(12) DEMANDE INTERNATIONALE PUBLIÉE EN VERTU DU TRAITÉ DE COOPÉRATION EN MATIÈRE DE BREVETS (PCT)

(19) Organisation Mondiale de la Propriété Intellectuelle
Bureau international

(43) Date de la publication internationale
16 mai 2013 (16.05.2013)

WIPO | PCT

(10) Numéro de publication internationale
WO 2013/067629 A1

(51) Classification internationale des brevets : *H03M 7/00* (2006.01)

(21) Numéro de la demande internationale : PCT/CA2012/001042

(22) Date de dépôt international : 13 novembre 2012 (13.11.2012)

(25) Langue de dépôt : français

(26) Langue de publication : français

(30) Données relatives à la priorité :
2758243 10 novembre 2011 (10.11.2011) CA

(71) Déposant : **PARALLEL GEOMETRY INC.** [CA/CA]; Bureau 2, 1160, avenue Laurier ouest, Montréal, Québec H2V 2L5 (CA).

(72) Inventeur : **ROTGE, Jean-François**; App. 15, 1525, rue Bernard, Outremont, Québec H2V 1W7 (CA).

(74) Mandataire : **FASKEN MARTINEAU DUMOULIN LLP**; Tour de la Bourse, C.P. 242, 800, Square Victoria, bureau 3700, Montréal, Québec H4Z 1E9 (CA).

(81) États désignés *(sauf indication contraire, pour tout titre de protection nationale disponible)* : AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) États désignés *(sauf indication contraire, pour tout titre de protection régionale disponible)* : ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), eurasien (AM, AZ, BY, KG, KZ, RU, TJ, TM), européen (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV,

[Suite sur la page suivante]

(54) Title : METHOD AND SYSTEM FOR CODING INFORMATION

(54) Titre : MÉTHODE ET SYSTÈME POUR CODER DE L'INFORMATION

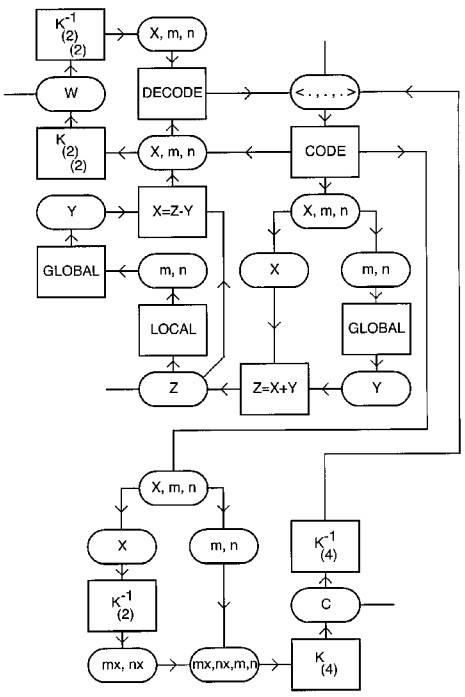


FIG.4

(57) **Abstract** : A method and an apparatus are described for coding information, the method comprising obtaining a list of integers to be coded; determining a hyperspatial pyramid having a size adapted for coding the list of integers, the hyperspatial pyramid having a plurality of vertices, the number of which is determined by the degree of the hyperspatial pyramid which is equal to the sum of the integers of the list of integers and by the dimension of the hyperspatial pyramid which is equal to the number of integers of the list of integers minus one; indexing the list of integers in this pyramid by means of an indexation system; and providing an indication of the indexation in the hyperspatial pyramid.

(57) **Abrégé** : Une méthode et un appareil sont décrits pour coder de l'information, la méthode comprenant obtenir une liste d'entiers à coder; déterminer une pyramide hyperspatiale ayant une taille adaptée pour coder la liste d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets dont le nombre est déterminé par le degré de la pyramide hyperspatiale qui est égal à la somme des entiers de la liste d'entiers et par la dimension de la pyramide hyperspatiale qui est égale au nombre d'entiers de la liste d'entiers moins un; indexer la liste d'entiers dans cette pyramide au moyen d'un système d'indexation; et fournir une indication de l'indexation dans la pyramide hyperspatiale.

WO 2013/067629 A1

MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Publiée :

— *avec rapport de recherche internationale (Art. 21(3))*

— *avant l'expiration du délai prévu pour la modification des revendications, sera republiée si des modifications sont reçues (règle 48.2.h))*

MÉTHODE ET SYSTÈME POUR CODER DE L'INFORMATION

Référence croisées à des demandes correspondantes

Cette demande de brevet revendique la date de priorité créé par le dépôt de la demande canadienne n°2,758,243, dont le titre est "*Méthode et appareil pour coder de l'information*" qui a été déposée le 10 novembre 2011.

Domaine d'application

Cette invention a trait au domaine du codage. Plus précisément, cette innovation a trait à une méthode et à un système pour coder de l'information au moyen d'une pyramide hyperspatiale.

Art antérieur

Soit $x_{0,\ldots,n}$ des nombres entiers d'une liste de nombre entiers et X le nombre entier généré; coder une liste d'entiers naturels en un seul entier et son inverse reviennent à trouver des fonctions de codage et décodage assurant la correspondance suivante :

Code $(x_{0,\ldots,n}) \rightarrow X \Leftrightarrow$ Decode $(X) \rightarrow x_{0,\ldots,n}$

Un système de codage consiste à établir une loi de correspondance appelée code entre les informations à représenter et les nombres obtenus, de telle sorte qu'à chaque information corresponde un et généralement un seul nombre à partir duquel il sera possible de retrouver les informations initiales. Certaines stratégies de codage reposent sur des mécanismes bijectifs fonctionnels contenus par exemple dans le théorème fondamental de l'arithmétique ou dans des méthodes de génération polynomiale des nombres. Dans le premier cas, l'unicité de la décomposition d'un nombre en un produit de facteurs premiers est la clé du mécanisme bijectif. Dans le second cas, l'unicité de la décomposition d'un nombre en une somme de nombres élémentaires doit être préalablement établie.

Par exemple, le codage de Fibonacci permet de concaténer une liste d'entiers en un nombre binaire tel que les séparateurs entre les nombres entiers soient représentés par le pattern binaire 11.

$$Fibonacci([1,2,3,9,8,7]) \rightarrow \{1101100111000110000110101 1\}_2 \rightarrow \{57088107\}_{10}$$

$$\overbrace{1}^{1}\overbrace{1}^{2}\overbrace{0}^{3}\overbrace{1}^{9}\overbrace{1}^{8}\overbrace{0}^{7}\,1\,0\,0\,0\,1\,1\,0\,0\,0\,0\,1\,1\,0\,1\,0\,1\,1$$

La personne versée dans le domaine appréciera que de manière générale, au delà, du choix théorique de la méthode de codage, la longueur de la liste à coder et la taille des nombres entiers à traiter doivent être considérées comme un problème essentiel du point de vue pratique et applicatif. La personne versée dans le domaine appréciera aussi qu'en dehors du problème de compression de la liste d'entiers, la taille du code généré, c'est à dire son nombre de digits représentatifs, est un élément fondamental dans le choix des méthodes de codage. Compte tenu de la taille des données à traiter et des niveaux de complexité algorithmique rencontrés, les phénomènes d'écroulement des performances, particulièrement lors des phases de décodage, sont pratiquement immédiats et disqualifient certaines méthodes théoriquement intéressantes en ce qui concerne la phase initiale de codage.

Les différentes familles de codage abordant un certain nombre de problématiques reliées par exemple aux thématiques de compression, de transmission, de structuration, ou de cryptage des données, les propriétés des codes générés peuvent être dans ce cas très différentes. Dans les systèmes de codage à indexation, la loi de correspondance est une relation d'ordre qui calcule le rang d'une liste ou séquence de nombres entiers, et qui permet de classer ou de trier cette liste à partir d'un nombre entier unique qui pourra à son tour être décodé pour restituer la liste.

La personne versée dans le domaine appréciera qu'il y ait de nombreuses applications possibles au codage. En particulier, l'ordonnancement de certaines séquences d'entiers est généralement le premier objectif des méthodes de codage par indexation. Les séquences à ordonner relèvent essentiellement de la combinatoire et sont par exemple des combinaisons ou des permutations qui doivent être ordonnées. Les applications sont vastes allant de l'algèbre linéaire, géométrie algébrique, théorie des graphes au classement lexicographique et alphabétique de dictionnaires ou à l'énumération et au positionnement de combinaisons en théorie des jeux.

L'indexation des treillis est particulièrement importante dans les problèmes de routage et d'algorithmique parallèle. Ainsi le routage peut être défini comme un mécanisme logiciel ou matériel qui achemine les informations entre la source et la destination lors d'une communication dans un réseau incomplètement connecté de type statique ou dynamique : il effectue le choix des chemins lorsqu'il y en a plusieurs, il gère les conflits, enfin il assure que les messages arrivent bien et dans le bon ordre.

L'adressage des données et de leurs zones de mémorisation dans des bases ou des structures de données est à la base de l'accès à l'information numérique et à sa structuration. L'intérêt du codage par indexation de l'information est de se libérer de toute organisation physique de la mémoire, préalable à la manipulation des données. Cette organisation physique est typiquement articulée autour des notions de tableaux ou de listes dont la structure, la taille et la localisation mémoire doivent être explicitement déclarées. À l'opposé, le codage par indexation privilégie une description implicite de l'information au travers des propriétés du nombre encodé. Par exemple, la représentation de l'information dans des tableurs utilise des formats standards de description qui spécifient les conventions d'organisation des fichiers. Ces formats, comme par exemple le Comma Separated Value, décrivent de manière explicite de quelle manière l'information doit être enregistrée.

La personne versée dans le domaine appréciera de plus que les fonctions d'indexation des treillis peuvent être utilisées comme de puissantes fonctions de hachage. Afin d'optimiser l'accès à l'information, les techniques de hachage permettent à partir de fonctions mathématiques dîtes de hachage de calculer directement l'indice d'une table à partir d'une chaine alphanumérique appelée la clé. La zone mémoire directement indexée contient alors les informations associées à la clé ou un nouveau jeu d'adresses permettant d'accéder indirectement à ces adresses.

La personne versée dans le domaine appréciera aussi que l'identification électronique est de plus en plus utilisée pour assurer la traçabilité des objets, des aliments ou des êtres vivants. Les différentes typologies d'identifiants sont les identifiants pour l'immatriculation, pour l'indexation de l'information : la clé primaire, ou alors pour la connexion. Les systèmes de codage par indexation sont particulièrement bien adaptés à

ces types de problème car ils facilitent entre autres les tris alphanumériques et permettent une identification rapide dans les bases de données.

Enfin, les fonctions d'encodage ou de couplage mathématique sont souvent à la base des systèmes de cryptage et peuvent être à la base de la relation biunivoque entre l'opération de codage et de décodage. L'utilisation des propriétés des nombres codés, comme par exemple la décomposition en produit de nombres premiers, peut également servir d'approche indépendante ou complémentaire.

Dans le codage positionnel, les systèmes de numération positionnelle sont utilisés pour coder et décoder des vecteurs de nombres entiers à partir d'une base de numération supérieure au cardinal k - 1 de l'alphabet de codage. Le codage d'un vecteur $(\alpha_0, \bullet\bullet\bullet \alpha_K)$ en base k sera alors

$$I_B = \alpha_0.B^k + \alpha_1.B^{k-1} + \cdots + \alpha_k.B^0$$

Si par exemple les entiers (1, 2, 3, 4, 5, 6, 7, 8, 9) sont associés aux lettres de l'alphabet (A, B, C, D, E, F, G, H, I), le code suivant sera obtenu $2.10^6 + 1.10^5 + 2.10^4 + 2.10^3 + 1.10^2 + 7.10^1 + 5.10^0 \rightarrow \{2122175\}_{10}$ correspondant au mot BABBAGE codé en base 10. Dans ce cas, l'alphabet ne dépassant pas le nombre de digits disponible, le code obtenu correspond à une simple substitution des lettres du mot par les digits associés. En enrichissant l'alphabet, le calcul polynomial devient nécessaire, les lettres pouvant être codées sur plus de un digit, par exemple en numération hexadécimale avec $A = 10$, $B = 11 \bullet\bullet\bullet$.

La personne versée dans le domaine appréciera que le codage d'une liste d'entiers puisse être effectué à partir du système de représentation modulaire des nombres entiers ou Residue Number System (RNS). Le système RNS est défini à partir d'un ensemble de nombres premiers deux à deux nommés modules et noté $\{m_0, m_1, \bullet\bullet\bullet, m_n\}$. Par conséquent $PGCD(m_i, m_j) = 1$, *avec* $i \neq j$. Un nombre entier X peut être alors représenté par un ensemble d'entiers appelés résidus $\{r_0, r_1, \bullet\bullet\bullet, r_n,\}$ tel que $x_i = X \ mod \ m_i$ avec $0 \leq x_i < m_i$. Cette représentation est d'autre part unique pour $X \leq M$ - 1 avec $M = \prod_{i=0}^{n-1} m_i$. Le principe de codage est donc de considérer la liste des nombres entiers à coder comme

une liste de résidus et de trouver la représentation décimale associée. On peut pour trouver des modules supérieurs aux entiers respectifs de la liste, utiliser une séquence de nombres premiers successifs tous supérieurs à leurs résidus respectifs ou débutant par le nombre premier immédiatement supérieur au nombre entier maximum de la liste à coder. L'exemple ci-après permet de coder le point projectif [69, 78, 73, 71, 77, 65] en dimension 5 à partir du module 73. La fonction Chinois de codage est essentiellement une fonction permettant de résoudre un système de congruences à partir d'une liste d'entiers résidus.

Chinois ([69, 78, 73, 71, 77, 65], [73, 79, 83, 89, 97, 101]) → {252617134512}

252617134512 *mod* 73 = 69 • • • 252617134512 *mod* 101 = 65

Le décodage du nombre décimal n'étant pas unique, il en résulte qu'il faut connaître le triplet (*Code, module initial, longueur de la liste*) pour effectuer le décodage. Le triplet nécessaire sera donc noté $\{252617134512\}_{(73,5)}$. La liste pourrait être codée avec une infinité d'autres triplets comme par exemple $\{412251249819\}_{(79,5)}$, 79 étant dans ce cas le premier nombre entier supérieur à 78.

Le codage de Cantor est associé à la notion de fonction de couplage développée pour répondre au problème d'énumérabilité des entiers naturels. La fonction de couplage de Cantor se ramène à un polynôme du second degré ayant les abscisses et les ordonnées du plan cartésien comme variable. La figure 1 présente l'ordre dans lequel le polynôme associe les couples de coordonnées aux indices selon la formule suivante. Par exemple $I_{K_2}(1,2) \rightarrow 8$. Il est possible d'obtenir un second polynôme de couplage en substituant x à y et dans ce cas $I_{K_2}(1,2) \rightarrow 7$.

$$I_{K_2} = \frac{(x+y)*(x+y+1)}{2} + y$$

La variante illustrée à la figure 1b présente une variante boustrophédonique.

La fonction de couplage de Cantor permettant le codage et le décodage d'une paire d'entiers naturels a été étendue à un nombre quelconque d'entiers de deux manières

différentes. La première revient à regrouper les entiers en constituant de manière successive des paires de paires d'entiers. Cette méthode a l'avantage de la simplicité de la fonction de codage initiale de Cantor : un polynôme de degré 2 mais génère des polynômes de degré n^2. Cette croissance exponentielle affecte en particulier la taille des indices calculés. Une alternative à cette approche est l'extension à n-dimensions de la fonction de couplage. Une expression à base de coefficients binomiaux est présentée par Cegielski et Richard sous la forme suivante :

$$K(x_1,\ldots,x_k)=\binom{x_1+\ldots+x_k+k-1}{k}+\binom{x_1+\ldots+x_{k-1}+k-2}{k-1}+\cdots$$

$$\cdots+\binom{x_1+x_2+1}{2}+\binom{x_1}{1}.$$

Malheureusement le codage hyperspatial de Cantor n'est pas bijectif. Des vecteurs de dimensions différentes peuvent générer le même code. Ce problème est aussi existant avec le couplage successif par paires. Dans ce cas, on peut par convention ajouter l'arité du vecteur à coder comme coordonnée supplémentaire.

Dans le codage d'Ernesto Pascal, un entier N peut être représenté d'une manière unique pour un certain degré n de la manière suivante :

$$N=\binom{x_n}{n}+\cdots+\binom{x_2}{2}+\binom{x_1}{1}, x_n>\cdots>x_2>x_1\geq 0.$$

Le nombre 1000 sera par exemple représenté par la séquence $\{12,9,8,7,6,5,3\}^7_{Cns}$ correspondant aux sept premières colonnes du triangle de Pascal ou alors par la séquence $\{19,8,3\}^3_{Cns}$ correspondant aux trois premières colonnes. Le codage s'applique donc à des séquences d'entiers strictement décroissantes et est utilisé pour solutionner le problème d'ordonnancement lexicographique des r-combinaisons de $a_1,a_2,\cdots,a_k$ de $\{1,2,\cdots,r\}$. La formule peut être légèrement modifiée pour calculer l'indice d'une combinaison donnée sous la forme d'une séquence d'entiers croissante et il est obtenu :

$$\binom{r}{k}-\binom{r-a_1}{k}-\binom{r-a_2}{k-1}-\cdots-\binom{r-a_{k-1}}{2}-\binom{r-a_k}{1}.$$

Il y a donc un besoin pour une méthode de codage de l'information qui résout au moins un problème de l'art antérieur.

Liste des figures

La figure 1a illustre l'ordre cantorien alors que figure 1b illustre l'ordre cantorien boustrophédonique.

La figure 2 illustre le triangle de De Gua de Malves et son extension projective.

La figure 3 illustre un treillis pyramidal et l'étiquetage des coordonnées projectives.

La figure 4 illustre une réalisation du système de codage et de décodage.

La figure 5 illustre une implémentation de l'algorithme GLOBAL et une implémentation de l'algorithme LOCAL.

La figure 6 illustre des arbres récursifs pour $\binom{4}{2}$.

La figure 7 illustre l'indexation positionnelle pour $T_{3,2}$ en base quaternaire.

La figure 8 illustre l'expansion des grands entiers.

La figure 9 illustre l'ordre numérique pour le treillis T4,3.

La figure 10 illustre une implémentation de l'algorithme NUMORD.

La figure 11 illustre une décomposition pyramidale d'un nombre.

La figure 12 illustre une implémentation de l'algorithme ORDNUM.

La figure 13 illustre le codage lexicographique du treillis T4,3.

La figure 14 illustre une implémentation de l'algorithme LEXORD.

La figure 15 illustre une implémentation de l'algorithme ORDLEX.

La figure 16 illustre le codage de Gray projectif (4,3).

La figure 17 illustre une implémentation de l'algorithme GRAORD.

La figure 18 illustre le parcours de T4,3 en Gray projectif.

La figure 19 illustre une implémentation de l'algorithme ORDGRA.

La figure 20 illustre une implémentation de l'algorithme VN2VM et de l'algorithme VM2VN.

La figure 21 illustre l'arbre combinatoire $\binom{6}{4}$.

La figure 22 illustre l'arbre $\binom{6}{4}$ et les étiquettes numériques.

La figure 23 illustre l'arbre $\binom{6}{2}$ et les étiquettes lexicographiques.

La figure 24 illustre un exemple de sous-espace projectif combinatoire.

La figure 25 illustre un exemple de bijection entre points projectifs et combinaisons.

La figure 26 illustre un exemple de génération projective et binaire des combinaisons.

La figure 27 illustre les coordonnées euclidiennes et affines d'un point cantorien.

La figure 28 illustre la taxonomie des ordres.

La figure 29 illustre des chemins cantoriens.

La figure 30 illustre une implémentation possible de la méthode pour coder de l'information.

La figure 31 illustre des chemins en spirale.

La figure 32 illustre une implémentation de l'algorithme SPIORDc et de l'algorithme ORDSPIc.

La figure 33 illustre une implémentation de l'algorithme SPIORDa et de l'algorithme ORDSPIa.

La figure 34 illustre une chaine en spirale pour $T_{6,3}$.

La figure 35 illustre un mécanisme de convolution.

La figure 36 illustre des convolutions de Vandermonde pour $\binom{9}{3}$.

La figure 37 illustre une implémentation de l'algorithme CRKREC.

La figure 38 illustre une implémentation de l'algorithme XENUME et de l'algorithme YENUME.

La figure 39 illustre une implémentation de l'algorithme COMBO.

La figure 40 illustre le calcul optimisé du coefficient binomial.

La figure 41 illustre une implémentation de l'algorithme PRODYN.

La figure 42 illustre une implémentation de l'algorithme MATPRT.

Les figures 43a et 43b illustrent une multiplication matricielle entre colonnes.

La figure 44 illustre une courbe qui représente pour une dimension donnée les valeurs des coefficients binomiaux.

La figure 45 illustre une implémentation de l'algorithme ENCDEC et de l'algorithme DIGBIN.

La figure 46 illustre une implémentation de l'algorithme NUMFCT.

La figure 47 illustre une implémentation de l'algorithme DRVREC.

La figure 48 illustre une implémentation de l'algorithme DRVSYM.

La figure 49 illustre une implémentation de l'algorithme BINRP4.

La figure 50 illustre une implémentation de l'algorithme BINRPNi et de l'algorithme RPNBINi.

La figure 51 illustre une implémentation de l'algorithme GLOBALo et de l'algorithme LOCALo.

La figure 52 illustre une implémentation de l'algorithme ORDNUMo et de l'algorithme ORINUMo.

La figure 53 illustre une implémentation de l'algorithme INDCOM et de l'algorithme COMIND.

La figure 54 illustre une implémentation de l'algorithme MPLNUM.

Sommaire

Selon un aspect de l'invention, une méthode est décrite pour coder de l'information, la méthode comprenant obtenir une liste d'entiers à coder; déterminer une pyramide hyperspatiale ayant une taille adaptée pour coder la liste d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets dont le nombre est déterminé par le degré de la pyramide hyperspatiale qui est égal à la somme des entiers de la liste d'entiers et par la dimension de la pyramide hyperspatiale qui est égale au nombre d'entiers de la liste d'entiers moins un; indexer la liste d'entiers dans cette pyramide au moyen d'un système d'indexation; et fournir une indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale afin de coder ainsi ladite liste d'entiers à coder.

Selon une réalisation de la méthode décrite ci-avant, la méthode comprend de plus fournir une liste de codes alphanumériques et transformer la liste de codes alphanumériques en liste d'entiers à coder.

Selon une réalisation de la méthode décrite ci-avant, le système d'indexation permet d'indexer chacun de la pluralité de sommets de ladite pyramide hyperspatiale.

Selon une réalisation de la méthode décrite ci-avant, l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale comprend un entier représentant l'indice du point projectif dans le système d'indexation, m le degré total de ladite liste d'entiers et n le nombre d'entiers de la liste d'entiers.

Selon une réalisation de la méthode décrite ci-avant, la méthode comprend de plus transformer l'entier représentant l'indice du point projectif par des coordonnées combinatoires mx et nx.

Selon une réalisation de la méthode décrite ci-avant, la méthode comprend de plus la transformation de m le degré total de la liste d'entiers et n le nombre d'entiers de la liste d'entiers par un entier Y représentant un indice global.

Selon une réalisation de la méthode décrite ci-avant, la méthode comprend de plus effectuer la sommation de l'entier Y représentant un indice global à l'entier représentant l'indice du point projectif pour fournir un indice total Z, l'indication de l'indexation dans la pyramide hyperspatiale comprenant l'indice total Z.

Selon une réalisation de la méthode décrite ci-avant, la liste d'entiers est obtenue d'un fichier.

Selon une réalisation de la méthode décrite ci-avant, l'étape de fournir l'indication de l'indexation dans la pyramide hyperspatiale comprend le stockage de ladite indication de l'indexation dans un fichier.

Selon une réalisation de la méthode décrite ci-avant, l'étape de fournir l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale comprend la transmission de ladite indication de l'indexation de la liste d'entiers à une unité de traitement par l'intermédiaire d'un réseau.

Selon une réalisation de la méthode décrite ci-avant, la méthode est utilisée pour coder une liste d'entiers relatifs, la méthode comprenant fournir la liste d'entiers relatifs et générer la liste d'entiers au moyen de la liste d'entiers relatifs.

Selon une réalisation de la méthode décrite ci-avant, l'étape de fournir l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale comprend l'affichage de ladite indication de l'indexation de la liste d'entiers sur un écran.

Selon une autre réalisation, la méthode décrite ci-avant est utilisée pour compresser une chaine de caractères dans laquelle la chaine de caractères est associée à ladite liste d'entiers.

Selon une autre réalisation, la méthode décrite ci-avant est utilisée pour coder une liste de listes, la méthode comprenant l'utilisation itérative de la méthode décrite ci-avant pour coder une liste de listes.

Selon une autre réalisation, la méthode décrite ci-avant est utilisée pour crypter une liste d'entiers, dans laquelle l'indexation de la liste d'entiers correspond au cryptage de la liste d'entiers.

Selon un aspect large de l'invention une méthode est décrite pour décoder de l'information codée au moyen de la méthode pour coder de l'information décrite ci-avant, la méthode comprenant obtenir l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale; obtenir une indication d'un système d'indexation utilisé pour générer ladite indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale et déterminer au moyen de l'indexation et de l'indication d'un système d'indexation utilisé ladite liste d'entiers.

Selon un aspect large de l'invention, un dispositif de stockage d'information est décrit pour stocker des instructions exécutables par un processeur, qui quand elles sont exécutées vont causer l'exécution par le processeur d'une méthode pour coder de l'information, la méthode comprenant obtenir une liste d'entiers à coder; déterminer une pyramide hyperspatiale ayant une taille adaptée pour coder la liste d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets dont le nombre est déterminé par le degré de la pyramide hyperspatiale qui est égal à la somme des entiers de la liste d'entiers et par la dimension de la pyramide hyperspatiale qui est égale au nombre d'entiers de la liste d'entiers moins un; indexer la liste d'entiers dans cette pyramide au moyen d'un système

d'indexation; et fournir une indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale afin de coder ainsi ladite liste d'entiers à coder.

Selon un aspect large de l'invention, un ordinateur est décrit, l'ordinateur comprenant une unité d'affichage; un processeur; une unité de stockage des données comprenant un programme configuré pour être exécuté par le processeur, le programme permettant de coder de l'information, le programme comprenant des instructions pour obtenir une liste d'entiers à coder; des instructions pour déterminer une pyramide hyperspatiale ayant une taille adaptée pour coder la liste d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets dont le nombre est déterminé par le degré de la pyramide hyperspatiale qui est égal à la somme des entiers de la liste d'entiers et par la dimension de la pyramide hyperspatiale qui est égale au nombre d'entiers de la liste d'entiers moins un; des instructions pour indexer la liste d'entiers dans cette pyramide au moyen d'un système d'indexation; des instructions pour fournir une indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale afin de coder ainsi ladite liste d'entiers à coder.

Un avantage du système décrit est qu'il s'agit d'un système polymorphe "plusieurs en un" car il s'agit d'un système combiné de codage et d'indexation mais également de transcodage entre systèmes de codage combinatoire utilisant les coefficients binomiaux.

Un avantage du système décrit est qu'il s'adresse au codage/indexation/transcodage de listes d'entiers quelconques sans structures combinatoires prédéfinies aussi bien qu'à des listes particulières avec structures combinatoires prédéfinies comme les listes de coefficients de polynômes ou les listes de combinaisons sans répétitions.

Un autre avantage du système décrit est qu'il permet de travailler en mode codage bijectif Code (X) $\rightarrow$ Y et Decode (Y) $\rightarrow$ X ou en mode cryptage Code (X) $\rightarrow$ Y Decode (Y, clé 1, clé 2) $\rightarrow$ X.

Un autre avantage du système décrit est qu'il peut apporter une solution à différents problèmes tels la nature et la taille des données, la taille du programme de codage/décodage, la taille du code généré et les propriétés du code généré.

Ainsi pour ce qui est de la nature et la taille des données, la méthode décrite peut permettre l'utilisation de données structurées ou non de taille quelconque.

Pour ce qui est de la taille du programme de codage/décodage, il est possiblement implémentable au moyen de quelques dizaines d'instructions et d'une librairie standard de traitement des entiers à multiprécisions, comme par exemple GMP.

Pour ce qui est de la taille du code généré, il n'y a pas d'explosion de la taille que ce soit en mode compressé ou non du fait qu'il n'y a pas de factorisation par nombres premiers ou de calculs utilisant les exponentielles.

Pour ce qui est des performances du codage et du décodage, il peut être possible d'utiliser des algorithmes optimisés en utilisant des algorithmes de nouvelle génération pour le calcul et la recherche des coefficients binomiaux dans le triangle de Pascal.

Pour ce qui est des propriétés du code généré, le code généré peut être indexé multi-usage pour par exemple au moins un parmi l'ordonnancement des listes d'entier, les architectures de routage, l'adressage et le hachage de données, l'identification électronique et la cryptographie.

