LANGAGE CONVERSATIONNEL POUR LE TRAITEMENT DES MULTI-SIGNAUX

J. LIENARD

CEPHAG-ENSIEG
Domaine Universitaire bp 46 - 38402 St-Martin d'Hères (France)

Résumé

Nous décrivons un système informatique destiné au traitement du signal : à l'expérimentation d'algorithmes ou à l'enseignement. Ce système est "non procédurial" : le traitement est décrit par un graphe de dépendance des sorties à partir des entrées

Il est interactif. Le graphe de traitement (qui remplace un programme classique) peut être étendu ou modifié après son évaluation : seuls les calculs nécessaires sont exécutés.

Il est multidimentionnel. Les entités manipulées sont des signaux appartenant à un espace possédant un nombre quelconque de dimensions. Des parties du graphe peuvent être répétées automatiquement le long de certaines d'entre elles.

Enfin, pour faciliter l'interactivité du système, nous avons choisi de représenter le graphe du traitement par un véritable diagramme dessiné sur un écran de dialogue

Summary

We describe a computer system designed for Signal Analysis : either algorithm tests or teaching. This system is not "procedural" : the analysis to perform is described by a dependance graph of outputs from inputs.

It is interactive. The analysis graph (replacing a classical program) can be enlarged or changed after having been evaluated : only necessary calculations are performed.

It is multidimensional : the objets considered are signals belonging to a space of any finite dimension. Parts of the graph can be automatically repeated along chosen dimensions.

Finally, to facilitate the interactivity of the system, we have chosen to represent the analysis graph by a real diagram drawn on a dialog screen.epeated along chosen dimensions.

Introduction

Nous proposons un système destiné au traitement interactif des signaux mono ou multi-dimensionnels.

L'interactivité nécessite que l'utilisateur puisse aussi bien effectuer des traitements élémentaires pour de simples contrôles que des suites d'ordres ("programmes"). Ces programmes doivent pouvoir être modifiés simplement et rapidement. Ceci exclut l'utilisation des langages de programmation comme Fortran ou Pascal sous leur forme habituelle compilée. Des langages traditionnement interprétés comme Basic, Forth ou Lisp sont plus adaptés mais ils sont généralement trop lents pour les besoins du traitement du signal. Une solution est de n'utiliser ces langages que pour appeler des sous-programmes déja compilés.

Une autre exigence d'un bon système interactif est d'être à la fois simple et sûr. Pour cela, il faut pouvoir manipuler des objets qui ne soient pas de simples tableaux de valeurs mais des structures, afin de diminuer le nombre des paramètres des sous-programmes et de permettre des vérifications de cohérence. Nous utilisons un tel système, avec un BASIC, depuis plusieurs années au CEPHAG [1]. L'expérience montre que si le passage du mode "programmé" au mode "exécution immédiate" répond assez bien aux besoins d'interactivité, il n'est pas totalement sûr, car les objets manipulés sont toujours des zones mémoires dont le contenu est fonction de la suite des opérations effectuées : exécutions de programmes et actions manuelles.

Il est souhaitable de disposer d'un langage véritablement spécialisé en traitement du signal. Une approche possible est de décrire le traitement par son diagramme de flux. Le langage SIGNAL [2] fait ainsi mais sa forme n'est pas adaptée au traitement interactif. Malheureusement, le diagramme de flux est restreint aux traitement de type filtrage, rythmés par une horloge d'échantillonnage unique et ne permet pas, par exemple, de décrire une analyse spectrale par tranformée de Fourier

Une réalisation très intéressante est proposée par Kopec [3] : il ne considère pas les signaux comme des zônes mémoire mais comme des entités abstraites, qui peuvent être multi-dimensionelles, liées entre elles par des fonctions. Ce langage est contruit à partir du langage LISP. Cela assure un fonctionnement interactif, mais impose à l'utilisateur l'apprentissage d'une syntaxe un peu rebutante.

Dans le système que nous proposons ici, nous avons également supprimé la notion de variable, au sens d'emplacement mémoire ou d'accumulateurs. Le traitement est décrit par un graphe de dépendance des sorties à partir des entrées. Ce graphe est beaucoup plus général qu'un diagramme de flux. Le graphe peut être étendu ou modifié après calcul : seuls les calculs nécessaires serons refaits, ce qui assure une bonne interactivité, avec sécurité puisque le graphe reflète toujours la dépendance entrées-sorties. Nous avons introduit une possibilité de multiplication des opérateurs qui correspond aux boucles des langages procéduriaux. Il est ainsi facile de traiter des signaux multi-dimensionnels.

