Mesures probabilistes de l'adéquation algorithme architecture

Algorithm-Architecture Matching Metrics

par Jean-Philippe DIGUET*, Olivier SENTIEYS†, Jean-Luc PHILIPPE* et Eric MARTIN*

* LESTER - Université de Bretagne Sud,
10 Rue Le Coat Saint Haouen, 56100 Lorient, France
† LASTI - ENSSAT - Université de Rennes,
6 rue de Kérampont, 22300 Lannion, France
eMail : diguet@iuplo.univ-UBS.fr

résumé et mots clés

Le champ d'action de la synthèse d'architecture s'avère trop vaste pour qu'un outil puisse offrir une solution optimale quelque soit l'algorithme cible. C'est pourquoi l'étude préalable de l'algorithme spécifié apparaît comme incontournable. Nous présentons ici, une nouvelle approche d'estimation dynamique des ressources, appliquée aux architectures pipelines sous contrainte de Latence. Nous employons une méthode probabiliste prenant en compte réellement les contraintes entre opérations, dans le but de guider le choix des transformations et des algorithmes impliqués dans la spécification. Les propriétés analysées sont la concurrence dans le temps des opérateurs, bus, registres et interconnexions et les statistiques de liens entre opérateurs. Des métriques sont également proposées pour l'interprétation des courbes d'estimation obtenues.

Synthèse de haut niveau, estimation de complexité, guidage, métriques

abstract and key words

The high level synthesis question is too wide to be optimaly addressed by a single and general CAD tool. So, interactive transfers of information are required between the tool and the designer, in order to make tractable the optimization of the synthesis task in a reasonnable time. This paper introduces an appoach which aims to provide the designer with information to quantify the hardware complexity in order to guide him in during his transformation choices. The method is based on probabilities, focuse the whole set of ressources and takes into account the real dependencies between operations. The method is characterized by a high level of abstraction. It firstly enables to combine the estimation with the most powerful algorithmic-transformations and secondly to be easily independent from the architectural model.

High level synthesis, complexity estimation, guidance, metrics

1. introduction

Un consensus semble s'installer au sein de la communauté de la synthèse architecturale, à propos du rôle et de la définition des outils à développer. Au lieu de créer des systèmes totalement automatiques, pouvant être utiles pour des modèles d'architectures spécifiques, il apparaît beaucoup plus judicieux de fournir au concepteur un outil interactif qui le guide dans ses choix [1] et lui permette de laisser l'outil réaliser les optimisations dans la bonne direction. Plusieurs techniques, comme les transformations structurelles [2] ou la sélection des composants [3,4] sont potentiellement efficaces pour optimiser une architecture dédiée. Cependant l'espace de *design* aux niveaux algorithmique, fonctionnel et structurel, est tellement vaste que l'outil, et encore moins le concepteur ne peuvent explorer l'ensemble des possibilités existantes. Le pouvoir réel d'optimisation des transformations est donc sous-utilisé. C'est pourquoi, il est nécessaire d'effectuer avant synthèse une analyse de l'application à traiter, de manière à avoir une vue d'ensemble de ses caractéristiques et propriétés. Les informations recueillies par cette étude préliminaire doivent être suffisamment précises pour permettre de réduire l'espace de recherche et d'identifier les étapes critiques de l'algorithme au sens des ressources à allouer pour respecter la contrainte de temps d'exécution. Le but

est de mettre en lumière la nature des techniques à utiliser pour améliorer l'adéquation algorithme-architecture. Un « module estimation » a été développé dans cet esprit. Il s'intègre dans le cycle de développement de GAUT [5], sous la forme d'une étude probabiliste de la répartition temporelle des ressources matérielles dont la régularité traduit l'adéquation algorithme architecture (AAA). Le module est donc à la fois un estimateur et un outil de guidage pour la synthèse d'architecture. Après avoir commenté les travaux effectués dans le domaine de l'estimation et de la caractérisation, nous développerons le principe de l'estimation probabiliste. Nous détaillerons plus particulièrement le cas des opérateurs à mettre en œuvre pour exécuter les opérations décrites dans la spécification. Notez qu'à chaque opérateur est associé un temps d'exécution, donc un nombre de périodes d'horloge (unité de temps [5]). En raison de la taille limité du document, l'estimation des bus, registres et interconnexions sera juste résumée, une description complète est fournie dans [9]. Cette partie s'achèvera, par un exemple significatif montrant l'intérêt de la méthode. Enfin, nous terminerons par un exposé des métriques statistiques que nous calculons pour quantifier la régularité des algorithmes testés. Des mesures effectuées à partir de divers types de FFT et de l'estimation de complexité de l'exemple précédent, accompagneront ces définitions.

2. état de l'art

Les travaux de Jain et Sharma [6] traitent de l'estimation des ressources et des performances en synthèse d'architecture. La méthode qu'ils proposent, estime le nombre de ressources minimum en évaluant dans chaque unité de temps, la limite inférieure du nombre d'opérations impliquées. Cependant, les relations de précédence ne sont prises en compte que partiellement à travers les intervalles ALAP (date au plus tôt - ALAP (date au plus tard) (voir fig. 1). De plus, la complexité de la méthode devient rapidement prohibitive : $O(N^2C)$ ou $O(NC^2)$ où N est le nombre de nœuds du graphe, i.e. le nombre d'opérations du graphe flot de données (GFD) et C le nombre d'unités de temps imposé ou prévu pour l'exécution de l'algorithme. Enfin, l'évolution dans le temps des régions critiques n'est pas évoquée. Potkonjak et Rabaey ont également étudié la question. Ils présentent dans [7], un modèle, qui traite l'estimation des ressources et performances de manière duale. Elle conduit à une complexité en $Nlog(N)$. La méthode est plus rapide mais moins précise que la précédente, de plus elle n'aborde pas le problème de la répartition temporelle des ressources. Enfin, comme précédemment, les dépendances sont résolues par le biais des intervalles ASAP-ALAP, donc sous-estimées. Une nouvelle approche est proposée dans [8], les auteurs se placent dans l'optique de guider l'utilisateur en cherchant à extraire les propriétés de l'algorithme avant synthèse. Ces derniers sont dans l'esprit de ce que nous décrivons par la suite, à la différence que nous traitons les propriétés de concurrence et de durée de vie des variables différemment, en incluant l'aspect temporel et en prenant réellement en compte les relations de précédence, le problème de régularité est lui aussi traité dynamiquement et s'accompagne de métriques statistiques.