Description détaillée

La personne versée dans le domaine appréciera que le triangle de Pascal possède de nombreuses propriétés. En particulier, pour le dénombrement des monômes d'un polynôme homogène de degré *m* en dimension n :

$$Cr(m,n)=\binom{m+n}{n}=\frac{(m+1)\cdots(m+n)}{n!}.$$

Il est aussi à noter que $Cr(m,n)=Cr(n,m)$ et conséquemment l'écriture du coefficient binomial central de la ligne *r*, $Cr(r,r)$. Par ailleurs, la somme des coefficients binomiaux de cette ligne a pour valeur :

$$\sum_{k=0}^{r}\binom{r}{k}=2^r.$$

Le calcul d'un coefficient binomial à partir des coefficients binomiaux situés dans la colonne adjacente de gauche ou alors dans la diagonale juste supérieure est respectivement :

$$\binom{r}{k}=\sum_{i=0}^{r-1}\binom{i}{k-1}, \qquad \binom{r}{k}=\sum_{i=0}^{r-1}\binom{k+i}{i}.$$

De plus, la formule par récurrence calculant un coefficient binomial à partir de ses deux voisins supérieurs permet de construire de manière naïve le triangle de Pascal et s'exprime :

$$\binom{r}{k}=\binom{r-1}{k-1}+\binom{r-1}{k} \quad avec \quad r\geq 1.$$

Les nombres pyramidaux sont des nombres figurés, c'est à dire des nombres décomposés et visualisés à partir de points répartis sur des polytopes réguliers. Le triangle de Pascal est formé à partir de nombres triangulaires, tétraédriques et généralement hypertétraédriques.

Le système divulgué est conçu à partir de l'idée d'utiliser les propriétés combinatoires du triangle de Pascal et les propriétés algébriques des extensions projectives hyperspatiales du triangle de De Gua de Malves comme système général d'indexation et de codage des listes d'entiers naturels. Pour cela, la correspondance biunivoque entre les monômes d'un polynôme $P_{m,n}$ et les points d'un treillis $T_{m,n}$ va être utilisée. L'ordonnancement des monômes de ce polynôme sera par conséquent équivalent à l'ordonnancement des points de ce treillis. La figure 2 montre la transition entre le triangle original décrit par De Gua et son extension projective en deux dimensions pour une courbe du troisième degré.

Définition et propriétés des treillis pyramidaux

Il sera apprécié par la personne versée dans le domaine que directement lié à la notion de nombre pyramidal, un treillis pyramidal est autant une structure combinatoire qu'une structure géométrique hyperspatiale dont l'étude relève de la géométrie algébrique. Les treillis se présentent sous forme de pyramides hyperspatiales, comme par exemple le

tétraèdre de la figure 3 ou le triangle de la figure 2. Un treillis pyramidal tétraédrique, noté $T_{m,n}$ correspondant à la ligne $m+n$ et à la colonne m du triangle de Pascal, est associé à chaque coefficient binomial $\binom{m+n}{m}$. C'est donc une configuration projective hypertétraédrique de dimension n possédant $\binom{m+n}{m}$ points homogènes, linéairement dépendants. Les points du treillis sont organisés de manière combinatoire, en sous-treillis de dimension $n-1$ et de degré allant de 0 à $m-1$ (règle de la sommation de la diagonale). Chaque treillis est localisé dans le triangle de Pascal par le couple (m,n) ou par la sommation de tous les points de tous les treillis précédents, ce que nous nommerons l'indice global. Le nombre de points, le nombre pyramidal associé, est égal par définition au nombre de monômes d'un polynôme homogène de degré m en n dimensions. Les coordonnées homogènes projectives des points d'un treillis $T_{m,n}$ peuvent donc être interprétées comme les exposants des monômes d'un polynôme associé de degré m et de dimension n.

Chaque case du triangle de Pascal correspond à une pyramide hyperspatiale dont le nombre de sommets est égal au nombre indiqué dans la case.

Ordre des monômes des polynômes homogènes

Un monôme x_n^m en $x_0, x_1, ..., x_k$ est défini comme le produit $x_n^m = x_{n_0}^{m_0} x_{n_1}^{m_1} ... x_{n_k}^{m_k}$ et le degré total de x_n^m est défini comme $|m| = m_0 + m_1 + ,..., + m_k$ si $vm = \langle m_k, m_{k-1}, ..., m_0 \rangle$, représente le vecteur des exposants, et $vn = \langle n_0, n_1, ..., n_k \rangle$, représente le vecteur des indices des variables avec $n_0 \le n_1 ... \le n_k$.

Les définitions euclidiennes des ordres des monômes des polynômes non homogènes sont basées sur la notion d'ordre lexicographique et de degré total des monômes, c'est à dire la somme des exposants des variables des monômes. En combinant ces deux notions, on obtient en particulier, l'ordre lexicographique pur, l'ordre total puis lexicographique, l'ordre total puis lexicographique inverse. Dans le cas des polynômes homogènes, tous les

monômes ont par définition le même degré total et l'ordonnancement des monômes peut être réalisé à partir de deux ordres projectifs lexicographiques duals, l'un sur les exposants et l'autre sur les indices, les monômes étant représentés en notation tensorielle. Les ordres lexicographiques inverses sont dans ce cas de simples variantes ascendantes ou descendantes des ordres lexicographiques.

Sera nommé ordre numérique l'ordre lexicographique sur les exposants tel que $vm_\alpha \succ_{num} vm_\beta$ si et si seulement la valeur la plus à gauche non nulle de $vm_\alpha - vm_\beta$ est négative.

Sera nommé ordre numlex l'ordre numérique sur les exposants tel que $vm = \langle m_k, m_{k-1}, ..., m_0 \rangle$ avec $m_0 \geq m_1 ... \geq m_k$.

Sera nommé ordre combinatoire l'ordre numérique sur les exposants tel que $vm = \langle m_k, m_{k-1}, ..., m_0 \rangle$ avec $m_0 > m_1 ... > m_k$.

Sera nommé ordre lexicographique l'ordre lexicographique sur les indices des variables tel que $vn_\alpha \succ_{lex} vn_\beta$ si et si seulement la valeur la plus à gauche non nulle de $vn_\alpha - vn_\beta$ est négative.

Si le polynôme de De Gua de Malves de la figure 2 correspondant à une courbe euclidienne du troisième degré est ordonné dans l'ordre lexicographique de ses coefficients, il est obtenu :

$$a, b.y, c.x, e.y^2, f.x.y, g.x^2, h.y^3, i.x.y^2, k.x^2.y, l.x^3.$$

Après avoir substitué x par $x0/x2$ et y par $x1/x2$, ramené l'ensemble des termes au même dénominateur et finalement conservé les numérateurs, la représentation homogène du polynôme est obtenue :

$$a.x_2^3, b.x_1x_2^2, c.x_0x_2^2, e.x_1^2x_2, f.x_0x_1x_2, g.x_0^2x_2, h.x_1^3, i.x_0x_1^2, k.x_0^2x_1, l.x_0^3.$$

En notant $vm_a = \langle 3,0,0 \rangle$, $vm_b = \langle 2,1,0 \rangle$ ... $vm_l = \langle 0,0,3 \rangle$, il est possible d'obtenir en ordre numérique inverse et en ordre numérique les séquences de monômes suivants :

$\langle a,b,c,e,f,g,h,i,k,l\rangle$

$\langle l,k,i,h,g,f,e,c,b,e\rangle$.

En notant $vn_a = \langle 2,2,2\rangle$, $vn_b = \langle 1,2,2\rangle$ ... $vn_l = \langle 0,0,0\rangle$, il est obtenu en ordre lexicographique : $\langle l,k,g,i,f,c,h,e,b,a\rangle$.

Principes de codage combinatoire

En utilisant les propriétés combinatoires et projectives des treillis, il est possible de définir un système général d'indexation des monômes des polynômes et simultanément l'utiliser comme système de codage de listes d'entiers naturels : les coordonnées. Il sera apprécié par la personne versée dans le domaine que par transposition des entiers en codes alphanumériques quelconques, le système de codage pourra être étendu à l'encodage de texte et de chaînes de caractères alphanumériques.

Le système divulgué se base donc sur les principes suivants :

1. A chaque point d'un treillis (m,n), aussi appelé sommet d'une pyramide hyperspatiale, est associé un vecteur de coordonnées homogènes de type $[a_0 : a_1 : ... : a_n]$. La somme des coordonnées est égale à m. La position d'une case dans le triangle de Pascal et par conséquent de la pyramide hyperspatiale et de l'hypersurface associée est donnée par la ligne et la colonne du triangle de Pascal. Les lignes et les colonnes associées à une case du triangle de Pascal correspondent aux degrés et aux dimensions algébriques des polynômes associés. Chaque sommet d'une même pyramide : sa pyramide de support, possède une signature formée d'un nombre égal de coordonnées dont la somme représente le degré et la longueur la dimension moins un. Cette signature peut être interprétée en théorie combinatoire soit comme les coordonnées projectives du point sommet appartenant à la pyramide, soit comme les puissances des monômes du polynôme représentant l'hypersurface associée, soit encore comme les indices des variables composant les monômes de ce même polynôme.

2. Le vecteur de coordonnées homogènes de type $[a_0 : a_1 : ... : a_n]$ est considéré comme le vecteur des exposants d'un monôme de degré total m, appartenant à un polynôme de degré m en n dimensions.

3. A chaque treillis est associé un ordre de parcours, aussi appelé indication de l'indexation, des points sur des chemins hamiltoniens particuliers passant par tous les points du treillis selon une réalisation de l'invention.

4. L'indice local de tout monôme est considéré comme l'indice du point homogène associé, c'est à dire le numéro du point dans l'ordre de parcours du chemin choisi.

5. Selon une réalisation de l'invention, tout vecteur de coordonnées est codé ou décodé en mode local grâce au triplet (indice local, degré total du vecteur, dimension $(n+1)$).

6. Selon une réalisation de l'invention, tout vecteur de coordonnées est codé ou décodé, en mode global, grâce à l'indice total obtenu en additionnant l'indice local et l'indice global.

7. Selon une réalisation de l'invention, la notion de vecteur de coordonnées est étendue à tout ensemble de nombres entiers naturels représentant des codes alphanumériques quelconques.

8. Le codage revient à trouver des systèmes bijectifs de numérotation ou d'indexation des messages en effectuant une double numérotation d'abord dans le triangle de Pascal, de la pyramide hyperspatiale associée au message donné : la pyramide de support, puis ensuite du point-sommet de la pyramide dont la signature représente le message. Le nombre entier ou le n-uplet, généralement le triplet de nombres entiers, résultant de cette double numérotation représente le message codé sous forme indexée. Le codage peut être basé sur différentes méthodes de numérotation et de parcours des points-sommets sur une même pyramide de support et le code résultant diffère alors d'une méthode à l'autre.

9. L e décodage revient à retrouver à partir du code la pyramide de support du message puis la signature du point-sommet appartenant à la pyramide. Cette double localisation ne peut s'effectuer qu'en connaissant à priori la méthode de parcours choisie pour la numérotation des points-sommets sur la pyramide de support.

Ainsi et tel qu'expliqué ci-après il est possible de définir une méthode pour coder de l'information, la méthode comprenant obtenir une liste d'entiers à coder; déterminer une pyramide hyperspatiale ayant une taille adaptée pour coder la liste d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets tel que le degré de la pyramide hyperspatiale est égal à la somme des entiers de la liste d'entiers et la dimension de la pyramide hyperspatiale est égale au nombre d'entier de la liste d'entiers moins un; indexer la liste d'entiers dans cette pyramide au moyen d'un système d'indexation; et fournir une indication de l'indexation dans la pyramide hyperspatiale.

Dans une réalisation, la liste d'entiers est obtenue d'un fichier. La personne versée dans le domaine appréciera que divers types de fichiers peuvent être utilisés pour stocker la liste d'entiers.

Il sera aussi apprécié que l'étape de fournir l'indexation de la liste d'entiers dans la pyramide hyperspatiale pourra comprendre dans une réalisation le stockage de indexation dans un fichier. La personne versée dans le domaine appréciera que divers types de fichiers pourront être utilisés pour ce stockage.

Alternativement, il sera apprécié par la personne versée dans le domaine que l'étape de fournir l'indexation de la liste d'entiers dans la pyramide hyperspatiale pourra comprendre dans une réalisation la transmission de la liste d'entiers à une unité de traitement par l'intermédiaire d'un réseau. La personne versée dans le domaine appréciera que divers types de réseaux pourront être utilisés. En fait, le réseau pourra comprendre au moins un d'un réseau local, d'un réseau de type MAN et d'un réseau de type WAN. Dans une réalisation, la transmission aura lieu à travers le réseau Internet.

Il sera aussi apprécié que dans une réalisation, une indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale comprendra l'affichage de ladite indication de l'indexation de la liste d'entiers sur un écran.

Il sera apprécié par la personne versée dans le domaine qu'une méthode est décrite pour décoder de l'information codée au moyen de la méthode pour coder de l'information décrite ci-avant. Dans ce cas, la méthode comprendra les étapes d'obtenir l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale. La méthode comprendra de plus l'étape d'obtenir une indication d'un système d'indexation utilisé pour générer ladite indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale. La méthode comprendra de plus l'étape de déterminer au moyen de l'indexation et de l'indication d'un système d'indexation utilisé ladite liste d'entiers, et enfin, l'étape de fournir ladite liste d'entiers.

Le diagramme de la figure 4 illustre une réalisation permettant de fournir plusieurs options de codage et décodage pour une liste d'entiers $<.,...,.>$.

En particulier la fonction *CODE* est une famille de fonctions de codage qui indexent la liste d'entiers $<.,...,.>$ par rapport au treillis $T_{m,n}$ où m est le degré total de la liste et $n-1$ sa longueur. Le résultat est un triplet $<X,m,n>$, X, représentant le nombre entier correspondant à l'indice du point projectif ayant la liste initiale d'entiers en entrée comme vecteur de coordonnées homogènes.

La personne versée dans le domaine appréciera que l'indice calculé est différent selon la méthode de codage adoptée. Selon les cas de figure, ce triplet pourra être fourni à la fonction *DECODE* qui restituera la liste initiale ou alors séparée en deux parties afin de constituer un seul entier Z qui pourra être décodé de manière bijective sans aucune autre information.

La partie de gauche du diagramme de la figure 4 fait apparaître un autre moyen de constituer un seul entier W, permettant un décodage bijectif. Il s'agit d'une famille de fonctions notées K_{2_2}, qui couplent le triplet deux à deux, m et n produisant un nombre entier, lui-même couple à X. Des fonctions de décodage en deux dimensions appliquées successivement permettent alors de restituer le code initial.

La partie droite du diagramme de la figure 4 correspond quant à elle à un schéma d'encodage-décodage bijectif couplé à un mécanisme de compression combinatoire. Dans

ce cas, le triplet $<X,m,n>$ est d'abord transformé en quadruplet, l'entier X étant remplacé par des coordonnées combinatoires particulières mx et nx. Le quadruplet $\langle mx,nx,m,n\rangle$ est ensuite codé par un schéma de codage en quatre dimensions pour produire un code C. Ce type de code bijectif, peut être décodé par un schéma de décodage inverse en quatre dimensions. Le principe de compression réside dans la diminution importante du degré du quadruplet par rapport au degré du triplet qui compense très largement l'augmentation d'un caractère supplémentaire entre le triplet et le quadruplet.

L'indexation d'un treillis permet de coder avec un seul entier le degré et la dimension du treillis. L'indexation d'un treillis $T_{m,n}$ peut être réalisée en dénombrant le nombre de cases du triangle de Pascal précédant la case de coordonnées $x=m$ et $y=m+n$ correspondant au coefficient binomial $\binom{m+n}{m}$. On effectue donc la somme du nombre de cases du triangle précédant la m, ligne du treillis à laquelle on ajoute le degré du treillis correspondant à sa colonne. La formule trouvée correspond à l'indexation de Cantor des entiers naturels dans le plan cartésien.

$$I_c = \frac{(m+n)*(m+n+1)}{2} + m$$

Indexation globale et totale

L'indexation globale permet d'assurer la bijection entre les points d'un treillis et leur indexation dans un espace à n-dimensions. Au lieu de calculer le nombre de cases précédant le coefficient binomial du treillis, le nombre total de points de tous les treillis précédents est calculé. À partir de l'équation $\sum_{k=0}^{r}\binom{r}{k}=2^r$, et en notant que $2^0+\cdots+2^r=2^{r+1}$, la formulation de l'indice global I_g est obtenue en sommant le nombre des points des treillis des lignes allant de 0 à $m+n-1$ 1 et des points des treillis de la ligne $m+n$ allant de 0 à m :

$$I_g = 2^{m+n} + \sum_{i=0}^{m-1} Cm(i, m+n-1).$$

Le nombre total plus un, de points précédant le treillis $T_{7,5}$ sera obtenu de la manière suivante :

$$I_g(5,2) \rightarrow (128-1)+(1+7+21+35+35)+(1) = 227.$$

L'indice global ajouté à l'indice local, le numéro du point sur le treillis, donne l'indice total du point qui permet de retrouver le vecteur initial codé sans mémoriser le degré et la dimension du treillis. Il est à noter que le caractère bijectif du codage et du décodage a un prix à payer en termes de taille des indices générés et de temps de calcul. Sur ce dernier point, des versions optimisées des algorithmes *GLOBAL* et *LOCAL* qui suivent seront présentées plus loin de même que plusieurs méthodes de calcul de l'indice local.

Il sera appréciée par la personne versée dans le domaine que dans l'algorithme *GLOBAL* illustré à la figure 5, la variable *ind* est équivalente au nombre total de points des treillis précédant la ligne du treillis $T_{m+n,m}$ et la variable *acc* équivalente à la sommation des coefficients binomiaux sur la ligne du treillis. La fonction de décodage *LOCAL* illustrée à la figure 5 effectue la recherche inverse en localisant sur le triangle de Pascal, la ligne précédant le treillis puis en isolant le treillis sur sa ligne.

Les différentes formulations récursives des coefficients binomiaux permettent de considérer les arbres récursifs d'appel des fonctions comme des systèmes d'ordonnancement des points d'un treillis. Dans tous les cas, les conditions d'arrêt de la récursivité seront soit $\binom{r}{k} = 1, k = r$, soit $\binom{r}{k} = 1, k = 0$, et dans ce cas, le nombre de feuilles terminales toutes égales à un sera nécessairement égal au coefficient binomial recherché. La figure 6 illustre les appels récursifs des fonctions correspondant aux équations $\binom{r}{k} = \sum_{i=0}^{r-1} \binom{i}{k-1}$, $\binom{r}{k} = \sum_{i=0}^{r-1} \binom{k+i}{i}$ et $\binom{r}{k} = \binom{r-1}{k-1} + \binom{r-1}{k}$ *avec* $r \geq 1$ pour le calcul du coefficient binomial $\binom{4}{2}$. Les coefficients binomiaux correspondant à

chaque nœud de l'arbre fournissent le nombre de feuilles terminales de leurs sous-arbres respectifs. Les différents algorithmes de lecture et parcours des arbres fournissent donc des algorithmes d'indexation des feuilles terminales. En établissant une correspondance entre la production des feuilles terminales et des points du treillis, il est possible d'obtenir un ordonnancement de ces derniers. En particulier, les parcours d'arbres en profondeur consistent à traiter la racine de l'arbre et à parcourir récursivement les sous-arbres gauche et droit de la racine. Dans le parcours en profondeur préfixe ou lexicographique, la racine est traitée avant les appels récursifs sur les sous-arbres gauche, puis droit.

Codage projectif positionnel

La représentation des nombres sous forme positionnelle est utilisée pour coder la séquence des coordonnées d'un point homogène. Soit N ce nombre, a_i le symbole i et b la base de numération, par définition :

$$N = \sum_{i=0}^{n} a_i b^i \text{ avec } 0 \le a_i \le b-1.$$

La base de numération $m+1$ est utilisée où m est le degré du treillis de dimension n, et a_i est la coordonnée homogène de rang i.

L'indice positionnel I_p du point de coordonnées $[x_0 : \ldots : x_n]$ du treillis $T_{m,n}$ est :

$$I_p = \frac{1}{m} \sum_{i=0}^{n} x_i (m+1)^{n-i} = \sum_{i=0}^{n} \left(x_i \sum_{j=1}^{n-i} \binom{n-i}{j} m^j \right) + 1.$$

La figure 7 reprend l'exemple de l'indexation de la courbe cubique en 2 dimensions de la figure 2. Les indices sont calculés en base quaternaire. Par exemple, le point $[2:0:1]$ a pour indice $\frac{2.(4)^2 + 0.(4)^1 + 1.(4)^0}{3} = 11$. L'indice maximum du treillis étant égal à 16 pour un nombre total de 10 points, la figure fait apparaître une indexation creuse c'est à dire présentant des trous dans la numérotation.

La personne versée dans le domaine appréciera que le phénomène d'indexation creuse est l'un des désavantages du codage positionnel. En effet, pour un treillis $T_{m,n}$, la croissance du rapport $\rho_{m,n}$ entre l'indice maximum $(m+1)^n$ du codage positionnel et l'indice maximum correspondant d'un codage combinatoire est :

$$\rho_{m,n} = \frac{(m+1)^n}{\binom{m+n}{m}} = \frac{n!(m+1)^{n-1}}{(m+2)(m+3)...(m+n)}.$$

La figure 8 illustre les courbes représentatives de ρ en fonction de la croissance de m et n. Celles-ci permettent de visualiser l'expansion du nombre de digits nécessaires pour le codage positionnel. On obtient par exemple une valeur de $\rho_{10,10}$ égale à $\frac{285311670611}{184756}$.

La structure algébrique en anneau des polynômes permet immédiatement de montrer l'homomorphisme additif du codage positionnel avec $I_p(A+B) = I_p(A) + I_p(B)$. Soient en effet les points A de coordonnées $[a_0 :...: a_n]$ et B de coordonnées $[b_0 :...: b_n]$, grâce aux propriétés d'associativité et de distributivité, il est alors possible de vérifier la relation :

$$I_p(A+B) = \frac{1}{m}\sum_{i=0}^{n} a_i(m+1)^i + \frac{1}{m}\sum_{i=0}^{n} b_i(m+1)^i = \frac{1}{m}\sum_{i=0}^{n}(a+b)_i(m+1)^i.$$

Ces propriétés d'homomorphisme additif s'étendent de la même manière à des combinaisons linéaires de type $\lambda_0 A + \lambda_1 B$. Le codage du point 11 de la figure 7 à partir de la combinaison linéaire des points 1 et 16, avec des poids respectifs de 1 et 2 donnera ainsi :

$$\frac{1.1+2.16}{1+2} \rightarrow 11;$$

$$\frac{1.[0:0:3]+2.[3:0:0]}{1+2} \rightarrow [2:0:1].$$

Codage en ordre numérique

La définition de l'ordre numérique suggère un algorithme simple de calcul de l'indice d'un point projectif, basé sur le décompte colonne après colonne de la position des coordonnées de ce point. La différence la plus à gauche et non nulle des vecteurs d'exposants de deux points projectifs, $vm_\alpha - vm_\beta$ étant négative, il suffit de compter de gauche à droite, colonne après colonne, l'indice de chaque coordonnée et d'en faire la sommation pour trouver l'indice final du point. Pour calculer par exemple l'indice du point $[1:2:1:0]$ du treillis $T_{4,3}$ de la figure 11, 15 zéros avant le 1 de la première colonne seront comptés, 4 zéros et 3 uns avant le 2 de la seconde colonne et 1 zéro avant le 1 de la troisième colonne. L'indice 23 ou $11=(34-23)$ sera obtenu selon la convention d'indexation choisie, ascendante ou descendante.

Il sera apprécié que les treillis $T_{m,n}$ sont décomposés récursivement en sous-treillis $T_{m,n}$, $T_{m-1,n-1}$, $\bullet\bullet\bullet$, $T_{0,n-1}$, ce qui revient à décomposer le polynôme représentatif d'une hypersurface en polynômes d'hypersurfaces de degrés décroissants dans la dimension inférieure.

De plus, les vecteurs d'exposants sont regroupés en $m+1$ tableaux correspondant aux monômes des polynômes des hypersurfaces de degré décroissant. La figure 11 présente la décomposition par tableaux, d'un treillis (4,3). Les trois dernières colonnes d'exposants représentent les vecteurs d'exposants de treillis (4,2), (3,2), (2,2), (1,2) et (0,2).

Enfin, les tableaux générés sont à leur tour décomposés en tableaux de degrés inférieurs en dimension $n-2$ et ainsi de suite. Les deux dernières colonnes des vecteurs d'exposants de la figure représentent donc les monômes de polynômes homogènes en dimension 1.

$$I_n = \sum_{i=1}^{n} Cm(m-1-\sum_{j=1}^{i} vn[j], n+1-i).$$

L'algorithme *NUMORD*, illustré à la figure 10, correspond à l'équation précédente contractée. La seconde boucle permettant l'accumulation itérative du degré du vecteur

d'exposants est en effet absorbée à l'intérieur de la première boucle par l'instruction 4. Afin d'alléger la présentation du code, la dimension projective égale à la longueur du vecteur moins un est fournie en argument. La figure 11 permet au travers d'un exemple de visualiser le déroulement pas à pas de la fonction *NUMORD*.

$$NUMORD([1,0,2,1,1,1],5) \rightarrow \{210\}_{num}$$

L'indice $\{210\}^5_{num}$ est le résultat de la sommation des nombres pyramidaux effectuée au niveau des lignes 1 à 6 de l'algorithme de la figure 10.

$$\left\langle \binom{9}{5}, \binom{8}{4}, \binom{5}{3}, \binom{3}{2}, \binom{1}{1} \right\rangle$$

Cette séquence fait apparaître la notion de composition pyramidale de l'indice numérique, illustrée à partir du tableau et du triangle de Pascal de la figure 11. Le décodage des indices calculés, présenté plus en avant, sera donc essentiellement basé sur le principe de décomposition pyramidale du nombre proposé, c'est à dire de la recherche de la séquence en cascade des nombres pyramidaux le composant.

Les exemples d'indexation qui suivent pour des points appartenant à des treillis $T_{4,3}$, $T_{6,4}$, $T_{55,9}$, $T_{54,9}$ illustrent en particulier le caractère non bijectif du codage numérique

$$NUMORD([1,2,1,0],3) \rightarrow \{11\}_{num}$$

$$NUMORD([4,0,1,0,1],4) \rightarrow \{11\}_{num}$$

$$NUMORD([1,2,3,4,5,6,7,8,9,10],9) \rightarrow \{22724037001\}_{num}$$

$$NUMORD([0,2,3,4,5,6,7,8,9,10],9) \rightarrow \{22724037001\}_{num}$$

i	$x[i]$	ind	m	n
1	1	84	6	5
2	0	14	5	4
3	2	4	5	3
4	1	1	3	2
5	1	0	2	1

Décodage en ordre numérique

L'algorithme de décodage est basé sur l'inversion du processus d'indexation figurant entre les lignes 1 à 6 de la fonction *NUMORD*. L'objectif pour calculer la signature numérique vm des exposants est de retrouver tout d'abord les nombres pyramidaux successifs qui ont permis de calculer l'indice numérique. Pour retrouver le plus grand nombre pyramidal entrant dans la composition de I_m^{num}, il faut chercher le plus grand nombre pyramidal qui lui est inférieur ou égal dans la colonne m du triangle de Pascal. Une fois ce nombre trouvé et son rang dans la colonne, le premier exposant de la signature recherchée est obtenu et on procède ainsi de colonne en colonne en décrémentant à chaque fois l'indice de départ. Ces étapes sont illustrées par le tableau précédent, avec le calcul de la signature numérique de $\{210\}_5$. Le dernier exposant de la signature est calculé simplement en soustrayant la somme des exposants trouvés du degré de l'indice.

Les exemples suivants illustrent l'énumération des points du treillis $T_{4,3}$ de la figure 11 indexés de 0 à $34 = Cr(4,3)-1$ puis l'énumération des points du treillis $T_{3,2}$ de la figure 2 indexés de 0 à $9 = Cr(3,2)-1$.

$$ORDNUM([34-k,4,3]) \rightarrow [0,0,0,4],[0,0,1,3],\cdots[4,0,0,0], k = 0,\cdots,34$$

$$ORDNUM([9-k,3,2]) \rightarrow [0,0,3],[0,1,2],\cdots[3,0,0], k = 0,\cdots,9$$

Codage en ordre lexicographique

Afin d'effectuer un codage en ordre lexicographique, le treillis $T_{m,n}$ est décomposé en sous-treillis $T_{m-1,n-1}$, $T_{m-1,n-2},\ldots,T_{m-1,0}$, ce qui revient à décomposer le polynôme

représentatif d'une hypersurface en polynômes d'hypersurfaces d'un degré inférieur en dimensions inférieures.

Ensuite, les vecteurs *vm* d'indices sont regroupés en *n* tableaux correspondant aux monômes des polynômes des hypersurfaces de degré décroissant. La figure 13 présente la décomposition par tableaux, d'un treillis $T_{4,3}$. Les 4 tableaux correspondent respectivement aux treillis $T_{3,3}$, $T_{3,2}$, $T_{3,1}$ et $T_{3,0}$, de cubiques dans les dimensions $n = 3,\cdots,0$.

