Ordonnancement des réseaux de capteurs sans fil embarqués

Anis Mezni

N°d'ordre NNT : 2019LYSEI030

THESE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de
INSA-Lyon
En cotutelle internationale avec
Faculté des Sciences de Tunis

Ecole Doctorale N° 160
Electronique, Electrotechnique et Automatique

Spécialité / discipline de doctorat : Automatique

Soutenue publiquement le 14/03/2019, par :
Anis Mezni

Ordonnancement des réseaux de capteurs sans fil embarqués

Devant le jury composé de :

Ben Hadj Alouane, néjib	Professeur	ENIT- Université Tunis Elmanar	Président
Rutten, éric	Chargé de Recherche HDR	INRIA Grenoble Rhône-Alpes	Rapporteur
Robbana, riadh	Professeur	INSAT-Université de Carthage	Rapporteur
Ben Ayed Jemni, leila	Professeur	ENSI-Université de la Manouba	Examinatrice
Niel, éric	Professeur	Ampère-INSA de Lyon	Directeur de thèse
Ben Ahmed, samir	Professeur	FST-Université Tunis Elmanar	Co-directeur de thèse
Dumitrescu, emil	Maître des conférences	Ampère-INSA de Lyon	Invité

Département FEDORA – INSA Lyon - Ecoles Doctorales – Quinquennal 2016-2020

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** **http://www.edchimie-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage **secretariat@edchimie-lyon.fr** INSA : R. GOURDON	M. Stéphane DANIELE Institut de recherches sur la catalyse et l'environnement de Lyon IRCELYON-UMR 5256 Équipe CDFA 2 Avenue Albert EINSTEIN 69 626 Villeurbanne CEDEX **directeur@edchimie-lyon.fr**
E.E.A.	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** **http://edeea.ec-lyon.fr** Sec. : M.C. HAVGOUDOUKIAN **ecole-doctorale.eea@ec-lyon.fr**	M. Gérard SCORLETTI École Centrale de Lyon 36 Avenue Guy DE COLLONGUE 69 134 Écully Tél : 04.72.18.60.97 Fax 04.78.43.37.17 **gerard.scorletti@ec-lyon.fr**
E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** **http://e2m2.universite-lyon.fr** Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 INSA : H. CHARLES **secretariat.e2m2@univ-lyon1.fr**	M. Philippe NORMAND UMR 5557 Lab. d'Ecologie Microbienne Université Claude Bernard Lyon 1 Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69 622 Villeurbanne CEDEX **philippe.normand@univ-lyon1.fr**
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** **http://www.ediss-lyon.fr** Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 INSA : M. LAGARDE **secretariat.ediss@univ-lyon1.fr**	Mme Emmanuelle CANET-SOULAS INSERM U1060, CarMeN lab, Univ. Lyon 1 Bâtiment IMBL 11 Avenue Jean CAPELLE INSA de Lyon 69 621 Villeurbanne Tél : 04.72.68.49.09 Fax : 04.72.68.49.16 **emmanuelle.canet@univ-lyon1.fr**
INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** **http://edinfomaths.universite-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 **infomaths@univ-lyon1.fr**	M. Luca ZAMBONI Bât. Braconnier 43 Boulevard du 11 novembre 1918 69 622 Villeurbanne CEDEX Tél : 04.26.23.45.52 **zamboni@maths.univ-lyon1.fr**
Matériaux	**MATÉRIAUX DE LYON** **http://ed34.universite-lyon.fr** Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bât. Direction **ed.materiaux@insa-lyon.fr**	M. Jean-Yves BUFFIÈRE INSA de Lyon MATEIS - Bât. Saint-Exupéry 7 Avenue Jean CAPELLE 69 621 Villeurbanne CEDEX Tél : 04.72.43.71.70 Fax : 04.72.43.85.28 **jean-yves.buffiere@insa-lyon.fr**
MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** **http://edmega.universite-lyon.fr** Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bât. Direction **mega@insa-lyon.fr**	M. Jocelyn BONJOUR INSA de Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69 621 Villeurbanne CEDEX **jocelyn.bonjour@insa-lyon.fr**
ScSo	**ScSo*** **http://ed483.univ-lyon2.fr** Sec. : Viviane POLSINELLI Brigitte DUBOIS INSA : J.Y. TOUSSAINT Tél : 04.78.69.72.76 **viviane.polsinelli@univ-lyon2.fr**	M. Christian MONTES Université Lyon 2 86 Rue Pasteur 69 365 Lyon CEDEX 07 **christian.montes@univ-lyon2.fr**

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Avant-propos

Ce travail s'inscrit dans le cadre d'une thèse en cotutelle internationale pour obtenir le grade de docteur de l'Université de Lyon délivré par l'INSA de Lyon, spécialité Automatique et le grade de docteur de l'université de Tunis El Manar délivré par la Faculté des Sciences de Tunis, spécialité Informatique.

Le travail que nous présentons dans cette thèse a été réalisé au sein du Laboratoire Ampère de l'INSA de Lyon en collaboration avec le Laboratoire Informatique pour les Systèmes Industriels (LISI-INSAT) en Tunisie. Il fait suite d'une thèse de doctorat qui s'est déjà déroulée au laboratoire Ampère, dans le cadre d'une coopération similaire (thèse de Mme Sajeh ZAIRI).

Cette thèse a été co-dirigée par le Professeur Eric Niel et le Maître de conférences Emil Dumitrescu à l'INSA et le Professeur Samir Ben Ahmed à la FST.

À mon directeur bien-aimé !

À mon co-directeur bien-co-aimé aussi !

Je dédie également ce travail
à tous ceux qui le méritent

Ordonnancement des réseaux de capteurs sans fil embarqués

Résumé

Les réseaux de capteurs sans fil ont attiré beaucoup d'activités de recherche et développement au cours de la dernière décennie. Pourtant, leur utilisation est restreinte, à ce jour, à la surveillance et l'acheminement des informations détectées. Cette thèse vise à introduire un nouvel aspect intéressant de point de vue fonctionnel. Partant d'une exigence spécifiée initialement, mettre en œuvre une synergie à plusieurs niveaux entre un ensemble des nœuds, tout en se basant sur une interaction adéquate. Ceci est réalisé par la génération automatique de code (correct par construction) et sa distribution par la suite en s'appuyant sur la théorie de contrôle par supervision. La synthèse de contrôleurs discrets SCD est une application de ce cadre théorique. Dans ce manuscrit, nous montrons comment la technique de la SCD peut être utilisée dans le domaine de réseaux de capteurs sans fil. Ainsi son potentiel se décline à deux niveaux. L'ordonnancement intra-cluster (groupe redondant de capteurs) avec des spécifications exprimant l'exclusion mutuelle lors de l'activation d'un capteur au sein d'un cluster, essentielle pour économiser l'énergie du réseau ainsi que la génération automatique d'un algorithme de routage optimal et multicritères pour les réseaux de capteurs. Spécifiquement, un chemin optimal devrait avoir à la fois une longueur minimale en termes de distance tout en évitant des chemins composés de nœuds dits « maillons faibles » (ayant un niveau d'énergie plus bas que la majorité).

Les outils formels cités s'appuient sur une démarche de modélisation basée sur des machines à états finis communicantes. Les verrous scientifiques sont liés à la nature d'un réseau de capteurs ainsi qu'à sa taille. La SCD génère des contrôleurs monobloc, alors qu'au sein d'un réseau de capteurs le traitement est essentiellement distribué. La problématique est celle de la distribution d'un contrôleur global, qui se présente sous forme d'une contrainte logique exprimée sur l'état global du réseau, sur chacun des nœuds du réseau, en ajoutant la synchronisation nécessaire pour garantir un fonctionnement distribué équivalent au contrôleur initialement généré.

Mots clés : réseaux de capteurs, synthèse de contrôleurs discrets, ordonnancement, génération automatique, correct par construction

: Laboratoire Ampère
25 Avenue Jean Capelle, Villeurbanne 69621 Cedex – France

Automatic Generation of Wireless Sensor Networks Scheduling

Abstract

Wireless Sensor networks are attracted many activities of research and development during the last decade. Yet, the distributed behavior of a WSN remains centered on two main objectives: sensing and routing. This thesis advocates the introduction of an additional feature, which can be considered interesting from a functional point of view and potentially from the power consumption one: starting from a designer-specified requirement, implement a multiple level synergy between (groups of) nodes, based on adequate interaction. This is achieved by automatic generation and distribution of correct-by-construction code, relying on the Supervisory Control Theory. The Discrete Controller Synthesis (DCS) technique is an application of this theoretic framework. In this thesis, we show how DCS can be used for WSN. Thus, its potential is at two levels. The intra-cluster scheduling of a redundant group of sensors with specifications expressing the mutual exclusion during the activation of a sensor within a cluster, essential to save the energy within the network and then a multicriteria automatic generation of an optimal routing functionality. Specifically, an optimal path should have both a minimal length and go through nodes having maximal residual energy.
The cited formal tools lean on a modelling approach based on communicating finite state machines (CFSM). The scientific challenges are generally related to the nature of the WSN as well as to its size. The DCS can only generates a monoblock controllers, while the WSN's behavior is essentially distributed. The issue is how to distribute a global controller, who appears in the form of a logical constraint expressed on the global state of the network, into local controllers while adding the necessary synchronization to guarantee a distributed functioning equivalent to the initially generated controller.

Keywords: wsn, discret control synthesis, scheduling, automatic generation, correct by construction

Remerciements

Premièrement, je tiens à remercier comme il se doit les membres du jury qui ont accepté de juger mon travail :

— Néjib Ben HADJ Alouane, Professeur à l'ENIT-Université de Tunis Elmanar, qui m'a fait l'honneur d'être président du jury ;

— Riadh Robbana, Professeur à l'INSAT-Université de Carthage et Eric Rutten, Chargé des recherches à INRIA-Gnenoble, pour avoir accepté de rapporter ma thèse. Leurs commentaires et questions me seront utiles pour préciser mes pistes de recherches futures ;

— Mme Leila Jemni Ben Ayed, Professeur à L'ENSI - Université de la Manouba, qui a bien voulu s'intéresser à mon travail et a accepté de faire partie du jury en tant qu'examinatrice ;

Je voudrais exprimer toute ma gratitude et ma reconnaissance à mes deux directeurs de thèse Eric Niel et Samir Ben Ahmed qui m'ont permis de préparer cette thèse dans le cadre d'une cotutelle internationale. J'ai bénéficié de leurs précieux conseils, et d'un soutien sans faille de leur part, que ce soit sur le plan académique ou administratif.