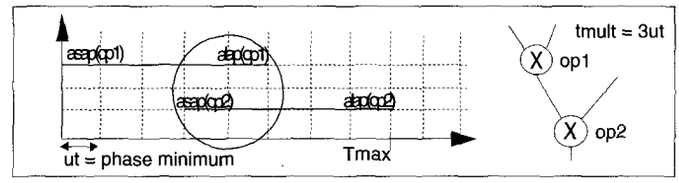


Figure 1. – Faux Parallélisme.

3. estimation probabiliste

3.1. introduction

Le module estimation, inséré dans l'outil GAUT se situe après les deux premières étapes que sont : la compilation générant le GFD et le module sélection qui optimise le choix des opérateurs et associe par la même, des temps de traversée aux opérations du GFD. Son rôle est d'effectuer une analyse des besoins de l'application à synthétiser suivant un modèle d'architecture décrit dans [5]. Celle-ci est réalisée pour chaque unité de temps située entre la date 0 et Tmax correspondant à la contrainte de temps, l'unité de temps étant définie par le module sélection. Elle fournit à l'utilisateur une estimation du nombre d'opérateurs, de registres, de bus et de connexions entre opérateurs (y compris la mémoire). L'estimation tient compte des précédences par un calcul de probabilités conditionnelles, elle fournit également une étude de faisabilité de synthèse des boucles et des statistiques sur les liaisons entre opérateurs. Les calculs sont détaillés dans [9].

3.2. estimation datée du nombre probable d'opérateurs

3.2.1. principe

Le nombre probable d'opérateurs du type n ordonnancés, à l'instant t s'obtient de la façon suivante :

$$\tilde{N}_n(t) = \sum_{i=0}^{\tilde{N}_{max}} i.P_n(\tilde{N}_n = i, t) \qquad (1)$$

où $P_n(\tilde{N}_n = i, t)$ est la probabilité que i opérations du type n soient ordonnancées à l'instant t, et $\tilde{N}_{max}$ la borne maximale du nombre d'opérateurs du type n à l'instant t. La complexité de (1)

la rend, en pratique, inexploitable. Par contre, une fois développée et simplifiée [9], elle se résume à l'expression suivante :

$$\tilde{N}_n(t) = \sum_{i \in Eop_n} P_{O_i}(t) \tag{2}$$

où Eop_n est l'ensemble des opérations réalisant une opération du type n et $P_{O_i}(t)$ la probabilité que l'opération i soit ordonnancée à l'instant t. Finalement, le nombre probable d'opérateurs exécutant une opération du type n à l'instant t est obtenu en prenant en compte les délais de calcul : ($\Delta_{op}(O_i)$: temps d'exécution de l'opérateur O_i) :

$$N_n(t) = \sum_{i \in Eop_n} \sum_{k=t-\Delta_{op}(0_i)+1}^{t} P_{O_i}(k) \tag{3}$$

3.2.2. *probabilités associées*

• **probabilités non conditionnées** : on considère dans ce cas la probabilité de présence d'une opération O_n comme indépendante des ses prédécesseurs. $P_{O_n}(t)$ ne dépend que des dates ASAP et ALAP, soit :

$$P_{O_n}(t) = \frac{1}{dp(O_n)} = \frac{1}{alap_{O_n} - asap_{O_n} + 1} \tag{4}$$

Ce type de probabilité prend en compte les dépendances de données à travers le simple calcul des dates ASAP et ALAP. Il s'agit d'une densité utilisée en synthèse d'architectures pour mesurer la mobilité d'une opération et ainsi effectuer des choix d'ordonnancement [10] ou de transformations [7]. Cependant celle-ci est incomplète puisqu'elle exploite un parallélisme potentiel qui n'a pas lieu d'être. Dans leurs intervalles respectifs de mobilité maximale [asap; alap] les prédécesseurs et successeurs sont considérés comme indépendants.

• **probabilités conditionnées** : La probabilité d'ordonnancement d'une opération O_n, dépend cette fois des prédécesseurs et des successeurs. $P_{O_n}(t)$ est calculée en fonction de t, de sa densité de probabilité et de celle de chacun des N_{pred} prédécesseurs et des N_{suc} successeurs. Il n'est pas possible d'évaluer simplement $P_{O_n}(t+1)$ en fonction de $P_{O_n}(t)$. En effet, de manière symétrique, les probabilités conditionnées par les prédécesseurs peuvent être exprimées en fonction des dates passées alors que les probabilités conditionnées par les successeurs sont, elles, dépendantes des dates futures. Il y a blocage, le calcul récursif n'est pas permis, quel que soit de sens du parcours. Pour exprimer $P_{O_n}(t)$ en fonction de la probabilité de ses prédécesseurs et successeurs, il faut pouvoir disposer des valeurs $\{P_{O_n}(asap), ..., P_{O_n}(t-1), P_{O_n}(t+1), ..., P_{O_n}(alap)\}$ ce qui n'est envisageable que par approximations successives.