Enfin, les tableaux générés sont à leur tour décomposés en tableaux d'un degré inférieur dans des dimension qui décroissent.

$$I_l = \sum_{i=0}^{m-1} C_n(m-i, n-1-vm[i+1])$$

La figure 13 illustre la décomposition par blocs d'un treillis (4, 3). Les signatures de longueur 4 commençant soit par x_0, x_1, x_2, ou x_3 forment quatre blocs de tailles respectives $\binom{4+3-1}{3} = 20$, $\binom{4+2-1}{2} = 10$, $\binom{4+1-1}{1} = 4$ et $\binom{4+0-1}{0} = 1$. La signature $vm = \langle 1,1,2,3 \rangle$ a pour indice :

$$C(4,(3-1)-1)+C(3,(3-1)-1)+C(2,(3-1)-2)+C(1,(3-1)-3) = \{10\}_{lex}.$$

Le déroulement de la fonction $LEXORD([0,1,1,2],4,3) \rightarrow \{23\}_{lex}$ est illustré sur le tableau suivant :

i	acci	m-i	d-vm[i+1]	Comb (m,d-i+1)
0	15	4	2	$\binom{6}{2}=15$
1	19	3	1	$\binom{4}{1}=4$
2	22	2	1	$\binom{3}{1}=3$
3	23	1	0	$\binom{1}{0}=1$

L'algorithme *LEXORD* est décrit à la figure 14.

Décodage en ordre lexicographique

L'algorithme *ORDLEX* qui permet d'effectuer un décodage en ordre lexicographique est décrit à la figure 15.

Les exemples suivants illustrent l'énumération des points du treillis $T_{4,3}$ de la figure 13 indexes de 0 à $34 = Cr(4,3)-1$ puis l'énumération des points du treillis $T_{3,2}$ de la figure 2 indexes de 0 à $9 = Cr(3,2)-1$.

$ORDLEX([34-k,4,3]) \rightarrow [0,0,0,0],[0,0,0,1],\cdots[3,3,3,3], k = 0,\cdots,34.$

$ORDLEX([9-k,3,2]) \rightarrow [0,0,0],[0,0,1],\cdots[2,2,2], k = 0,\cdots,9.$

Codage de Gray projectif

La sommation des variations entre les codes consécutifs produits pour le codage numérique n'est pas constante. Par exemple, la sommation des variations entre les codes 34 et 33 du treillis $T_{4,3}$ est égale à $\langle 0,0,0,4\rangle - \langle 0,0,1,3\rangle \rightarrow 0$ et égale à $\langle 0,0,4,0\rangle - \langle 0,1,0,3\rangle \rightarrow 1$ entre les codes 30 et 29. Le codage de Gray projectif permet d'éliminer ces différences de variations et de manière générale les sauts entre des points non consécutifs d'un treillis. Ces points ont en fait une distance sur le treillis supérieure à un. Le principe du codage de Gray projectif présenté est essentiellement un remplacement

dans l'arbre récursif, des sous-arbres d'indexation impaire, par leurs symétriques. Ce remplacement étant récursif, l'opération se répètera pour chaque feuille autant de fois qu'elle possédera des parents successifs ayant des indexations impaires. Par conséquent, les sous-arbres possédant un nombre pair de parents successifs impairs n'auront pas à être modifiés, des symétries en nombre pair s'annulant.

Il sera apprécié que dans un codage de Gray projectif, la sommation des exposants du vecteur $Vn_{i+1} - Vn_i$ obtenue à partir de deux points consécutifs est nulle et la distance de Hamming entre deux vecteurs consécutifs d'exposants, est égale à 2.

La figure 16 illustre un codage de Gray projectif (4,3).

L'algorithme illustré à la figure 17 est une version modifiée de l'algorithme illustré à la figure 10. La ligne 4 teste la parité de chaque exposant de la signature fournie en entrée (fonction *type*()). Dans le cas d'un exposant impair l'indiçage est inversé à la ligne 7 et la variable s change de signe. Sa valeur égale à 1 ou -1 permet de prendre en compte le nombre pair ou impair des exposants déjà traités et identifier ainsi les sous-arbres possédant un nombre pair de parents successifs impairs et qui n'ont pas alors à être modifiés.

La figure 18 illustre le parcours des points du treillis $T_{4,3}$ en Gray projectif. Les différentes sections font apparaître une succession de parcours de type cantorien boustrophédonique.

L'indiçage en Gray projectif du point 10 pour le treillis $T_{6,4}$ et pour le treillis $T_{2,4}$ est obtenu de la manière suivante :

$$GRAORD([4,0,1,0,1],4) \rightarrow \{10\}_{gra};$$

$$GRAORD([0,0,1,0,1],4) \rightarrow \{10\}_{gra}.$$

Décodage de Gray projectif

Une implémentation de l'algorithme de décodage de Gray projectif est illustrée à la figure 19 par la fonction *ORDGRA* .

L'exemple suivant illustre l'énumération des points du treillis $T_{4,3}$ de la figure 16 indexés de 0 à $34 = Cr(4,3)-1$:

$$ORDGRA([34-k,4,3]) \rightarrow [0,0,0,4],\cdots[3,0,0,1],[4,0,0,0], k = 0,\cdots,34 .$$

Transcodage dual et direct

Il sera apprécié que le transcodage d'un système de codage à l'autre peut être réalisé en repassant d'un codage à l'autre par simple conversion entre le vecteur des exposants *Vn* et le vecteur des coordonnées *Vm*, soit par une équivalence directe entre les fonctions d'indexation. Dans le premier cas, on parlera de transcodage dual réalisé par les deux fonctions 20 *VN2VM* et *VM2VN* et nous aurons les équivalences suivantes formées à partir des signatures lexicographiques et numériques :

$$S_{num}[Cm(m,n)-i] = VmVn(S_{lex}[i-1]);$$

$$S_{lex}[Cm(m,n)-i] = VnVm(S_{num}[i-1]).$$

Par exemple si :

$S_{num}[Cm(7,4)-168] = [1,1,4,1,0]$ *et* $S_{lex}[168-1] = [0,1,2,2,2,2,3]$, il sera obtenu :

$$[1,1,4,1,0] \leftarrow VM2VN([0,1,2,2,2,2,3])$$

Le transcodage entre vecteur est illustré à la figure 20.

Production récursive d'un ordre

Compte tenu de la relation de symétrie du triangle de Pascal $\binom{r}{k} = \binom{r}{r-k}$ et des équivalences entre les décompositions récursives par colonne et diagonale, un arbre

récursif est associé à deux coefficients binomiaux symétriques. La figure 21 permet de visualiser la structure récursive de $\binom{6}{4}$ identique à celle de $\binom{6}{2}$. Les nœuds $a(i,j)$ sont indicés en largeur i et en profondeur j et les nombres de feuilles des sous-arbres correspondants sont directement calculés à partir de i et j. Toute fonction $I(i,j)$ permet alors d'étiqueter les nœuds de l'arbre et produit un certain type d'ordonnancement des feuilles terminales de l'arbre. Si l'on opte pour un même parcours lexicographique préfixe de l'arbre, on obtient alors l'indexation en ordre numérique d'une courbe quartique en deux dimensions $\binom{6}{4}$, ou alors l'indexation en ordre lexicographique d'une hyperquadrique en quatre dimensions $\binom{6}{2}$ ou finalement l'indexation en ordre lexicographique des 4-combinaisons sans répétition pour 6 objets. La production récursive d'un ordre revient par conséquent à associer à un coefficient binomial un triplet composé d'un ordre de lecture de l'arbre récursif, d'une fonction d'indexation des nœuds de cet arbre et d'un choix du type de combinaisons avec ou sans répétitions.

Fonctions d'indexation des arbres récursifs

Les fonctions d'indexation numérique, Numlex et combinatoire peuvent être déduites séquentiellement l'une de l'autre. L'interprétation géométrique qui en découle utilise les passages entre espaces projectif, affine puis euclidien.

Ordres numérique et numlex

L'algorithme d'indexation numérique consiste à incrémenter à partir de zéro, les indices des nœuds de même niveau tant qu'ils possèdent le même parent. Le parcours de l'arbre se fait donc de gauche à droite et de bas en haut. L'indice du dernier niveau est obtenu en déduisant la somme des indices des niveaux précédents du degré de l'arbre, ce qui garantit l'homogénéité des points numériques produits. Pour passer de l'espace de codage numérique à l'espace de codage numlex, des matrices de passage $N2X_n$ qui associent aux sommets de simplexes d'espaces projectifs numériques les sommets de sous-espaces projectifs aux coordonnées croissantes au sens large. Par exemple, le sommet $[1:0:0:0]$

du simplexe projectif en 3 dimensions sera associé au sommet [1,1,1,1]. Le produit matriciel d'un point projectif n-dimensionnel avec la matrice $N2X_{n-1}$ générera dans tous les cas un point aux coordonnées croissantes. En notant $X2N_n$, la matrice de passage inverse permettant de passer de l'espace numlex à numérique, les matrices de passage suivantes vont être obtenues.

$$N2X = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 0 & 1 & 1 & \dots & 1 \\ 0 & 0 & 1 & \dots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix}, X2N = \begin{bmatrix} 1 & -1 & 0 & \dots & 0 \\ 0 & 1 & -1 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix}$$

A titre d'exemple, le point numérique projectif $[\alpha_0 : \alpha_1 : \alpha_2 : \alpha_3]$ sera donc transformé en point numlex $[\alpha_0, \alpha_0 + \alpha_1, \alpha_0 + \alpha_1 + \alpha_2, \alpha_0 + \alpha_1 + \alpha_2 + \alpha_3]$ par le produit matriciel avec la matrice $N2X_3$. Le point [5:2:0:4] sera quant à lui transformé en point numlex [5:7:7:11] et son indice lexicographique sera calculé à partir de ses coordonnés comme suit : $\binom{4+(11-1)}{4} + \binom{3+(7-1)}{3} + \binom{2+(7-1)}{2} + \binom{1+(5-1)}{1} = 1001 + 84 + 28 + 5 = 1118$.

Ordres numérique et combinatoire

On passe de l'ordre numérique à combinatoire en effectuant une combinaison linéaire entre les points numlex et le point $[0,1,2,\cdots,n-2,n-1]$. Cette opération revient à additionner chaque coordonnée du point numlex avec son rang, équivalent à son niveau dans l'arbre récursif. La figure 24 illustre la présence du sous-espace combinatoire des 3-combinaisons dans le simplexe de référence de l'espace projectif en 3 dimensions. Ce sous-espace est représenté par les sommets de son simplexe de référence obtenus par transformation linéaire projective des sommets du simplexe de référence de l'espace projectif initial (le simplexe unité). La transformation linéaire suivante est obtenue :

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 & 3 & 4 \\ 0 & 2 & 3 & 4 \\ 0 & 1 & 3 & 4 \\ 0 & 1 & 2 & 4 \end{bmatrix}.$$

Les points projectifs appartenant à l'espace combinatoire et obtenus par combinaison quelconque de deux des sommets du simplexe ont donc des coordonnées strictement croissantes. Il est possible d'établir le théorème d'équivalence suivant :

L'indexation d'une k-combinaison d'un r-alphabet $\{0,\cdots,r-1\}$ est équivalente par transformation projective à l'indexation numérique du treillis $T_{r-k,k}$.

La figure 25 illustre l'indexation numérique d'un treillis $T_{2,3}$ équivalent à l'indexation des $\binom{5}{3}$ 3-combinaisons à partir du 5-alphabet $0,\cdots,4$. En dimension supérieure, par exemple, le point numérique 2172, de coordonnés [2,4,1,2,0,3] appartenant au treillis $T_{12,5}$ est transformé en point numlex [2,6,7,9,9,12] par multiplication matricielle avec la matrice $N2X_5$, puis est additionné avec le point [0,1,2,3,4] pour donner le point combinatoire [2,7,9,12,13,17]. On vérifie alors que la position de la combinaison [2,7,9,12,13] pour l'alphabet $\langle 0,1,\cdots,17-1\rangle$ est bien :

$$2172 = \binom{17-3}{5} + \binom{17-8}{4} + \binom{17-10}{3} + \binom{17-13}{2} + \binom{17-14}{1}.$$

L'équivalence entre l'ordre numérique et combinatoire permet d'utiliser les algorithmes de codage numérique *ORDNUM* comme algorithmes rapides de générations de combinaisons. Ceux-ci font appel en particulier à des algorithmes énumératifs du type $HAKMEM_{175}$ (R. Schroeppel, M. Beeler, R.W. Gosper. Hakmem. Technical report, Massachusetts Institute of Technology, 1972), générant les combinaisons à partir de la combinaison précédente représentée sous forme binaire. Comme ce type d'algorithme, pour générer les combinaisons, énumère de manière croissante les nombres ayant le même nombre de bits 1 dans leur représentation binaire, il en résulte que l'algorithme

ORDNUM optimisé peut être également utilisé pour cet usage. La figure 26 illustre la génération énumérative des points projectifs d'un treillis $T_{2,3}$ correspondant à une quadrique en 3 dimensions. Les 10 points, par transformation projective dans le sous-espace combinatoire, forment les 3-combinaisons prises parmi 5 à partir du 5-alphabet 0,...,4. La relation avec la représentation binaire des combinaisons calculées par l'algorithme *HAKMEM* s'en suit avec une transposition des indices des combinaisons énumérées de manière décroissante dans le sens de la lecture.

Ordres numérique et cantorien

Il est possible à l'aide des résultats précédents, de transcoder le codage cantorien à partir du codage numérique en effectuant des changements d'espaces projectifs à euclidiens. En utilisant le formalisme matriciel, le passage d'un point numérique $[\alpha_0 : \alpha_1 : \ldots : \alpha_n]$ de l'espace projectif à l'espace euclidien est en effet réalisé par le produit de ce point par la matrice de passage $P2E$ où $|\alpha|$ est le degré du point. La matrice $E2P$ permettra l'opération inverse pour passer de l'espace euclidien à l'espace projectif :

$$P2E = \begin{bmatrix} 1 & 0 & 0 & \ldots & \frac{1}{|\alpha|} \\ 0 & 1 & 0 & \ldots & \frac{1}{|\alpha|} \\ 0 & 0 & 1 & \ldots & \frac{1}{|\alpha|} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & \frac{1}{|\alpha|} \end{bmatrix}, E2P = \begin{bmatrix} 1 & 0 & 0 & \ldots & -1 \\ 0 & 1 & 0 & \ldots & -1 \\ 0 & 0 & 1 & \ldots & -1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & |\alpha| \end{bmatrix}.$$

Le résultat est donc le point homogène suivant :

$$[\alpha_0 : \alpha_1 : \cdots : \frac{(\alpha_0 + \alpha_1 + \cdots + \alpha_n)}{|\alpha|}] \equiv [\alpha_0 : \alpha_1 : \cdots : \alpha_{n-1} : 1].$$

Le point euclidien obtenu en éliminant la dernière coordonnée égale à 1, $[\alpha_0 : \alpha_1 : \cdots : \alpha_{n-1}]$ est alors le point cantorien en $n-1$ dimensions. Une équivalence est donc établie entre le codage cantorien de la liste inverse $K(x_k,...,x_1)$ et le codage numérique. Pour se faire, il sera noté que l'arbre récursif cantorien est obtenu en éliminant

le dernier niveau de l'arbre récursif numérique, c'est à dire le niveau permettant l'homogénéisation du point numérique.

$$K(x_n,...,x_1) = I_{num}(x_1,...,x_{n)} + \binom{m+n-1}{n}, m = \sum_{i=1}^{n} x_i .$$

En reprenant l'exemple :

$$K(1,3,2,4) \rightarrow \binom{10+4-1}{4} + \binom{6+4-2}{3} + \binom{4+4-3}{2} + \binom{1+4-4}{1} = 782,$$

il est possible de bien vérifier que $I_{num} = 67 + \binom{10+4-1}{4} = 782$. Il est à noter que le premier terme de l'équation :

$$K(1,3,2,4) \rightarrow \binom{10+4-1}{4} + \binom{6+4-2}{3} + \binom{4+4-3}{2} + \binom{1+4-4}{1} = 782$$

doit être soustrait pour obtenir une indexation des points du treillis comprise entre 0 et le nombre total de points.

L'équation $\binom{r-a_i}{k} = \binom{m-1-\sum_{j=1}^{i} vn[j]}{n+1-i}$ permet d'autre part d'obtenir la seconde équivalence, $K(x_n,...,x_2) = I_{num}(x_1,...,x_n)$ qui fait correspondre au codage de Cantor d'une liste, le codage numérique de la liste inverse précédée d'un entier quelconque. Cette équivalence est illustrée par l'exemple suivant qui permet de déterminer le codage de Cantor K_3 à partir de la fonction $NUMORD([x0,x1,x2,x3],3)$ utilisée en mode symbolique :

$$K_3 = \frac{m_0}{1} + \frac{m_1(m_1+1)}{2} + \frac{m_2(m_2+1)(m_2+2)}{6};$$

$$avec\ m_0 = x3,\ m_1 = m_0 + x2,\ m_2 = m_1 + x1.$$

Taxinomie et ordre hamiltonien

Les équivalences précédentes permettent d'établir une taxinomie des systèmes de codage correspondants à différents ordres d'indexation. La figure 28 présente les ordres numlex, et combinatoire comme des cas particuliers de l'ordre numérique, l'ordre cantorien, étant lui-même un cas particulier de l'ordre combinatoire. Il est possible d'étendre cette classification en introduisant un ordre hamiltonien qui englobera également l'ordre de Gray projectif, lui-même considéré comme une généralisation de l'ordre cantorien boustrophédonique. De nouveaux ordres basés sur des courbes spirales remplissant l'espace seront décrits ci-après.

Recherche de chaînes hamiltoniennes sur les treillis

La personne versée dans le domaine appréciera que l'indiçage local pour un treillis est tributaire du type de parcours de l'ensemble de ses points. Parmi les méthodes disponibles, figurent en particulier celles attachées à la théorie des courbes remplissant l'espace (Space Filling Curves).

Fonction cartésienne de couplage en spirale

La fonction de couplage spiralée présentée à la figure 29 permet d'indexer les entiers relatifs deux à deux. Le plan cartésien est parcouru en spirale selon la figure 29. Dans ce cas les polynômes d'indexation des paires de coordonnées du plan cartésien sont recherchés.

Il sera apprécié que la génération de la spirale s'effectue de manière itérative à l'aide d'une équerre rectangle isocèle, tournée de $\pi/2$, puis changée d'échelle puis translatée de une unité de manière à assurer la continuité de la courbe.

$$Z(x,y)=\begin{cases} 4x^2-3x+y & si\ x>0\ \ et\ \ |y|<=|x| \\ 4y^2-y-x & si\ y>0\ \ et\ \ |x|<=|y| \\ 4x^2-x-y & si\ x<0\ \ et\ \ |y|<=|x| \\ 4y^2-3y+x & si\ y<0\ \ et\ \ |x|<=|y| \end{cases}$$

Le point 44 a pour racine carrée entière 6, ce qui correspond à la sixième équerre de la spirale ayant le point 36 = 6.6 comme sommet. Le point 44, est situé entre le sommet 36 (carré pair) et le sommet suivant 49 (carré impair). Étant supérieur au point angulaire intermédiaire 42 = 7.6, il est donc situé sur la section horizontale basse de la spirale. Les variables i et j sont alors calculées à partir du point angulaire intermédiaire 42, i étant donc calculé à partir de la formule de l'indice du coin inférieur gauche de la spirale $I_{Bg} = 2i(2i+1) = 42$, soit $i = 3$ et j à partir de la différence entre le nombre donné et l'indice inférieur gauche, soit $j = 44 - 42 = 2$. Le calcul du couple $[x, y]$ est complété en portant les valeurs de i et j dans les formules précédentes c'est à dire $j = x + i$ puis $i = -y$. Le résultat final est donc $[x, y] = [j - i, -i] = [-1, -3]$. L'exemple suivant illustre l'utilisation des deux fonctions de codage et décodage cartésien en spirale *SPIORDc* et *ORDSPIc* .

SPIORDc([−1509123456789,82739987654321]) →
$\{27383622228148684752295686632\}_S e$

ORDSPIc(27383622228148684752295686632) → [−1509123456789,82739987654321]

La figure 32 décrit une implémentation d'un algorithme de codage cartésien *SPIORDc* et de décodage cartésien en spirale *ORDSPIc* .

Fonction affine de couplage en spirale

$$Z(x,y) = \begin{cases} \dfrac{y(9y-7)}{2} + x & si\ y < 0\ \ et\ \ x <= 2y \\ \dfrac{x(9x-7)}{2} + y & si\ x > 0\ \ et\ \ y <= 2x \\ \dfrac{(9(x-y)^2 - x - y)}{2} & autrement \end{cases}$$

La figure 33 décrit une implémentation d'un algorithme de codage affine en spirale *SPIORDa* et de décodage affine en spirale *ORDSPIa* .

Fonction projective de couplage en spirale

Il est possible de construire un ordre spiralé en n-dimensions en définissant récursivement des chaînes de chaînes de spirales, les premières spirales étant unidimensionnelles, puis bidimensionnelles et ainsi de suite. La figure 34 présente un ordre spiralé pour un treillis (6,4) de sextique en 3 dimensions. Les jonctions entre les spirales bidimensionnelles empilées sont alternées, tantôt entre les débuts, tantôt entre les fins des spirales. Pour obtenir des chaînes spiralées en 4 dimensions, il faut chaîner des treillis spiralés en 3 dimensions les uns aux autres en commençant par un point, treillis (0,4) puis un treillis (1,4), un treillis (2,4), etc.

Optimisations du calcul des coefficients binomiaux

La personne versée dans le domaine appréciera que le calcul rapide des coefficients binomiaux est essentiel dans toutes les fonctions de codage et décodage présentées ci-avant. Les fonctions itératives ou récursives basées sur les relations simples attachées au triangle de Pascal sont d'excellentes candidates pour le calcul rapide des coefficients binomiaux de petite taille. Elles ne sont plus suffisantes pour des calculs faisant intervenir des grands entiers et doivent être améliorées ou remplacées. Aussi des variantes et des améliorations algorithmiques indispensables à la réduction des temps de calcul seront présentées ci-après.

Formulation récursive

La formulation récursive $\binom{r}{k}=\binom{r-1}{k-1}+\binom{r-1}{k}$ *avec* $r \geq 1$ de type additif, est à la base de la construction du triangle de Pascal. Dans ce cas, l'arbre récursif binaire associé possède de nombreuses branches redondantes et a d'autre part une profondeur maximale de $r-1$. Ces propriétés confèrent à l'algorithme de calcul une complexité temporelle dans l'ordre du coefficient binomial $\binom{r}{k}$ lui-même. En conséquence les temps de calcul deviennent immédiatement prohibitifs et ne peuvent être contrôlés informatiquement qu'en utilisant des tables d'appel qui mémorisent les fonctions appelées, leurs arguments

et leurs résultats. Ce mécanisme emprunté à des optimisations en théorie de la compilation figure en particulier dans le langage *Maple* et peut casser la complexité algorithmique en supprimant les appels redondants présents dans le calcul récursif des coefficients binomiaux ou également dans celui des nombres de Fibonacci. En passant à une formulation de type additif-multiplicatif généralisant la formule $\binom{r}{k}=\binom{r-1}{k-1}+\binom{r-1}{k}$ *avec* $r\geq 1$, une nouvelle formulation récursive de profondeur maximale $r-k$ minimisant les redondances va être présentée. Il sera montré que cette approche peut-être considérée comme une variante récursive optimisée de la convolution de Vandermonde. En appliquant la formule $\binom{r}{k}=\binom{r-1}{k-1}+\binom{r-1}{k}$ *avec* $r\geq 1$ de manière répétitive sur les coefficients binomiaux d'un triangle rectangle isocèle à d côtés, la relation suivante est obtenue :

$$\binom{r}{s}=\sum_{i=0}^{k\leq s}\binom{r-k}{s-i}\binom{k}{i}.$$

Celle-ci est obtenue en remarquant, comme l'illustre la figure 34, que le coefficient binomial à la pointe inférieure du triangle peut être calculé à partir des coefficients binomiaux de n'importe laquelle des lignes au dessus. Il suffit pour cela d'affecter les coefficients binomiaux de cette ligne d'un facteur multiplicatif approprié, par exemple $a0=1.d0+3.d1+3.d2+1.d3$. Ces facteurs multiplicatifs dus aux redondances de la formule récursive binaire 8 appliquée ligne après ligne et coefficient binomial après coefficient binomial sont simplement les coefficients binomiaux du développement $(x+y)^d$. La figure 34 présente l'arbre binaire d'application de la formule $\binom{r}{k}=\binom{r-1}{k-1}+\binom{r-1}{k}$ *avec* $r\geq 1$ sur un triangle à 4 côtés. Les duplications successives des coefficients binomiaux fournissent les suites de facteurs multiplicatifs suivants $\{1\}$, $\{1,1\}$, $\{1,2,1\}$, $\{1,3,3,1\}$ et les égalités suivantes $a0=1.a0$, $a0=1.b0+1.b1$, $a0=1.c0+2.c1+1.c2$, $a0=1.d0+3.d1+3.d2+1.d3$. En posant $s=k$

dans la formule $\binom{r}{k}=\binom{r-1}{k-1}+\binom{r-1}{k}$ *avec* $r\geq 1$, la forme récursive intermédiaire suivante est obtenue :

$$\binom{r}{k}=\sum_{i=0}^{k}\binom{r-k}{k-i}\binom{k}{i}.$$

Afin de garantir l'existence d'un triangle rectangle associé à chaque coefficient binomial du triangle de Pascal, le coefficient symétrique le plus à gauche sur une ligne doit être conservé en utilisant la propriété de symétrie des coefficients binomiaux $\binom{r}{k}=\binom{r}{r-k}$.

L'équation suivante est alors obtenue pour $k\leq r$:

$$\binom{r}{k}=\sum_{i=0}^{k}\binom{r-k}{min(k-i,r-k-(k-i))}\binom{k}{min(i,k-i)}.$$

Cette dernière peut être finalement considérée comme une variante récursive optimisée de la convolution de Vandermonde qui s'exprime en effet par :

$$\binom{p+q}{c}=\sum_{k=0}^{c}\binom{p}{c-k}\binom{q}{k}.$$

L'algorithme de la figure 37, utilise trois conditions d'arrêt selon les valeurs de k. Elles ont été desimbriquées pour une lecture plus facile du code. Du point de vue de la programmation purement théorique, l'algorithme fonctionne également sans avoir recours à la fonction *min*(), inutile dans la formulation classique de Vandermonde, mais utile dans la réduction des appels récursifs.

En effectuant les substitutions successives suivantes, $k=i$, $c=k$, $q=k$, et $r=p+k$, l'équivalence entre les relations $Z(x,y)=\begin{cases}\frac{y(9y-7)}{2}+x & si\ y<0\ \ et\ \ x<=2y\\ \frac{x(9x-7)}{2}+y & si\ x>0\ \ et\ \ y<=2x\\ \frac{(9(x-y)^2-x-y)}{2} & autrement\end{cases}$ et

$\binom{r}{k}=\sum_{i=0}^{k}\binom{r-k}{k-i}\binom{k}{i}$ et par suite entre les relations $\binom{r}{k}=\sum_{i=0}^{k}\binom{r-k}{k-i}\binom{k}{i}$ et $\binom{r}{s}=\sum_{i=0}^{k\le s}\binom{r-k}{s-i}\binom{k}{i}$ sont obtenues. La figure 35 présente les différentes convolutions possibles parmi les lignes du triangle de Pascal, la récursivité commençant à partir du couple de lignes $\{r-k,k\}$ soit $\{6,3\}$. Pour pouvoir utiliser dans la convolution de Vandermonde, les lignes intermédiaires entre les lignes $r-k$ et k, il faudrait choisir un couple $\{r-k-i,k+i\}$ satisfaisant $\frac{r-2k}{2}\ge i\ge 0$ et par conséquent introduire une division entière par 2 dans l'algorithme de récursivité. Dans cet exemple il y aura ainsi $i=1$ et la possibilité de commencer la récursivité dans l'algorithme modifié avec le couple de lignes $\{5,4\}$.

Formulations itératives

Il y a trois formulations itératives permettant de calculer les coefficients binomiaux à partir d'additions, de soustractions, de multiplications et de divisions. Il sera apprécié par la personne versée dans le domaine que la formulation par colonne se retrouve dans toutes les implémentations algorithmiques de base :

$$C_y(m,n)=\prod_{i=1}^{n}(m+i)/i\,.$$

La formulation par ligne est quant à elle principalement utilisée pour le calcul de l'expansion des binômes de Newton :

$$C_x(m,n)=\prod_{i=1}^{m}(m+n+1-i)/i\,.$$

Enfin la formulation par diagonale est issue de la propriété de symétrie des coefficients binomiaux à répétitions :

$$C_d(m,n)=C_y(n,m)=\prod_{i=1}^{m}(n+i)/i\,.$$

La figure 38 illustre une réalisation d'un algorithme de calcul des binômes par ligne *XENUME* et par colonne *YENUME*.