Enfin, pour donner au système une approche agréable, nous avons choisi, tout comme dans celui proposé par Bentz [4], d'introduire le graphe du traitement par un véritable dessin. Bien que ce choix ne soit pas indispensable et que l'on puisse définir une version "littéraire" du langage, nous commencerons par exposer ce mode de présentation afin de faciliter la présentation. Nous montrerons ensuite comment utiliser des signaux multi-dimensionnels, puis nous décrirons succinctement le fonctionnement interne.

PRESENTATION GRAPHIQUE DU TRAITEMENT

Le traitement est décrit par un graphe définissant les relations entre les entrées et les sorties. Ce graphe est formé d'opérateurs interconnectés. Ces opérateurs peuvent être des opérateurs arithmétiques élémentaires ou des opérateurs globaux (comme la tranformée de Fourier). Ils sont représentés par des "boites" possédant des bornes d'entrée-sortie et caractérisées par un symbole ou un nom . A la différence d'un diagramme de flux, ce diagramme décrit des relations entre les signaux considérés comme entités globales et non pas entre les éléments de ces signaux. Ce diagramme est "spatial" [4]. Il ne comporte pas de boucles, sinon il ne serait pas calculable (c'est un graphe de dépendance). Les dimensions des signaux en entrée et en sortie des opérateurs ne sont pas obligatoirement égales (exemple : l'opérateur "moyenne d'un signal" avec une entrée vectorielle et une sortie scalaire).

Ces opérateurs sont disponibles dans une fenêtre "bilbliothèque" et déplacés à l'aide d'un moyen de désignation graphique ("souris",crayon...) dans la fenêtre de travail. Le choix de la position sur l'écran est faite manuellement, laissant à l'utilisateur le choix de la présentation. Le graphe se construit en reliant les bornes entre elles. Nous ne décrivons pas ici le fonctionnement de l'éditeur graphique ; plusieurs solutions sont possibles en fonction du matériel.

Un traitement quelque peu compliqué correspond à un graphe assez encombrant. Pour le condenser, nous proposons de pouvoir remplacer un sous-ensemble du graphe par un seul petit opérateur. Cela est équivalent à la notion de "macro" utilisé dans beaucoup de langages. Il est toujours possible de visualiser le contenu de ces macro-opérateurs dans une nouvelle fenêtre (comme si on utilisait une loupe pour en inspecter l'intérieur) et ce contenu peut être modifié. Le processus est itératif : une macro peut contenir d'autres macros. On obtient ainsi un fonctionnement analogue à celui de langages interactifs comme LISP où on peut toujours demander d'imprimer le contenu des fonctions. Les macros peuvent être sauvegardées indépendamment et regroupées en bibliothèque d'application. Celle-ci peut facilement être hiérarchisée à l'aide de macros imbriquées. On peut aussi utiliser le même mécanisme pour la documentation en ligne des opérateurs, les pages de textes étant également condensées en pseudo-macros. La figure 1 illustre ce fonctionnement.

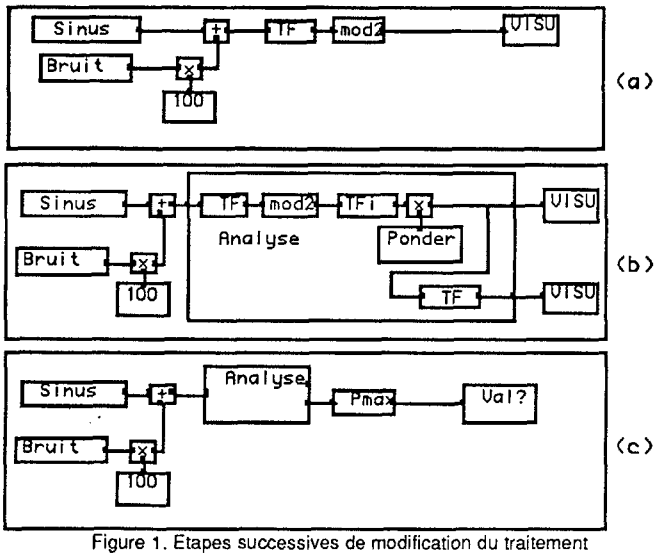

Figure 1. Etapes successives de modification du traitement

TRAITEMENT DE MULTI-SIGNAUX

Nous voulons pouvoir traiter des signaux multi-dimensionnels. De tels signaux peuvent être des images (fonction de deux variables spatiales), des ensembles de signaux temporels comme les sorties d'une antenne réseau (fonction du temps et d'une ou deux variables spatiales). Chaque "fils" du graphe de description du traitement propage donc un signal $S(u_1,u_2,.....u_N)$ où N est le nombre de dimensions (ou de variables) du signal (que nous appelerons l'ordre du signal), et est compris entre 0 (pour une constante) à quelques unités. Nous n'imposons pas de limites autres que celles des moyens de traitement. L'ordre du signal de sortie peut changer entre l'entrée et la sortie d'un opérateur. La nature des dimensions peut également changer (exemple : la transformée de Fourier fait passer du temps à la fréquence). L'étendue du signal suivant chaque dimension sera considérée comme constante pour tout le traitement

Il n'est pas possible d'avoir une bibliothèque d'opérateurs adaptée à chaque combinaison de nombre de dimensions. Nous utilisons des règles qui permettent de sélectionner automatiquement le bon opérateur s'il existe ou d'utiliser un opérateur adapté à un ordre plus petit.