Je remercie tout particulièrement Emil Dumitrescu, mon encadrant de thèse, pour sa disponibilité et ses multiples conseils. Les longues discussions que nous avons eues ensembles ont permis de faire progresser mes travaux de recherche. Sans lui, rien de tout cela n'aurait été possible. Le mot merci me semble insuffisant, que Dieu te bénisse pour tout !

Je souhaite en outre remercier l'ensemble de mes collègues du laboratoire Ampère ainsi que ceux du laboratoire LISI-INSAT pour leur accueil et l'ambiance de travail depuis mes premières années de thèse jusqu'à aujourd'hui.

Je profite également de ces quelques lignes pour remercier mes parents. Pour leur soutien moral ainsi que matériel. Qu'ils sachent à travers ces mots, l'expression de mes remerciements les plus sincères.

A ma chère fiancée "Biba", pour son soutien, qui a été à mes côtés dans les moments de joie et de difficultés et qui m'a enveloppé de tendresse et d'affection. Qu'elle sache que je l'aime beaucoup.

Table des matières

Liste des tableaux

Table des figures

Introduction générale

Contexte et Problématique de la thèse

Les avancées technologiques réalisées au cours de ces dernières années, dans le domaine des réseaux sans fil, de la micro-mécanique et de la micro-électronique ont fait naître des petits appareils autonomes, communicants et équipés de capteurs à un coût raisonnable. Ces nouveaux appareils, appelés noeuds ou capteurs, constituent les briques de base des réseaux de capteurs sans fil à grande échelle et ont été à l'origine de leur émergence. Due aux caractéristiques intrinsèques de capteurs telle que la miniaturisation, la capacité de traitement et la communication sans fil, les réseaux de capteurs ont ainsi de nombreuses perspectives applicatives dans plusieurs domaines. Les applications militaires ont été le moteur initial dans le développement de ces technologies pour le suivi de déplacement des troupes ennemies par exemple ou l'analyse et la surveillance de terrains dangereux. Les réseaux de capteurs ont été aussi utilisés tant pour des applications environnementales, pour la détection de feux de forêts, la surveillance d'activité volcanique ou sismique, ou encore le suivi du déplacement d'animaux. Ainsi, pour des applications médicales comme le suivi des patients ou la veille épidémiologique, ou dans un but commercial, pour l'optimisation des processus de stockage, ou encore pour la construction de maisons intelligentes.

Tous ces applications se caractèrisent par des contraintes spécifiques. Cependant, le rôle d'un réseau de capteurs est à peu près, toujours le même. Les noeuds de ce réseau sont déployés de manière aléatoire dans une zone géographique, appelée zone d'intérêt dans le but de collecter un ensemble de paramètres de

l'environnement, telles que la température ou la pression, et autres. Les données récoltées sont acheminées grâce à des communications sans fil en multi-saut vers des points de collecte ou de traitement appelés station de base ou puits dotés de puissance de calcul et quantités d'énergie supérieure. Ce puits possède un lien radio qui lui permet d'envoyer les informations récoltées à l'utilisateur final.

Malgré, les perspectives attrayantes d'utilisation des réseaux de capteurs, les problématiques qu'engendrent ces réseaux n'en sont pas moins nombreuses. A priori, leur fonctionnement ne se base sur aucune infrastructure et, au moment de leur déploiement, les capteurs n'ont aucune information relative au réseau auquel ils appartiennent. De plus, un réseau de capteurs subit des contraintes qui lui sont propres et font de sa mise en œuvre un véritable challenge, telles que la portée réduite de communication des capteurs, la durée de vie limitée de leurs batteries, la faible capacité de stockage ainsi qu'une puissance de calcul réduite. Afin de surmonter ces contraintes, plusieurs problématiques de recherche ont vu le jour récemment, dont on peut citer ceux qui visent à optimiser la consommation énergétique des noeuds ce qui améliore en conséquence la durée de vie du réseau.

Les concepteurs des réseaux de capteurs sans fil ont établi une classification des traitements embarqués selon la consommation électrique qu'ils engendrent. Il en ressort que les opérations «locales» (mesure, calcul, ...) sont moins coûteuses en énergie que les opérations de communication, qu'il convient de minimiser, afin de prolonger la durée de vie du réseau. Par ailleurs, il s'avère que ces traitements locaux ne nécessitent pas d'être exécutés en continu, mais plutôt de manière périodique. Ainsi, le fonctionnement de chaque nœud est une alternance d'états de veille (les plus longs possibles), de calculs locaux, et de communications. Cette alternance est loin d'être arbitraire, car pour garantir la satisfaction de l'objectif global du réseau, une synergie doit exister entre les différents noeuds : en résumé, il faut être localement «endormi» le plus longtemps possible, mais se «réveiller» suffisamment souvent pour ne pas perdre une mesure importante, ou pour relayer la transmission d'une mesure importante effectuée par un autre nœud. C'est la problématique de l'ordonnancement des capteurs.

Objectifs de la thèse

Étant donné les perspectives applicatives prometteuses des réseaux de capteurs ainsi que les problématiques soulevées, le comportement distribué d'un réseau de capteurs reste, néanmoins, générique centré sur deux objectifs principaux : la détection et l'acheminement des informations captées.

La problématique traitée dans le cadre de ce travail de thèse s'inscrit dans ces tendances, mais la contribution comporte une prise de risque, par son positionnement en rupture vis-à-vis des approches largement adoptées par la grande communauté de chercheurs dans ce domaine. Cette rupture est la conséquence d'un souhait d'apporter des garanties de correction multi-échelle (locale et globale), lors de la conception d'un réseau de capteurs, et avant son déploiement. Ceci nécessite de s'appuyer sur des outils formels.

Ainsi, le travail présenté préconise l'introduction d'un nouvel aspect intéressant pour les concepteurs : partant d'une exigence globale spécifiée par le concepteur, mettre en œuvre une synergie à plusieurs niveaux entre un ensemble des nœuds, tout en créant une interaction adéquate entre ceux-ci. Ceci est réalisé par la génération automatique de code *correct par construction* et sa distribution par la suite, le tout en s'appuyant sur la théorie du contrôle par supervision. La synthèse de contrôleurs discrets SCD est une application de ce cadre théorique. Ainsi, notre objectif est de proposer une démarche de conception de l'ordonnancement d'un réseau de capteurs en s'appuyant sur la SCD, garantissant à la fois la correction de cet ordonnancement, et l'utilisation minimale des ressources d'énergie du réseau. En effet, l'implémentation manuelle d'une politique d'ordonnancement peut être complexe, coûteuse, et implique des séries de tests et de corrections successives. Par contre, la SCD offre la capacité de générer automatiquement et correctement des programmes en conformité avec des spécifications exprimées par l'utilisateur sous forme d'assertions. Le fragment de programme généré, connu sous le nom de "contrôleur", renforce la satisfaction de ces spécifications. Dans le contexte des réseaux de capteurs, un tel outil permet de générer des règles d'ordonnancement multi-échelles. Dans ce travail, son potentiel se décline à deux niveaux :

1. capteur/ "cluster" (groupe redondant de capteurs) avec des spécifications

exprimant l'exclusion mutuelle lors de l'activation d'un capteur au sein d'un cluster, essentielle pour économiser l'énergie au sein du réseau ;

2. routage au sein du réseau, un routage optimal selon des critères hiérarchisés : plus court chemin, évitement des chemins composés de noeuds dits "maillons faibles" (ayant un niveau d'énergie plus bas que la majorité).

Les outils formels cités s'appuient sur une démarche de modélisation basée sur des machines d'états finis communicantes.

Principales contributions

Nous résumons, dans cette section, les contributions présentées dans cette thèse.

Le regroupement des nœuds en "clusters" est l'une des techniques les plus adoptées dans le domaine de réseaux de capteurs pour satisfaire l'objectif de passage à l'échelle et prolonger la durée de vie du réseau. Dans cette optique, nous intéressons dans la première contribution de cette thèse à un ordonnancement intra-cluster d'un réseau de capteurs. En se basant sur l'algorithme GAF [158], la zone cible sera divisée en plusieurs grilles virtuelles, chaque nœud appartient à la grille correspondante en fonction de ses coordonnées géographique. Les nœuds appartenant à la même grille forment ainsi un cluster. Par ailleurs, nous proposons un modèle formel abstrait pour chaque nœud du réseau. Ce modèle permet d'associer le comportement d'un nœud à la consommation électrique estimée pour ce comportement. Par application de la technique de la SCD, on génère un superviseur centralisé qui garantit une condition d'exclusion mutuelle lors de l'activation de nœuds d'un même cluster : un seul capteur actif à un moment donné au sein d'un cluster. Cependant, cet ordonnanceur est centralisé (sous forme d'une contrainte logique) : il requiert une vue globale sur l'ensemble du réseau. Ceci est incompatible avec le fonctionnement distribué, inhérent à un réseau de capteurs. Nous proposons ainsi une manière appropriée de distribution locale d'un ordonnancement global par la décomposition structurelle du superviseur centralisé. Cette approche répond au besoin de contrôle localisé, mais induit un surplus significatif de communication au sein du réseau,

nuisant à sa durée de vie. Ce surplus de communication se justifie sur le plan fonctionnel : il faut garantir une synchronisation entre les nœuds communicants. Nous proposons de réaliser cette exigence par l'intermédiaire d'une «barrière» de synchronisation : un mécanisme logiciel obligeant un ensemble de capteurs à s'attendre mutuellement avant de prendre une décision locale à chacun, et avant désactivation.

L'approche proposée est amenée à agir sur la structure de l'ordonnanceur. Celui-ci est transformé d'abord par la décomposition structurelle, puis par l'ajout d'une barrière de synchronisation. Ces opérations qui visent à raffiner le comportement de l'ordonnanceur, modifient donc le code de celui-ci, alors qu'il avait initialement été généré automatiquement. Ceci engendre un besoin de validation du modèle raffiné obtenu, par rapport à l'ordonnanceur abstrait généré initialement par synthèse.

Comme la plupart des techniques formelles, la taille d'un réseau de capteurs est un problème difficile pour l'application de la SCD. Cette technique est limitée par une complexité spatiale exponentielle en nombre de variables d'état du système exploré, ce qui la rend a priori incompatible avec l'application envisagée. Nous proposons ainsi une démarche méthodologique étayée par un algorithme, qui permet d'exploiter la symétrie au sein d'un réseau de capteurs et déduire des règles inductives permettant d'étendre la formule d'un contrôleur distribué à un nombre de nœuds arbitrairement grand.