Le coefficient binomial $\binom{9}{6}=84$ sera obtenu en sixième itération, par $C_y(6,3)$ dans la colonne 3, à la fin de la séquence $YENUME(3,9)\rightarrow\langle 1,4,10,20,35,56,84\rangle$ ou alors en quatrième itération, par $C_x(6,3)$ dans la ligne 9, à la fin de la séquence $XENUME(3,9)\rightarrow\langle 1,9,36,84\rangle$. Le temps de calcul proportionnel aux itérations conduira à effectuer de manière générale le calcul rapide au moyen de la fonction C_x dans le cas où $m>n$ et au moyen de C_y si $m\leq n$. Ce calcul pourra être finalement effectué par la fonction C_d qui nécessite le même nombre d'itérations que C_x par construction du triangle de Pascal. La séquence pour obtenir le coefficient binomial $\binom{9}{6}=84$ sera alors $\langle 1,7,28,84\rangle$. La généralisation de cet exemple amène donc à utiliser la propriété de symétrie pour optimiser le nombre d'itérations du calcul d'un coefficient binomial et utiliser une seule et même fonction de calcul dans les deux cas de figure :

$$C_r(m,n)=\begin{cases}C_y(m,n) & si\ m>n\\ C_y(n,m) & si\ m\leq n\end{cases}.$$

Cette première optimisation sur le nombre d'itérations étant faite, il est possible de réduire à nouveau ce nombre de moitié en mettant l'équation $C_y(m,n)=\prod_{i=1}^{n}(m+i)/i$ sous la forme d'un produit de différences de carrés. Soit le produit suivant ayant un nombre impair $2k+1$ de termes,

$$\frac{(m+1)}{1}\frac{(m+2)}{2}\cdots\frac{(m+2k)}{2k}\frac{(m+2k+1)}{2k+1}$$ après avoir effectué le changement de variables $m=M-k-1$,

$$\frac{(M-k)}{1}\frac{(M-k+1)}{2}\cdots\frac{M}{k+1}\cdots\frac{(M+k-1)}{2k}\frac{(M+k)}{2k+1}.$$

En regroupant tous les termes à l'exception de M sous la forme de la différence de deux carrés :

$$\frac{(M)}{1}\frac{(M^2-1^2)}{2.3}\frac{(M^2-2^2)}{4.5}\cdots\frac{(M^2-k^2)}{2k.(2k+1)}.$$

La même opération est obtenue pour un produit ayant un nombre pair de $2k$ termes et il est obtenu alors :

$$\frac{(M)}{1}\frac{(M-k)}{1}\frac{(M^2-1^2)}{2.3}\frac{(M^2-2^2)}{4.5}\cdots\frac{(M^2-(k-1)^2)}{2k.(2k+1)}.$$

En divisant à chaque itération par $2i(2i+1)$, la fonction de l'algorithme optimisé du calcul d'un coefficient binomial est alors présentée par la figure 39.

Divisions différées

Selon les performances relatives de la multiplication et de la division, il est possible d'envisager des variantes des algorithmes précédents en différant et limitant les divisions exactes. Il suffit pour cela de calculer indépendamment les dénominateurs qui sont des factorielles et de diviser par ce résultat à la toute fin des algorithmes ou de manière intermédiaire si l'on veut contrôler l'expansion des grands entiers en cours de calcul. Cette approche est également intéressante dans la conception d'algorithmes parallèles.

La figure 39 illustre une réalisation d'un algorithme de calcul par produit de différences de carrés.

Calcul par programmation dynamique

Le calcul des coefficients binomiaux peut être effectué sans multiplications et divisions en utilisant le schéma de récurrence $\binom{r}{k}=\binom{r-1}{k-1}+\binom{r-1}{k}$ et en faisant appel à de la mémoire de stockage. Les opérations nécessaires sont l'addition et l'affectation et le nombre de passages requis pour calculer un coefficient binomial est $mn+\frac{1}{2}(m(m+3))+n+1$.

Le principe est qu'un schéma de calcul des coefficients binomiaux soit établi et soit basé sur une représentation rectangulaire horizontale du triangle de Pascal. L'évaluation des coefficients binomiaux s'effectue alors colonne par colonne au lieu de ligne par ligne. L'algorithme consiste donc à parcourir les cases d'un seul rectangle comme illustré par la figure 40. Le coefficient binomial recherché est situé à la pointe du rectangle et calculé à partir des coefficients binomiaux des colonnes précédentes.

Une implémentation de l'algorithme de programmation dynamique est illustrée à la figure 34.

Méthode additive et multiplicative

En utilisant l'opérateur de multiplication, il est possible à partir de la représentation rectangulaire du triangle de Pascal, de développer des méthodologies de calcul des coefficients binomiaux basées sur l'exponentiation rapide de matrices triangulaires particulières. Il est fait pour cela appel à la formule du binôme de Newton étendu aux matrices carrées commutables en l'occurrence $(M+I)^r$ où I est la matrice identité. L'application de la formule permet d'exprimer les puissances de matrices triangulaires formées exclusivement de 1, à partir de matrices exprimées à partir des coefficients binomiaux et il est alors obtenu :

$$M=\begin{bmatrix}1&0&0&\dots&0\\1&1&0&\dots&0\\1&1&1&\dots&0\\\vdots&\vdots&\vdots&\ddots&\vdots\\1&1&1&\dots&1\end{bmatrix}^{m+1}=\begin{bmatrix}1&0&0&\dots&0\\\binom{m+1}{1}&1&0&\dots&0\\\binom{m+2}{2}&\binom{m+1}{1}&1&\dots&0\\\vdots&\vdots&\vdots&\ddots&\vdots\\\binom{m+n}{n}&\binom{m+n-1}{n-1}&\binom{m+n-2}{n-2}&\dots&1\end{bmatrix}.$$

Compte tenu de la structure symétrique de la matrice M par rapport à sa diagonale, le produit matriciel de deux matrices pour des puissances quelconques se simplifie sous la forme de la fonction MATPRT (voir figure 42) qui n'utilise pour les calculs que deux vecteurs $Vn0$ et $Vn1$ correspondant aux premières colonnes des deux matrices, et stocke le résultat dans le vecteur Wn.

Ainsi, par exemple, le produit matriciel de la 3 ième et de la 31 ième colonne du triangle de Pascal sera représenté sous forme rectangulaire. Le dernier nombre du vecteur résultant Wn sera donc évalué à partir du produit matriciel entre deux puissances de la matrice M. Dans ce cas, le nombre 8436 sera donc évalué à partir des puissances 4 et 32 de la matrice initiale ayant [1,1,1,1] pour vecteur représentatif.

$$MATPRT([1,4,10,20],[1,32,528,5984],3) \rightarrow [1,36,666,8436]$$

La méthode naïve consiste donc à calculer toutes les puissances successives de M jusqu'à la puissance recherchée selon la séquence suivante :

$$MATPRT([1,1,1,1],[1,1,1,1],3) \rightarrow W_M^2 = [1,2,3,4];$$

$$MATPRT([1,1,1,1],[1,2,3,4],3) \rightarrow W_M^3 = [1,3,6,10];$$

$$MATPRT([1,1,1,1],[1,r,\cdots,\cdots],3) \rightarrow W_M^r = [1,1+r,\cdots,\cdots].$$

Minimiser le nombre de produits matriciels pour obtenir le coefficient binomial cherché revient à trouver la meilleure décomposition de la puissance correspondante en une séquence de nombres entiers. La stratégie algorithmique est alors de calculer un coefficient binomial quelconque par un nombre minimum de multiplications matricielles entre les colonnes. Diverses méthodes pour choisir les colonnes peuvent être utilisées dont certaines empruntées aux techniques d'exponentiation rapide. Celles ci reposent en effet sur la relation suivante applicable aux nombres entiers mais également aux matrices carrées :

$$a^{e=\alpha_0+\cdots+\alpha_k} = a^{\alpha_0} a^{\cdots} a^{\alpha_k}.$$

L'exposant e est décomposé en une séquence de nombres entiers par l'intermédiaire de différentes bases de numération ou d'autres techniques de décomposition comme celle de Zeckendorf à base de nombres de Fibonacci . Les figures 43a et 43b illustrent ce principe pour le calcul matriciel des coefficients binomiaux avec une décomposition binaire de l'exposant (figure 43a) ou une décomposition de Zeckendorf (figure 43b). Pour une

décomposition binaire de la puissance, seules les colonnes $2^0,2^1,2^2,\cdots,2^{2^k}$ vont être utilisées pour effectuer les produits matriciels. Le coefficient binomial $\binom{90}{4}=2555190$ sera ainsi calculé selon les étapes suivantes :

- Le coefficient binomial est tout d'abord exprimé dans le triangle de Pascal représenté sous forme rectangulaire :

$$\binom{90}{4}\rightarrow\binom{90-4}{4}_R\rightarrow\binom{86}{4}_R;$$

- L'indice de la colonne compté à partir de 1 est ensuite représenté sous forme binaire ce qui permet de faire apparaître les puissances de 2 utilisées dans la décomposition binaire :

$$86+1=87\rightarrow\{1010111\}_2\rightarrow\{2^6,0,2^4,0,2^2,2^1,2^0\};$$

- Les produits matriciels successifs des puissances de matrices correspondantes sont effectués :

$$M^{87}=M^{64}\times M^{16}\times M^4\times M^2\times M^1$$

pour finalement fournir la matrice à la puissance recherchée faisant apparaître le coefficient binomial recherché comme dernier élément de la première colonne de la matrice :

$$M^{64+16+4+2+1}=M^{87}=\begin{bmatrix}1 & 0 & 0 & 0 & 0\\ 87 & 1 & 0 & 0 & 0\\ 3828 & 87 & 1 & 0 & 0\\ 113564 & 3828 & 87 & 1 & 0\\ 2555190 & 113564 & 3828 & 87 & 1\end{bmatrix}.$$

L'exercice peut se répéter en utilisant cette fois une décomposition de Zeckendorf :

$$87\rightarrow\{55,21,8,3\}_{Zk}$$

avec le gain d'une multiplication matricielle par rapport à la décomposition binaire :

$$M^{87} = M^{55} \times M^{21} \times M^{8} \times M^{3}.$$

Dans les deux cas de figure, l'efficacité et la minimisation des calculs requièrent un schéma itératif simple de calcul des puissances de matrice intermédiaires. De manière générale, chaque matrice utilisée dans le produit matriciel résultant sera obtenue à partir de la matrice ou des matrices calculées précédemment. Un schéma itératif de type $M_i = M_i \times M_i$ permettra ainsi de calculer le produit matriciel à la volée pour une décomposition binaire alors qu'un schéma itératif de type $M_i = M_i \times M_{i-1}$ sera utilisé pour une décomposition de Zeckendorf avec des puissances égales aux nombres successifs de Fibonacci.

Une implémentation de l'algorithme de produit matriciel simplifié est illustrée à la figure 42.

Calcul par factorisation

Il sera apprécié par la personne versée dans le domaine que les critères de divisibilité des coefficients binomiaux peuvent être exploités pour trouver des factorisations rapides permettant d'obtenir le coefficient binomial par multiplication des facteurs trouvés. Pour le calcul des larges coefficients binomiaux, l'algorithme le plus connu est celui de Goetgheluck (P. Goetgheluck. Computing binomial coefficients. *The American Mathematical Monthly*, 94(4) :360-365, 1987.), basé sur un théorème de Lucas concernant la factorisation des coefficients binomiaux en nombres premiers.

Position d'un nombre dans le triangle de Pascal

La recherche de la position d'un nombre donné, c'est à dire l'identification des deux coefficients binomiaux adjacents de ce nombre dans une ligne ou dans une colonne donnée, est à la base des techniques de décodage présentées. Ce problème est indéterminé lorsque la ligne ou la colonne d'appartenance n'est pas spécifiée car il peut y avoir plusieurs entrées identiques dans le triangle de Pascal. Dans le cas où la colonne n est

connue, cela revient à rechercher pour une certaine valeur y_0 et $y = 0$ la racine positive de la fonction polynomiale et équation suivante :

$$y = y_0 - \frac{(m+1)(m+2)\cdots(m+n)}{n!}.$$

Sous cette forme le graphe de la fonction est translaté pour obtenir la racine cherchée sur l'axe des y. La figure 44 illustre la recherche de la position du nombre 20 dans la quatrième colonne du triangle de Pascal avec un graphe de la fonction cette fois sans translation. Le résultat graphique montre que l'abscisse de l'intersection de la droite d'ordonnée 20 avec la courbe est située entre les valeurs 2 et 3. Le nombre 20 est donc compris entre les coefficients binomiaux $\binom{4+2}{4} = 15$ et $\binom{4+3}{4} = 35$.

Méthode naïve

La méthode naïve est l'application de la formule itérative $C_r(m,n) = \begin{cases} C_y(m,n) & si\ m > n \\ C_y(n,m) & si\ m \leq n \end{cases}$

avec un test d'arrêt lorsque la valeur du coefficient binomial calculé dépasse la valeur y_0 fournie en entrée. Une colonne est donc parcourue de manière itérative depuis son origine jusqu'à la découverte d'un coefficient binomial supérieur au nombre donné. Si la complexité théorique est linéaire, l'estimation des coûts réels de calcul doit prendre en compte les opérations d'addition, de multiplication et de division exacte pour des grands nombres entiers.

Méthode compte-gouttes

Certains types d'équations diophantiennes possèdent la caractéristique commune de posséder une seule racine supérieure à un, et d'être dérivables de manière continue. Un algorithme de recherche de cette racine est proposé pour toute équation diophantienne de forme $y_0 = P(x)$ où $P(x)$ est un polynôme à coefficients entiers positifs ou nuls et qui possède une seule racine positive. Parmi les équations particulières répondant à ces critères, figurent, notamment les équations suivantes :

$P_m(x) = x(x+k.1)\cdots(x+k(m-1)),\quad S_m(x) = 1^m + 2^m + \cdots + (x-1)^m$.

$P_m(x)$ lorsque $k=1$ se retrouve dans de nombreux problèmes liés à l'existence de puissances exactes égales à une suite d'entiers consécutifs et au coeur du théorème de Erdös et Selfridge (P. Kirschenhofer, À. Pintér, R. F. Tichy Yu, F. Bilu, B. Brindza and A. Schinzel. Diophantine equations and bernoulli polynomials. *Compositio Mathematica*, 131(2) :173-188, 2002.). Lorsque $k=0$, $P_m(x)$ se ramène à l'équation exponentielle classique et à la recherche des racines *n*ièmes entières, $S_m(x)$ est relative aux nombres de Bernouilli.

Principes

Le principe consiste à améliorer la méthode naïve décrite précédemment en recherchant tout d'abord de manière rapide le nombre de digits de la racine positive de l'équation diophantienne, puis en calculant l'un après l'autre les digits manquants. La recherche du nombre de digits se fait en calculant le signe de la fonction pour des puissances successives de 10 de la variable x. L'ordre de convergence s'adapte donc automatiquement au degré de la fonction. Les digits manquant nécessitent au plus 10 fois le nombre de digits de la racine, évaluations de la fonction.

La figure 45 illustre une implémentation d'un algorithme de la méthode compte-goutte *ENCDEC*.

En notant que l'indice renvoyé par $ENCDEC(1,x)$ est 0, il est obtenu par exemple $ENCDEC(496,5) \to 6$, et de manière générale il est vérifié que la position du coefficient binomial central respecte bien l'identité :

$ENCDEC(Comb(x,x),x) \to x$.

En remplaçant d'autre part la fonction *Comb* par la fonction $NUMFCT(x,n,0)$ (algorithme présente à la figure 46) dans les fonctions *ENCDEC* et *DIGBIN*, et en considérant $NUMFCT(x,n,0)$, il est possible de réaliser l'extraction de la racine *n*ième

entière de x, utilisant la généralisation de la fonction *ENCDEC* à la recherche des solutions de certaines familles d'équations diophantiennes.

Méthodes analytiques

Les méthodes analytiques pour $n=1,2,3,4$ peuvent être intéressantes, selon la performance de l'extraction de racines carrées et cubiques entières. C'est le cas par exemple en deux dimensions pour la fonction inverse de la fonction de couplage de Cantor $I_{K_2}=\frac{(x+y)*(x+y+1)}{2}+y$ qui permet de déterminer les coordonnées d'un point identifié par son indice. Soit l'équation $y_0-\frac{(m+1)*(m+2)}{2!}=0$, nous recherchons tout d'abord la valeur réelle positive de $m_0=\frac{-3+\sqrt{(1+8y0)}}{2}$ qui indique entre quelles diagonales du schéma de Cantor le point recherché se trouve. La partie entière de m_0 qui est égale à l'indice de la diagonale précédant le point recherché, permet alors d'établir les correspondances suivantes avec l'équation $I_{K_2}=\frac{(x+y)*(x+y+1)}{2}+y$ et de déterminer les coordonnées du point donné :

$$y=y_0-\frac{(\lfloor m_0\rfloor+1)*(\lfloor m_0\rfloor+2)}{2!};$$

$$x=\lfloor m_0\rfloor+1-y.$$

Le point d'indice 11 donnera ainsi une valeur de $\lfloor m_0\rfloor=3$ pour les coordonnées [3,1]. Les formules analytiques pour les valeurs $n=3,4$ seront illustrées ci-après dans les méthodes de décomposition binomiale d'un nombre.

Méthode numérique

La convergence du calcul de la position d'un point peut être accélérée de manière significative en recherchant les racines de la fonction binomiale avec un schéma itératif utilisant les dérivées successives de la fonction. Un algorithme de type Raphson-Newton

dont les principes pourraient être étendus à des algorithmes de type Householder possédant des ordres de convergence supérieurs est présenté ci-après. Quel que soit l'algorithme choisi, celui-ci doit être conçu pour le calcul en nombres entiers et s'adapter à une fonction de degré quelconque. Pour cela un algorithme récursif de calcul de la dérivée *k*-ième de la fonction est déterminé qui calcule la valeur de la dérivée sans avoir à recourir à une fonction de dérivation symbolique précalculée.

La formulation récursive de la dérivée k-ième de la fonction binomiale $\binom{m+n}{m}$ est :

$$n!f_{(n)}^{(k)} = \begin{cases} 0 & si\ n<0\ ou\ k<0 \\ 1 & si\ n=1\ et\ k=1 \\ f_{(n-1)}^{(k)}(m+n)+k f_{(n-1)}^{(k-1)} & autrement \end{cases}.$$

La figure 47 illustre une implémentation d'un algorithme de calcul récursif de la dérivée k-ième *DRVREC* .

$$!f(m) = (m+1)(m+2)\cdots(m+7)$$

$$DRVREC(7,3) \rightarrow 210m^4 + 3360m^3 + 19320m^2 + 47040m + 40614$$

La figure 48 illustre une implémentation d'un algorithme de type Raphson-Newton de degré n.

$$DRVSYM(4) \rightarrow x - \frac{((x+1)(x+2)(x+3)(x+4)-24y_0)}{(((2x+3)(x+3)+(x+1)(x+2))(x+4)+(x+1)(x+2)(x+3))}$$

Représentation pyramidale des nombres entiers

Munis de fonctions de calcul pour positionner un nombre entier quelconque dans une colonne donnée du triangle de Pascal, il est possible de réaliser la fonction générale de décodage d'un nombre entier en représentation pascalienne. Le décodage d'un nombre en représentation quaternaire, étant une opération fréquente et que nous verrons essentielle dans le processus de compression des listes générales d'entiers, un décodage de nature purement analytique sera décrit en premier lieu, utilisant les principes présentés

précédemment. Dans ce cas, un décodage séquentiel est obtenu où les coefficients binomiaux successifs $\beta_{3,\cdots,0}$ sont calculés à partir des racines entières positives de l'équation $y = y_0 - \frac{(m+1)(m+2)\cdots(m+n)}{n!}$ pour les degrés $4,\cdots,1$ de la variable m. Conformément en deux dimensions à l'indice de la diagonale précédant le point correspondant au nombre donné, ces valeurs entières représentent l'indice de l'hyperplan, plan, droite et point diagonal précédant le point recherché.

$$\beta_0 = \binom{1+(y-1)}{1}$$

$$\beta_1 = \binom{2+\lfloor(\frac{3}{2}+\frac{1}{2}\sqrt{1+8y})\rfloor}{2}$$

$$\beta_2 = \binom{3+\lfloor(\frac{1}{3}P^{\frac{1}{3}}+\frac{1}{P^{\frac{1}{3}}}-2)\rfloor}{3}, P = 81y+3\sqrt{-3+729y^2}$$

$$\beta_3 = \binom{4+\lfloor(\frac{-5}{2}+\frac{1}{2}\sqrt{5+4\sqrt{1+24y}})\rfloor}{4}$$

En reprenant l'exemple précédent où le code combinatoire était 1118, le décodage s'effectuera selon la séquence suivante :

$$\beta_3(y) = \beta_3(1118) = \binom{4+10}{4} = 1001$$

$$\beta_2(y) = \beta_2(1118-1001) = \binom{3+6}{3} = 84$$

$$\beta_1(y) = \beta_1(1118-1001-84) = \binom{2+6}{2} = 28$$

$$\beta_0(y) = \beta_0(1118-1001-84-28) = \binom{1+4}{1} = 5$$

Le vecteur $[(4+1),(6+1),(6+1),(10+1)]$ étant alors multiplié par la matrice $X2N_3$ pour restituer le code original $[5:2:0:4]$.

Si à présent la méthode compte-gouttes est utilisée, la fonction $BINRP4$, illustrée à la figure 49, sera définie qui recherche la position d'un nombre entier quelconque dans la colonne 4, puis décrémente ce nombre de l'indice trouvé afin de rechercher la position de ce nouveau nombre dans la colonne 3, ce processus étant poursuivi jusqu'à la toute première colonne. Les indices trouvés sont équivalents aux coordonnées affines cantoriennes en 4 dimensions. Pour l'exemple précédent, $BINRP4(1118) \rightarrow \{5,7,7,11\}$ correspondant au vecteur précédemment obtenu par la méthode analytique. De la même manière la représentation binomiale quaternaire du nombre 100 sera $\{100\}_{bin}^{4} = \{0,4,4,5\}$ et sa décomposition quaternaire sera $\left\{\binom{0}{1},\binom{5}{2},\binom{6}{3},\binom{8}{4}\right\}$ soit $\{100\}_{Bin}^{4} = \{0,10,20,70\}$.

La figure 49 illustre une implémentation d'un algorithme de décomposition combinatoire quaternaire $BINRP4$ d'un nombre.

Il sera alors possible d'étendre la fonction $BINRP4$ à la décomposition n-aire d'un nombre pour obtenir la fonction générale $BINRPN$ et sa fonction inverse $RPNBIN$ qui recompose un nombre, donné sous forme pascalienne, en nombre décimal.

La figure 50 illustre une implémentation d'un algorithme (BINRPNi) de représentation pyramidale n-aire.

Par exemple $BINRPNi(125,5) \rightarrow \{4,4,4,4,4\}$ soit $\{125\}_{bin}^{5} = \{4,4,4,4,4\}$, et inversement $RPNBINi([4,4,4,4,4],5) \rightarrow 125$.

D'autre part, compte-tenu des équivalences démontrées précédemment, il est possible de remplacer la fonctionnalité de décodage lexicographique $ORDLEX()$ par :

$$ORDLEX([x,m,n]) \equiv \overbrace{[n,\cdots,n]}^{m} - BINRPNi(x,m).$$

Enfin, il est possible d'obtenir les fonctions de codage et décodage du système de numération combinatoire du paragraphe 4.4.1, en remplaçant a par $a+ind$ dans les deux lignes de $BINRPNi$ et $Comb(vn[n-i+1]-1,i)$ par $Comb(n-vn[m+1-i]-i,i)$ dans $RPNBINi$.

Optimisations algorithmiques

La personne versée dans le domaine appréciera que l'élimination de certaines boucles soit le premier niveau d'optimisation algorithmique. Le principe de base est de remplacer une double boucle l'appel aux fonctions de calcul d'un coefficient binomial dans une boucle par une simple boucle calculant le coefficient de manière itérative.

Application aux listes générales d'entiers

Principe de compression combinatoire

Différents types de codage sont présentés qui, conformément au diagramme divulgué ci-avant, permettent d'encoder toute liste d'entiers par l'intermédiaire d'un nombre entier unique, l'indice total ou par un triplet constitué de l'indice local de la liste, de son degré total et de sa longueur. L'indice total correspond à la numérotation absolue d'un point projectif représentant la liste à coder dans l'espace projectif hyperspatial. La personne versée dans le domaine appréciera qu'il est algorithmiquement pur mais génère très rapidement de très grands entiers. Suite aux équivalences démontrées entre le codage numérique et le codage cantorien, il est également possible d'opter pour une autre stratégie et d'encoder le triplet $\{X,m,n\}$ en un seul nombre entier assurant le caractère bijectif du codage décodage. Il sera possible pour ce faire utiliser par exemple un codage cantorien tridimensionnel, ou un codage cantorien bidimensionnel en deux itérations successives. La connaissance implicite de la méthode de codage choisie permettra de réaliser le décodage. Une autre méthode consiste à remplacer le nombre X par ses coordonnées cantoriennes bidimensionnelles m_X,n_X puis de coder le quadruplet $\{m_X,n_X,m,n\}$ avec un codage cantorien en quatre dimensions. En remplaçant le triplet

par un quadruplet il y aura un effet d'expansion limité de la taille du code généré : la longueur de la liste à coder étant simplement incrémentée de un, mais un effet de compression significatif dû à l'abaissement du degré total de la liste à coder. Soit en effet, un poids total du triplet de $\frac{(m_X+n_X)*(m_Xn_X+1)}{2}+m_X+m+n$, en exprimant X en fonction de ses coordonnées cantoriennes, le degré total du quadruplet sera abaissé à m_X+n_X+m+n.

La figure 51 illustre une implémentation d'un algorithme d'optimisation de codage global (GLOBALo) et local (LOCALo).

La figure 52 illustre une implémentation d'un algorithme de décodage numérique (ORDNUMo et ORINUMo).

La figure 53 illustre une implémentation d'un algorithme de codage et de décodage compressé (INDCOM).

En reprenant le point projectif [69,78,73,71,77,65] de degré 433 en dimension 5 de l'exemple 4.1, les indices suivants sont obtenus :

$I_{Num} = \{55014101437\}_{(433,5)}$

$I_{Tot} = \{14196068833898572081041480622812588561594557825924180864872855452746861095964803189964668959253194639858643000122386287763569638590 10\}_Z$

et sous forme compressée,

$I_{Com} = INDCOM([69,78,73,71,77,65],5) \rightarrow \{507098669008328947049\}_C$.

Représentation de listes de listes

L'application récursive de la fonction *INDCOM* permet de coder des listes de listes. Soit $L_0, L_1, \cdots, L_n$ un ensemble de listes d'entiers de dimensions respectives $n_0, n_1, \cdots, n_n$, un codage de premier niveau sera défini ainsi :

$INDCOM([INDCOM(L_0,n_0),INDCOM(L_1,n_1),\cdots,INDCOM(L_n,n_n)],n)$.

Représentation des nombres entiers relatifs

Outre la représentation des entiers relatifs par les fonctions de couplage spiralées, il est possible de considérer la représentation théorique des entiers signés, par paire. Cette approche amène par exemple à doubler la longueur du vecteur initial en faisant précéder chaque entier par un entier signifiant son signe et par extension son type. Il est possible par exemple d'utiliser la convention du 0 pour un entier positif, du 1 pour un entier négatif, du 2 pour un irrationnel, du 3 pour un imaginaire, etc. La liste $\{-1,2,7,-3,5\}$ sera ainsi codée sous forme entrelacée $\{1,1,0,2,0,7,1,3,0,5\}$ avec un indice compressé de $\{6592155939510\}_C$.

La personne versée dans le domaine appréciera donc que des entiers relatifs pourront être codés au moyen d'une liste d'entiers. Dans un tel cas, une liste d'entiers à coder sera générée au moyen de la liste d'entiers relatifs.

Tris alphanumériques

L'ordre numérique permet de créer des dictionnaires par tris alphanumériques. Pour cela, le principe consistera à ramener l'ensemble des mots du dictionnaire à des longueurs et des poids identiques. Les mots à trier seront associés à des points projectifs d'un même treillis. Il sera pris pour longueur, celle du mot le plus grand, et pour poids, celui du mot le plus lourd. Pour trier par exemple, les mots ART, ARA, BRAS, ARABE, ARABICA, codés avec un alphabet de 26 lettres, l'indice de chaque lettre correspondant à sa position dans l'alphabet, les points suivants qui leur correspondent sur le treillis $T_{40,7}$ seront ordonnancés. La longueur maximale sera héritée du mot ARABICA et le poids maximum du mot BRAS :

$ART \rightarrow [1,18,20,0,0,0,0,1] \rightarrow \{45609855\}_{I_{num}}$;

$ARA \rightarrow [1,18,1,0,0,0,0,20] \rightarrow \{45662979\}_{I_{num}}$;

$BRAS \rightarrow [2,18,1,19,0,0,0,0] \rightarrow \{38531317\}_{I_{num}}$;

$ARABE \rightarrow [1,18,1,2,5,0,0,13] \rightarrow \{45658898\}_{I_{num}}$;

$ARABICA \rightarrow [1,18,1,2,9,3,1,5] \rightarrow \{45658530\}_{I_{num}}$.

En inversant le tri, pour respecter l'ordre descendant du codage numérique, la séquence triée alphabétiquement sera obtenue :

$\langle 45662979,45658898,45658530,45609855,38531317\rangle \rightarrow$
$\langle ARA, ARABE, ARABICA, ART, BRAS\rangle$.

Hachage

Il sera apprécié par la personne versée dans le domaine que les fonctions d'indexation présentées précédemment peuvent avantageusement servir de fonctions de hachage de la même manière que la fonction classique de hachage polynomial bâtie à partir de l'équation :

$$H(x) = (\alpha_0.B^k + \alpha_1.B^{k-1} + \cdots + \alpha_k.B^0) \bmod N .$$

Dans ce cas, B sera pris communément égal à une puissance de 2, par exemple 256 et N égal à un nombre premier. En couplant de la même manière les fonctions d'indexation combinatoire à l'opérateur modulo, les valeurs suivantes seront obtenues à partir de la séquence précédente. Le choix de N occasionnera un phénomène de collisions et un taux de remplissage de la table de hachage plus ou moins important.