Le calcul exact, après convergence doit aboutir aux égalités suivantes :

$$P_{O_n}(t) = \prod_{i=1}^{N_{Pred}} \left(\sum_{t_1=asap(p_i)}^{t-\Delta(p_i)} P((p_i,t_1)|(O_n,t)).P_{O_n}(t) \right) \cdot \prod_{j=1}^{N_{Suc}} \left(\sum_{t_2=t+\Delta(O_n)}^{alap(s_j)} P((s_j,t_2)|(O_n,t)).P_{O_n}(t) \right) \tag{5}$$

Où $P((A,t_1)|(B,t_2)).P_B(t_2)$ est la probabilité conditionnelle que A soit ordonnancée à t_1 sachant que B est ordonnancée à t_2.

Ce calcul devant être effectué pour tous les ordonnancements possibles de l'ensemble des opérations du GFD, sa complexité interdit de l'utiliser si l'on souhaite conserver le caractère rapide de l'estimation.

Il est cependant possible d'effectuer ce calcul de manière beaucoup plus rapide, en acceptant une approximation. Le problème est alors vu de la manière suivante : nous considérons que la probabilité $P_{O_n}(t_k)$ est une simple proportion du nombre de possibilités d'ordonnancement offerte à O_n en $t = t_k$ par rapport à la totalité des possibilités existant sur l'intervalle $[asap(O_n);\ alap(O_n)]$. Le calcul est donc effectué en associant une probabilité uniforme aux prédécesseurs et aux successeurs. Les probabilités exactes pourront être obtenues en répétant la méthode à partir des probabilités calculées à l'itération précédente. Pour des raisons de temps de calcul, que l'on souhaite courts, nous nous arrêtons au premier ordre.

Soit :

$$P_{O_n}(t) = \frac{1}{N_0} \cdot \prod_{i=1}^{N_{Pred}} max[1;\ t - (asap(p_i + \Delta(p_i)) + 1] \cdot \prod_{j=1}^{N_{Suc}} max[1;\ asap(s_j) - (t + \Delta(O_n) + 1)] \tag{6}$$

avec :

$$No = \sum_{t=asap(O_n)}^{alap(O_n)} \prod_{i=1}^{N_{Pred}} max[1;\ t - (alap(p_i + \Delta(p_i)) + 1] \cdot \prod_{j=1}^{N_{Suc}} max[1;\ alap(s_j) - (t + \Delta(O_n) + 1)] \tag{7}$$

La complexité de calcul C se trouve extrêmement réduite, d'autant plus que No et P_{O_n} peuvent être calculés simultanément, soit avec N le nombre d'opérations du GFD :

$$C < \mathrm{O}(N).max_n\{(alap(n) - asap(n))/\Delta_t\} .max_i\{N_{Pred}(i)\}.max_j\{N_{Suc}(j)\}$$

La méthode employée nous permet d'avoir une estimation à complexité arithmétique réduite. Elle offre également une exploration dans le temps de la complexité matérielle conditionnée par les relations de précédence du GFD.

3.3. estimation des bus, registres et interconnexions

Nous calculons en premier lieu, les probabilités de transfert pour chacune des variables de l'algorithme. Dans le GFD, une liaison entre deux nœuds traduit l'existence d'une variable communiquant le résultat d'une opération à une autre opération. La durée de vie des variables dépend donc à la fois de l'ordonnancement de l'opérateur producteur et de l'opérateur consommateur. Nous délimitons ensuite les intervalles de temps pendant lesquels les variables occuperont un bus, un registre, ou une connexion entre opérateurs. Le calcul des nombres probables s'effectue enfin, de la même manière que pour les opérateurs, par une somme des probabilités d'existence à l'instant t considéré.

Pour illustrer le propos, prenons l'exemple de la probabilité d'occupation d'un registre P_R lors du transfert d'une donnée produite par un opérateur O_0 à la date t_0 et consommée par un opérateur O_i à la date t_i. Les dates t_0 et t_i sont telles que la variable reste en registre dans l'unité de traitement.

$$P_R(t,i) = \begin{cases} P\{(O_i,t_i);\ (O_0,t_0)\} & \text{si } t \in [t_0+\Delta Op0;\ t_i+\Delta Opi] \\ 0 & \text{sinon} \end{cases} \quad (8)$$

$P\{(O_i,t_i);\ (O_0,t_0)\}$ est la probabilité que O_i débute à la date t_i et O_O à la date t_0, les deux événements n'étant pas indépendants, on obtient la formule suivante :

$$\begin{aligned} P\{(O_i,t_i);\ (O_0,t_0)\} &= P((O_i,t_i) \mid (O_0,t_0)).P_{O_0}(t_0) \\ &= P((O_0,t_0) \mid (O_i,t_i)).P_{O_i}(t_i) \\ &= \frac{p_{O_0}(t_0)}{\sum_{t_i-\Delta Op0}^{asap(O_0)} P_{O_0}(t)} . P_{O_i}(t_i) \end{aligned} \quad (9)$$

L'équation 9 traduit simplement le fait que tous les ordonnancements de O_0 ne sont pas permis pour un ordonnancement donné de O_i. En pratique, la complexité n'est pas augmentée par la dépendance des probabilités. Le terme $P\{(O_0,t_0) \mid (O_i,t_i)\}$ est calculé à partir des sommes cumulées : $S_0(t') = \sum_{t=asap(O_0)}^{t'} p_{O_0}(t)$ constituées au fur et à mesure que $p_{O_0}(t)$ est connue. Il n'y a donc pas de nouvelles boucles imbriquées, donc pas d'accroissement supplémentaire de la complexité de la méthode d'estimation.

$P_{O_i}(t)$ et $P_{O_0}(t)$ sont calculées de la même manière que précédemment pour les opérateurs, le calcul n'est d'ailleurs effectué qu'une seule fois. Elles peuvent donc être conditionnelles ou non, selon la prise en compte souhaitée des dépendances de données.