La seconde contribution de thèse, consacrée au problème du routage inter-cluster, en explorant la problématique de la distribution et la communication entre contrôleurs. Nous proposons de générer automatiquement un contrôleur global puis le distribuer de telle sorte qu'au niveau de chaque nœud, on obtient une « table de routage » spécifique à ce nœud. Notre algorithme de routage optimal se base sur deux critères hiérarchisés : le nombre de sauts minimal reliant chaque nœud à la station de base tout en évitant de choisir les nœuds critiques (nœuds avec des réserves énergétiques assez épuisées) comme nœuds relais lors de l'acheminement de messages. Cette proposition permet d'économiser les réserves énergétiques minimales des nœuds d'une part et d'avoir une consommation d'énergie équitable entre les différents nœuds du réseau d'autre part.

Organisation du document

Ce manuscrit est divisé en deux parties principales. La partie I présente le cadre de travaux et l'état de l'art, elle s'articule autour de deux chapitres.

Chapitre 1 : Les réseaux de capteurs sans fil

Ce chapitre constitue une introduction au domaine très vaste des réseaux de capteurs. Nous présentons les différents concepts liés à la mise en œuvre d'un réseau de capteurs, ses caractéristiques et les différents domaines d'application.

Chapitre 2 : Modèles, langages et outils pour la conception des Réseaux de Capteurs Sans Fil (RCSFs)

Il présente les modèles, les langages et les outils existants qui peuvent être utilisés pour résoudre les problèmes de conception ciblés. Parmi eux, nous identifions et introduisons les modèles synchrones à base d'automates, ainsi que les techniques de contrôle discrètes et les outils qui seront appliqués dans cette thèse.

La partie II expose les contributions de la thèse. Elle est aussi constituée de deux chapitres.

Chapitre 3 : Ordonnancement intra-cluster :

Ce chapitre traite de la problématique d'ordonnancement d'activité des noeuds dans les réseaux de capteurs. Après une introduction du problème, nous décrivons quelques solutions existantes centralisées et distribuées. Dans un second lieu, nous montrons l'intérêt d'utiliser la technique de la synthèse de contrôleurs discrets (DCS) afin d'obtenir un ordonnancement automatique correct par construction d'un ensemble de noeuds-capteurs communicants. Enfin, nous présentons quelques résultats de simulation pour évaluer les performances de notre approche d'ordonnancement.

Chapitre 4 : Génération automatique d'un routage optimal multi-critères pour les RCSFs

Au cours de ce chapitre, nous introduisons l'algorithme de routage développé après avoir présenté les principaux travaux récents de routage dans le domaine de réseaux de capteurs dont une synthèse justifiant le développement de l'algorithme proposé. Nous réalisons ensuite, des simulations approfondies sur l'algorithme proposé et les résultats sont comparés aux algorithmes exis-

tants pour démontrer leur supériorité en termes de quelques métriques de performance comme la durée de vie du réseau et la quantité totale d'énergie consommée.

Nous finalisons ce manuscrit par une conclusion générale. Nous rappelons les différentes contributions réalisées tout au long de ce travail de recherche et nous mettons en relief les perspectives de recherche de l'ensemble des travaux présentés. Les travaux contenus dans ce document ont été publiés dans des conférences internationales MobiWac 2016, HPCS 2017 et IWCMC 2018 dont ils sont disponibles dans la liste des publications.

Première partie

Cadre des travaux et état de l'art

Chapitre 1

Les réseaux de capteurs sans fil

1.1 Introduction

Ce chapitre présente une introduction au domaine de RCSFs. Il en rappelle la terminologie et les mécanismes spécifiques afin de faciliter la lecture des chapitres suivants. L'élément de base constituant un RCSF est le nœud. L'architecture d'un nœud ainsi que celle d'un réseau de capteurs sont détaillées dans la section 2. Une synthèse des applications potentielles de réseaux de capteurs est présentée dans la section 3. La section 4 décrit les différents facteurs qui influencent la conception de RCSFs. Enfin, la section 5 constitue une réflexion sur la conception des capteurs d'un réseau, offrant en guise de conclusion une ouverture vers la problématique traitée dans ce document.

1.2 Analyse structurelle des réseaux de capteurs sans fil

1.2.1 Architecture d'un nœud

Un nœud est composé de quatre composants principaux comme illustré dans la figure 1.1, à savoir : Unité d'acquisition de données, unité de traitement, unité de communication ainsi qu'une unité source d'énergie. Chaque nœud peut disposer également, d'autres dispositifs optionnels, tels qu'un système de mobilisation qui permet aux nœuds, si nécessaire, de se déplacer, un système de localisation

comme un GPS (Global Positioning System) [2], ainsi qu'un générateur d'énergie permettant de fournir une quantité d'énergie supplémentaire à l'unité d'énergie à l'instar comme une cellule photovoltaïque, par exemple). Les composants principaux sont encadrés par des traits pleins dans la figure 1.1 par contre les composants optionnels sont en pointillés.

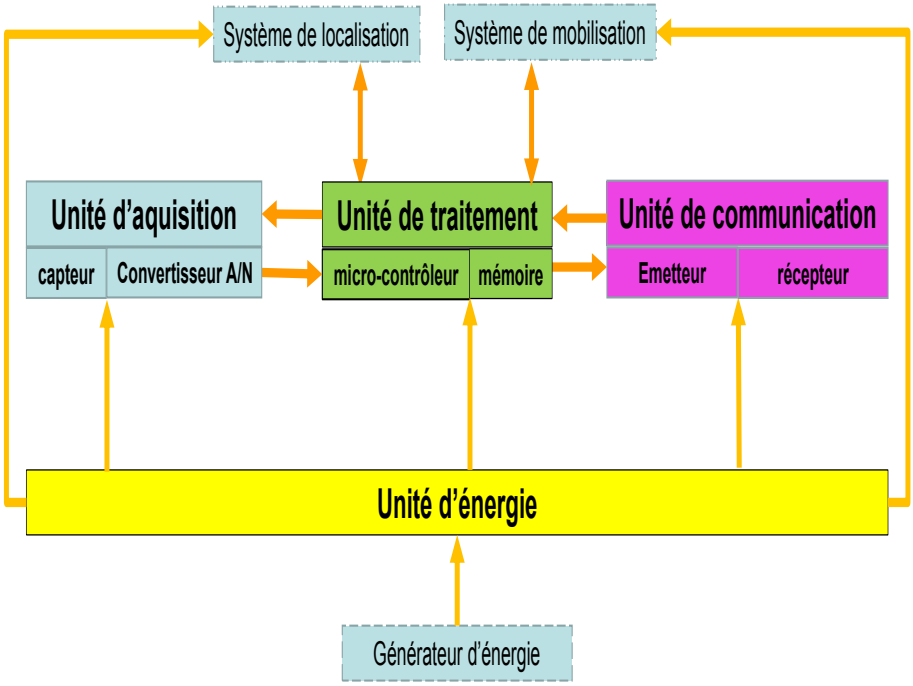

FIGURE 1.1 – Architecture d'un nœud

L'unité d'acquisition de données est composée de deux sous unités : le capteur et le convertisseur analogique-numérique ADC (Analog to Digital Converter). Ce capteur est chargé de mesurer une grandeur physique tandis que le convertisseur analogique-numérique convertit les signaux analogiques en données numériques à transmettre à l'unité de traitement.

L'unité de traitement comporte aussi deux sous unités : l'unité de stockage et l'unité de calcul (généralement un micro-contrôleur à faible consommation d'énergie). L'unité de stockage est pour enregistrer les données environnementales recueillies. Ces données seront traitées par la suite par le micro-contrôleur. Le rôle d'un micro-contrôleur ne se limite seulement qu'aux traitements de données, mais il commande aussi les autres unités notamment l'unité de com-

munication.

L'unité de communication est un émetteur-récepteur radio qui permet aux noeuds du réseau de communiquer entre eux par une liaison sans fil. Cette unité peut être en 4 modes de fonctionnement différents : "Transmit", "Receive", "Idle", et "Sleep". Les deux premiers modes sont les plus consommateurs en terme d'énergie. Malheureusement, comme tous les réseaux sans-fil, la quantité d'énergie nécessaire à une opération de transmission augmente aussi avec la distance. C'est pour cela que les nœuds d'un réseau de capteurs recourent souvent à un routage multi-sauts pour préserver l'énergie.

L'unité source d'énergie (généralement une batterie) représente la seule source qui permet de fournir l'énergie à toutes les autres unités, pour effectuer les tâches décrites précédemment. On peut noter aussi que les nœuds peuvent être rechargés grâce à des sources externes [134] telles que les cellules solaires, vibrations, etc. Cependant, ces techniques d'alimentation dépendent de plusieurs facteurs environnementaux [134] leur empêchant ainsi de fournir une alimentation en continu des nœuds mais elles peuvent être utilisés comme des sources complémentaires d'énergie. Ainsi, la consommation énergétique des noeuds est alors un des critères fondamentaux qu'il faut tenir en compte dans la conception d'un RCSF.

1.2.2 Structure d'un réseau de capteurs

Un RCSF est généralement composé d'un très grand nombre de nœuds voire des milliers, autonomes en terme d'énergie, déployés sans infrastructure dans des zones étendues. Ces nœuds sont capables de surveiller des phénomènes environnementaux (humidité, température, vibration, mouvement, etc..), de traiter les données collectées, et de collaborer pour les transférer à la station de base via des liaisons sans fil. La station de base s'engage ainsi à recueillir ces données et de les envoyer à son tour à l'utilisateur final soit via Internet ou satellite. Vu la portée radio limitée des nœuds, les communications dans un RCSF se font souvent selon une architecture multi-sauts. Comme illustré sur la figure 1.2, chaque nœud doit ainsi relayer les données collectées par d'autres noeuds afin de les acheminer de proche en proche jusqu'au centre de

traitement, où ces données doivent être traitées et éventuellement exploitées par l'utilisateur final. On s'intéresse uniquement aux nœuds, à la station de base, et au réseau d'interconnexion (encadré en pointillés dans la figure 1.2) puisque elle est la partie la plus complexe (communication entre les nœuds) et la plus contraignante (ressources limitées des nœuds, etc.), il ne s'agit pas d'étudier comment le puits est connecté à l'utilisateur final.