1-Certains opérateurs peuvent être prévus pour s'adapter à l'ordre. C'est le signal d'entrée qui détermine le fonctionnement.

2-Si l'ordre du signal d'entrée est supérieur à celui de l'opérateur, il y a répétition automatique de l'opérateur suivant les dimensions excédentaires. Ce que nous appellerons "bouclage implicite". A partir de deux dimensions, il est nécessaire de préciser sur quelle(s) dimension(s) travaille l'opérateur.

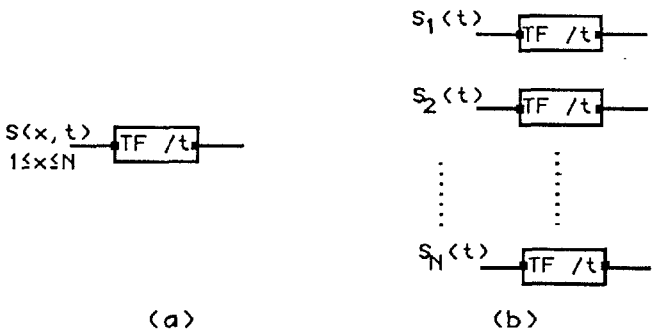

Figure 2. "Bouclage implicite"

3-Le cas des opérateurs à plus d'une entrée est plus compliqué si les signaux appliqués n'ont pas le même ordre. Le bouclage implicite effectue alors automatiquement un "épandage" le long de la dimension supplémentaire

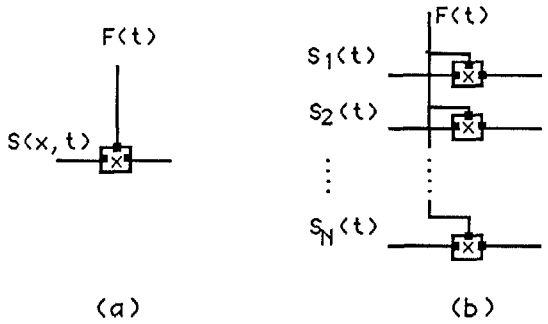

Figure 3. Epandage automatique

4-Il existe deux cas où le bouclage explicite ne suffit pas : quand l'opérateur n'a pas d'entrée et surtout quand les opérations ne sont pas indépendantes suivant les dimensions excédentaires. Dans ce cas on utilise un bouclage explicite avec une notation qui permet d'indiquer suivant quelles dimensions il est réalisé et quelles sont les entrées sorties concernées. Pour bien comprendre la notation, il faut voir que le bouclage revient à remplacer une chaine d'opérateurs identiques interconnectés (disposés en ligne ou en colonne) par un seul module. Le signal d'entrée du module dessiné est celui du premier module de la chaine (il correspond à une "condition initiale" ; sa sortie est la sortie du dernier de la chaine (figure 4). Le module unique est dessiné avec un bord épaissi dans la direction de la chaine (comme si les modules de la chaine, dessinés sur des cartes, étaient rassemblés en un seul paquet.

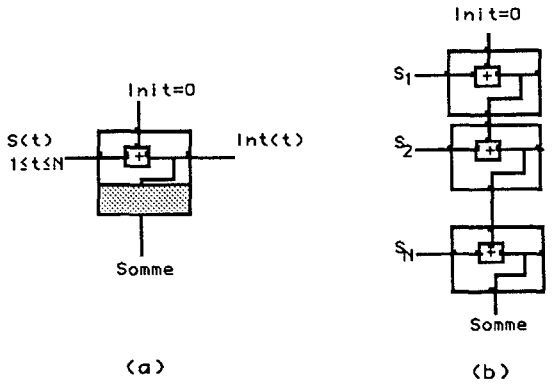

Figure 4. "Bouclage" explicite.

Ces mécanismes de "bouclages" inplicites ou explicites sont l'équivalent des boucles séquentielles des langages procéduriaux. Il est ainsi très facile d'étudier l'influence d'un paramètre évolutif en le considérant comme une dimension supplémentaire. La figure 5 montre comment le graphe de la figure 1 est complété pour effectuer un moyennage statistique (ajout d'une dimension au bruit) et étudier l'influence du rapport signal sur bruit.