L'estimation des ressources de transfert nécessite de parcourir les intervalles de temps d'un nœud du GFD et de ses successeurs, la complexité est donc plus élevée que dans le cas des opérateurs. Cela peut s'avérer critique dans les cas particuliers d'intervalles de temps très allongés. En pratique, une interpolation est effectuée, celle-ci est suffisante lorsque de pareilles situations sont rencontrées (voir [9]).

3.4. Statistiques des connexions

Durant l'exploration du GFD, nous calculons les statistiques correspondant à la fréquence des liens entre les différents types d'opérateurs. Celles-ci sont calculées dans le but de fournir au concepteur une métrique lui permettant de juger de la régularité des liaisons et de décider, par exemple, de la fusion de deux opérateurs (classiquement un multiplieur-additionneur). Elles reposent sur l'analyse de la durée de vie des variables. Si cette dernière est supérieure à une valeur seuil, la variable peut être stockée en mémoire avant d'être lue, sinon elle reste en registre. Le seuil est dit minimal lorsqu'il conduit à un nombre minimal de registres. Un lien faible (eq. 10) désigne une connexion s'effectuant à travers la mémoire, un lien fort (calcul semblable avec la condition : $t_j - t_i < seuil$) traduit une liaison à travers un registre. Dans le premier cas, le temps disponible est supérieur au seuil permettant d'écrire puis de lire la donnée en mémoire, dans le second cas c'est impossible, la donnée reste en registre.

$$lien\,faible_{type[i]->type[j]} = \frac{\sum_{connexions}^{type[i]->type[j]} \frac{\sum_{\{t_i,t_j\}} (t_i,t_j) | t_j - t_i \geq seuil}{Card\{(t_i,t_j)\}}}{Card\{successeurs(Op-type_{[i]})\}} \quad (10)$$

4. résultats d'estimations

4.1. calcul des moyennes

Différents algorithmes ont été testés, dans plusieurs configurations de contraintes de temps et de spécifications *(Elliptic5; IIR7; FFTs 1024pts; FIR16; LMS 1024pts; GAL 128pts, 30 cellules)*. Nous avons calculé les moyennes (partie entière supérieure) des nombres probables d'opérateurs sur l'intervalle [1...Latence]. On observe que les moyennes obtenues sont identiques avec et sans condition sur les probabilités, ce qui est normal puisque l'aire (Nb probable x temps) est invariable, seule la courbe subit une déformation (voir fig. 6 a1-a2). Les résultats obtenus sont proches de ceux fournis par la synthèse, le taux de corrélation étant de 0,95. Ceci justifie l'utilisation, dans GAUT d'une moyenne grossière pour initialiser la synthèse. Par contre, si l'on examine dans le détail les résultats, on note que la moyenne sous-estime la

complexité dès lors que l'on teste un algorithme contenant des régions critiques associées à une contrainte de temps sévère. Ces régions critiques, repérées par des pics sur les courbes d'estimation, traduisent un besoin en ressources, supérieur à la moyenne, très localisé dans le temps. De telles observations ont été faites notamment avec le filtre Elliptic5, la FFT géométrique et un filtre IIR du $7^{ème}$ ordre. C'est la raison pour laquelle, l'étude dynamique sur l'intervalle [1...Latence] est nécessaire, en effet, par l'analyse de la répartition temporelle des ressources, elle permet de mesurer l'AAA. L'exemple suivant en est l'illustration.

4.2. exemple d'un filtre IIR7

La figure 2 représente l'évolution de la spécification du filtre, la figure 6 fournit une partie des résultats, de la première spécification (figures a1...d1) et de la deuxième (figures a2...d2). La moyenne des multiplieurs nécessaire à la première spécification est de trois, alors que la synthèse en requiert quatre.

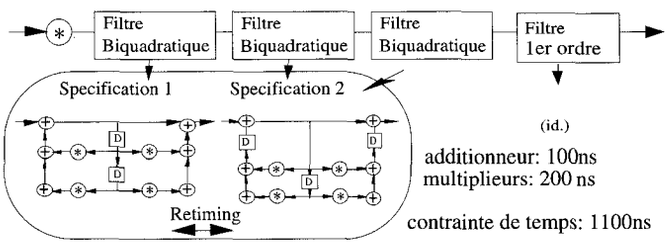


Figure 2. – Spécifications du IIR7.

Observons les courbes des additionneurs : (a1) et des multiplieurs : (b1) où les pointillés correspondent aux probabilités non conditionnelles. On remarque que les multiplications sont contraintes d'être exécutées avant la date 600ns et les additions plutôt après. De plus, on note que le degré de liberté, des nœuds relatifs aux multiplications, est très faible puisque que les courbes avec et sans probabilités conditionnelles sont quasiment jointes.

Nous appliquons donc une transformation de *retiming* (spécification 2) pour améliorer la distribution des nœuds dans l'intervalle de temps imparti. Comme le montre les figures (a2) et (b2), la transformation est efficace et elle conduit cette fois à une synthèse nécessitant trois multiplieurs ce qui est conforme à la moyenne obtenue.