$\langle 45662979,45658898,45658530,45609855,38531317\rangle \rightarrow \langle 11,12,8,5,6\rangle$

Application aux polynômes

Représentation littérale

Considérer le polynôme sous forme d'une chaîne de caractères présente certains avantages, particulièrement pour préserver la représentation initiale du polynôme, sous forme factorisée par exemple ou alors en préservant son type de notation, en polonaise inversée par exemple. En contrepartie, les traitements algorithmiques ultérieurs du polynôme doivent passer par une étape de mini compilation classique de type expression

arithmétique. Un codage élémentaire caractère par caractère pourra par conséquent être réalisé à partir d'un alphabet réduit de type $\{0,1,2,3,4,5,6,7,8,9,x,+,-,*,/,(,),\#\}$ celui-ci étant constitué des digits, des opérateurs arithmétiques, des parenthèses, de la variable x et du caractère dièse utilisé comme séparateur. L'alphabet sera étendu aux lettres de l'alphabet si le polynôme est destiné à des traitements symboliques. Un code numérique étant affecté à chaque symbole de l'alphabet, la table d'équivalence de l'alphabet à n caractères sera simplement la liste des $n+1$ entiers naturels. Le poids maximum de cette liste sera par conséquent $(n+1)^2$. L'exemple suivant présente l'équation homogène d'un tore de rayons a et b centré à l'origine du repère euclidien :

$$Tore = (x_0^2 + x_1^2 + x_2^2 - (b^2 - a^2) * x_3^2)^2 - 4 * a^2 * (x_0^2 + x_1^2) * x_3^2.$$

La liste de 56 entiers et de degré total 3789, représentant l'équation du tore dans le système de codage *UNICODE* est la suivante :

$$L_{Tore} = \{40,120,48,94,50,43,120,49,94,50,43,120,50,94,50,45,40,98,94,50,$$
$$45,97,94,50,41,42,120,51,94,50,41,94,50,45,52,42,97,94,50,42,40,120,48,94,$$
$$50,43,120,49,94,50,41,42,120,51,94,50\}_{(Unicode)}.$$

Le codage numérique de L_{Tore}, est effectué par $NUMORD(L_{Tore},56-1)$ qui donne un indice numérique local de :

$$I_{Num} = \{43049865424991594367802378012550962768849420931362077564740744794736889204295402592251903551063043291931247050023162038956 65\}_{(3789,55)}.$$

Représentation structurée

Les problèmes de performances algorithmiques sont étroitement liés aux représentations choisies, et les problèmes de mémoire sont en partie occasionnés par les pertes dues aux polynômes creux. Le vecteur L représentant l'expansion numérique du polynôme du tore pour des rayons $a = 2$ et $b = 1$ permet de visualiser les différents problèmes reliés à la représentation vectorielle : l'ordre des coefficients correspondant à l'ordre choisi sur les monômes, la présence des coefficients nuls propres aux polynômes creux et finalement la gestion des signes des coefficients qui échappent aux codages des entiers naturels.

$L = [1,0,2,0,1,0,0,0,0,2,0,2,0,0,1,0,0,0,0,0,0,0,0,0,0,-10,0,-10,0,0,6,0,\ 0,0,9]$

Après avoir codé les valeurs négatives du vecteur L, le code numérique du nouveau vecteur entrelacé du tore est alors :

$I_{Num} = \{19143865180361749932587381394851 3\}_{(46,69)}$.

Afin de palier à ce type de problème et en considérant que les coefficients nuls n'influent que sur la dimension du vecteur à coder et non sur son degré, il est possible d'adopter la stratégie d'adressage en cascade suivante. Une table de hachage sans collisions d'une taille égale au nombre de coefficients $Cm(m,n)$ est créée pour un polynôme de degré m en n dimensions. Cette table contient les adresses codées à leur tour sous forme de nombres entiers sur les informations propres aux monômes. Ces informations stockées séparément peuvent être par conséquent les valeurs numériques signées des coefficients, mais également des pointeurs sur des fonctions de calcul symbolique des coefficients ou encore des indices de bases de données relationnelles pour établir des contraintes géométriques entre différents polynômes (CADCAM variationnel).

$T_0 = [k_0,0,k_2,0,k_4,0,0,0,0,k_9,0,k_{11},0,0,k_{14},0,0,0,0,0,0,0,0,0,0,k_{25},0,k_{27},0,\ 0,k_{30},0,0,0,k_{34}]$

Codage des vecteurs et polynômes creux

Le codage des vecteurs creux peut être réalisé de plusieurs manières différentes. La méthode la plus commune est celle dite de représentation «rank-and-value» décrite par Lehmer (D.H. Lehmer. The machine tools of combinatorics. In E.F. Beckenbach, editor, Applied Combinatorial Mathematics, pages 5-31. Wiley, New York, 1964). Dans ce cas, le codage est réalisé par une suite de paires d'entiers formées de la position de chaque entier dans la liste creuse, puis de sa valeur. Un zéro est mis en fin de liste pour signifier la fin du message. La table précédente T_0 est donc codée de la manière suivante :

$T_{00} = \langle 0,k_0,2,k_2,4,k_4,9,k_9,11,k_{11},14,k_{14},25,k_{25},27,k_{27},30,k_{30},34,k_{34},0\rangle$.

Les positions des valeurs non nulles peuvent être aussi codées séparément en utilisant par exemple la technique de compression combinatoire. Il y aura alors le code compressé suivant pour mémoriser les emplacements des valeurs non nulles de T_0 :

$$\langle 35,0,2,4,9,11,14,25,27,30,34\rangle \rightarrow \{1836997097013699578176773581776\}_C .$$

Il est possible également d'utiliser le codage de Pascal pour les séquences décroissantes mais en mémorisant de manière séparée la longueur du vecteur :

$$\langle 34,30,27,25,14,11,9,4,2,0\rangle \rightarrow \{148139661\}_{Pas} .$$

Produit symbolique de polynômes à plusieurs variables

Les fonctions rapides d'indexation de monômes qui permettent de réaliser le produit de deux polynômes à plusieurs variables de manière vectorielle par indexation directe doivent être alors utilisées. L'algorithme *MPLNUM* illustré à la figure 54 utilise l'indexation par ordre numérique en multipliant chaque monôme du premier polynôme par chaque polynôme du second, ce qui est convenu d'appeler l'algorithme naïf de multiplication de deux polynômes. Un des intérêts de l'algorithme présenté est qu'il peut être facilement étendu aux multiplications de vecteurs creux.

L'algorithme de multiplication symbolique de deux polynômes multi variés est basé sur l'opération élémentaire du produit de deux monômes, de l'indexation du monôme résultant et de la sommation des monômes de même rang. La fonction *MPLNUM* réalise le produit symbolique de deux polynômes dans une dimension donnée. Le degré du polynôme résultant dgr calculé en ligne 1 et les nombres des monômes des deux polynômes multipliés et du polynôme résultant sont calculés de la ligne 2 à la ligne 9. L'initialisation de tous les termes du vecteur résultant à zéro est réalisée à la ligne 10. Une boucle est faite sur le nombre *lga* des monômes du polynôme a en ligne 11 et sur le nombre *lgb* des monômes du polynôme b en ligne 12. Les signatures en ordre numérique des deux polynômes a et b sont obtenues aux lignes 13 et 14. La multiplication des monômes est réalisée par l'addition des signatures *siga* et *sigb* à la ligne 15. L'indice du monôme résultant est calculé à la ligne 16. Finalement, les monômes résultants de même

rang sont sommés et accumulés dans le vecteur associé aux polynômes résultant à la ligne 17.

L'exemple suivant illustre la multiplication en deux dimensions d'une courbe du second degré projective (conique) par une courbe du premier degré (droite), ce qui donne une courbe du troisième degré (cubique dégénérée). Soit $y2a$, la conique d'équation $y2a = a_0x_0^2 + a_1x_0x_1 + a_2x_1^2 + a_3x_0x_2 + a_4x_1x_2 + a_5x_2^2$ et $yb1$, la droite d'équation $yb1 = b_0x_0 + b_1x_1 + b_2x_2$, les $10 = \binom{5}{3}$ coefficients de la cubique $y3c$ en ordre numérique $\langle 0,0,3\rangle, \langle 0,1,2\rangle, \cdots, \langle 3,0,0\rangle$ sont obtenus,

$MPLNUM([a0,a1,a2,a3,a4,a5],[b0,b1,b2],2,1,2) \rightarrow$

[$a0$ * b0, $a0$ * $b1$ + $a1$ * $b0$, $a1$ * $b1$ + $a2$ * $b0$, $a2$ * $b1$, $a0$ * $b2$ + $a3$ * $b0$, $a1$ * $b2$ + $a3$ * $b1$ +$a4$*$b0$, $a2$*$b2$ +$a4$*$b1$, $a3$*$b2$+ $a5$*$b0$, $a4$*$b2$ +$a5$*$b1$, $a5$*$b2$].

Implémentation de la méthode divulguée

Il sera apprécié par la personne versée dans le domaine que la méthode pour coder de l'information divulguée ci-avant ainsi que la méthode pour décoder de l'information codée au moyen de la méthode divulguée ci-avant pourront être implémentées de diverses façons.

En particulier, ces méthodes pourront être implémentées par exemple dans un ordinateur 300 tel qu'illustré à la figure 30. L'ordinateur 300 comprendra par exemple un processeur 310, un bus de données 312, une unité de stockage des données 314, un port de communication 316, une unité d'affichage 318 et un clavier 320.

Dans cet exemple, le processeur 310 est adapté pour traiter des instructions. La personne versée dans le domaine appréciera que divers types de processeurs peuvent être utilisés pour effectuer le traitement des instructions.

L'unité de stockage des données 314 est adaptée pour stocker des données. La personne versée dans le domaine appréciera que divers types d'unités de stockage des données pourront être utilisés pour effectuer un tel stockage.

Le port de communication 316 est utilisé pour échanger de l'information entre l'ordinateur 300 et une autre unité de traitement non illustrée sur la figure 30. La personne versée dans le domaine appréciera que le port de communication 316 dépend du type de connexion utilisé entre l'ordinateur 300 et l'autre unité de traitement.

L'unité d'affichage 318 est utilisée pour afficher des données à un usager. La personne versée dans le domaine appréciera que divers types d'unité d'affichage pourront être utilisés.

Le clavier 320 est utilisé pour permettre à un usager de fournir des données à l'ordinateur 300. Là encore, la personne versée dans le domaine appréciera que divers types de claviers pourront être utilisés.

Il sera apprécié que le processeur 310, l'unité de stockage de données 314, le port de communication 316, l'unité d'affichage 318 et le clavier 320 sont opérativement connectés ensemble au moyen du bus de données 312.

En fait et dans une réalisation, l'unité de stockage des données 314 comprend un système d'exploitation 322. Le système d'exploitation 322 pourra être de différents types.

L'unité de stockage des données 314 comprend de plus un programme 324 pouvant être exécuté par le processeur 310 et dont l'exécution permet de coder de l'information. Il sera apprécié par la personne versée dans le domaine qu'un programme dont l'exécution permettra de décoder de l'information pourra aussi être stocké dans l'unité de stockage des données 314.

Le programme permettant de coder de l'information comprend des instructions pour obtenir une liste d'entiers à coder. Il sera apprécié que la liste d'entiers à coder pourra être obtenue de diverses manières. Dans une réalisation, la liste d'entiers sera entrée par un usager au moyen du clavier 320. Dans une autre réalisation la liste d'entiers sera obtenue d'un fichier localisé dans l'unité de stockage de données 314 ou reçu par l'intermédiaire du port de communication 316.

Le programme permettant de coder de l'information comprend de plus des instructions pour déterminer une pyramide hyperspatiale ayant une taille adaptée pour coder la liste

d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets dont le nombre est déterminé par le degré de la pyramide hyperspatiale qui est égal à la somme des entiers de la liste d'entiers et par la dimension de la pyramide hyperspatiale qui est égale au nombre d'entiers de la liste d'entiers moins un.

Le programme permettant de coder de l'information comprend des instructions pour indexer la liste d'entiers dans cette pyramide au moyen d'un système d'indexation. Tel que mentionné ci-avant, divers systèmes d'indexation pourront être utilisés.

Le programme permettant de coder de l'information comprend des instructions pour fournir une indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale afin de coder ainsi ladite liste d'entiers à coder. Il sera apprécié que l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale pourra être fournie de diverses façons. Par exemple, l'indication de l'indexation de la liste d'entiers pourra être affichée sur l'unité d'affichage 318 (par exemple un écran). Alternativement, l'indication de l'indexation de la liste d'entiers pourra être stockée dans un fichier localisé dans l'unité de stockage de données 314 ou elle pourra être transmise par l'intermédiaire du port de communication 316 à une autre unité de traitement.

Il sera apprécié qu'un dispositif de stockage d'information pourra être fourni pour stocker des instructions exécutables par un processeur. Ces informations exécutables par un processeur vont causer le processeur d'exécuter une méthode comprenant obtenir une liste d'entiers à coder; déterminer une pyramide hyperspatiale ayant une taille adaptée pour coder la liste d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets dont le nombre est déterminé par le degré de la pyramide hyperspatiale qui est égal à la somme des entiers de la liste d'entiers et par la dimension de la pyramide hyperspatiale qui est égale au nombre d'entiers de la liste d'entiers moins un; indexer la liste d'entiers dans cette pyramide au moyen d'un système d'indexation; et fournir une indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale afin de coder ainsi ladite liste d'entiers à coder.

Clause 1. Une méthode pour coder de l'information, la méthode comprenant :

obtenir une liste d'entiers à coder;

déterminer une pyramide hyperspatiale ayant une taille adaptée pour coder la liste d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets dont le nombre est déterminé par le degré de la pyramide hyperspatiale qui est égal à la somme des entiers de la liste d'entiers et par la dimension de la pyramide hyperspatiale qui est égale au nombre d'entiers de la liste d'entiers moins un;

indexer la liste d'entiers dans cette pyramide au moyen d'un système d'indexation; et

fournir une indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale afin de coder ainsi ladite liste d'entiers à coder.

Clause 2. La méthode revendiquée à la clause 1, comprenant de plus fournir une liste de codes alphanumériques et transformer la liste de codes alphanumériques en liste d'entiers à coder.

Clause 3. La méthode revendiquée à la clause 1, dans laquelle le système d'indexation permet d'indexer chacun de la pluralité de sommets de ladite pyramide hyperspatiale.

Clause 4. La méthode revendiquée à la clause 1, dans laquelle l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale comprend un entier représentant l'indice du point projectif dans le système d'indexation, m le degré total de ladite liste d'entiers et n le nombre d'entiers de la liste d'entiers.

Clause 5. La méthode revendiquée à la clause 4, comprenant de plus transformer l'entier représentant l'indice du point projectif par des coordonnées combinatoires mx et nx.

Clause 6. La méthode revendiquée à la clause 4, comprenant de plus la transformation de m le degré total de la liste d'entiers et n le nombre d'entiers de la liste d'entiers par un entier Y représentant un indice global.

Clause 7. La méthode revendiquée à la clause 6, comprenant de plus effectuer la sommation de l'entier Y représentant un indice global à l'entier représentant l'indice du point projectif pour fournir un indice total Z, l'indication de l'indexation dans la pyramide hyperspatiale comprenant l'indice total Z.

Clause 8. La méthode revendiquée dans un des clauses 1 à 7, dans laquelle la liste d'entiers est obtenue d'un fichier.

Clause 9. La méthode revendiquée dans une des clauses 1 à 8, dans laquelle l'étape de fournir l'indication de l'indexation dans la pyramide hyperspatiale comprend le stockage de ladite indication de l'indexation dans un fichier.

Clause 10. La méthode revendiquée dans une des clauses 1 à 8, dans laquelle l'étape de fournir l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale comprend la transmission de l'indexation de la liste d'entiers à une unité de traitement par l'intermédiaire d'un réseau.

Clause 11. La méthode revendiquée dans une des clauses 1 à 10 pour coder une liste d'entiers relatifs, la méthode comprenant fournir la liste d'entiers relatifs et générer la liste d'entiers au moyen de la liste d'entiers relatifs.

Clause 12. La méthode revendiquée dans une des clauses 1 à 8, dans laquelle l'étape de fournir l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale comprend l'affichage de ladite indication de l'indexation de la liste d'entiers sur un écran.

Clause 13. L'utilisation de la méthode revendiquée dans une des clauses 1 à 3 et 4 à 7 pour compresser une chaine de caractères dans laquelle la chaine de caractères est associée à ladite liste d'entiers.

Clause 14. Une méthode pour coder une liste de listes, la méthode comprenant l'utilisation itérative de la méthode revendiquée dans une des clauses 1 à 3 et 4 à 7 pour coder une liste de listes.

Clause 15. L'utilisation de la méthode revendiquée à la clause 1 pour crypter une liste d'entiers, dans laquelle l'indexation de la liste d'entiers correspond au cryptage de la liste d'entiers.

Clause 16. Une méthode pour décoder de l'information codée au moyen de la méthode pour coder de l'information revendiquée dans une des clauses 1 à 7, la méthode comprenant :

obtenir l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale;

obtenir une indication d'un système d'indexation utilisé pour générer ladite indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale;

déterminer au moyen de l'indexation et de l'indication d'un système d'indexation utilisé ladite liste d'entiers.

fournir ladite liste d'entiers.

Clause 17. Un ordinateur comprenant :

une unité d'affichage;

un processeur;

une unité de stockage des données comprenant un programme configuré pour être exécuté par le processeur, le programme permettant de coder de l'information, le programme comprenant :

des instructions pour obtenir une liste d'entiers à coder;

des instructions pour déterminer une pyramide hyperspatiale ayant une taille adaptée pour coder la liste d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets dont le nombre est déterminé par le degré de la pyramide hyperspatiale qui est égal à la somme des entiers de la liste d'entiers et par la dimension de la pyramide hyperspatiale qui est égale au nombre d'entiers de la liste d'entiers moins un;

des instructions pour indexer la liste d'entiers dans cette pyramide au moyen d'un système d'indexation;

des instructions pour fournir une indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale afin de coder ainsi ladite liste d'entiers à coder.

Clause 18. Un dispositif de stockage d'information pour stocker des instructions exécutables par un processeur, qui quand elles sont exécutées vont causer l'exécution par le processeur d'une méthode pour coder de l'information, la méthode comprenant obtenir une liste d'entiers à coder; déterminer une pyramide hyperspatiale ayant une taille adaptée

pour coder la liste d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets dont le nombre est déterminé par le degré de la pyramide hyperspatiale qui est égal à la somme des entiers de la liste d'entiers et par la dimension de la pyramide hyperspatiale qui est égale au nombre d'entiers de la liste d'entiers moins un; indexer la liste d'entiers dans cette pyramide au moyen d'un système d'indexation; et fournir une indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale afin de coder ainsi ladite liste d'entiers à coder.

Revendications :

1. Une méthode pour coder de l'information, la méthode comprenant :

obtenir une liste d'entiers à coder;

déterminer une pyramide hyperspatiale ayant une taille adaptée pour coder la liste d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets dont le nombre est déterminé par le degré de la pyramide hyperspatiale qui est égal à la somme des entiers de la liste d'entiers et par la dimension de la pyramide hyperspatiale qui est égale au nombre d'entiers de la liste d'entiers moins un;

indexer la liste d'entiers dans cette pyramide au moyen d'un système d'indexation; et

fournir une indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale afin de coder ainsi ladite liste d'entiers à coder.

2. La méthode revendiquée à la revendication 1, comprenant de plus fournir une liste de codes alphanumériques et transformer la liste de codes alphanumériques en liste d'entiers à coder.

3. La méthode revendiquée à la revendication 1, dans laquelle le système d'indexation permet d'indexer chacun de la pluralité de sommets de ladite pyramide hyperspatiale.

4. La méthode revendiquée à la revendication 1, dans laquelle l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale comprend un entier représentant l'indice du point projectif dans le système d'indexation, m le degré total de ladite liste d'entiers et n le nombre d'entiers de la liste d'entiers.

5. La méthode revendiquée à la revendication 4, comprenant de plus transformer l'entier représentant l'indice du point projectif par des coordonnées combinatoires mx et nx.

6. La méthode revendiquée à la revendication 4, comprenant de plus la transformation de m le degré total de la liste d'entiers et n le nombre d'entiers de la liste d'entiers par un entier Y représentant un indice global.

7. La méthode revendiquée à la revendication 6, comprenant de plus effectuer la sommation de l'entier Y représentant un indice global à l'entier représentant l'indice du point projectif pour fournir un indice total Z, l'indication de l'indexation dans la pyramide hyperspatiale comprenant l'indice total Z.

8. La méthode revendiquée dans une des revendications 1 à 7, dans laquelle la liste d'entiers est obtenue d'un fichier.

9. La méthode revendiquée dans une des revendications 1 à 8, dans laquelle l'étape de fournir l'indication de l'indexation dans la pyramide hyperspatiale comprend le stockage de ladite indication de l'indexation dans un fichier.

10. La méthode revendiquée dans une des revendications 1 à 8, dans laquelle l'étape de fournir l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale comprend la transmission de l'indexation de la liste d'entiers à une unité de traitement par l'intermédiaire d'un réseau.

11. La méthode revendiquée dans une des revendications 1 à 10 pour coder une liste d'entiers relatifs, la méthode comprenant fournir la liste d'entiers relatifs et générer la liste d'entiers au moyen de la liste d'entiers relatifs.

12. La méthode revendiquée dans une des revendications 1 à 8, dans laquelle l'étape de fournir l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale comprend l'affichage de ladite indication de l'indexation de la liste d'entiers sur un écran.

13. L'utilisation de la méthode revendiquée dans une des revendications 1 à 3 et 4 à 7 pour compresser une chaine de caractères dans laquelle la chaine de caractères est associée à ladite liste d'entiers.

14. Une méthode pour coder une liste de listes, la méthode comprenant l'utilisation itérative de la méthode revendiquée dans une des revendications 1 à 3 et 4 à 7 pour coder une liste de listes.

15. L'utilisation de la méthode revendiquée à la revendication 1 pour crypter une liste d'entiers, dans laquelle l'indexation de la liste d'entiers correspond au cryptage de la liste d'entiers.

16. Une méthode pour décoder de l'information codée au moyen de la méthode pour coder de l'information revendiquée dans une des revendications 1 à 7, la méthode comprenant :

obtenir l'indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale;

obtenir une indication d'un système d'indexation utilisé pour générer ladite indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale;

déterminer au moyen de l'indexation et de l'indication d'un système d'indexation utilisé ladite liste d'entiers.

fournir ladite liste d'entiers.

17. Un ordinateur comprenant :

une unité d'affichage;

un processeur;

une unité de stockage des données comprenant un programme configuré pour être exécuté par le processeur, le programme permettant de coder de l'information, le programme comprenant :

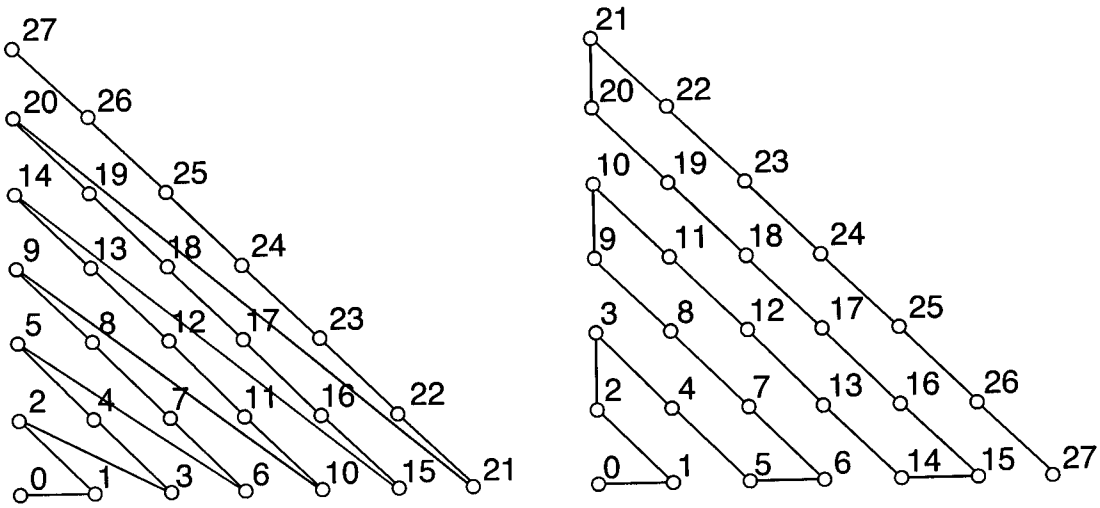
des instructions pour obtenir une liste d'entiers à coder;

des instructions pour déterminer une pyramide hyperspatiale ayant une taille adaptée pour coder la liste d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets dont le nombre est déterminé par le degré de la pyramide hyperspatiale qui est égal à la somme des entiers de la liste d'entiers et par la dimension de la pyramide hyperspatiale qui est égale au nombre d'entiers de la liste d'entiers moins un;

des instructions pour indexer la liste d'entiers dans cette pyramide au moyen d'un système d'indexation;

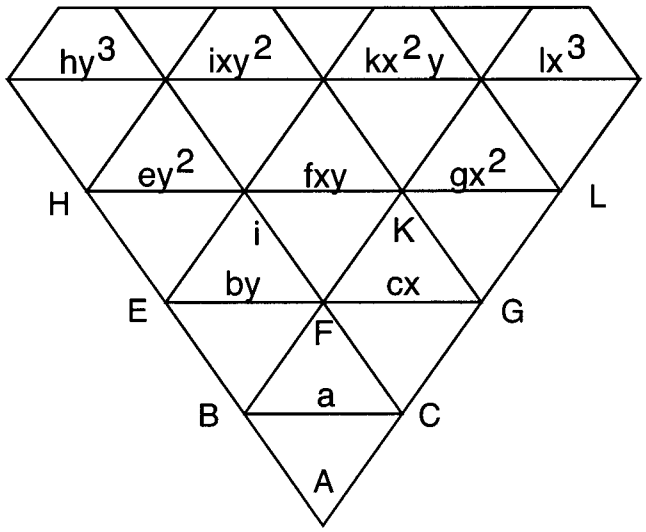
des instructions pour fournir une indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale afin de coder ainsi ladite liste d'entiers à coder.

18. Un dispositif de stockage d'information pour stocker des instructions exécutables par un processeur, qui quand elles sont exécutées vont causer l'exécution par le processeur d'une méthode pour coder de l'information, la méthode comprenant obtenir une liste d'entiers à coder; déterminer une pyramide hyperspatiale ayant une taille adaptée pour coder la liste d'entiers, la pyramide hyperspatiale ayant une pluralité de sommets dont le nombre est déterminé par le degré de la pyramide hyperspatiale qui est égal à la somme des entiers de la liste d'entiers et par la dimension de la pyramide hyperspatiale qui est égale au nombre d'entiers de la liste d'entiers moins un; indexer la liste d'entiers dans cette pyramide au moyen d'un système d'indexation; et fournir une indication de l'indexation de la liste d'entiers dans la pyramide hyperspatiale afin de coder ainsi ladite liste d'entiers à coder.