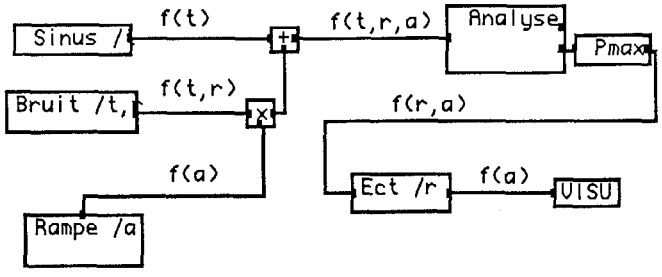

Figure 5. Augmentation du nombre de dimensions

FONCTIONNEMENT INTERNE.

Le graphe du traitement possède des modules de calcul possédant au moins une entrée et une sortie, des modules sources ne possédant que des sorties (générateurs, lecture de fichiers...), et des modules terminaux ne possédant que des entrées (imprimeurs, visualisation de courbes, écriture de fichier...). En phase d'édition, aucun calcul n'est fait. Seules quelques vérifications de cohérences sont effectuées.

Sur demande de calcul, le graphe est parcouru en sens inverse, à partir des modules terminaux activés. Ce parcours détermine les seuls modules sources utiles. Puis l'interprétation du graphe s'effectue à partir de ceux-ci. Elle se fait en deux phases :

a) test des cercles vicieux, vérification des cohérences des caractéristiques des signaux et détermination des bouclages implicites. Pour pouvoir faire cela on associe à chaque borne un descripteur du signal qui mémorise les grandeurs correspondant à chaque dimension, le mode de représentation des valeurs (entiers, réels, complexes...), l'unité, etc.. Les caractéristiques de chaque dimension (nombre de points, origine, échelle) sont mémorisées globalement .

b) calculs. Dans le cas de bouclages multiples, l'ordre des boucles est choisi de façon à minimiser le volume de stockage intermédiaire nécessaire, indépendamment de l'imbrication graphique des ordres de bouclage. Certains résultats intermédiaires sont conservés au cours du calculs, même s'il ne sont pas indispensables, afin de pouvoir les réutiliser ensuite en cas de modification interactive du traitement. La stratégie utilisée est basée sur le "coût" des calculs.

Après calcul, toute modification du graphe (insertion ou suppression d'un opérateur, modification d'un paramètre), le graphe est parcouru en sens direct pour annuler les mémorisations situées en aval qui ne correspondent plus au graphe modifié. A la demande de calcul suivante, le parcours inverse du graphe s'arrête aux bornes dont le signal est mémorisé et ce sont ces bornes qui se comportent comme de nouvelles sources. Cette stratégie permet un fonctionnement interactif à la fois rapide, puisque tous les calculs ne sont pas refaits à chaque fois, et sûr, puisque les résultats visualisés dépendent toujours rigoureusement de l'état présent du graphe.

CONCLUSIONS.

Nous pensons que ce langage répond bien aux nécessités du traitement interactif (simplicité, rapidité de manipulation et d'exécution) de signaux multi-dimensionnels. Cette aptitude au traitement multi-dimensionnel peut être mise à profit pour tracer très simplement des courbes d'évolution en fonction de paramètres qui sont considérés comme des dimensions suplémentaires. Dans ces utilisations interactives, pour obtenir une bonne vitesse d'éxécution, ont pourra disposer d'une bibliothèque assez complète d'opérateurs écrits en un autre langage et déja compilé.

La réalisation de l'éditeur graphique du langage nécessite un écran graphique et une intelligence locale. Une station de travail de CAO ou même un simple micro-ordinateur convienne bien. Mais une utilisation intensive en multi-dimentionnel nécessite la connection à une machine plus puissante ou l'utilisation d'une carte spécialisée.

Ce langage pourrait également être utilisé pour décrire un traitement en vue d'une implantation sur un processeur spécialisé et en particulier sur une structure systolique. La description simple du traitement devrait faciliter la réalisation du compilateur. Dans ce dernier cas, le traitement doit entierement pouvoir se décrire de façon homogène à l'aide de quelques opérateurs de base.

Références

[1] J. Liénard, " Notice d'utilisation du Système Fourier", Rapport Interne CEPHAG

[2] P.Le Guernic, A.Benvéniste, "SIGNAL, un langage pour le traitement du signal", Traitement du Signal, oct 1984

[3] G.Kopec, "The Signal Representation Langage", IEEE Trans ASSP, August 1985

[4] B.Bentz, "An automatic programming system for signal processing applications", Patern Recognition N°6, pp 491, 1985

[5] J. Liénard, "Contribution à l'étude des moyens de mise en oeuvre des traitements numériques du signal", Thèse d'état, Grenoble, 1983