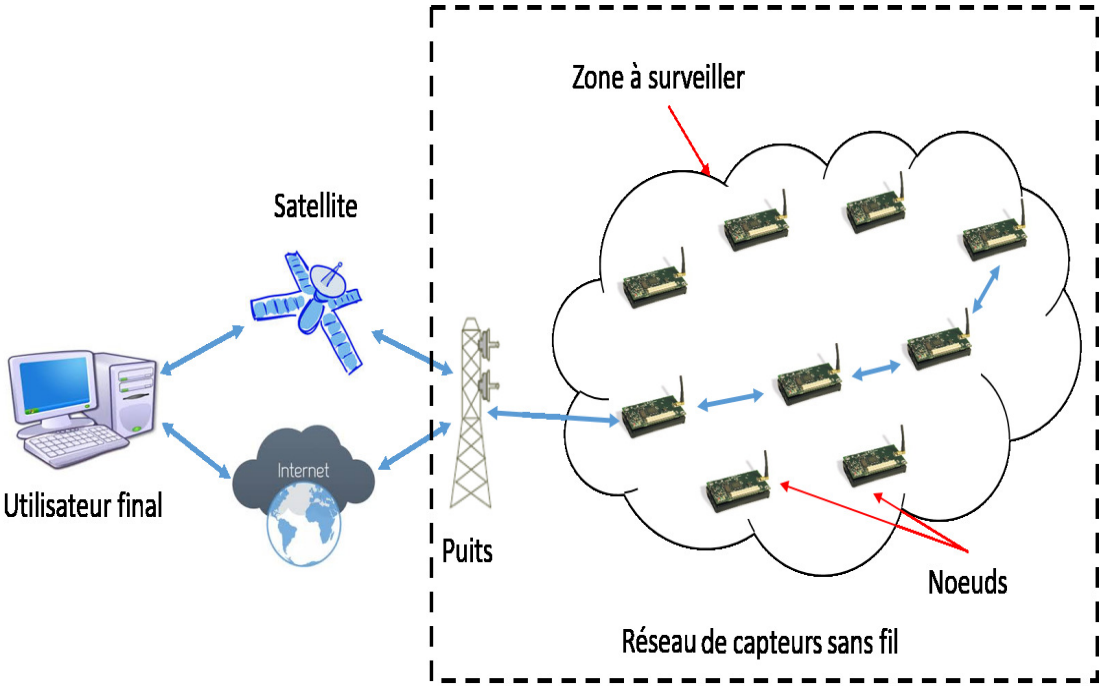

FIGURE 1.2 – Architecture d'un réseau de capteurs sans fil

1.2.3 Architectures des réseaux de capteurs sans fil

Dans le domaine des réseaux de capteurs, on trouve deux architectures possibles :

Architecture plate

Un réseau de capteurs sans fil plat est un réseau composé d'un ensemble des noeuds homogènes et identiques de point de vue batterie et complexité du matériel, à l'exception de puits. Cette architecture plate se caractèrise par une

densité assez élevée de capteurs où les noeuds se communiquent en multi-saut. Cependant, en présence d'un très grand nombre de noeuds, le passage à l'échelle devient critique. Ainsi, les protocoles dédiés à ce type d'architecture (routage, MAC, etc..) doivent gérer et organiser les noeuds d'une manière très efficace en termes d'énergie.

Architecture hiérarchique

Une architecture hiérarchique était proposée pour réduire essentiellement la communication sur des longues distances et par conséquent la préservation de l'énergie en regroupant les noeuds en clusters. Pour chaque cluster, un seul nœud, appelé clusterhead, est déterminé pour coordonner les activités de autres membres du cluster tels que le routage,la synchronisation et/ou l'agrégation des données, etc. Les membres d'un cluster peuvent passer en mode veille afin de préserver leurs énergies s'il y a une redondance de couverture au sein de cluster. Sur détection d'un événement ou sur requête envoyé de la station de base, les noeuds d'un même cluster qui demeurent actifs transmettent ainsi leurs données (grandeurs physiques mesurées : température, pression ...) au cluster-head. Ce dernier s'occupe de relayer ces données jusqu'au puits.

Afin de réduire la consommation énergétique des noeuds et augmenter par conséquent la durée de vie du réseau tout en garantissant la propriété de passage à l'échelle, nous adoptons au cours de ce travail une architecture hiérarchique. En effet, le réseau de capteurs étudié est partitionné alors en des sous-réseaux (clusters) tout en respectant les exigences de connectivité. Chaque cluster est ainsi composé d'un nombre réduit de noeuds redondants. Cette redondance va être par la suite exploitée pour conserver les énergies de noeuds et par conséquent l'augmentation de la durée de vie du réseau.

1.3 Domaines d'application des réseaux de capteurs

La taille réduite des nœuds-capteurs, leur coût de fabrication faible ainsi que leur faculté de communication sans fil permettent aux réseaux de capteurs d'envahir un large éventail d'applications telles que le contrôle militaire, in-

dustriels, agricole, biomédical, surveillance de l'environnement et beaucoup d'autres applications.

Applications militaires : comme pour plusieurs autres technologies de l'information, les applications militaires ont été aussi le moteur principal de développement des réseaux de capteurs [37]. La détection des troupes d'ennemis était le sujet d'une des premières exploitations des réseaux de capteurs. L'emploi des réseaux de capteurs dans le domaine militaire peut aller aussi jusqu'à la surveillance des terrains de bataille, ou bien encore les frontières géographiques d'un pays avec ses voisins.

Applications environnementales : on peut créer un réseau de capteurs autonome en déployant les nœuds dans la nature. Des événements naturels peuvent ainsi être détectés tel que un début de feux dans les forêts, inondations, Tsunami, etc.. Ceci permet de sauver des vies humaines par des interventions beaucoup plus rapides et efficaces des secours [163] [155] [73] grâce aux capteurs déployés dans ces zones de risque. un fameux exemple d'application environnementale pour les réseaux de capteurs est le projet ARGO là où des nœuds équipés par des capteurs de température et de salinité sont utilisés pour surveiller l'eau de l'océan, sa température, sa salinité ainsi que sa vélocité.

Applications urbaines et domotiques : les capteurs entrent de plus en plus dans nos vies quotidiennes. La mise en réseau de ces capteurs forme un environnement dit pervasif qui permet de fournir toutes les informations nécessaires aux applications de sécurité, de maintenance et de confort dans une maison [53]. Le déploiement d'un tel réseau permet de déduire des actions intelligentes suivant le comportement des occupants créant ainsi une maison "intelligente".

Applications médicales : L'implantation de certains nombre de capteurs dans le corps humain permet de contrôler les problèmes médicaux d'un individu. Ils servent à surveiller les patients et dégagent leurs états d'avancement dans un hôpital. A titre indicatif, l'implantation des mini capteurs vidéo sous la peau permet de recevoir des images en temps réel d'une partie de corps sans aucune chirurgie et pendant environ 24h. Ainsi, un réseau de capteurs pour la surveillance des «signes vitaux» est réalisé dans [11].

Applications agricoles : Le déploiement d'un réseau de capteurs dans un champs agricole permet de prendre des mesures sur certains paramètres tels que

la température, l'humidité etc. Ces paramètres sont ainsi envoyés à un centre de traitement pour optimiser l'irrigation en eau ou pour déterminer les zones les plus sèches par exemple. L'utilisation d'un tel réseau dans le domaine agricole permet l'apparition de ce qu'on appelle l' « agriculture intelligente ».

1.4 Comportement d'un réseau de capteurs sans fil

L'une des spécificités qui caractérise les réseaux de capteurs est l'application à laquelle ils sont dédiés. Ceci rend difficile la définition d'un comportement applicatif générique de ces réseaux. Néanmoins, après une étude de quelques applications dédiées aux réseaux de capteurs sans fil, on peut distinguer deux grandes familles génériques des leurs comportements applicatifs possibles :

Collecte de données : Les noeuds du réseau captent de manière périodique des grandeurs environnementales (température, pression, etc.) et les transmettent par la suite à la station de base. En effet, pour déterminer l'état de la zone de couverture à un instant t, le puits diffuse une requête pour que les capteurs déployés dans cette zone remontent leur dernier relevé. Les informations sont ainsi routées de saut en saut jusqu'au puit.

Détection d'événements : La zone d'intérêt est surveillée en permanence par un réseau de capteurs mais aucune alerte n'est acheminée à la station de base que lorsque un événement se produit en un point quelconque dans cette zone tel que une détection de début de feu dans une forêt ou la détection d'un mouvement. Ceci amène les capteurs situés à proximité à remonter les informations captées vers le puits.

1.5 Déploiement d'un réseau de capteurs sans fil

Le déploiement consiste à placer un certains nombre de capteurs dans une zone d'intêret afin de la surveiller. Ainsi, il peut se réaliser de manière prédéterminée ou aléatoire dépendant essentiellement de l'application du réseau et la nature de la zone à surveiller.

— **déploiement prédéterminé** : Quand l'environnement est accessible, il est

possible de placer les noeuds à des positions prédéfinies (placement individuel de chaque nœud). Ce type de déploiement est souvent applicable aux applications de surveillance urbaine, industrielle ou à la domotique. c'est ainsi que sont déployés les noeuds chargés de réguler la climatisation d'un immeuble ;

— **déploiement aléatoire** : Ce type de déploiement est utilisé quand la zone à surveiller est inaccessible ou hostile. Ainsi, les noeuds sont placés à des positions que l'on ne connaît pas à l'avance (largués par avion, par exemple, sur la zone d'intérêt). Dans un tel réseau, les noeuds doivent être capables d'organiser de façon coopérative les tâches de détection, et de transmission des données sans aucune intervention humaine mais aussi s'auto-adapter par rapport aux conditions changeantes du réseau. Au cours de ce travail, le mode de déploiement aléatoire est considéré. Une fois disséminés, les noeuds sont aussi supposés être statiques.

1.6 Facteurs influençant la conception de réseaux de capteurs

Un RCSF subit néanmoins des contraintes qui lui sont propres et font de sa mise en œuvre un véritable challenge. Ces contraintes sont les suivantes :

— **consommation d'énergie :** L'économie d'énergie est un challenge important dans les réseaux de capteurs puisque les noeuds sont équipés par des batteries à capacité limitée et leur rechargement une fois les noeuds sont déployés est une opération très coûteuse et difficile. Un réseau de capteurs ne peut pas survivre si la perte de nœuds est très importante car ceci engendre des pertes de communications dues à très grande distance entre les nœuds restants. Par conséquent, les noeuds devraient économiser au maximum leurs énergies afin de pouvoir fonctionner pour une longue période ;

— **passage à l'échelle :** Le nombre de nœuds déployés pour une application donnée peut être à l'ordre des centaines voire des milliers de nœuds. Un

nombre aussi important de nœuds engendre beaucoup de messages entre ces noeuds. Afin de garantir le bon fonctionnement du réseau, les protocoles dédiés aux RCSFs doivent ainsi prendre en compte le facteur "grande échelle" ;

— **contraintes matérielles :** On exige généralement un coût très faible pour permettre un déploiement de réseaux de capteurs à forte densité. Le challenge posé par certains industriels est qu'un nœud doit être placé dans une petite surface n'excédant pas, généralement, un centimètre cube ($1cm^3$). Cependant, les noeuds ont des capacités de calcul, de traitement et de stockage limitées, à cause de ces contraintes de taille (miniaturisation), de coût et de faible consommation d'énergie ;

— **topologie du réseau :** La topologie d'un réseau de capteurs est souvent très dynamique, puisque certains noeuds dans ce réseau peuvent devenir indisponibles (défaillance, épuisement de batteries, perte des liens sans fil, etc.). De même, le déploiement de nouveaux capteurs dans la zone d'intérêt, rend aussi la topologie du réseau fréquemment instable.