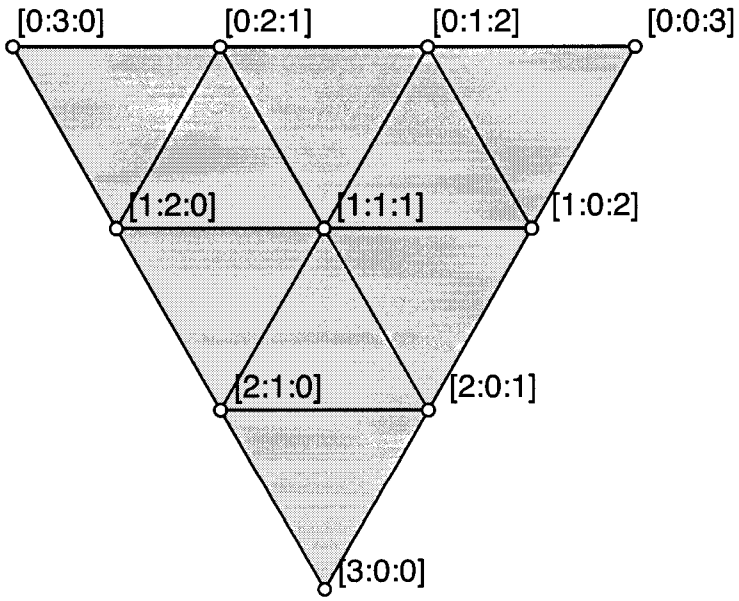
Ordre cantorien et cantorien boustrophédonique

FIG.1



Triangle original

FIG.2A

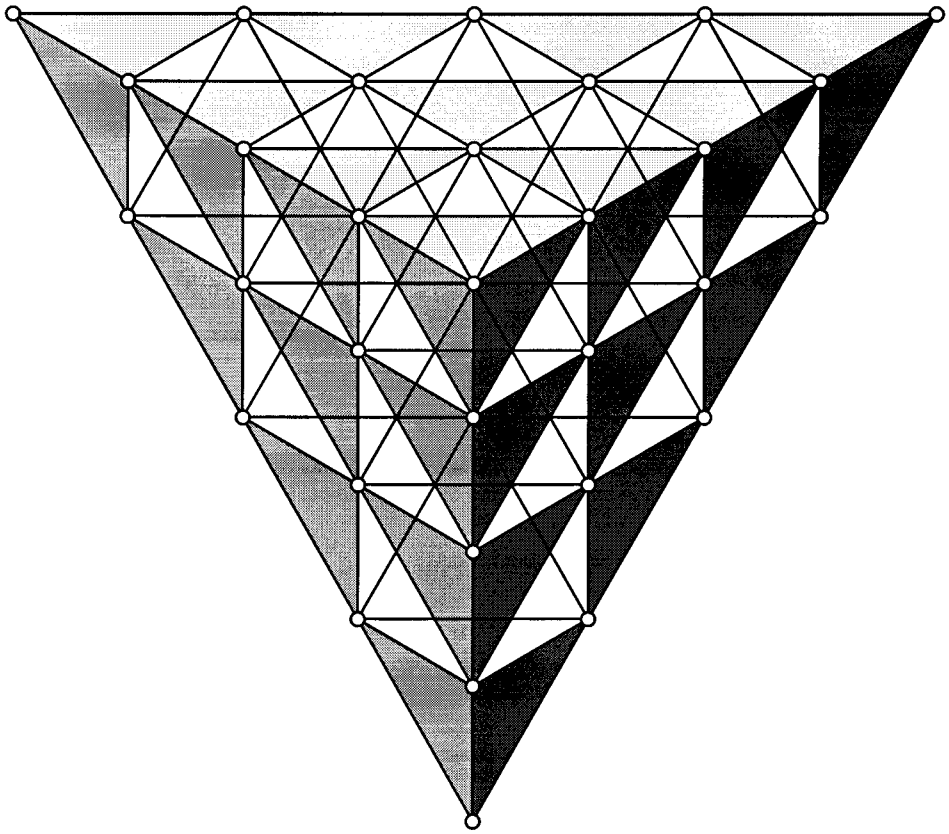


[0:0:3] => l
[0:1:2] => k
[0:2:1] => i
[0:3:0] => h
[1:0:2] => g
[1:1:1] => f
[1:2:0] => e
[2:0:1] => c
[2:1:0] => b
[3:0:0] => a

Zigzag de Malves

Triangle de De Gua de Malves et son extension projective

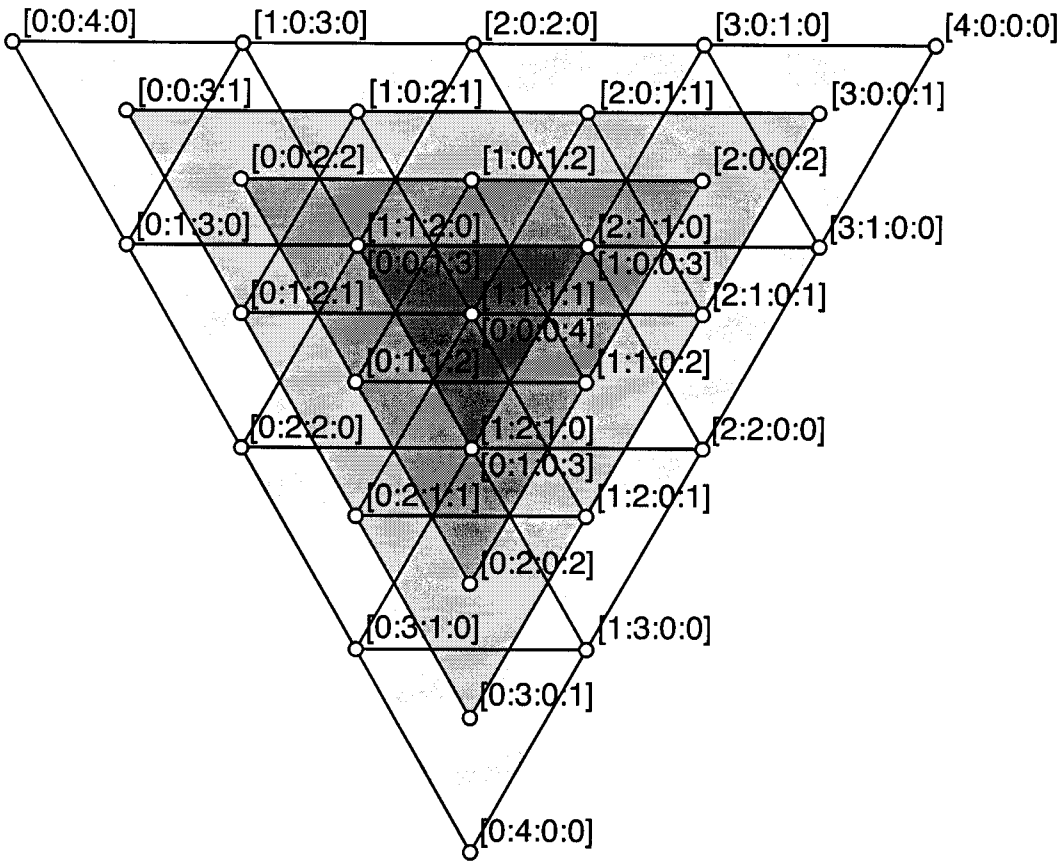
FIG.2B



[Treillis $T_{4,3}$]

Treillis pyramidal et étiquettage des coordonnées projectives

FIG.3A



Coordonnées
Treillis pyramidal et étiquettage des coordonnées projectives

FIG.3B

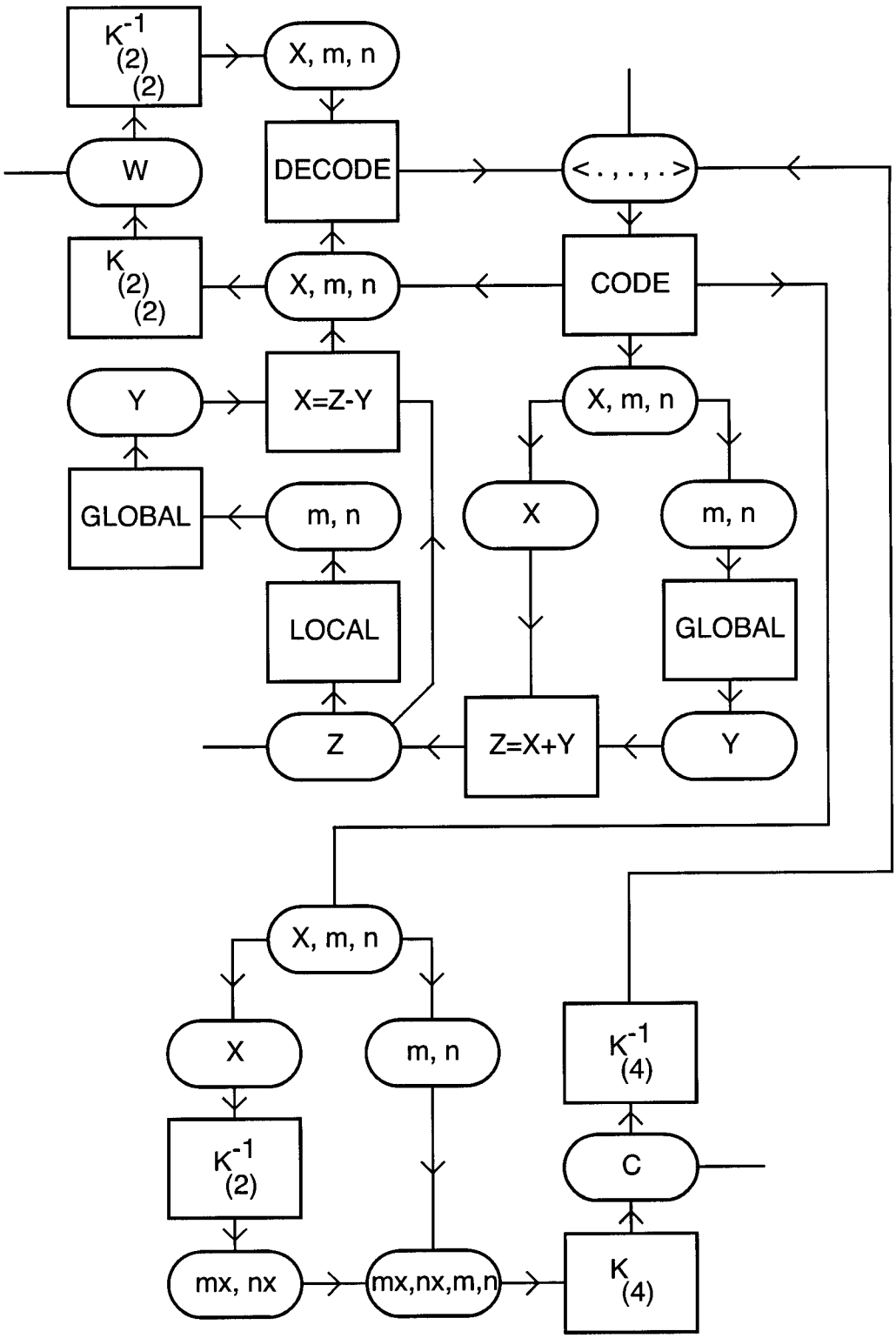

K-1 (2) (2)
X, m, n
W
DECODE
<.,.,.>
K (2) (2)
X, m, n
CODE
Y
X=Z-Y
X, m, n
GLOBAL
m, n
X
m, n
LOCAL
GLOBAL
Z
Z=X+Y
Y
X, m, n
X
m, n
K-1 (4)
K-1 (2)
C
mx, nx
mx,nx,m,n
K (4)

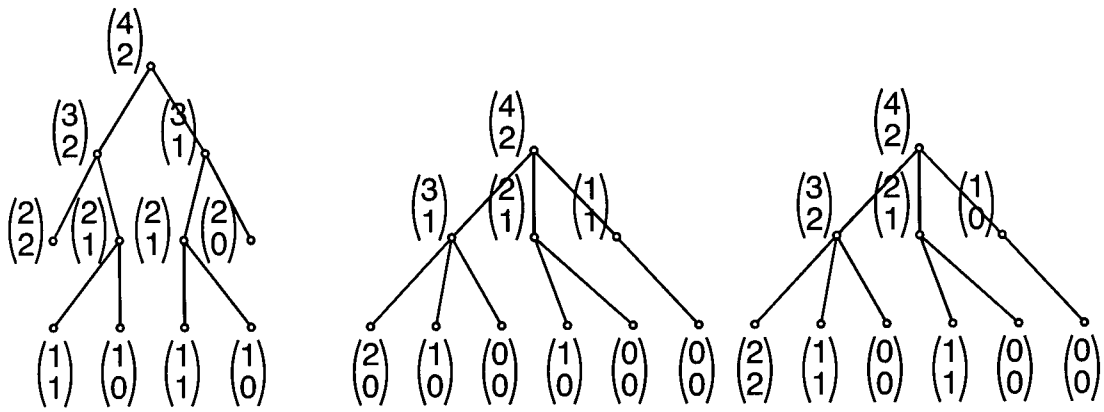
FIG.4

```
> GLOBAL:=proc(v2)
> local m,n,i,acc,ind;
> m:=v2[1];
> n:=v2[2];
> acc:=0;
> ind:=2^(m+n)-1;
> for i from 0 to m-1 do
>  acc:=acc+Comb(i,m+n-i);
> end do;
> ind+acc+1;
> end proc:
```

```
> LOCAL:=proc(x)
> local acc,i,j,ind;
> acc:=0;
> i:=0;
> j:=0;
> while x > = acc do
>    acc:=2^i;
>    i:=i+1;
> end do;
> ind:=x-2^(i-2);
> acc:=0;
> while ind > = acc do
>    acc:=acc+Comb(j,i-2-j);
>    j:=j+1;
> end do;
> [j-1,i-j-1];
> end proc:
```

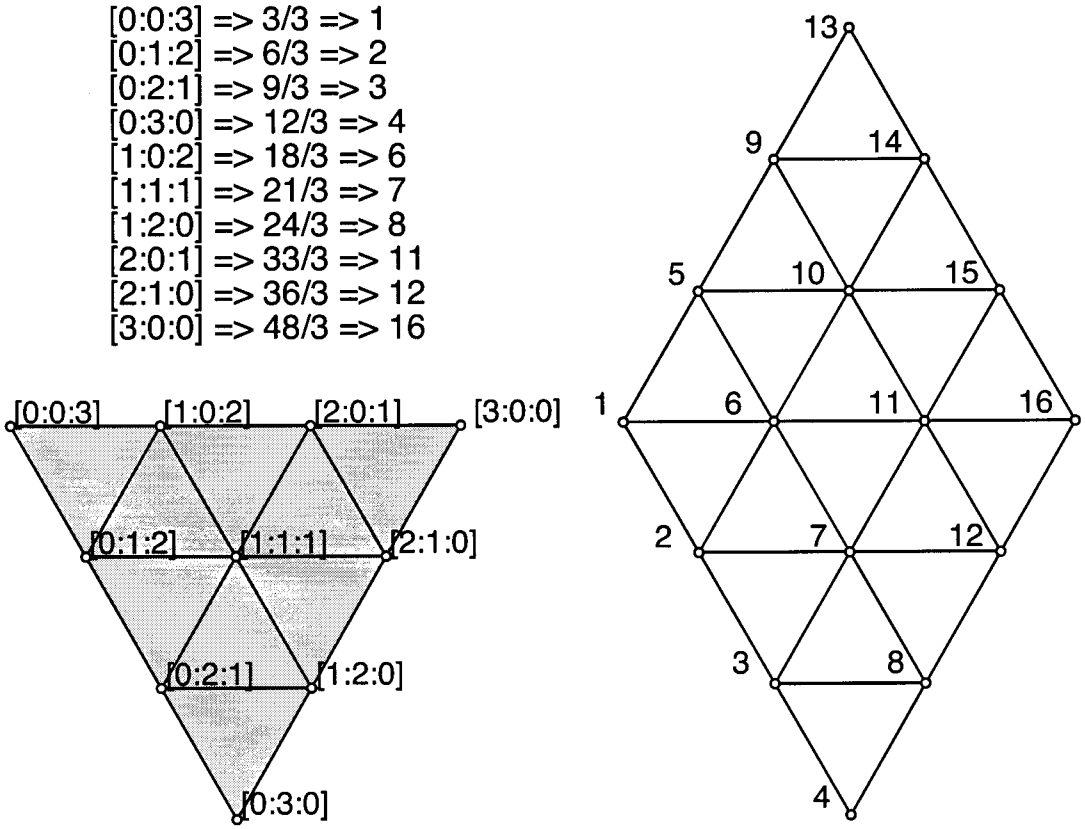
Codage et décodage de l'indice global

FIG.5



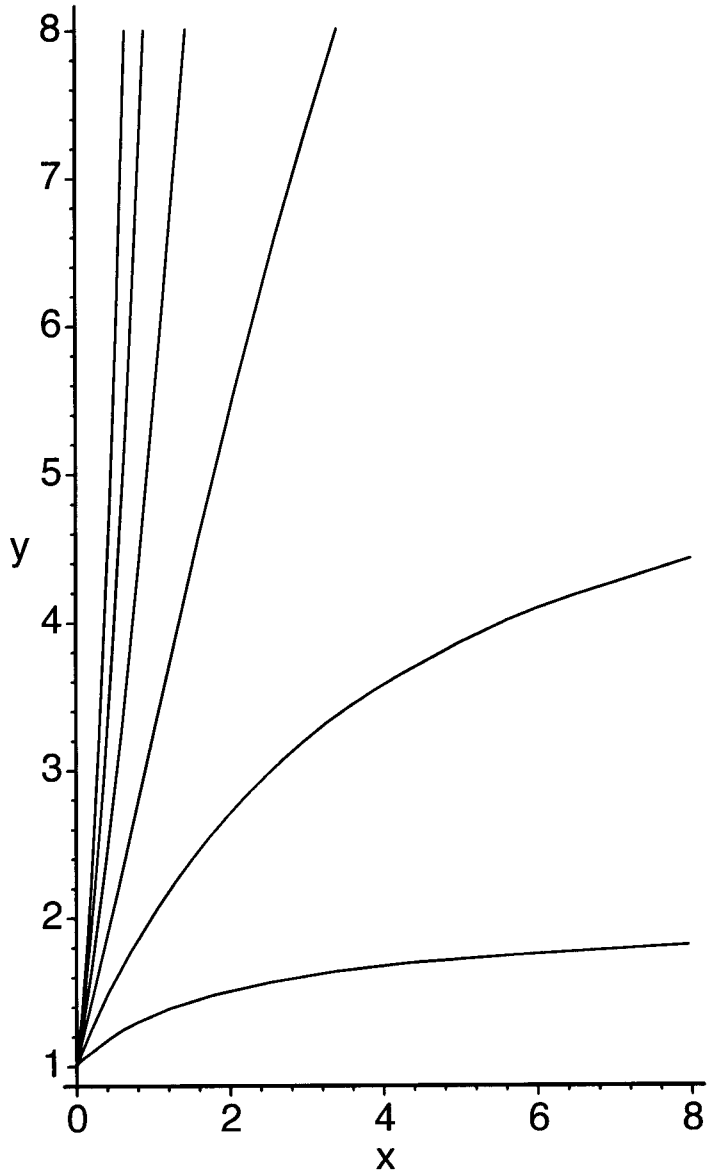
Arbres récursifs pour $\binom{4}{2}$

FIG.6



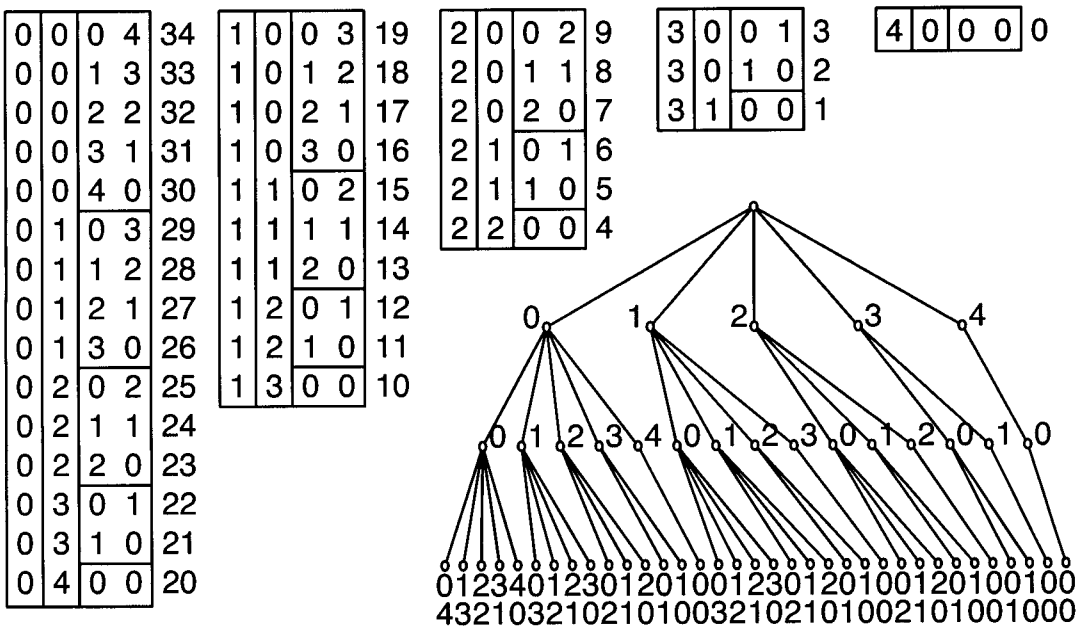
Indexation positionnelle pour $T_{3,2}$ en base quaternaire

FIG.7



Expansion des grands entiers

FIG.8



Ordre numérique pour le treillis $T_{4,3}$

FIG.9

```
> NUMORD:=proc(vn,n)
> local i,j,k,m,acci,accj,d;
> m:=0;
> d:=n;
> for k from 1 to d+1 do
>  m:=m+vn[k];
> end do;
1 acci:=0;
2 m:=m-1;
3 for i from 1 to n do
4  m:=m-vn[i];
5  acci:=acci+Comb(m,d-i+1);
6 end do;
> acci;
> end proc:
```

Codage Ordre Numérique

FIG.10

i	*acci*	*m*	*d - i + 1*	*Comb(m, d - i + 1*
1	126	4	5	$\binom{9}{5}$=126
2	196	4	4	$\binom{8}{4}$= 70
3	206	2	3	$\binom{5}{3}$= 10
4	209	1	2	$\binom{3}{2}$ =3
5	210	0	1	$\binom{1}{1}$ =1

	0	1	2	3	4	5	6	7	8	9
0	1									
1	1	1								
2	1	2	1							
3	1	3	3	1						
4	1	4	6	4	1					
5	1	5	10	10	5	1				
6	1	6	15	20	15	6	1			
7	1	7	21	35	35	21	7	1		
8	1	8	28	56	70	56	28	8	1	
9	1	9	36	84	126	126	84	36	9	1

Composition binomiale d'un nombre

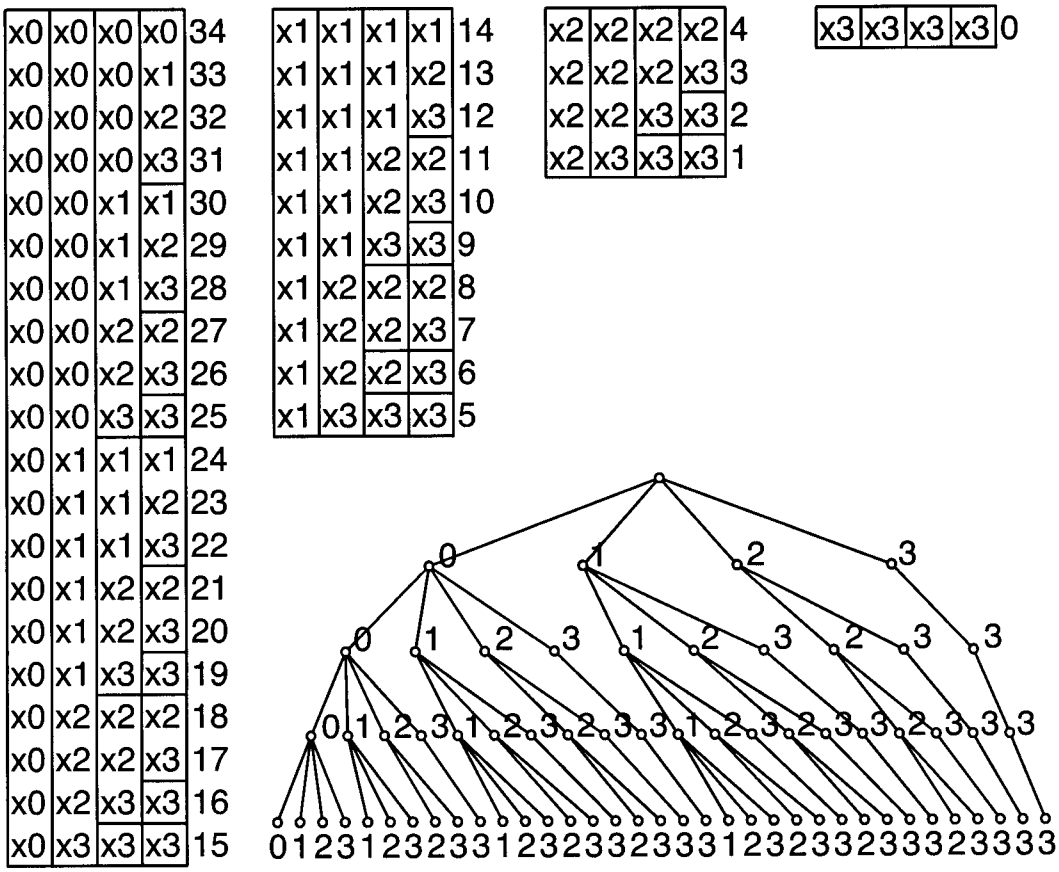
FIG.11

```
> ORDNUM:=proc(v3)
> local ind,m,n,d,i,x,v2;
> ind:=v3[1];
> m:=v3[2];
> n:=v3[3];
> d:=n;
> for i from 1 to d+1 do
>  x[i]:=0;
> end do;
> for i from 1 to d do
>  v2:=ORINUM(ind,m,n);
>  x[i]:=v2[1];
>  ind:=v2[2];
>  m:=m-x[i];
>  n:=n-1;
> end do;
> x[d+1]:=m;
> convert(x,array);
> end proc:
```

```
> ORINUM:=proc(ind,m,n)
> local j,t0,t1,exp;
> t1:=0;
> for j from 0 to m do
>  t0:=t1;
>  t1:=Cm(j,n);
>  if t1 > ind then
>    exp:=m-j;
>    j:=m;
>  end if;
> end do;
> [exp,ind-t0];
> end proc:
```

Décodage Ordre Numérique

FIG.12



Ordre numérique pour le treillis $T_{4,3}$

FIG.13

```
> LEXORD:=proc(vm,m,n)
> local i,acci,d;
> acci:=0;
> d:=n-1;
> for i from 0 to m-1 do
>   acci:=acci+Comb(m-i,d-vm[i+1]);
> end do;
> acci;
> end proc:
```

Codage ordre lexicographique

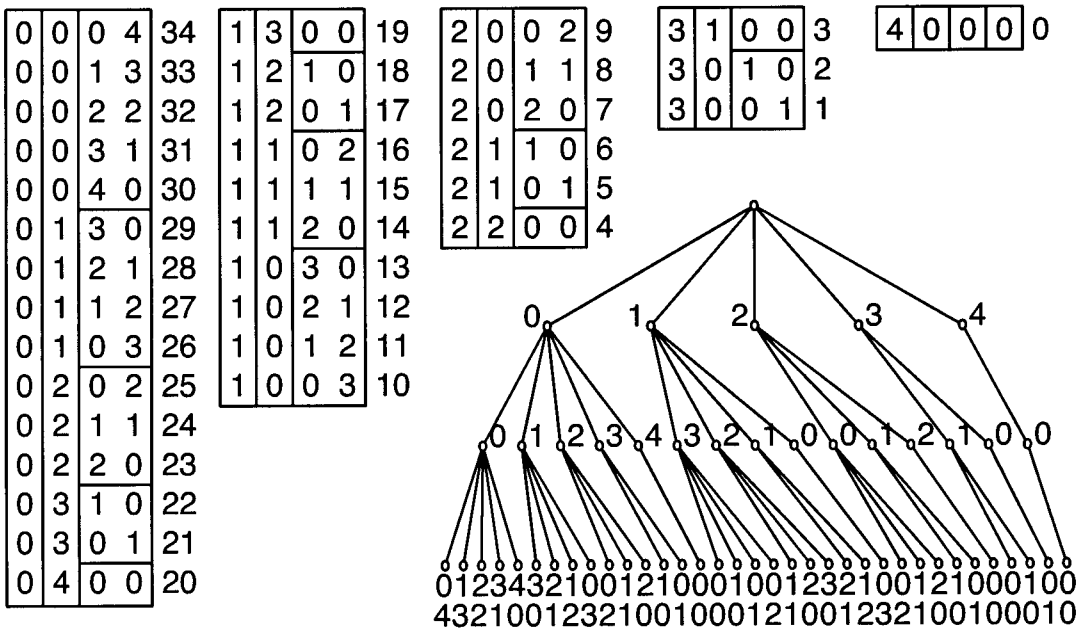
FIG.14

```
> ORDLEX:=proc(v3)
> local ind,i,m,n,x,v2,d,bloc;
> ind:=v3[1];
> m:=v3[2];
> n:=v3[3];
> d:=n+1;
> for i from 1 to m do x[i]:=0; end do;
>  for i from 1 to m do
>   v2:=ORILEX([ind,m,n]);
>   bloc:=v2[1];
>   ind:=v2[2];
>   x[i]:=d-bloc;
>   m:=m-1;
>   n:=bloc-1;
>  end do;
> convert(x,array);
> end proc:
```

```
> ORILEX:=proc(v3)
> local ind,m,n,i,acc,acci,v2;
> ind:=v3[1];
> m:=v3[2];
> n:=v3[3];
> acci:=0;
> for i from 0 to n do
>  acc:=acci;
>  acci:=Comb(m,i);
>  if ind < acci then
>   v2:=[i+1,ind-acc];
>   i:=n;
>  end if;
> end do;
> v2;
> end proc:
```