Les figures (c1) et (c2) permettent d'apercevoir les effets du partage de données (courbes pointillées) sur le nombre de registres nécessaires. On constate, sans partage de données, que les moyennes obtenues respectivement 14 pour (c1) et 12 pour (c2) sont très proches des nombres de registres (respectivement 14 et 13) trouvés par la synthèse ce qui laisse supposer que la phase de fusion des registres s'est déroulée correctement. Enfin, les figures (d1) et (d2) restituent les résultats de l'estimation des liaisons en sortie des additionneurs avec des multiplieurs (lignes pointillées et croix), la mémoire (lignes pointillées et carrés) ou d'autres additionneurs (ligne pleine). Les trois pics visibles sur (c2) sont corrélés avec ceux apparaissant sur (d2). En observant la courbe du partage de données, nous pouvons remarquer qu'une différence de quatre registres est répétée trois fois. Si l'on se rapporte au schéma de la figure 6, on comprend qu'il s'agit des liaisons entre la première addition de chaque filtre d'ordre deux et les quatres multiplications suivantes. La courbe (carrés) des liaisons entre les additionneurs et la mémoire nous indique que le résultat de l'addition provoquant quatre liaisons est renvoyé en mémoire. Nous savons donc qu'il est possible de guider efficacement la synthèse en évitant l'écriture en mémoire des données provenant des additionneurs avant la date 800ns. Le résultat est une diminution du nombre de bus et de registres si le partage de données est imposé.

5. métriques et statistiques de régularité

L'estimation obtenue, nous donne l'évolution de la complexité de l'algorithme en fonction du temps. La distribution des ressources peut varier d'une spécification à l'autre. Pour pouvoir juger de cette régularité de manière quantitative et non plus seulement qualitative, nous calculons différentes métriques à partir des courbes obtenues. Ainsi l'AAA est associée à une mesure globale. Concernant l'estimation d'opérateurs, la régularité de la courbe est corrélée avec le nombre exact d'opérateurs utilisés, une courbe régulière se traduira par l'emploi du nombre moyen d'opérateurs. Dans le cadre des interconnexions, registres et bus, la régularité est directement liée au coût de ces ressources, une courbe irrégulière se concrétisera par des accès simultanés importants à la mémoire ou aux opérateurs nécessitant un nombre de connexions ponctuelles élevé, lesquelles sont sous utilisées par ailleurs.

Soit $N_i(t)$ la courbe analysée, décrivant le nombre probable de ressources i à l'instant t. Il y a alors deux façons d'appréhender la mesure de la régularité : soit considérer $N_i(t)$ comme un processus aléatoire, soit comme la distribution d'un processus aléatoire.

5.1. $N_i(t)$ est un processus aléatoire

Nous pouvons alors calculer en plus de sa moyenne, son écart type celui-ci reflétant la variation de l'estimation autour du nombre moyen. Nous avons également la possibilité de calculer l'entropie, le désordre traduisant l'irrégularité. Le calcul de l'entropie est effectué en normalisant celle-ci par l'entropie d'une courbe idéale correspondant à une équi-répartition des ressources sur l'intervalle de temps. Une entropie égale à 1 correspondra donc à un

nombre différent de ressources dans chaque unité de temps, et une entropie nulle à une constante égale au nombre moyen.

5.2. $N_i(t)$ est la distribution d'un processus aléatoire

Les statistiques d'ordre 3 et 4 permettent de quantifier respectivement la dissymétrie et l'aplatissement d'une distribution, elles sont donc candidates à la caractérisation d'une courbe que l'on souhaite la plus plate possible. Dans un premier temps, nous reproduisons un processus aléatoire $x(n)$ ayant pour distribution $N_i(t)$. Nous calculons ensuite le moment d'ordre 3 ($Skewness$) en imposant comme moyenne de $x(n)$ la moitié du temps de latence : $T/2$, de manière à savoir si les ressources ont plutôt tendance à se concentrer dans la première ($S < 0$) ou la seconde partie du temps ($S > 0$). Le moment d'ordre 4 ($Kurtosis$) de $x(n)$ est analysé en fonction de son écart par rapport à la valeur résultant d'une distribution uniforme : 1.8 (valeur asymptotique obtenue avec un nombre infini d'événements).

La figure 3 résume la signification de ces métriques.

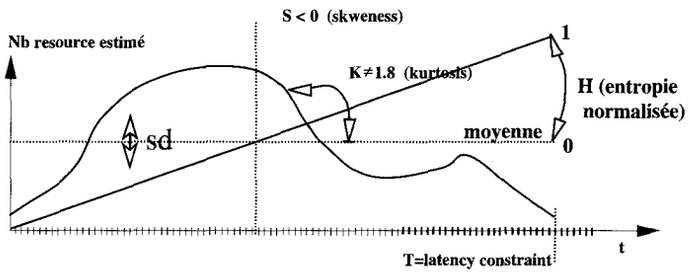


Figure 3. – Caractérisations statistiques des courbes d'estimation.

5.3. quantification des spécifications du Filtre IIR7

Les quatre métriques proposées ont été appliquées aux cas des multiplieurs pour les deux spécifications du filtre IIR7. Les mesures et distributions du nombre probable de multiplieurs de la figure 5 sont issues des courbes b1 et b2 de la figure 6. Les mesures associées à une courbe idéale correspondant à l'équi-répartition du nombre moyen de multiplieurs (3) sont fournies en comparaison. Les conclusions tirées au 4.2 se confirment au regard des résultats obtenus. La spécification 2 possède une entropie et un écart type plus faibles, ce qui traduit une dispersion moins importante. Son moment d'ordre 4 est plus proche de la valeur idéale, ce qui reflète un meilleur aplanissement. Enfin le moment d'ordre 3 de la deuxième spécification est faiblement positif contrairement à celui de la première plus fortement négatif, ce qui vérifie la tendance de la disposition temporelle des opérations, avant $T/2$ pour la première et après pour la seconde.

5.4. exemple des FFTs

La figure 4 expose les résultats obtenus avec quatre types différents de FFT 16pts sous une contrainte de 960ns (soit 64 FFT 16pts à 16kHz correspondant à une application de filtrage en sous-bandes). Sur la gauche sont fournies les courbes d'estimation probabiliste appliquée aux multiplieurs (c) et à l'ensemble des connexions (a). Sur la droite se trouvent les distributions associées aux courbes d'estimation (b pour a et d pour c). Celles-ci servent de support au calcul de l'entropie.