— **l'auto-configuration :** est la capacité des nœuds, à partir de moment de leur déploiement, à découvrir leur voisinage et à s'organiser en un réseau structuré et fonctionnel sans intervention humaine ;

— **l'auto-reconfiguration :** est la capacité d'un réseau de capteurs à changer sa configuration initiale dans le but d'atteindre un objectif exigé. Ces objectifs peuvent être par exemple la reconstruction des chemins entre les différents noeuds du réseau et la station de base ou l'ordonnancement d'activités des noeuds pour réduire la consommation d'énergie d'un groupe de noeuds redondant et par conséquent l'augmentation de la durée de vie du réseau ;

— **tolérance aux pannes :** Dans un RCSF, certains nœuds peuvent être bloqués ou tomber en panne à cause d'un épuisement de sa batterie ou d'un dégât matériel. Ces facteurs rendent ce type de réseaux très vulnérables. Ainsi, la perte d'un nœud ou d'un lien ne doit pas affecter le fonctionnement global du réseau.

1.7 Techniques de conservation d'énergie dans les réseaux de capteurs

Les noeuds d'un réseau de capteurs sont supposés fonctionner de manière autonome, ils doivent s'auto-reconfigurer afin d'assurer la tâche à laquelle est dédié le réseau. Dans ce travail, la reconfiguration des noeuds inclut deux éléments essentiels : soit au niveau local par l'ordonnancement d'activité de noeuds soit au niveau global par la détermination d'un routage optimal. Cette reconfiguration doit assurer non seulement un fonctionnement économe en énergie mais correct aussi de point de vue comportemental.

1.7.1 Niveau local : ordonnancement d'activité de noeuds

Le premier élément du problème de reconfiguration des réseaux de capteurs est l'ordonnancement d'activité des nœuds. De nombreux travaux ont établi une classification des traitements embarqués selon la consommation électrique qu'ils nécessitent. Il en ressort que les opérations «locales» (mesure, calcul, ...) sont moins coûteuses en énergie que les opérations de communication. Par ailleurs, il s'avère que ces traitements locaux ne nécessitent pas d'être exécutés en continu, mais plutôt de manière périodique. Ainsi, le fonctionnement de chaque nœud est une alternance d'états de veille (les plus longs possibles), de calculs locaux, et de communications. Cette alternance est loin d'être arbitraire, car pour garantir la satisfaction de l'objectif global du réseau, une synergie doit exister entre les différents nœuds : en résumé, il faut être localement « endormi » le plus longtemps possible, mais se « réveiller » suffisamment souvent pour ne pas perdre une mesure importante, ou pour relayer la transmission d'une mesure importante effectuée par un autre nœud. Les travaux d'ordonnancement des activités des noeuds ont trouvé leur justification dans cette optique [33] [166]. Mettre en veille les noeuds redondants permet de préserver leurs énergies et ainsi augmenter la durée de vie du réseau [55] [153].

1.7.2 Au niveau global : routage optimal

Déterminer les chemins optimaux permettant de connecter un nœud à la station de base, tout en assurant une consommation énergétique assez réduite et équilibrée entre les différents noeuds de réseau, est une fonctionnalité très importante pour un réseau de capteurs. L'algorithme de routage proposé est adapté aux caractéristiques des réseaux de capteurs (faible capacité de stockage, énergie limitée, etc ..). Il est généré automatiquement, offrant une méthode pour déterminer les chemins optimaux suivant deux métriques indépendantes : le minimum de nombre de sauts intermédiaires pour atteindre la station de base et le maximum de réserves énergétiques des noeuds choisis tout au long des chemins menant à la station de base. Les détails de cet algorithme sont introduits dans le chapitre 4.

1.8 Conclusion

Comme la plupart des systèmes embarqués, les réseaux de capteurs sans fil sont soumis à des exigences de conception spécifiques telles que les contraintes d'énergie et la correction fonctionnelle. Ces exigences sont efficacement gérées par des approches de conception qui ont acquis une maturité industrielle au cours des dernières décennies [63] [124] [47]. Ce chapitre identifie et énumère les tendances en matière de conception d'utilisation des réseaux de capteurs. Très nombreuses, ces tendances évoquent un potentiel de développement important des réseaux de capteurs, aussi bien en termes d'améliorations matérielles (électroniques), logicielles, et des services fournis. Minimiser la consommation d'énergie d'un nœud, et de ce fait maximiser la durée de vie du réseau, reste néanmoins un leitmotiv qui contraint tous les travaux dans ce domaine.

Il est ainsi d'usage de concevoir la fonction d'un nœud, qu'il s'agisse de l'acquisition des données environnantes, ou de l'ordonnancement, ou encore sa communication avec son voisinage, de la manière la plus générique possible. Les nœuds d'un réseau sont ainsi identiques du point de vue logiciel (i.e même architecture logicielle), et se distinguent par les données qu'ils contiennent et qui évoluent au fil du temps par le biais d'un processus d'auto-apprentissage.

En d'autres termes, on conçoit un seul capteur et on le duplique par la suite pour obtenir un réseau homogène. Ainsi, pour une application de surveillance environnementale, chaque nœud doit assurer les mêmes fonctions (détection et transmission des alertes vers la station de base) dès son déploiement initial jusqu'à l'extinction de sa batterie. Ce travail de conception est effectué *manuellement*, selon une logique globale qui doit mettre en œuvre un comportement distribué et communicant, et nécessite de ce fait un grand niveau d'expertise.

Le travail effectué dans cette thèse préconise une démarche aux vertus complémentaires : renforcer la qualité du travail du concepteur en s'appuyant sur des outils formels capables de générer du code *correct par construction*.

Cette démarche entraîne la possibilité de *reconfigurer* le réseau régulièrement par la génération d'une fonction de contrôle globale répondant éventuellement à des contraintes de réactivité (demande de réactivation d'un nœud, perte d'un nœud et autres). Cette fonction globale sera par la suite concrétisée par des logiques locales obéissant à l'architecture distribuée du réseau de capteurs. Ce travail présume la capacité de récupérer la topologie du réseau et de le reconfigurer en transmettant le bon contrôleur vers le bon nœud.

Les outils formels sont un élément clé au sein de cette démarche de conception ; ils apportent aux concepteurs une aide précieuse dans la gestion efficace et correcte de la complexité conceptuelle manuelle : découvrir des erreurs subtiles de conception [104], générer des modèles de test [59] ainsi que la génération automatique de code correct-par construction [22].

Malgré leur efficacité reconnue, il y a peu de tentatives d'application de méthodes formelles dans le domaine de réseaux de capteurs. Le prochain chapitre fait état des approches formelles jugées intéressantes pour la modélisation et l'analyse de réseaux de capteurs sans fil.

Chapitre 2

Modèles, langages et outils pour la conception des RCSFs

2.1 Introduction

Les réseaux de capteurs sans fil s'apparentent à des *systèmes réactifs* : ils interagissent continuellement avec leur environnement extérieur et qui s'exécutent au rythme imposé par ce dernier. Ils sont concurrents par nature. Ces systèmes implémentent des fonctions critiques qui doivent être correctes avant leur mise en exploitation. Dans ce sens, d'importantes garanties de correction sont offertes par les méthodes de conception et les outils formels. Ce chapitre offre une analyse critique des démarches de conception de réseaux de capteurs sans fil (RCSFs) basées sur des outils et langages formels, et introduit la synthèse de contrôleurs discrets (DCS) comme outil de conception complémentaire, offrant un potentiel intéressant.

Parmi les solutions formelles proposées pour l'étude et la conception de réseaux de capteurs et des systèmes réactifs en général, les approches à base d'automates tels que les STATECHARTS [66] et les langages synchrones [17] trouvent un intérêt particulier : elles offrent des mécanismes de modélisation à la fois intuitifs et expressifs pour représenter la séquentialité, la concurrence, la communication et pour structurer les modèles.

La section 2.2 rappelle les approches formelles courantes pour la modé-

lisation et l'analyse des RCSFs. Les formalismes de modélisation à base des automates sont présentés dans la section 2.3. Nous présentons dans la section 2.4, la synthèse de contrôleurs discrets (DCS) issue de la théorie de contrôle par supervision, une technique que peut être appliquée à ces formalismes pour contrôler le franchissement des transitions. Quelques travaux existants appliquant la théorie de contrôle par supervision SCT sont finalement présentés dans la section 2.5.

2.2 Approches existantes de modélisation et d'analyse des réseaux de capteurs sans fil

2.2.1 Modélisation pour la vérification formelle

Plusieurs approches de vérification formelle sont utilisées. Cette section rappelle leurs spécificités respectives, en vue d'une identification de l'approche la plus adéquate dans le contexte de cette étude.

Vérification formelle

La vérification formelle est une technique qui consiste à prouver une propriété ou une spécification qui concerne le comportement d'un système. En cas de non respect de cette propriété, la vérification formelle fournit un contre-exemple. La complexité incessante de conception des systèmes embarqués tels que les réseaux de capteurs sans fil, l'avionique et les centrales nucléaires, accroît l'importance d'utilisation des techniques de vérification formelles. Bien que vaste, le domaine de la vérification se divise en deux branches principales : la preuve de théorèmes et le "model checking".

1. **La preuve de théorèmes** consiste à traiter le modèle formel comme un ensemble d'axiomes, et les propriétés à vérifier sur le modèle sont exprimées sous la forme de théorèmes. Elle applique des règles d'inférence et d'induction sur le modèle formel du système, en essayant de prouver le théorème cible [74].

   Les avantages de la preuve de théorèmes sont les suivants :

— elle est peu impactée par la taille de l'espace d'états du modèle exploité ;

— elle est capable de gérer les types de données abstraites et aussi l'induction, ce qui la rend en fait utilisable pour les systèmes d'états finis ou infinis, en raison de ces qualités, elle représente un outil efficace pour évaluer la sûreté des composants arithmétiques ;

— les théorèmes sont potentiellement plus expressifs que la logique temporelle.