Décodage lexicographique

FIG.15



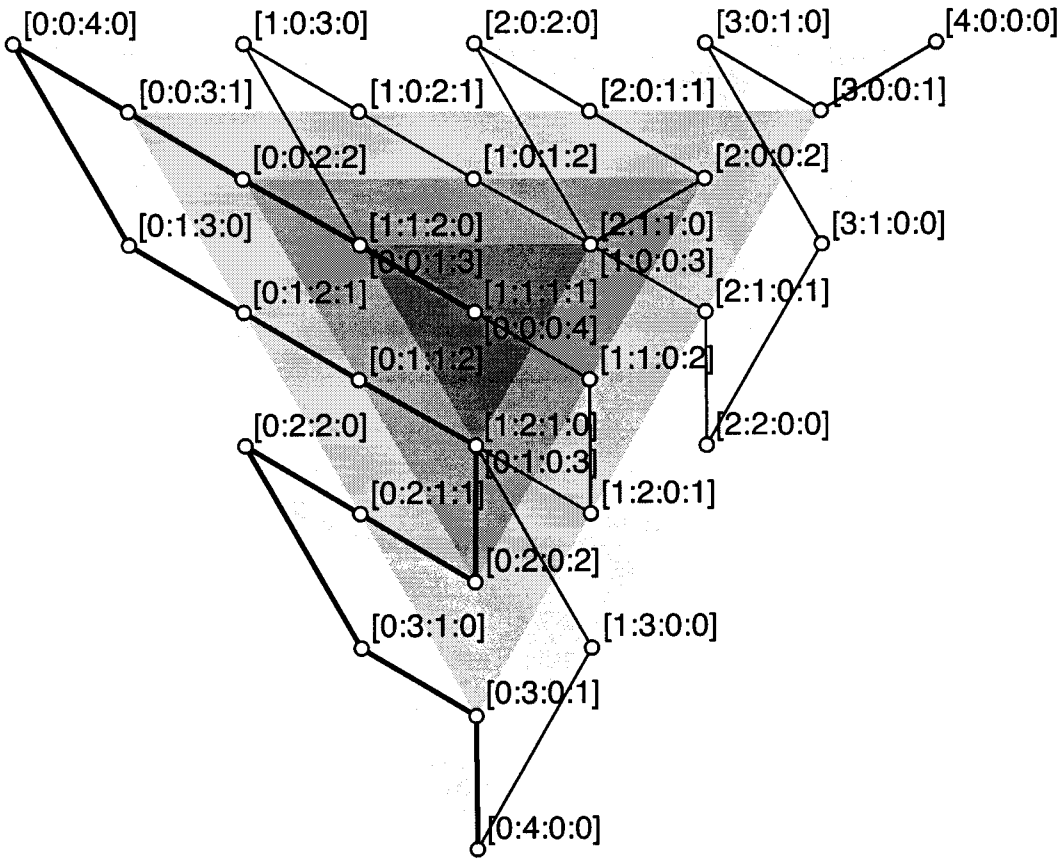
Codage de Gray projectif (4,3)

FIG.16

```
> GRAORD:=proc(vn,d)
> local i,j,k,m,acci,accj,n,accv,par,s;
> m:=0;
> n:=d;
> s:=1;
> for k from 1 to d+1 do
>  m:=m+vn[k];
> end do;
> acci:=0;
> m:=m-1;
> for i from 1 to n do
>  m:=m-vn[i];
>  n:=d-i;
>  accj:=Comb(m,n+1);
1  if (i=1)
2   then acci:=acci+accj;
3   else
4    par:=type(vn[i-1],even);
5    if par
6     then acci:=acci+s*accj;
7     else acci:=acci+s*(accv-1-accj);
8          s:=-1*s;
9     end if;
10  end if;
>  accv:=Comb(m+1,n);
> end do;
> acci;
> end proc:
```

Codage de Gray projectif

FIG.17



Parcours de $T_{4,3}$ *en Gray projectif*

FIG.18

```
> ORDGRA:=proc(v3)
> local ind,m,n,d,i,x,v2;
> ind:=v3[1];
> m:=v3[2];
> n:=v3[3];
> d:=n;
> for i from 1 to d+1 do
>  x[i]:=0;
> end do;
> for i from 1 to d do
>  v2:=ORIGRA(ind,m,n);
>  x[i]:=v2[1];
>  ind:=v2[2];
>  m:=m-x[i];
>  n:=n-1;
> end do;
> x[d+1]:=m;
> convert(x,array);
> end proc:
```

```
> ORIGRA:=proc(ind,m,n)
> local j,t0,t1,exp,p;
> p:=ind;
> if type(m,odd)
>  then p:=Comb(m,n)-ind-1;
> end if;
> t1:=0;
> for j from 0 to m do
>  t0:=t1;
>  t1:=Comb(j,n);
>  if t1 > p then
>   exp:=m-j;
>   j:=m;
>  end if;
> end do;
> if type(m,odd)
>  then [exp,Comb(m-exp,n-1)-(p-t0)-1];
>  else [exp,p-t0];
> end if
> end proc:
```

Décodage de Gray projectif

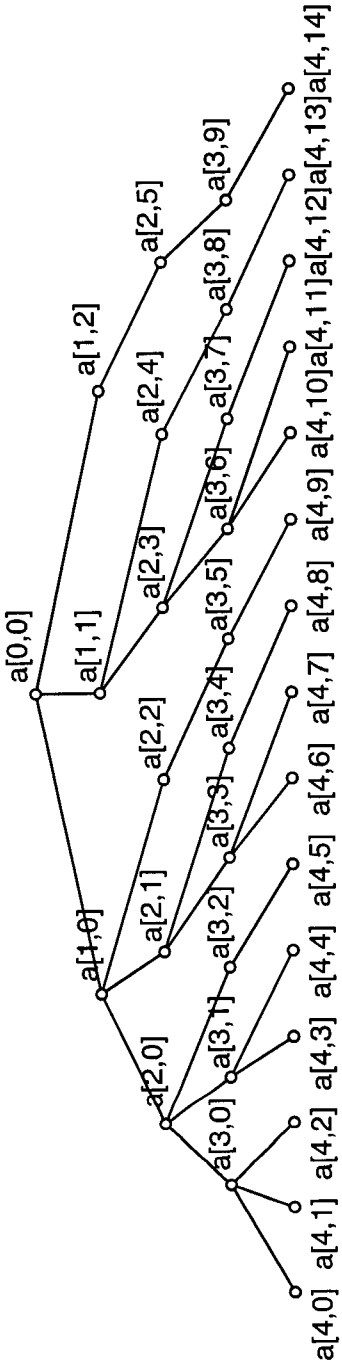
FIG.19

```
> VN2VM:=proc(vn,m,n)
> local i,j,acc,vm,exp,res;
> for i from 1 to m do
>  vm[i]:=0;
> end do;
> acc:=0;
> for i from 0 to n do
>  for j from 1 to vn[n+1-i] do
>   acc:=acc+1;
>   vm[acc]:=i;
>  end do;
> end do;
> convert(vm,array);
> end proc:
```

```
> VM2VN:=proc(vm,m,n)
> local i,j,acc,vn,coo;
> for i from 1 to n+1 do
>  vn[i]:=0;
> end do;
> acc:=0;
> for i from 1 to m do
>  coo:=vm[i];
>  vn[n-coo+1]:=vn[n-coo+1]+1;
> end do;
> convert(vn,array);
> end proc:
```

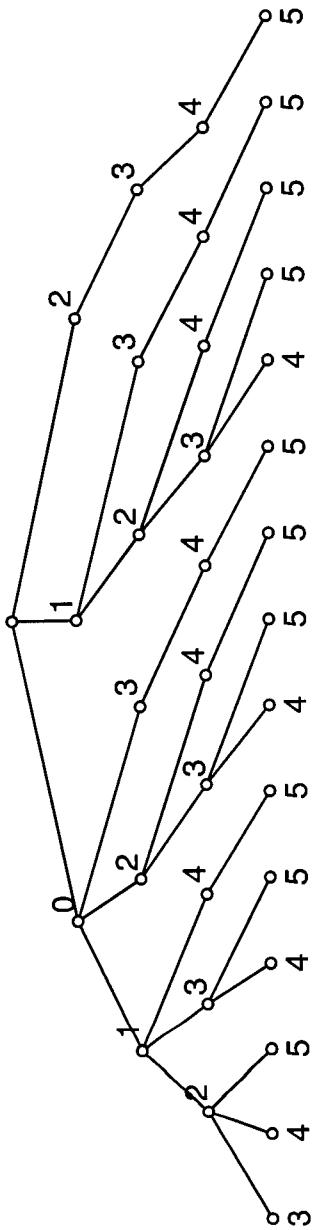
Transcodage entre vecteurs

FIG.20



Arbre générique

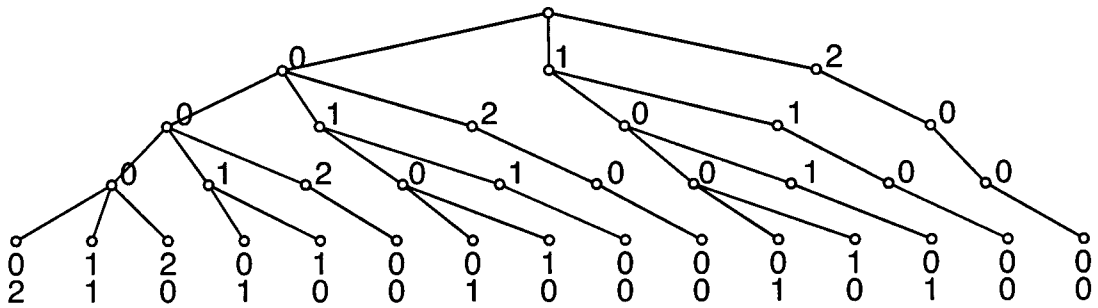
FIG.21A



Arbre des combinaisons

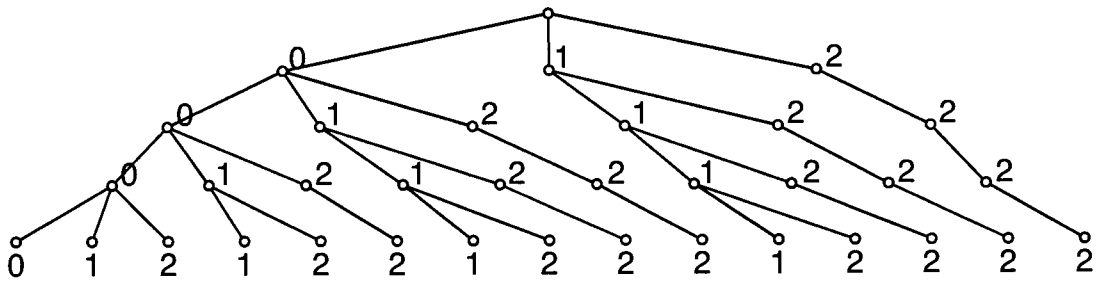
Arbre générique et combinatiore $\binom{6}{4}$

FIG.21B



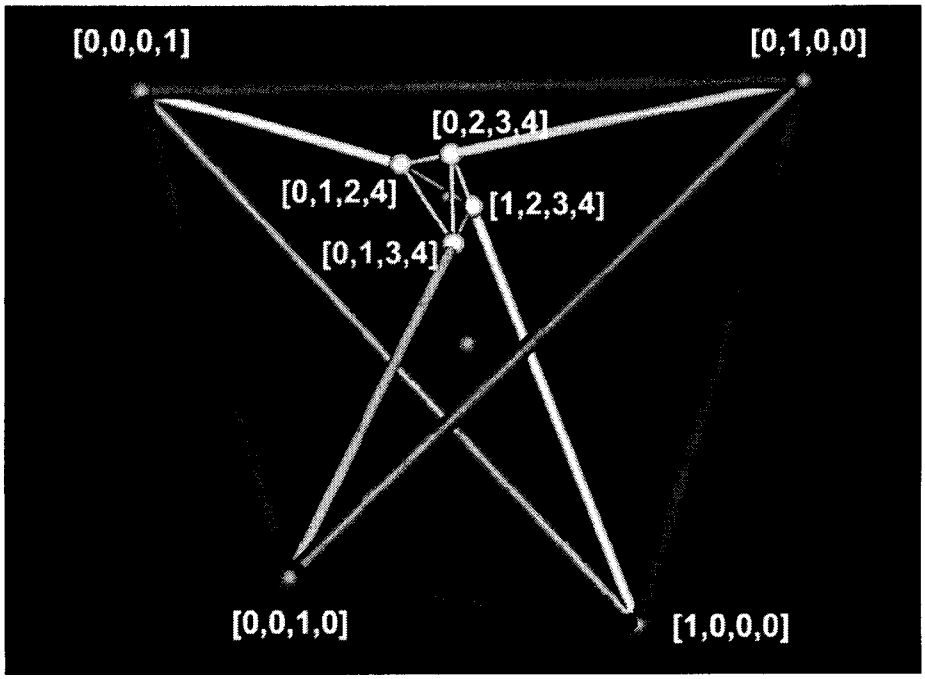
Arbre $\binom{6}{4}$ et étiquettes numériques

FIG.22



Arbre $\binom{6}{2}$ et étiquettes lexicographiques

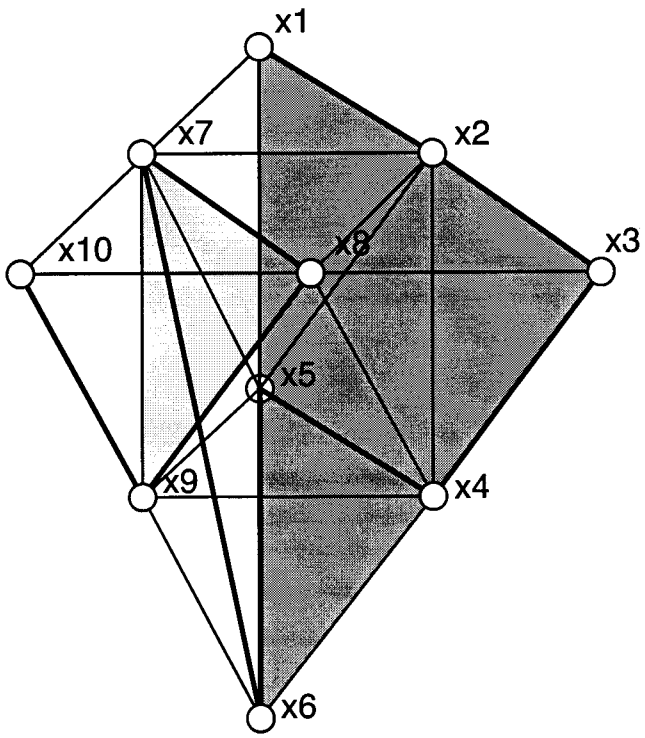
FIG.23



Sous-espace projectif combinatoire

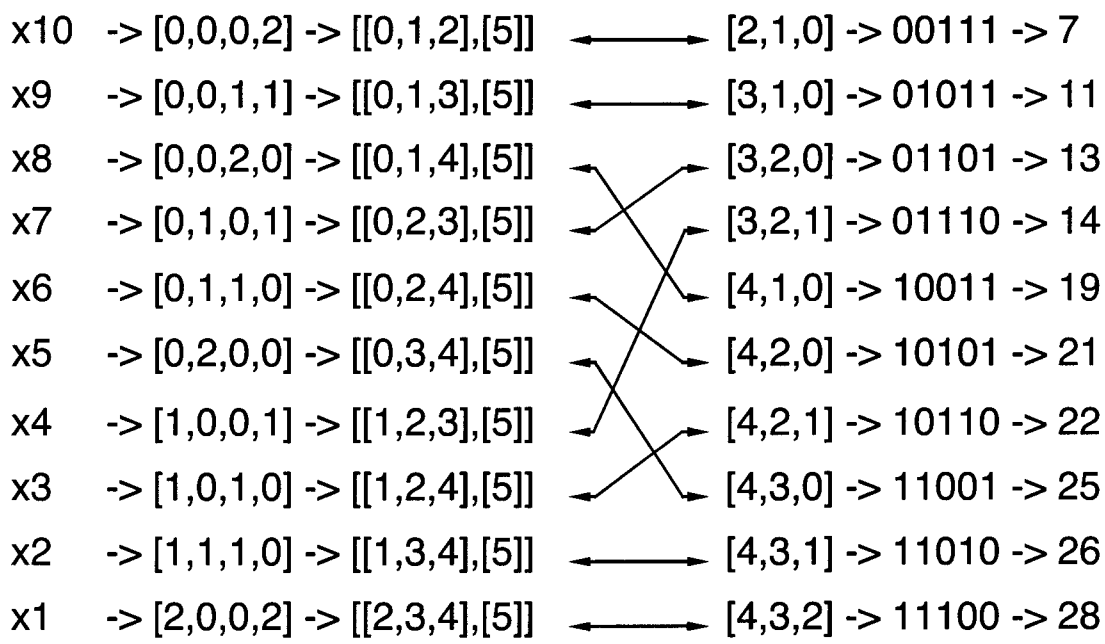
FIG.24

x10 -> [0,0,0,2] -> [[0,1,2],[5]]
x9 -> [0,0,1,1] -> [[0,1,3],[5]]
x8 -> [0,0,2,0] -> [[0,1,4],[5]]
x7 -> [0,1,0,1] -> [[0,2,3],[5]]
x6 -> [0,1,1,0] -> [[0,2,4],[5]]
x5 -> [0,2,0,0] -> [[0,3,4],[5]]
x4 -> [1,0,0,1] -> [[1,2,3],[5]]
x3 -> [1,0,1,0] -> [[1,2,4],[5]]
x2 -> [1,1,0,0] -> [[1,3,4],[5]]
x1 -> [2,0,0,0] -> [[2,3,4],[5]]



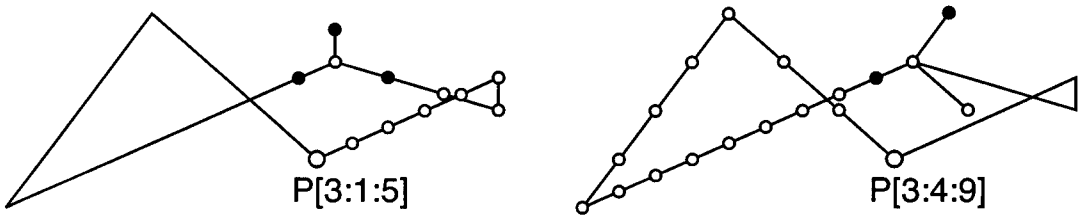
Bijection entre points projectifs et combinaisons

FIG.25



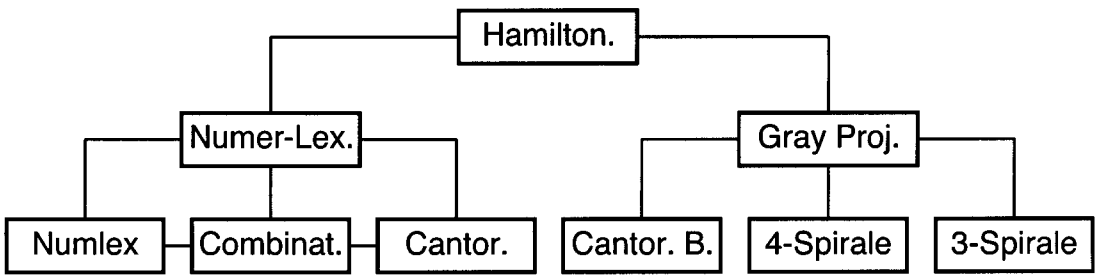
Génération projective et binaire des combinaisons

FIG.26



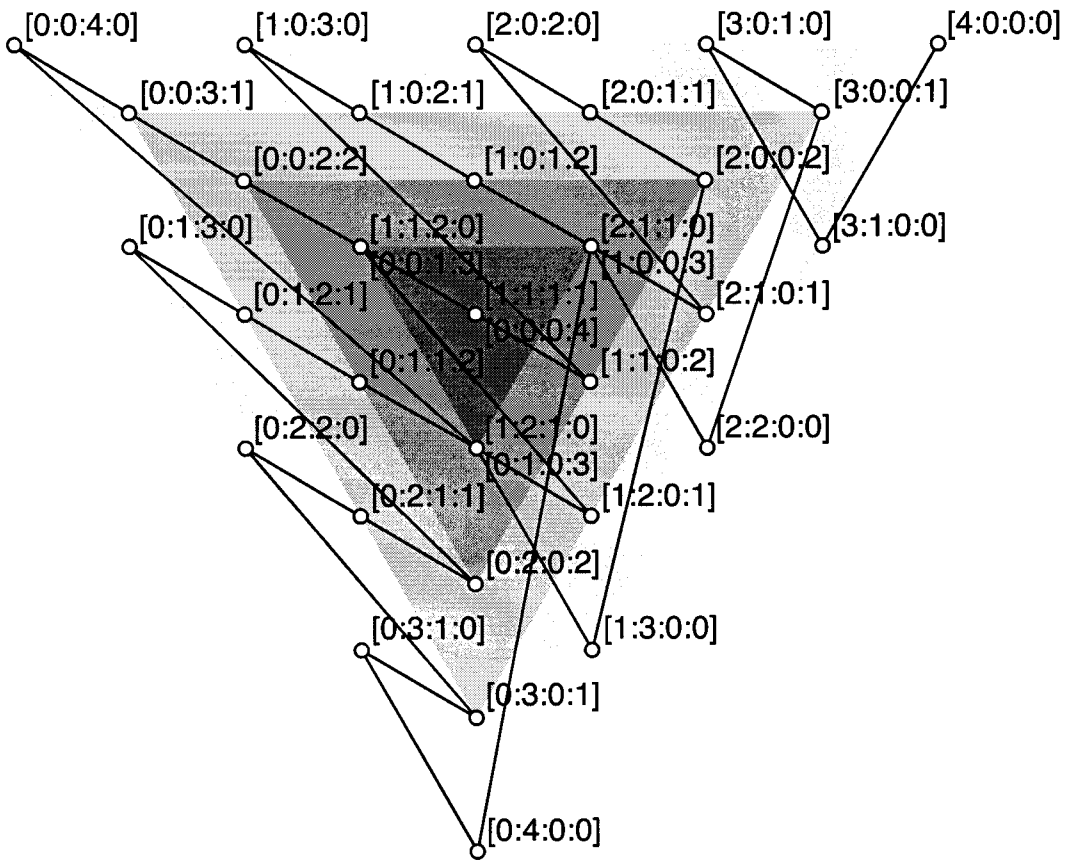
Coordonnées euclidiennes et affines d'un point cantorien

FIG.27



Taxinomie des ordres

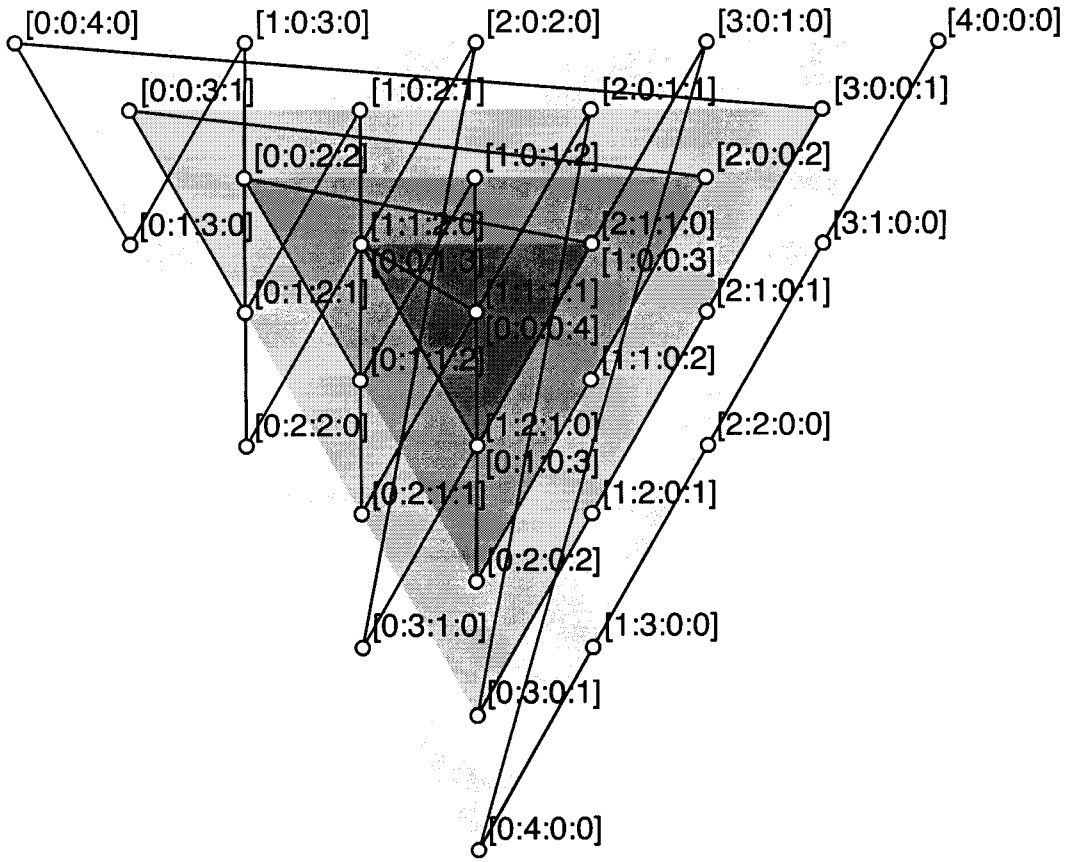
FIG.28



Chemins hamiltoniens cantoriens

Ascendant

FIG.29A



Chemins hamiltoniens cantoriens

Descendant

FIG.29B

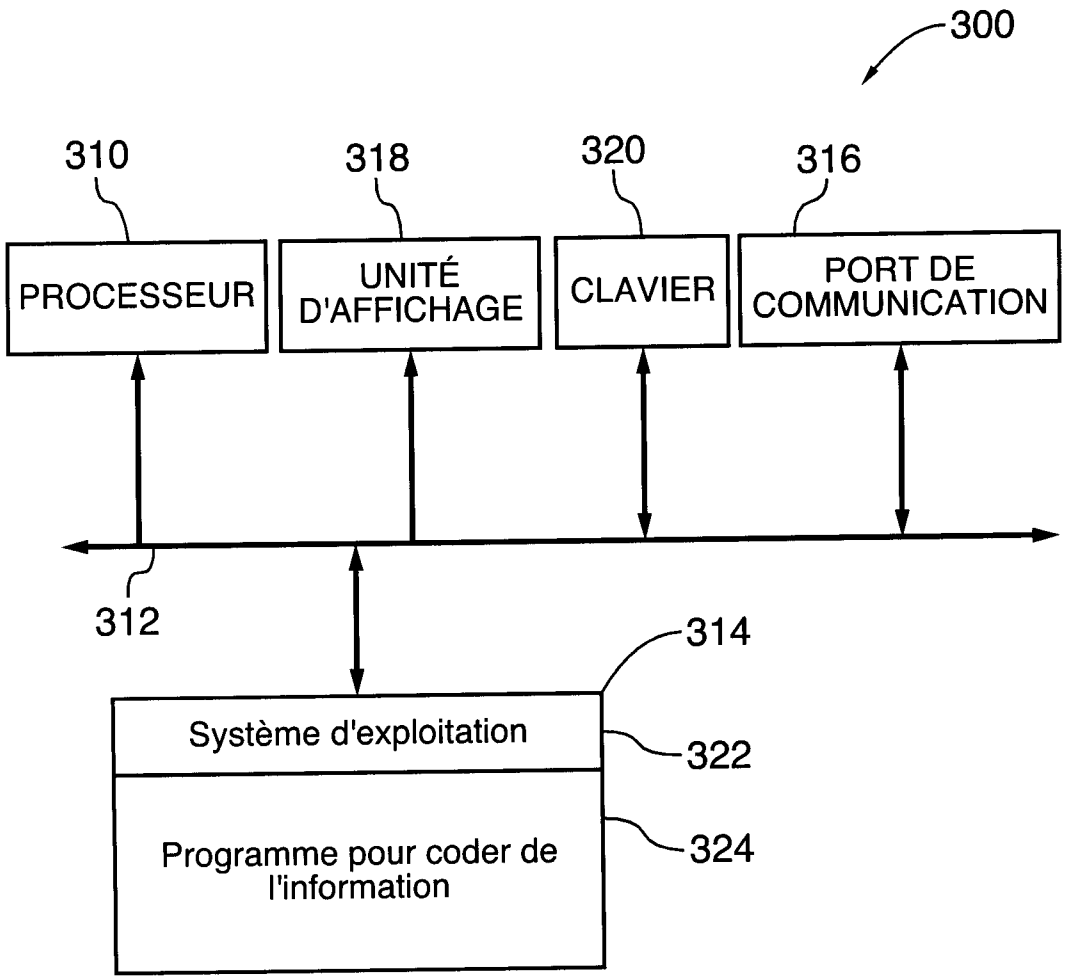
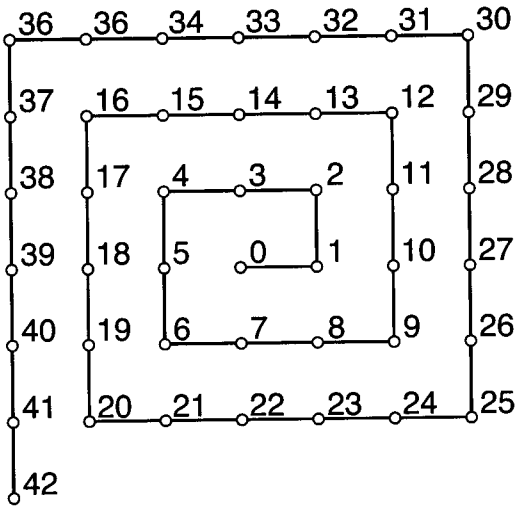
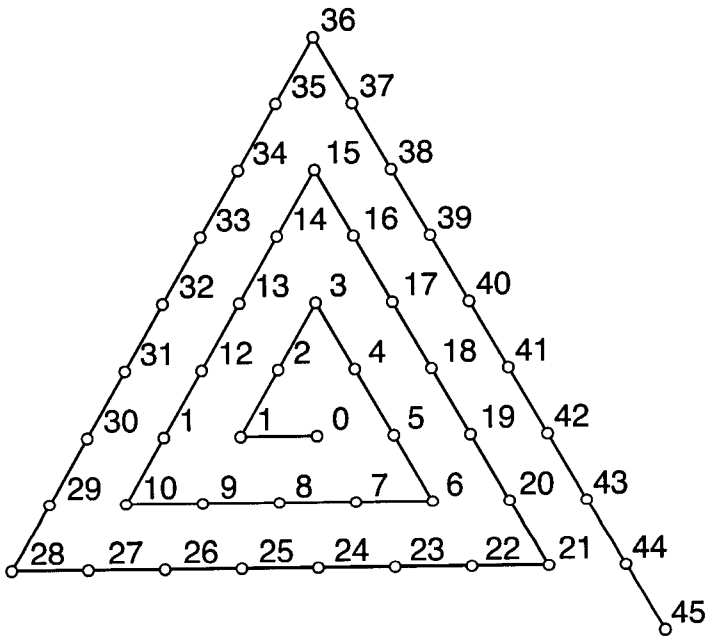