Si on analyse le cas de la FFT de type géométrique, on s'aperçoit que sa régularité se reflète très bien dans les métriques issues de l'estimation de ses multiplieurs. En effet, des quatre, elle possède l'entropie la plus faible, un coefficient d'asymétrie très faiblement négatif, un coefficient d'aplatissement très proche de 1.8, enfin l'écart type le moins élevé. On note de plus une distribution centrée sur la moyenne. Les valeurs des métriques de régularité sont résumées dans le tableau 1.

Tableau 1. - Métriques de régularité de différentes FFT

	Nb. Mult.				Nb. Connex.			
	DIF4	GEOM	DIT2	DIF2	DIF4	GEOM	DIT2	DIF2
Moy.	6	8	8	8	15,7	42,7	19,1	18,7
σ	3,9	1,5	4,5	5,6	7,2	13,7	10,8	9,2
$-\gamma_1^{T/2}$	0,09	-0,07	-0,51	-0,38	-0,15	-0,03	0,37	-0,28
γ_2	2,3	1,88	2,4	2,45	2,46	1,5	2,6	2,5
H_N	0,76	0,34	0,86	0,9	0,66	0,746	0,753	0,69

La régularité des opérateurs se paie, par contre, au niveau des interconnexions. La moyenne est beaucoup plus élevée dans le cas de la FFT géométrique. Cela traduit une mobilité des opérations du GFD plus réduite que dans le cas des trois autres algorithmes. En conséquence, l'algorithme nécessite plus de transferts simultanés. Ces restrictions sur la mobilité des nœuds du GFD se retrouvent dans les statistiques de liens. En effet, on note que 42% des connexions de la FFT géométrique doivent être réalisés à travers les registres de l'unité de traitement contre moins de 28% seulement pour les autres types de spécification.

La FFT DIF4 offre le meilleur compromis entre le coût des interconnexions et des multiplieurs. Elle offre les moyennes les plus basses donc le coût théorique minimal le plus faible. De plus, ces moyennes s'accompagnent de métriques de régularité également plus avantageuses que celles des algorithmes DIF2 et DIT2, donc d'une facilité plus grande d'optimisation. Enfin, les statistiques de liens montrent que la spécification possède des interconnexions moins contraintes (lien fort minimum : 24%).

L'estimation de l'évolution du coût global d'un algorithme est obtenue en combinant celles des interconnexions et des opérateurs. Ce type de métrique fournit la première information analysée, avant d'étudier en détail la répartition de chaque type de ressources. Elle est notamment utilisée pour trier divers choix algorithmiques. Le coût des registres est retenu (voir [9]) pour caractériser celui des interconnexions et ainsi obtenir le coût probable. La figure 4-e illustre le coût probable des différentes FFT,