La preuve des théorèmes a aussi des inconvénients :

— comme elle peut être appliquée dans un cadre général, avec éventuellement des modèles comportant un nombre infini d'états, des problèmes de décidabilité peuvent survenir ;

— le processus de vérification nécessite une grande expertise et de l'effort de la part de l'utilisateur car les outils qui supportent la preuve de théorèmes ne sont pas totalement automatiques. C'est qui la rend parfois difficile à adopter par l'industrie ;

— la preuve de théorèmes est destinée principalement à être utilisée pour la vérification de systèmes matériels.

Certains outils tels que Coq [21] et PVS [115], ont été développés pour automatiser le processus de la preuve de théorèmes, afin de résoudre les problèmes dans des délais raisonnables et d'améliorer les performances du processus de vérification (complexité, efficacité, etc.). Coq fournit un langage formel pour écrire des définitions mathématiques, des algorithmes et théorèmes, ainsi qu'un environnement pour le développement semi interactif de preuves de théorèmes. PVS est une plate-forme supportée par un langage de spécification et un outil de vérification basé sur une méthode de preuve des théorèmes. Il a été utilisé avec succès pour vérifier les conceptions matérielles [75] ou un protocole d'arrêt d'un système nucléaire [85]. Bien que le processus de preuve de théorèmes automatique a pu résoudre le problème du temps de vérification, son principal inconvénient est le besoin d'assistance de l'utilisateur pour guider la preuve. En outre, si une

preuve de théorème échoue, il est difficile d'obtenir des informations de diagnostic / débogage.

2. **Le "Model checking"** : Le model checking est une autre technique de vérification formelle de systèmes à états finis. Elle a été introduite dans [39] en 1983. Elle consiste à réaliser une exploration exhaustive de l'espace-états d'un modèle formel (construit sous la forme d'un automate à état finis ou produit à l'aide d'un langage de spécification dédié comme les réseaux de Petri [119]), afin de vérifier la satisfaction d'une exigence définie en logique temporelle. Dans le cas où la modélisation ne satisfait pas l'exigence, le model cheking fournit un contre-exemple qui illustre la violation de l'exigence. Le rôle de l'utilisateur est alors réduit à la modélisation formelle de son système et à l'énoncé de la propriété, ce qui demande une moindre expertise vis à vis la preuve de théorèmes.

   Le Model Checking est apparu pour résoudre le problème de vérification des programmes concurrents, où les erreurs de concurrence sont difficiles à trouver, car ils ne se reproduisent pas très souvent dans le programme. Au début, les preuves ont été réalisées à la main en utilisant le formalisme logique Floyd-Hoare. Dans [113] Owicki et Gries ont proposé un système de raisonnement sur les régions critiques conditionnelles.

   Les travaux de Pnueli [121], Owicki et Lamport [114] ont proposé l'utilisation de la logique temporelle pour la spécification de programmes concurrents. Bien qu'ils préconisaient toujours des preuves construites à la main, leurs travaux démontraient de manière convaincante que la logique temporelle était idéale pour exprimer des concepts tels que l'exclusion mutuelle, l'absence de blocage et de famine. Dans [40], Clarke et Emerson ont proposé un algorithme qui raisonne automatiquement sur les propriétés temporelles d'un système d'états finis en explorant l'espace d'état.

   Dans sa thèse de doctorat, McMillan a développé la technique du model cheking symbolique, ainsi que l'outil CMU-SMV [100] pour vérifier la satisfaction des propriétés temporelles dans le système modélisé. L'outil est basé sur la structure du diagramme de décision binaire (BDD) au lieu du graphe d'état et utilise un langage spécial pour représenter les modèles et la

logique temporelle pour les propriétés spécifiées. La méthode symbolique couplée aux BDD a montré des performances exceptionnelles, et la plupart des outils de vérification formels commerciaux exploitent ce principe.

Edmund M. Clarke explique dans [38] les avantages du model checking, nous citerons ici certains d'entre eux :

— méthode entièrement automatique : l'utilisateur d'un Model Checker n'a pas besoin de construire une preuve de correction. Il n'a qu'à décrire un circuit ou un programme à vérifier ainsi que la spécification à vérifier et le processus de vérification se fait alors de manière automatique ;

— rigoureux : lorsqu'une propriété spécifiée n'est pas satisfaite par le modèle étudié, le model checker fournit un contre-exemple qui montre où la propriété a été violée. Il s'agit de l'une des caractéristiques les plus importantes de l'utilisation de la technique de model checking ;

— puissant : utiliser la logique temporelle permet d'exprimer facilement des propriétés complexes et aide à raisonner sur des systèmes concurrents alors qu'il est très difficile de vérifier manuellement tous les cas possibles.

Néanmoins, la technique du model checking souffre du problème de l'explosion combinatoire de l'espace d'états du système, dû à la croissance exponentielle de la taille de l'espace d'états d'un système concurrent en fonction du nombre de processus et du nombre de composants par processus [42]. En effet, Un système avec n variables d'état booléennes a 2^n états possibles. Cette explosion se produit lors de la construction / manipulation de BDD. Bien que les BDD permettent une manipulation efficace de grands ensembles d'états, leur complexité théorique est toujours exponentielle en termes de nombre de variables booléennes gérées. Il est parfois impossible d'explorer l'ensemble de l'espace d'états avec des ressources limitées de temps et de mémoire [41].

Applications de la vérification formelle pour les réseaux de capteurs

A notre connaissance, la majorité des travaux de modélisation et de vérification des réseaux de capteurs se limitent à modéliser une instance de réseau ou un aspect de ce réseau (un protocole de communication ou un algorithme de routage) à l'aide d'un langage formel et à vérifier le modèle à l'aide d'un "model checker" supportant le langage utilisé. Leur objectif ainsi est soit de valider la capacité du model checker à modéliser des réseaux de capteurs sans fil, soit de vérifier qu'une instance de réseau de capteurs sans fil satisfait effectivement des propriétés données.

Dans [112], un algorithme appelé OGDC (Optimal Geographical Density Control) est modélisé, simulé et vérifié en "Real-Time Mode" [111], dont le but de maintenir une couverture optimale de la zone d'intérêt par des noeuds d'un réseau de capteurs sans fil. Sur un réseau de six noeuds, ils prouvent que toute la zone est couverte au premier passage de l'algorithme.

Watteyne et al. s'intéressent seulement à la modélisation de la couche MAC dans [154]. Ils cherchent à déterminer les durées dans les pires cas de différentes phases d'un protocole MAC temps réel conçu pour les réseaux de capteurs. Après une première validation par simulations, les auteurs vérifient la propriété à l'aide de l'environnement de modélisation et de validation formelle UPPAAL [149]. La vérification de la propriété est faite pour sept topologies contenant jusqu'à six noeuds. Pour chacune des topologies étudiées,ils ont vérifié que, quel que soit le nœud émetteur, le message arrive au puits à temps.

Tschirner et al [148] spécifient un réseau de capteurs sans fil biomédical en utilisant les automates temporisés [5] puis vérifient et simulent leur modèle avec UPPAAL [16]. L'étude de cas se base sur un émetteur/récepteur spécifique (Chipcon CC2420) et la vérification de propriétés qualitatives et quantitatives pour un réseau faisant de la collecte de données. Le focus est porté sur la couche MAC et routage. Les collisions sont prises en comptes, ainsi que les acquittements de paquets.

Mounier et al [107] modélisent un réseau de capteurs sans fil, qui détecte un nuage de pollution, en utilisant le langage IF [28]. Ils utilisent le model checker Kronos [27] pour comparer la consommation énergétique pire cas de deux

protocoles de routage différents : inondation dirigée et inondation contrôlée. La description de l'application est très abstraite (simple relai du signal environnemental). Ils modélisent donc deux configurations d'un réseau de capteurs sans fil, chacune avec un protocole de routage différent. Une configuration spécifie plusieurs aspects du réseau de capteurs sans fil : l'environnement, les capteurs, l'application, la couche MAC et la couche routage. Ensuite, ils calculent pour ces deux configurations la durée de vie dans le pire des cas du réseau.

Malgré le succès de la vérification formelle dans plusieurs domaines les tentatives d'application de cette techniques dans le domaine des réseaux de capteurs restent peu nombreuses à cause du problème de l'explosion combinatoire. En effet, lorsque le système est complexe et distribué à l'instar d'un réseau de capteurs, le nombre de comportements possibles devient énorme, ce qui rend alors impossible d'explorer en un temps raisonnable le modèle de comportements du système. De plus, les model checkers interviennent tard dans le cycle de développement, ce qui augmente le coût, en cas de détection d'erreur.

Ce que l'on peut retenir toutefois, c'est que malgré les limitations intrinsèques, dues à l'explosion combinatoire, la nature régulière des réseaux de capteurs permet de mettre en place des raisonnements semblables à de l'induction, aussi bien sur le plan structurel que comportemental, permettant d'exploiter une portion maniable d'un réseau de capteurs, et d'étendre ensuite les résultats vers une implémentation plus large. Ce sera d'ailleurs une des directions d'exploration menée dans le cadre de ce travail.

2.3 Modélisation à base d'automates d'états finis

La modélisation par automates d'états offre un cadre à la fois visuel, intuitif et rigoureux. Ces modèles permettent d'exprimer facilement des comportements dynamiques comme les séquences, la concurrence, et les interactions. Ils fournissent aussi un cadre adéquat pour appliquer les outils formels (e.g vérification et/ou synthèse). Par ailleurs, le désormais classique cadre formel des automates d'états finis a été exploité avec succès dans de nombreux travaux, dont [4], [47] et [48], ce qui justifie le choix de ce même cadre dans la suite de ce travail. Bien que des différences aient été soulignées dans la littérature, les termes "automates"

et "machines à états finis" sont utilisés de manière interchangeable dans la suite de ce document, après rappel de la définition formelle retenue. L'abréviation anglophone FSM [1] est retenue et utilisée dans la suite de ce document.

Dans la suite de ce document, $Bool$ dénote l'ensemble $\{0,1\}$ de valeurs Booléennes. Pour tout ensemble E de variables Booléennes, $Bool(E)$ dénote l'ensemble de vecteurs dont les composantes correspondent à la totalité des variables de E.