300
310
318
320
316
PROCESSEUR
UNITÉ D'AFFICHAGE
CLAVIER
PORT DE COMMUNICATION
312
314
Système d'exploitation
322
Programme pour coder de l'information
324

FIG.30



4-Spirale orthogonale

FIG.31A



3-Spirale triangulaire
Chemins en spirale

FIG.31B

```
> SPIORDc:=proc(v2)
> local x,y,ind;
> x:=v2[1];
> y:=v2[2];
> if (x=0) and (y=0)
>  then
>   ind:=0;
> end if;
> if (x > 0) and (abs(y) <= abs(x))
>  then
>   ind:=4*x^2-3*x+y;
> end if;
> if (y > 0) and (abs(x) <= abs(y))
>  then
>   ind:=4*y^2-y-x;
> end if;
> if (x < 0) and (abs(y) <= abs(x))
>  then
>   ind:=4*x^2-x-y;
> end if;
> if (y < 0) and (abs(x) <= abs(y))
>  then
>   ind:=4*y^2-3*y+x;
> end if;
> ind;
> end proc:
```

```
> ORDSPIc:=proc(num)
> local eq,eq0,eq1,x,y;
> eq:=isqrt(num);
> eq0:=(eq-1)*eq;
> eq1:=eq*eq;
> if type(eq,odd)
>  then
>   if (num < eq1)
>    then
>     y:=1-(eq+1)/2;
>     x:=num-eq0+y;
>    else
>     x:=(eq+1)/2;
>     y:=num-eq1-x+1;
>   end if;
>  else
>  if (num < eq1)
>   then
>     y:=eq/2;
>     x:=eq0+y-num;
>   else
>     x:=-eq/2;
>     y:=eq1-x-num;
>     end if;
> end if;
> [x,y];
> end proc:
```

Codage et décodage cartésien en spirale

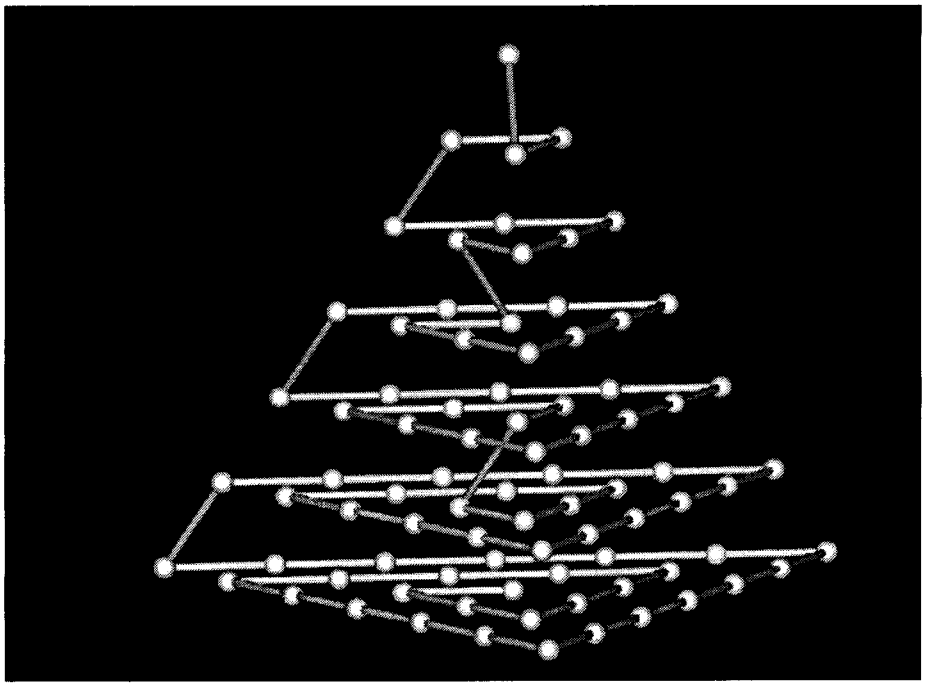
FIG.32

```
> SPIORDa:=proc(v2)
> local ind,x,y;
> x:=v2[1];
> y:=v2[2];
> if (x > 0) and (y <= 2*x)
>   then
>     ind:=x*(9*x-7)/2+y;
>   else
>    if (y < 0) and (x <= 2*y)
>      then
>        ind:=y*(9*y-7)/2+x;
>      else
>        ind:=(9*(x-y)^2-x-y)/2;
>    end if;
> end if;
> ind;
> end proc:
```

```
> ORDSPIa:=proc(z)
> local ind,i,j,ior,imod,x,y;
> i:=iquo((isqrt(1+8*z)-3),2)+1;
> ior:=i*(i+1)/2;
> j:=z-ior;
> imod:=i mod 3;
> if (imod = 2)
>   then
>    x:=i/3-j+1/3;
>    y:=2*i/3-j+2/3;
>  else
>    if (imod = 1)
>     then
>      x:=(i-1)/3+1;
>      y:=j-(i-1)/3;
>    else
>      x:=j-2*i/3;
>      y:=-i/3;
>    end if;
> end if;
> [x,y];
> end proc:
```

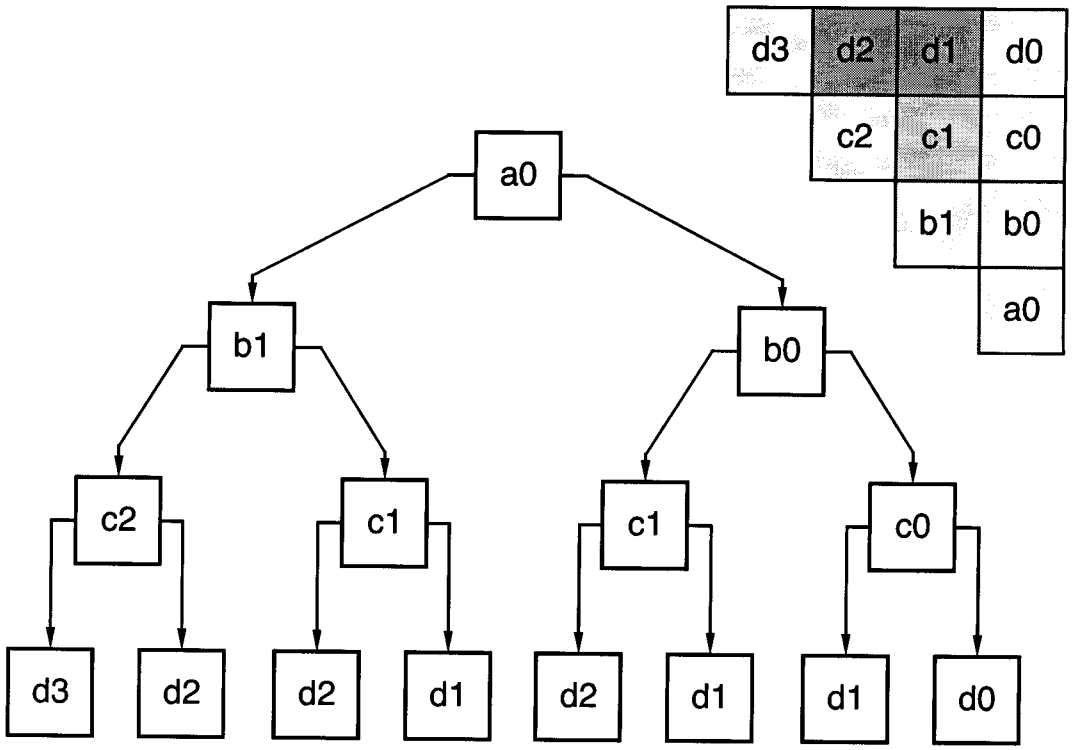
Codage et décodage affine en spirale

FIG.33



Chaîne en Spirale pour $T_{6,3}$

FIG.34



Mécanisme de convolution

FIG.35

0									
1	1	1							
2	1	2	1						
3	1	3	3	1					
4	1	4	6	4	1				
5	1	5	10	10	5	1			
6	1	6	15	20	15	6	1		
7	1	7	21	35	35	21	7	1	
8	1	8	28	56	70	56	28	8	1
9	1	9	36		126	126	84	36	9	1

Convolutions de Vandermonde pour $\binom{9}{3}$

FIG.36

```
> CRKREC:=proc(r,k)
> local i,acc;
> if k=0 then return 1; end if;
> if k=1 then return r; end if;
> if k=r
>  then 1;
> else
>  acc:=0;
>  for i from 0 to k do
>   acc:=acc+CRKREC(r-k,min(k-i,r-k-k+i))*CRKREC(k,min(i,k-i));
>  end do;
> return acc;
> end if;
> end proc:
```

Récursivité par addition de produits

FIG.37

```
> XENUME:=proc(x,y)
> local i,alpha,res;
> alpha:=1;
> res[1]:=1;
> for i from 1 to x do
>   alpha:=(alpha*(y+1-i))/i;
>   res[i+1]:=alpha;
> end do;
> convert(res,array);
> end proc:
```

```
> YENUME:=proc(x,y)
> local acc,i,m,n,res;
> m:=x;
> n:=y-x;
> acc:=1;
> res[1]:=1;
> for i from 1 to n do
>   acc:=(acc*(m+i))/i;
>   res[i+1]:=acc;
> end do;
> acc;
> convert(res,array);
> end proc:
```

Calcul des binomes par ligne et par colonne

FIG.38

```
> Combo:=proc(m,n)
> local mt,nt,xx,i,acc,res,den;
> if n < 0 then return 0; end if;
> if type(n,odd)
>   then
>    mt:=m+(n+1)/2;
>    acc:=mt;
>    nt:=(n-1)/2;
>   else
>    mt:=m+(n+2)/2;
>    acc:=mt*(m+1);
>    nt:=(n-2)/2;
> end if;
> xx:=mt*mt;
> for i from 1 to nt do
>  den:=4*i*i+2*i;
>  acc:=acc*(xx-i*i)/den;
> end do;
> if type(n,odd)
>  then acc;
>  else acc/n;
> end if;
> end proc:
```

Calcul par produit de différences de carrés

FIG.39

	0	1	2	3	4	5	6	7	8	9
0	1									
1	1	1								
2	1	2	1							
3	1	3	3	1						
4	1	4	6	4	1					
5	1	5	10	10	5	1				
6	1	6	15	20	15	6	1			
7	1	7	21	35	35	21	7	1		
8	1	8	28	56	70	56	28	8	1	
9	1	9	36	84	126	126	84	36	9	1

Changement d'indiçage du triangle de Pascal

FIG.40A

	0	1	2	3	4	5	6	7	8	9
0	1	1	1	1	1	1	1	1	1	1
1	1	2	3	4	5	6	7	8	9	10
2	1	3	6	10	15	21	28	36	45	55
3	1	4	10	20	35	56	84	120	165	220
4	1	5	15	35	70	126	210	330	495	715
5	1	6	21	56	126	252	462	792	1287	2002
6	1	7	28	84	210	462	924	1716	3003	5005
7	1	8	36	120	330	792	1716	...	...	...
8	1	9	45	165	495	1287	3003	...	...	...
9	1	10	55	220	715	2002	5005	...	...	...

Changement d'indiçage du triangle de Pascal

FIG.40B

```
> PRODYN:=proc(m,n)
> local i,j,T;
> for i from 1 to n+1 do
>   T[i]:=1;
> end do;
> for i from 1 to m do
>    for j from 2 to n+1 do
>     T[j]:=T[j]+T[j-1];
>   end do;
> end do;
> end proc:
```

Programmation Dynamique

FIG.41

```
> MATPRT:=proc(vn0,vn1,n)
> local i,j,acc,wn;
> acc:=0;
> wn:=array(1..n+1);
> for i from 0 to n do
>    for j from i to 0 by -1 do
>      acc:=acc+vn0[j+1]*vn1[i-j+1];
>    end do;
> wn[i+1]:=acc;
> acc:=0;
> end do;
> convert(wn,list);
> end:
```

Produit matriciel simplifié

FIG.42

2^0	2^1		2^2				2^3		2^4		2^5	
1	1	1	1	1	1	1	1	1	1	1	1	1
1	2	3	4	5	6	7	8	...	16	...	32	...
1	3	6	10	15	21	28	36	...	...	...	...	...
1	4	10	20	35	56	84	120	...	...	...	...	...
1	5	15	35	70	126	210	330	...	...	...	...	...
1	6	21	56	126	252	462	792	...	...	...	...	...
1	7	28	84	210	462	924	1716	...	...	...	...	...

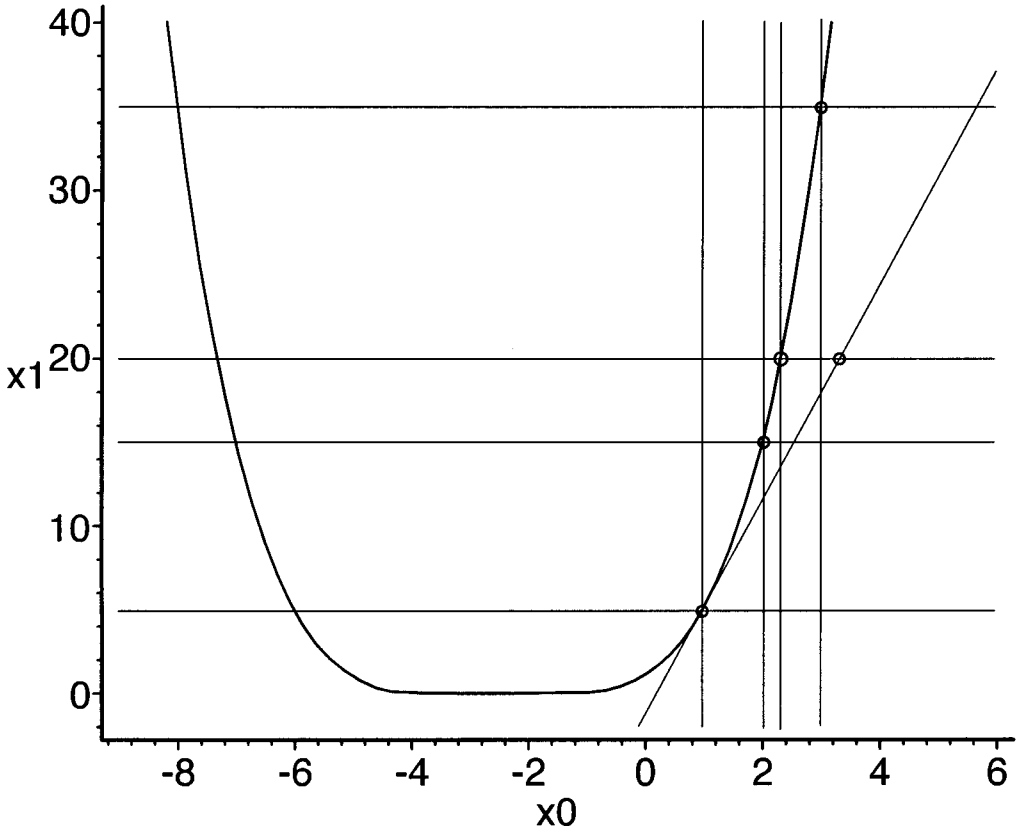
Adressage binaire

FIG.43A

1	2	3		5			8					13
1	1	1	1	1	1	1	1	1	1	1	1	1
1	2	3	4	5	6	7	8	...	...	...	...	13
1	3	6	10	15	21	28	36	...	...	...	...	...
1	4	10	20	35	56	84	120	...	...	...	...	...
1	5	15	35	70	126	210	330	...	...	...	...	...
1	6	21	56	126	252	462	792	...	...	...	...	...
1	7	28	84	210	462	924	1716	...	...	...	...	...

Adressage de Fibonacci

FIG.43B



$$y = \frac{(m+1)(m+2)(m+3)(m+4)}{4!}$$

FIG.44

```
> ENCDEC:=proc(y0,n)
> local i,j,prd,g,p,prdsv,dens;
> if y0 <= n
>   then return 0;
> end if;
> p:=DIGBIN(y0,n);
> g:=10^(p-1);
> for i from 1 to p do
> prd:=g;
> dens:=1;
>   for j from 1 to 10
>    while dens >= 0 do
>      prdsv:=prd;
>      prd:=g+j*10^(p-i);
>      dens:=y0-Comb(prd,n);
>   end do;
> g:=prdsv;
> end do;
> g;
> end proc:
```

```
> DIGBIN:=proc(y0,n)
> local x,prd,p,dens;
> x:=1;
> prd:=1;
> p:=-1;
> dens:=1;
> while dens >= 0 do
>   prd:=Comb(x,n);
>   dens:=y0-prd;
>   x:=x*10;
>   p:=p+1;
> end do;
> p;
> end proc:
```

Méthode compte-gouttes

FIG.45

```
> NUMFCT:=proc(x,n,g)
> local i,acc;
> acc:=1;
> for i from 1 to n do
>   acc:=acc*(x+g*i);
> end do;
> end proc:
```

Série diophantienne

FIG.46

```
> DRVREC:=proc(n,k)
>   if (n < 0) or (k < 0)
>    then
>     return 0:
>    else
>     if (n = 0) and (k = 0)
>      then
>       return 1:
>     else
>       return DRVREC(n-1,k)*(m+n)+k*DRVREC(n-1,k-1):
>   end if;
> end if;
> end proc:
```

Calcul récursif de la dérivée *k*-ième

FIG.47

```
> DRVSYM:=proc(n)
> local i,mi,num,den,prd,fac,drv,xi;
> drv:=1;
> prd:=x+1;
> fac:=1;
> for i from 1 to n do
>   num:=prd-fac*y0;
>   den:=drv;
>   xi:=(x+i+1);
>   drv:=drv*xi+prd;
>   prd:=prd*xi;
>   fac:=fac*(i+1);
> end do;
> x-num/den;
> end proc:
```

Schéma de Raphson-Newton de degré *n*

FIG.48

```
> BINRP4:=proc(x)
> local x0,a,b,c,d,res;
> a:=ENCDEC(x,4);
> x0:=x-Comb(a,4);
> b:=ENCDEC(x0,3);
> x0:=x0-Comb(b,3);
> c:=ENCDEC(x0,2);
> d:=x0-Comb(c,2);
> [d,c+1,b+1,a+1];
> end proc:
```

Décomposition binomiale quaternaire d'un nombre

FIG.49

```
> BINRPNi:=proc(x,n)
> local i,ind,a,x0,ls,res;
> x0:=x;
> for i from 1 to n-1 do
>   ind:=n+1-i;
>   a:=ENCDEC(x0,ind);
>   x0:=x0-Comb(a,ind);
>   if x0 < 0
>     then
>      x0:=0;
>       ls[i]:=a;
>     else
>       ls[i]:=a+1;
>   end if;
> end do;
> ls[n]:=x0;
> convert(ls,array);
> end proc:
```

```
> RPNBINi:=proc(vn,n)
> local i,acc,accsv;
>   acc:=0;
>   for i from 1 to n do
>     accsv:=Comb(vn[n-i+1]-1,i);
>     acc:=acc+accsv;
>   end do;
> end proc:
```

représentation binomiale n-aire

FIG.50

```
> GLOBALo:=proc(v2)
> local m,n,t0,acc,alpha,i,t2;
> m:=v2[1];
> n:=v2[2];
> t0:=m+n+1;
> acc:=0;
> alpha:=1;
> for i from 1 to m do
>   acc:=acc+alpha;
>   alpha:=(alpha*(t0-i))/i;
> end do;
> t2:=2^(m+n)+acc;
> end proc:
```

```
> LOCALo:=proc(x)
> local i,acc,j,ind,alpha;
> acc:=0;
> for i from 0 while x >= acc do
>   acc:=2^i;
> end do;
> ind:=x-2^(i-2);
> acc:=0;
> alpha:=1;
> for j from 1 while ind >= acc do
>   acc:=acc+alpha;
>   alpha:=(alpha*(i-1-j))/j;
> end do;
> [j-2,i-j];
> end proc:
```

Optimisation codage global et local

FIG.51

```
> ORDNUMo:=proc(v3)
> local ind,m,n,d,i,x,v2;
> ind:=v3[1];
> m:=v3[2];
> n:=v3[3];
> d:=n;
> for i from 1 to d+1 do
>   x[i]:=0;
> end do;
> for i from 1 to d do
>   v2:=ORINUMo(ind,m,n);
>   x[i]:=v2[1];
>   ind:=v2[2];
>   m:=m-x[i];
>   n:=n-1;
> end do;
> x[d+1]:=m;
> convert(x,array);
> end proc:
```

```
> ORINUMo:=proc(ind,m,n)
> local j,t0,t1,exp;
> if ind = 0
>  then [m,0];
>  else
>   t1:=1;
>    for j from 1 to m do
>      t0:=t1;
>      t1:=(t1*(n+j))/(j);
>      if t1 > ind then
>       exp:=m-j;
>        j:=m;
>      end if;
>    end do;
>   [exp,ind-t0];
> end if;
> end proc:
```

Optimisation décodage numérique

FIG.52

```
> INDCOM:=proc(vn,n)
> local m,x,v4,deg,a,b;
> m:=VECDEG(vn,n);
> x:=NUMORD(vn,n);
> a:=ENCDEC(x,2);
> b:=x-Combo(a,2);
> v4:=[b,a+1-b,n,m];
> deg:=VECDEG(v4,3);
> NUMORD(v4,3)+Combo(deg-1,4);
> end proc:
```

```
> COMIND:=proc(x)
> local y,v4,x0,x1,x2,m,n;
> v4:=BINRP4(x);
> m:=v4[1];
> n:=v4[2]-v4[1];
> x0:=v4[3]-v4[2];
> x1:=v4[4]-v4[3];
> x2:=1/2*(x0+x1)*(x0+x1+1)+x1;
> ORDNUMo([x2,m,n]);
> end proc:
```

Codage et décodage compressés

FIG.53

```
> MPLNUM:=proc(va,vb,dga,dgb,dim)
> local i,j,k,r,dgr,lga,lgb,lgr,v,siga,sigb,sigr,ind;
1  dgr:=dga+dgb;
2  lga:=1;
3  lgb:=1;
4  lgr:=1;
5  for k from 1 to dim do
6    lga:=lga*(dga+k)/k;
7    lgb:=lgb*(dgb+k)/k;
8    lgr:=lgr*(dgr+k)/k;
9  end do;
10  for k from 1 to lgr do v[k]:=0; end do;
11   for i from 1 to lga do
12    for j from 1 to lgb do
13      siga:=ORDNUM([lga-i,dga,dim]);
14      sigb:=ORDNUM([lgb-j,dgb,dim]);
15      sigr:=AddPol(siga,sigb,dim+1);
16      ind:=lgr-NUMORD(sigr,dim)-1;
17      v[ind+1]:=v[ind+1]+va[i]*vb[j];
18    end do;
19  end do;
20  convert(v,array);
21  end proc:
```

Produit de deux polynômes multivariés

FIG.54

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CA2012/001042

A. CLASSIFICATION OF SUBJECT MATTER

CIB: ***H03M 7/00*** (2006.01)

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H03M 7/00 (2006.01)

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Google (pyramide hyperspatiale, hyperspatial pyramid), Ieeexplore (pyramid, integer), Epodoc (pyramid, interger, treillis, multi+)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Craig, J.W.; , "Multidimensional trellis coding for MPSK," Military Communications Conference, 1990. MILCOM '90, Conference Record, A New Era. 1990 IEEE , vol., no., pp.369-374 vol.1, 30 Sep-3 Oct 1990 *Abrégé et fig.1*	1-18
A	WO2006110002A1, Chang et al., 19 October 2006 (19-10-2006) *document complet et fig. 5*	1-18

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
04 March 2013 (04.03.2013)	**06 March 2013 (06.03.2013)**
Name and mailing address of the ISA/ Facsimile No.	Authorized officer Jean Fortin (819) 997-7985 Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CA2012/001042

Patent Document Cited in Search Report	Publication Date	Patent Family Member(s)	Publication Date
WO2006110002A1	19 October 2006 (19-10-2006)	EP1869642A1	26 December 2007 (26-12-2007)
		EP1869642A4	29 June 2011 (29-06-2011)
		JP2008538435A	23 October 2008 (23-10-2008)
		JP4808771B2	02 November 2011 (02-11-2011)
		KR20060108499A	18 October 2006 (18-10-2006)
		KR100747489B1	08 August 2007 (08-08-2007)
		US2009263029A1	22 October 2009 (22-10-2009)
		US8131094B2	06 March 2012 (06-03-2012)

RAPPORT DE RECHERCHE INTERNATIONALE

Demande internationale n°
PCT/CA2012/001042

A. CLASSEMENT DE L'OBJET DE LA DEMANDE

CIB: ***H03M 7/00*** (2006.01)

Selon la classification internationale des brevets (CIB) ou à la fois selon la classification nationale et la CIB

B. DOMAINES SUR LESQUELS LA RECHERCHE A PORTÉ

Documentation minimale consultée (système de classification suivi des symboles de classement)

H03M 7/00 (2006.01)

Documentation consultée autre que la documentation minimale dans la mesure où ces documents relèvent des domaines sur lesquels a porté la recherche

Base de données électronique consultée au cours de la recherche internationale (nom de la base de données, et si cela est réalisable, termes de recherche utilisés)

Google (pyramide hyperspatiale, hyperspatial pyramid), Ieeexplore (pyramid, integer), Epodoc (pyramid, interger, treillis, multi+)

C. DOCUMENTS CONSIDÉRÉS COMME PERTINENTS

Catégorie*	Documents cités avec, le cas échéant, l'indication des passages pertinents	n° des revendications visées
A	Craig, J.W.; , "Multidimensional trellis coding for MPSK," Military Communications Conference, 1990. MILCOM '90, Conference Record, A New Era. 1990 IEEE , vol., no., pp.369-374 vol.1, 30 Sep-3 Oct 1990 *Abrégé et fig.1*	1-18
A	WO2006110002A1, Chang et al., 19 October 2006 (19-10-2006) *document complet et fig. 5*	1-18

[ ] Voir la suite du cadre C pour la fin de la liste des documents.

[X] Les documents de familles de brevets sont indiqués en annexe.

* Catégories spéciales de documents cités :

"A" document définissant l'état général de la technique, n'étant pas considéré comme particulièrement pertinent

"E" demande ou brevet antérieur, mais publié à la date de dépôt international ou après cette date

"L" document pouvant jeter un doute sur une revendication de priorité ou cité pour déterminer la date de publication d'une autre citation ou pour une raison spéciale (telle qu'indiquée)

"T" document ultérieur publié après la date de dépôt international ou la date de priorité et n'appartenant pas à l'état de la technique pertinent, mais cité pour permettre de comprendre le principe ou la théorie constituant la base de l'invention

"X" document particulièrement pertinent; l'invention revendiquée ne peut être considérée comme nouvelle ou comme impliquant une activité inventive par rapport au document considéré isolément

"Y" document particulièrement pertinent; l'invention revendiquée ne peut être considérée comme impliquant une activité inventive lorsque le document est associé à un ou plusieurs autres documents de même nature, cette combinaison étant évidente pour une personne du métier

Date à laquelle la recherche a été effectivement achevée	Date d'expédition du rapport de recherche
04 mars 2013 (04-03-2013)	06 mars 2013 (06-03-2013)
Nom et adresse postale de l'administration chargée de la recherche internationale/CA Office de la propriété intellectuelle du Canada Place du Portage I, C114 - 1er étage, Boîte PCT 50, rue Victoria	Fonctionnaire autorisé Jean Fortin (819) 997-7985

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/CA2012/001004

Patent Document Cited in Search Report	Publication Date	Patent Family Member(s)	Publication Date
WO2006110002A1	19 October 2006 (19-10-2006)	EP1869642A1	26 December 2007 (26-12-2007)
		EP1869642A4	29 June 2011 (29-06-2011)
		JP2008538435A	23 October 2008 (23-10-2008)
		JP4808771B2	02 November 2011 (02-11-2011)
		KR20060108499A	18 October 2006 (18-10-2006)
		KR100747489B1	08 August 2007 (08-08-2007)
		US2009263029A1	22 October 2009 (22-10-2009)
		US8131094B2	06 March 2012 (06-03-2012)