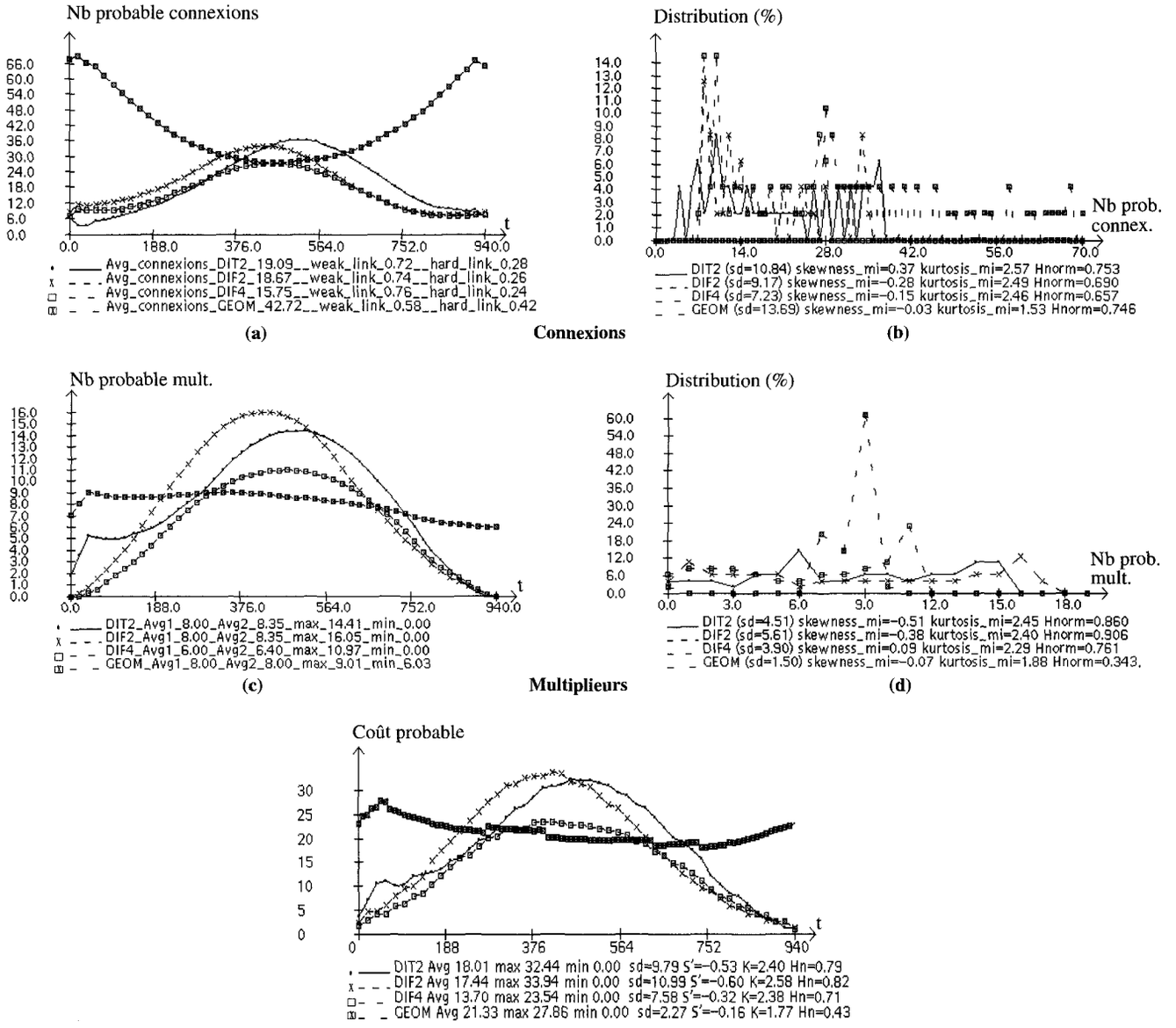


Figure 4. – Caractérisations statistiques de différents types FFT.

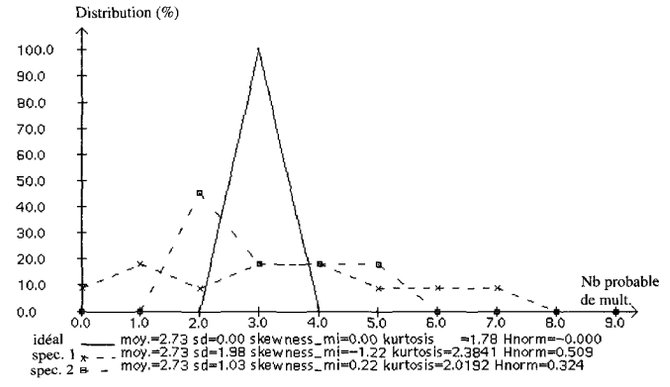


Figure 5. – Caractérisations statistiques du filtre IIR7.

le résultat obtenu confirme ce qui était pressenti, c'est à dire que l'algorithme DIF4 offre le meilleur compromis et finalement la solution à coût minimal.

6. conclusion

Nous avons décrit une nouvelle approche dans l'estimation de l'ensemble des ressources en synthèse architecturale d'unités de traitement. En utilisant les probabilités conditionnelles de placement de chaque nœud du GFD, la méthode permet une caractérisation rapide de la répartition du coût, donc de l'AAA. Celle-ci apporte une précision temporelle permettant de guider le concepteur

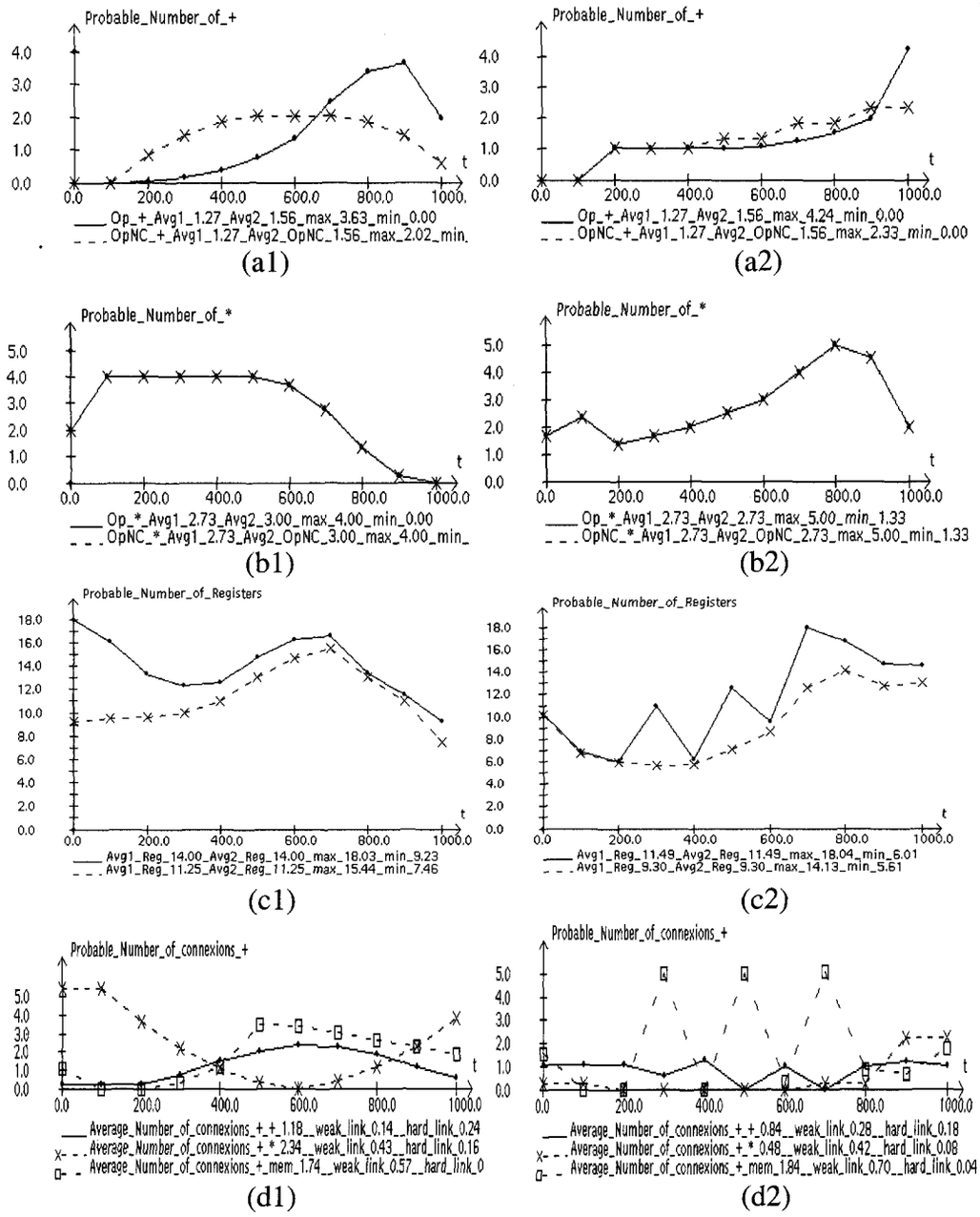


Figure 6. – Estimations du IIR7.

