# Optimisation d'exécution d'applications temps réelles sur plate-formes automobiles hétérogènes

Mohamed Benazouz[1], Paul Dubrule[2], Lilia Zaourar[1]

[1]Université Paris-Saclay, CEA, LIST [2]Alkalee

F-91120, Palaiseau nom.prénom@cea.fr, paul.dubrulle@alkalee.fr

**Mots-clés** : *affectation, ordonnancement, temps réel, automobile.*

## 1 Introduction

De nos jours, les Systèmes Cyber-Physiques (CPS) sont de plus en plus présents dans la vie quotidienne. Dans ces systèmes, les composants ont besoin d'une certaine quantité de données d'entrée pour produire une quantité connue de données de sortie, et certains d'entre eux doivent le faire en synchronisation avec une échelle de temps de référence.

La prochaine génération de véhicules autonomes s'appuiera fortement sur des systèmes CPS à base de fusion de capteurs pour faire fonctionner la voiture. Les capteurs et les actionneurs ont des fréquences spécifiques. Pour produire sa sortie, le noyau de fusion a besoin d'un certain nombre d'échantillons provenant de plusieurs sources, avec une corrélation temporelle entre eux.

La prédiction des performances est importante pour le concepteur du système lors de sa mise en oeuvre. Elle couvre différentes caractéristiques du système, notamment son débit, son empreinte mémoire et sa latence. Dans le cas d'exécutions distribuées de tels systèmes sur des plateformes matérielles, une analyse des communications entre les composants est nécessaire pour configurer un réseau capable de respecter le temps réel de l'application.

Prenons l'exemple de système de fusion de données qui pourrait être intégré à l'écran du cockpit d'une voiture, représenté sur La figure1. Celui-ci est composé de trois capteurs produisant des échantillons de données qui seront utilisés par un composant de fusion de données, et un composant d'affichage. La fonction des capteurs est de lire les données d'entrée, tandis que la fonction du composant de fusion de données est de calculer un résultat sur la base de ces données. La fonction du composant d'affichage est de rendre le résultat de la fusion sur un écran. Pour ce faire, les composants capteurs envoient les données au composant de fusion, et le composant de fusion envoie le résultat au composant d'affichage. Le premier composant capteur est une caméra vidéo produisant images. Les deux autres composants analysent les échantillons radar et lidar pour produire un descripteur des obstacles détectés les plus proches. Le composant de fusion utilise ces informations pour dessiner les descripteurs d'obstacles sur la trame correspondante.

Les formalismes de flux de données appelés *Data Flow (DF)* peuvent être utilisés pour effectuer ce type d'analyse des performances [1]. Les travaux de [2] présentent un formalisme de flux de données étendant les *Synchronous Data Flow (SDF)* avec une sémantique de synchronisation pour les acteurs ainsi que les propriétés correspondantes. Dans ce travail, nous présentons des stratégies d'allocation et d'ordonnancement partants de ce formalisme afin d'optimiser l'exécution d'applications temps réelles sur des platefrome de calculs hétérogènes. Elles constituent les futurs platefomes d'exécution d'applications sur les véhicules.

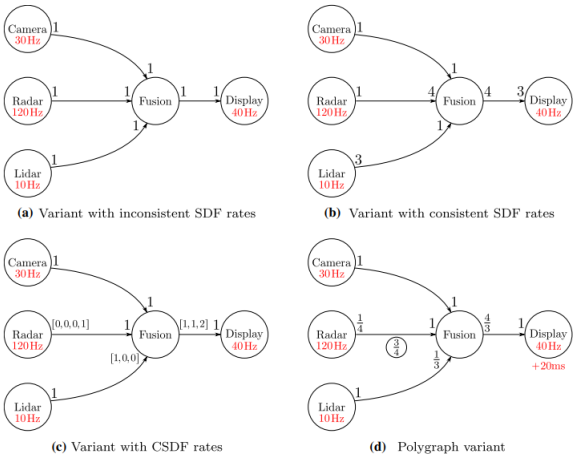


FIG. 1 – un exemple d'un système de fusion de données modélisé sous la forme d'un graphe de flux de données et un système de fusion de données modélisé comme un graphe de flux de données avec des fréquences sur un sous ensemble d'acteurs. Les variantes (a-d) spécifient une quantité de données échangées par les composants dans différentes variantes du modèle.

## 2 Modélisation et résolution

Nous partons d'une plateforme hétérogène dans laquelle $M$ est un ensemble de $m$ unités de traitement hétérogènes noté $PE$ (Processing Element). Un coût de communication fixe entre chaque paire $PE_k$ et $PE_l$ est noté $Cm_{kl}$. Une application $A$ de $n$ tâches est représentée par un graphe périodique orienté pondéré $G(V, E, w)$, chaque sommet $v_i \in V$ représente une tâche $t_i$ qui est caractérisée par son poids $w_i$, $i \in \{1, ..., n\}$. Chaque arc $e_{i,j}$ représente une contrainte de précédence entre deux tâches $t_i$ et $t_j$. Chaque arc $e_{i,j}$ est pondéré par un poids $Ct_{i,j}$ qui représente le volume de communications entre $t_i$ et $t_j$ si elles sont exécutées sur deux $PE_k$ différents. L'exécution de la tâche $t_i$ sur $PE_k$ engendre un temps d'exécution noté $execut_{i,k}$ et une puissance $P_{i,k}$. L'objectif ici est de minimiser le temps d'exécution total (*makespan*) tout en respectant une borne $D$ fixée pour la quantité d'énergie autorisée durant l'exécution.

Plusieurs méthodes de résolution existent dans la littérature pour le problème d'ordonnancement sur une plateforme hétérogène avec des paramètres et caractéristiques différentes. Nous nous appuyons dans ce travail sur les résultats de [3, 4]. Nous présenterons la modélisation mathématique détaillée de ce problème suivies de stratégies de résolution à la fois à base de solveurs et d'heuristiques spécifiques. Des comparaisons seront établies avec des instances industrielles issues du domaines de l'automobile.

## Références

[1] Benazouz, M., Munier-Kordon, A., Hujsa, T., Bodin, B.. *Liveness evaluation of a cyclo-static dataflow graph*. The 50th Annual Design Automation Conference 2013.

[2] Dubrulle, P., Kosmatov, N., Gaston, C., Lapitre, A.. *PolyGraph : a data flow model with frequency arithmetic*. International Journal on Software Tools for Technology Transfer, 2021.

[3] Honorat, A., Desnos, K., Bhattacharyya, S. S., Nezan, J. F. *Scheduling of synchronous dataflow graphs with partially periodic real-time constraints*. Proceedings of the 28th International Conference on Real-Time Networks and Systems, 2020.

[4] Ait Aba, M., Zaourar, L., Munier, A.. *Efficient algorithm for scheduling parallel applications on hybrid multicore machines with communications delays and energy constraint*. Concurrency and Computation : Practice and Experience Journal, 2020.