2.3.1 Définition : automate d'états finis

Dans ce document, nous utilisons la notion l'automate fini déterministe. Une machine à états finis déterministe est un 5-tuple

$$M = \langle Q, q_0, X, O, T \rangle$$

où :

— Q est l'ensemble fini d'états ;

— q_0 désigne l'état initial ;

— X est l'ensemble des variables d'entrée Booléennes ;

— O est l'ensemble des variables de sortie Booléennes ;

— T est l'ensemble $\{(q,x,o,q') : q,q' \in Q, i \in Bool(X)\}$ des transitions de M.

Chaque transition est symbolisée par une flèche dotée d'une étiquette de la forme g/a, où $g \in Bool$ est une expression portée par les variables O et doit être vraie pour que la transition puisse être franchie. L'étiquette g est dénommée *la garde* de la transition, tandis que $a \in BoolO$ est un vecteur de valeurs Booléennes affectées aux sorties O au moment où la transition est franchie. a est dite "l'action" associée d'une transition. Une transition (q,g,a,q') sera graphiquement notée $q \xrightarrow{g/a} q'$. Par la suite, nous utilisons ce niveau de description, avec une vue graphique, pour présenter nos modèles.

La composition parallèle synchrone [103] d'automates est un modèle de composition concurrente des machines d'état et est désignée par le symbole "||". Le

1. *Finite State Machine* : machine d'états finis

résultat d'une composition parallèle d'automates synchrones est lui-même un automate et cette opération conserve le déterminisme des automates composants (le résultat est déterministe si les composants sont déterministes). Pour deux automates $S_i = \langle Q_i, q_{i,0}, X_i, O_i, T_i \rangle$, avec i=1,2 et $Q_1 \cap Q_2 = \emptyset$. Leur composition synchrone est définie comme suit $S_1 \| S_2 = \langle Q_1 x Q_2, (q_{1,0}, q_{2,0}), X_1 \cup X_2, O_1 \cup O_2, T \rangle$ où $T = \left\{ (q_1, q_2) \xrightarrow{g_1 \wedge g_2 / a_1 \wedge a_2} (q_1', q_2') | q_1 \xrightarrow{g_1/a_1} q_1' \in T_1, q_2 \xrightarrow{g_2/a_2} q_2' \in T_2 \right\}$. L'état composé ($q_1$,$q_2$) est appelé un macro état, où q_1 et q_2 sont ses états composants. La figure 2.1 montre un exemple de composition synchrone.

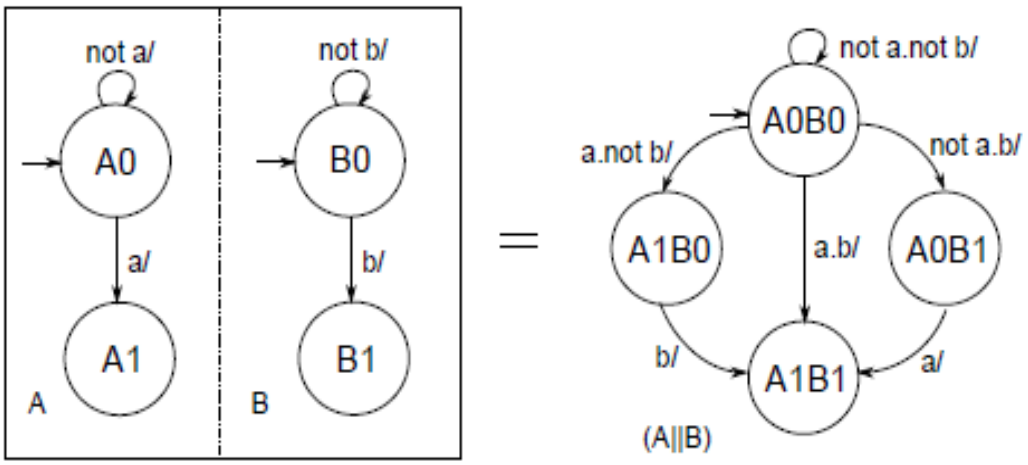

Figure 2.1 – Exemple de composition synchrone de deux automates

L'expression du produit synchrone d'automates d'états finis constitue le socle pour toute démarche de modélisation compositionnelle, permettant de modéliser des systèmes complexes par assemblage de composants simples, et en spécifiant éventuellement leurs interactions à l'aide d'associations entre entrées et sorties. La spécificité du modèle retenu et présenté ci-dessus réside dans le fait d'effectuer les "actions" (affectation des sorties), au moment où une transition est franchie, ce qui range ces modèles parmi les automates dits *de Mealy*.

Outre la capacité compositionnelle offerte par le produit synchrone, de nombreux langages de modélisation dits *synchrones* offrent des mécanismes d'expression supplémentaires, apportant des moyens raffinés pour la structuration, la généricité et la réutilisabilité. Le principal atout de ces langages réside dans leur

sémantique *formelle* équivalente à celle des automates d'états finis synchrones.

2.3.2 Langages synchrones à sémantique basée sur les automates

StateCharts [66] est probablement le premier langage formel proposé pour la conception du système réactif. Un tel langage à base des automates, supporte des descriptions modulaires et hiérarchiques du comportement du système, répondant aux descriptions multi-niveaux ainsi qu'à la concurrence. Sa sémantique a été définie à un niveau de détail suffisant dans [50]. Un autre avantage est qu'il existe un certain nombre d'outils commerciaux basés sur StateCharts, tels que StateMate et StateFlow. La plupart d'entre eux supportent la traduction de StateCharts vers des langages de descriptions de matériel, tels que VHDL, voire vers C. StateCharts a de nombreuses fonctionnalités de langages synchrones tels que le produit synchrone et le mécanisme de diffusion. Il est aussi à l'origine de certaines langages synchrones à base des automates comme Argos [95].

Argos est un langage synchrone basé sur des automates parallèles et hiérarchiques. Il prend son origine dans StateCharts, mais résout certains problèmes existants tels que ceux concernant les boucles de modularité et de causalité [64]. Sa sémantique est formalisée et compatible avec le point de vue synchrone adopté dans Esterel [63].

SyncCharts [34], créé par Charles Andre, est un autre exemple de langages synchrones à base d'automates. Il hérite de nombreuses fonctionnalités de StateCharts et Argos. SyncCharts se compose d'états et de transitions pour sa structure et de signaux pour sa dynamique. Tout programme SyncCharts peut être automatiquement traduit en un programme Esterel, ce qui permet aux utilisateurs de bénéficier de cet environnement logiciel développé pour la programmation synchrone.

Les *automates de mode* [96] sont une extension du modèle d'automates permettant de modéliser des "modes de fonctionnement" à travers une hiérarchisation des états. Ils supportent la programmation d'automates mixtes de style Argos avec une composition parallèle et hiérarchique de type StateCharts et des équations de flot de données de style Lustre étiquetées sur les états des automates. Certains autres langages synchrones utilisant les flux de données tel

que Signal [30] ont également été étendus avec des automates pour traiter des conceptions orientées contrôle.

Heptagon/BZR [47] est un autre langage synchrone de programmation qui permet de décrire un système à base de modèles mixant des automates et des équations de flot de données [44], avec une syntaxe permettant l'expression de structures de contrôle. Ce langage permet de modéliser des systèmes complexes (composés par des sous-systèmes) par la composition parallèle et hiérarchique des modèles des différents sous-systèmes. Les modèles évoluent en parallèle de manière synchrone de telle sorte qu'une réaction globale implique une réaction locale de chacun des sous-modèles. Semblable à Lustre, Heptagon structure les programmes en un ensemble de nœuds pour des raisons de passage à l'échelle et d'abstraction. Un nœud a un nom, un ensemble de flux d'entrée, un ensemble de flux de sortie et ainsi un corps définissant le comportement du composant. Son comportement de base est le suivant : à chaque étape, en fonction des entrées et des valeurs de l'état actuel, les équations associées à l'état actuel produisent des sorties et les conditions des transitions sont évaluées afin de déterminer l'état de la prochaine étape. Le corps des nœuds est exprimé sous la forme d'un ensemble d'équations qui déterminent les valeurs des sorties, en termes d'expressions sur des valeurs des flots d'entrée ou locaux. Ainsi, BZR [47] étend Heptagon par le mécanisme des contrats comportementaux pour définir des propriétés attendues pour le système modélisé. Bien qu'il est très utilisé pour la vérification et le contrôle des systèmes réactifs, ce langage n'est pas adapté pour la modélisation de nos capteurs. En fait, on a besoin de manipuler les variables d'état de modèles, alors que le compilateur Heptagon cache systématiquement la signification de l'état interne, c'est le parti pris des concepteurs de ce logiciel.

2.4 La théorie du contrôle par supervision

2.4.1 L'approche Ramadge et Wonham

Tout comme la vérification formelle, la théorie du contrôle par supervision [124] (SCT [2]) s'appuie elle aussi sur les modèles à base d'états/transitions.

2. de l'acronyme anglais *Supervisory Control Theory*

Cependant, plutôt que de vérifier si un système satisfait une exigence formellement spécifiée, cette technique renforce la satisfaction de cette exigence en générant si possible un *superviseur*. De nombreuses variantes ont été développées par la suite [12] [99], s'appuyant toujours sur principe de la SCT, mais s'appuyant sur des variantes différentes du modèle d'automate d'états finis. Ainsi, dans le cadre de la SCT, le modèle d'entrée est un automate d'états événementiel. L'approche de synthèse de contrôleurs discrets (DCS [3]) introduite par Marchand et al. [99] préconise quant à elle la modélisation par automates synchrones communicants *à variables* (état, entrées, sorties), avec comme points d'entrée au choix, les langages Signal [72], Targos [95] ou BZR [131].

L'architecture de contrôle s'en trouve elle aussi adaptée, pour prendre en considération cette spécificité, comme le montre la figure 2.2. Un *superviseur* SUP est généré automatiquement, à partir du modèle d'un *système* M et d'une exigence P à satisfaire sur le comportement de M.

Le calcul du superviseur dépend de la contrôlabilité de M. Pour réaliser ce contrôle, le superviseur doit agir sur la valeur de certaines variables d'entrée de M désignées comme étant contrôlables (dont la valeur peut être affectée par le superviseur). Ainsi, le vecteur d'entrées X de M doit être divisé en deux sous-ensembles disjoints : les entrées contrôlables X_c et les entrées incontrôlables X_{uc}, dont le comportement ne peut pas être influencé. Le superviseur SUP, s'il existe, contraint M en affectant des valeurs adéquates à ses entées *contrôlables* de manière à toujours satisfaire P.

L'exigence à renforcer par synthèse peut être soit de nature qualitative, soit quantitative.

2.4.2 Spécification des objectifs de contrôle

La figure 2.2 permet d'illustrer comment l'objectif est exprimé en fonction des signaux de sortie. Le superviseur est calculé par la technique définie par [99].