dans ses choix algorithmiques ou ses options de transformations. Un des intérêts majeurs des métriques proposées réside dans leur indépendance vis-à-vis de la synthèse et du modèle architectural utilisé. La méthode peut donc être appliquée de manière générique dans le cadre d'une bibliothèque d'architectures. Le fait que l'estimation soit située très tôt dans le cycle de conception lui confère un aspect de *haut-niveau* qui doit également être souligné et mis en relation avec les transformations. L'importance des estimations est en effet très liée à leur inter-action avec les transformations, dont le potentiel d'optimisation croît avec le niveau d'abstraction.

BIBLIOGRAPHIE

[1] Working groups, "Conclusions," in *Ieee Work. on Vlsi Signal Processing*, La Jolla, San Diego, Oct. 1994.

[2] M.Potkonjak and J.M.Rabaey, "Optimizing resource utilization using transformations," *Ieee Trans. on Computer-Aided Design*, vol. 13, no. 3, pp. 277–292, Mar. 1994.

[3] S.Note, F.Catthor, G.Goossens, and H.J.De Man, "Combined hardware selection and pipelining in high-performance data-path design," *Ieee Trans. on Computer-Aided Design*, vol. 11, no. 4, pp. 413–423, Apr. 1992.

[4] O.Sentieys, J.Ph.Diguet, J.L.Philippe, and E.Martin, "Hardware module selection for real time pipeline architectures using probabilistic cost estimation," in *Ieee ASIC conf.*, Rochester, USA, Sept. 1996.

[5] E.Martin, O.Sentieys, H.Dubois, and J.L.Philippe, "Gaut, an architectural synthesis tool for dedicated signal processors," in *EURO-DAC*, Hamburg, Oct. 1993, pp. 85–94.

[6] A.Sharma and R.Jain, "Estimating architectural resources and performance for high-level synthesis applications," *Ieee Trans. on Vlsi Systems*, vol. 11, no. 6, pp. 175–190, June 1993.

[7] J.M.Rabaey and M.Potkonjak, "Estimating implementation bounds for real time Dsp applications specific circuits," *Ieee Trans. on Computer-Aided Design*, vol. 13, no. 6, June 1994.

[8] L.Guerra, M.Potkonjak, and J.Rabaey, "System-level design guidance using algorithm properties," in *Ieee Work. on Vlsi Signal Processing*, San Diego, CA, Oct. 1994, pp. 73–82.

[9] J.Ph.Diguet, *Estimation de Complexité et Transformations d'Algorithmes de Traitement du Signal pour la Conception de Circuits Vlsi*, Ph.D. thesis, Université de Rennes I, Oct. 1996.

[10] P.G.Paulin and J.P.Knight, "Force-directed scheduling for the behavioral synthesis of asic's," *Ieee Trans. on Computer-Aided Design*, vol. 8, no. 6, pp. 661–679, June 1989.

Manuscrit reçu le 23 juillet 1997.

LES AUTEURS

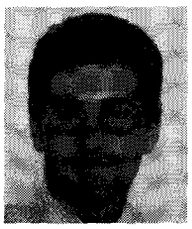
Jean-Phillipe DIGUET

Jean-Phillipe Diguet, ingénieur ESEO en 1992, DEA STIR de l'université de Rennes I en 1993 et le doctorat de l'université de Rennes I en traitement du signal et télécommunications en 1996. Il effectue sa thèse au LASTI à Lannion dans le domaine de la synthèse architecturale puis rejoint l'IMEC (Leuven, Belgique) pour une année post-doctorale sur le thème de l'optimisation des mémoires au niveau système. A présent maître de conférence à l'Université de Bretagne Sud et membre du LESTER, ses recherches s'articulent autour de la conception optimisée des systèmes VLSI pour le traitement du signal.

Jean-Luc PHILIPPE

Jean-Luc Phillipe a passé sa thèse en 1984, à l'université de Rennes I. Professeur à l'Université de Bretagne Sud depuis 1996, ses domaines d'intérêt portent sur l'inter-action algorithmes-architectures dans les domaines des applications orientées calculs (traitement du signal, télécom), et dans celles orientées contrôle (automatisme).

Olivier SENTIEYS



Olivier Sentieys diplômé de l'ENSSAT en 1990, il passe une thèse de l'Université de Rennes en 1993 sur les méthodes de conception des architectures hétérogènes. Maître de conférences à l'ENSSAT depuis 1993, il travaille au LASTI sur la synthèse de haut niveau des circuits VLSI dans le cadre des applications de traitement du signal. Il est un des responsables du projet GAUT, ainsi que dans ses évolutions pour les nouvelles contraintes technologiques et applicatives. Depuis 1996 il est co-responsable scientifique des activités du groupe Signal Architecture du LASTI.

Eric MARTIN

Eric Martin, né en 1961 est professeur agrégé de l'ENS de Cachan, Professeur des Universités et directeur du LESTER à l'Université de Bretagne Sud. Son domaine d'intérêt concerne l'adéquation algorithme architecture en traitement du signal et des images, et les méthodes de conception de haut niveau des circuits dédiés. Il pilote depuis 1988 le développement du projet GAUT.