Exigences qualitatives : le contrôle logique

Ce type de contrôle peut renforcer trois types d'exigences :

3. *Discrete Controller Synthesis*

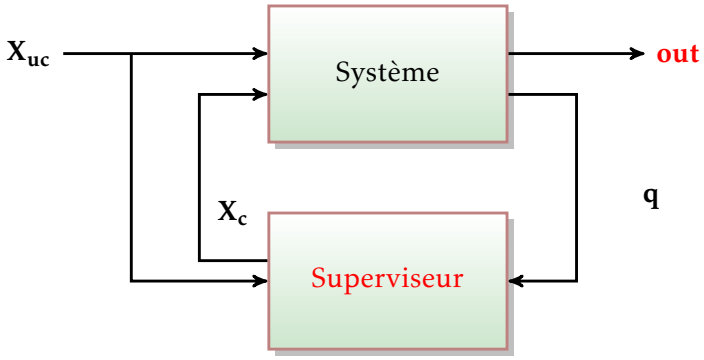

FIGURE 2.2 – Système contrôlé par un superviseur

— *invariance d'un sous ensemble d'états* : l'exigence P exprime un ensemble d'états au sein duquel M doit rester impérativement. La DCS calcule itérativement le plus grand ensemble d'états de M satisfaisant P, au sein duquel il est possible de rester indéfiniment en choisissant des valeurs X_c adéquates. Cet ensemble d'états est appelé *l'invariant sous contrôle* IUC de M pour P. L'ensemble $SUP \subseteq T$ des transitions de M menant à l'ensemble IUC constitue la solution du problème de contrôle. Cet ensemble solution est le superviseur renforçant P sur M. Le superviseur SUP n'est exploitable que si l'état initial de M est inclus dans IUC. Sinon, on considère que le problème de contrôle n'a pas de solution.

  Cette méthode de calcul garantit la production d'un superviseur dit *maximalement permissif* : un tel superviseur implémente la totalité des comportements satisfaisant P.

  Une fonction $S' = make\_invariant(S, P)$ synthétise un système contrôlé S' tel que les transitions contrôlables menant aux états $q_{i+1} \notin P$ sont inhibés, ainsi que ceux menant aux états à partir de lesquels une séquence de transitions incontrôlables peut mener à de tels états $q_{i+1} \notin E$;

— *atteignabilité* : L'exigence P exprime un ensemble d'états qui doivent rester atteignables. Une fonction $S' = keep\_reachable(S, P)$ qui synthétise et rend un système contrôlé S' dans lequel les transitions contrôlables entrant dans un sous-ensemble d'états d'où P n'est pas atteignable sont inhibées ;

— *attractivité* : L'exigence P exprime un ensemble d'états qui doivent être atteints en un nombre fini de pas, à partir de l'état initial, quel que soit le com-

portement incontrôlable subi par M. Une fonction $S' = make\_attractive(S, P)$ synthétise et un système contrôlé S' dont le FSM sous-jacent est un graphe d'états acyclique dont les transitions mènent inévitablement vers P.

Exigences quantitatives : le contrôle optimal

La théorie du contrôle telle que vue précédemment permet de calculer des contrôleurs lorsque l'objectif à assurer s'exprime de manière logique (invariance, atteignabilité, etc..). À ce niveau, une deuxième question peut se poser : quelle est la *meilleure* façon d'y arriver : la plus rapide, la moins coûteuse, etc. et ce malgré le comportement incontrôlable de l'environnement. La *synthèse optimale* peut répondre à cette questions. Le champ d'action de cette théorie couvre le contrôle de systèmes discrets où une performance optimale est souhaitée. Le principe est d'associer aux états et/ou transitions du système un coût ; le contrôle revient alors à choisir pour un état donné, les signaux d'entrée contrôlables faisant évoluer le système suivant une trajectoire de coût optimal. Le coût d'une trajectoire peut se définir de manières diverses, en agrégeant les coûts des états/transitions composant cette trajectoire selon des opérations simples telles que l'addition, le minimum et le maximum. A noter qu'en présence d'entrées incontrôlables, la synthèse optimale ne peut garantir de solution optimale mais souvent sous-optimale.

Ainsi le contrôle optimal peut être envisagé de deux manières :

— *optimisation à un pas* : Il s'agit d'optimiser localement sur chaque état, en choisissant parmi ses successeurs affichant des coûts optimaux [99]. Il peut y avoir plusieurs solutions équivalentes, car l'optimisation ne mène pas nécessairement au déterminisme. On peut noter que ceci nous donne seulement un choix d'une étape i.e. un optimum local et non un optimum global sur tous les comportements ;

— *optimisation sur les chemins* : l'objectif de contrôle optimal sur les chemins est d'optimiser les coûts accumulés tout au long des chemins menant de l'état initial vers un ou plusieurs états *cible*. Une mesure monotone et décroissante est d'abord calculée sur l'ensemble des trajectoires menant vers les états cibles. Affectée à chaque état, cette mesure exprime

le meilleur coût réalisable, malgré les valeurs des entrées incontrôlables, pour atteindre les cibles. Cette mesure est utilisée ensuite comme critère d'optimisation à un pas.

2.5 Expression formelle d'exigences qualitatives

L'identification des états P de M, préalablement au calcul d'un superviseur ne se fait pas de manière énumérative. Ces états correspondent à une configuration interne induite par une exigence de valeurs et/ou de séquences de valeurs des variables de sortie de M, exigence décrite en logique temporelle. Le standard IEEE PSL [77] étend les logiques linéaire LTL [120] et arborescente CTL* [62] et offre l'approche de spécification la plus mature aujourd'hui, supportée d'ailleurs par de nombreux outils industriels. La table 2.1 illustre la sémantique de quelques opérateurs de PSL , en s'appuyant sur la syntaxe LTL.

Tableau 2.1 – la sémantique LTL des quelques opérateurs de PSL

Syntaxe de PSL	Syntaxe de LTL
$next\ p$	$X\ p$
$always\ p$	$G\ p$
$never\ \ p$	$G\ \neg p$
$eventually\ p$	$F\ p$
$p\ until!\ q$	$p\ U\ q$
$p\ until\ q$	$(p\ U\ q)\ or\ G\ p$
$p\ before\ q$	$(\neg q\ U\ (p \wedge \neg q)) \vee G(\neg q)$

Malheureusement PSL est peu supporté par certains outils expérimentaux, dont ceux utilisés dans ce travail. Alternativement à la spécification logique et pour palier ce manque de support de PSL, l'expression de P peut se faire de manière *opérationnelle*, à travars l'écriture d'un observateur, qui n'est rien d'autre qu'un modèle FSM particulier, capable d'associer certaines séquences observées mais non désirées à l'activation d'un état dit d'erreur. Composé avec M, l'observateur identifie P comme l'ensemble des états globaux issus du produit cartésien, qui ne contiennent pas l'état d'erreur parmi leurs composantes.

A noter qu'un sous-ensemble dit "simple" de PSL [54] a été identifié. Les formules PSL appartenant à ce sous-ensemble peuvent être systématiquement traduites en observateurs. La restriction "au sous-ensemble simple" de PSL [54] assure que les propriétés de ce sous-ensemble peuvent être évaluées tant par simulation que par vérification formelle. Le mécanisme qui rend ce possible est la génération automatique d'observateurs équivalents, de sous-ensemble PSL formulæ.

Les observateurs utilisés dans ce travail expriment uniquement des assertions de sécurité invariantes qui peuvent être construites soit manuellement soit automatiquement [109] à partir du sous-ensemble simple de PSL.

Dans la suite, sans perte de généralité, nous nous focalisons uniquement sur des assertions de type *"always/never p"*, où p désigne une expression booléenne atomique portée par les variables de M.

2.6 Exemples d'application de la SCT dans les systèmes distribués

Plusieurs travaux récents se sont focalisés sur l'application de la théorie du contrôle par supervision (SCT) dans le domaine de réseaux de capteurs ainsi que dans les systèmes distribués en général.

La technique de SCT est combinée avec l'unité d'enregistrement de données dans [46] pour gérer les informations collectées par un réseau de capteurs et prendre des décisions en se basant sur ces informations. Dans [57], la coordination d'un réseau de capteurs hétérogène contenant tant des nœuds stationnaires que des nœuds mobiles est réalisée par un contrôleur d'événements discrets (DEC). Ce DEC est *centralisé*, il séquence les tâches les plus appropriées et assigne les ressources de capteurs selon la perception actuelle de l'environnement.

Une méthode d'implantation de contrôle global de ressources des plates-formes matérielles, tout en recherchant une solution à la gestion de l'énergie au niveau des noeuds de réseaux de capteurs sans fil, a été aussi proposée dans [19]. Ceci est basé sur un principe de synthèse de contrôleur, qui cherche à assurer des propriétés globales sur un ensemble de modèles, sans modifier leurs

comportements intrinsèques. Dans cette étude de cas, le contrôle est appliqué sur des machines de Mealy modélisant les pilotes de différents périphériques. Le contrôleur à construire permet ainsi d'empêcher les changements de modes de fonctionnement des périphériques qui mènent la plate-forme dans un état global violant des propriétés spécifiées. ces propriétés peuvent être par exemple des exclusions d'accès à des bus, ou assurent une stratégie de gestion de l'énergie dans le but de réduire les pics de consommation.

Le contrôle distribué des systèmes à événement discrets en utilisant la SCT est investigué dans [32], En se focalisant sur comment synthétiser des contrôleurs locaux pour des sous-systèmes individuels, tel que le comportement contrôlé résultant reste le même que celui obtenu initialement par la supervision globale.

Dans [84] une proposition de contrôle des systèmes distribués modélisés par des machines à états communicantes (CFSM) avec des FIFOs non bornées fiables est présentée ; un problème de contrôle distribué d'évitement d'état est formulé, et il est montré que l'existence d'une solution pour ce problème est non décidable.

Une procédure de calcul pour réaliser un superviseur distribué est présentée dans [146]. Au lieu de calculer un modèle centralisé du système entier, ils calculent pour chaque composant local une abstraction du système basé sur une bisimulation faible. Ainsi un superviseur local pour chaque composant est calculé en se basant sur le modèle de composant et le modèle abstrait de système, en utilisant la SCT. La collecte de tous les superviseurs locaux résultants forme un superviseur distribué. La correction et la perfection de cette solution distribuée dépendent fortement de la qualité de l'abstraction de système.

2.7 Synthèse et discussion

Ce chapitre présente les modèles, langages et outils existants pour la conception des systèmes embarqués et plus précisément pour les RCSFS, qui sont des systèmes embarqués réactifs.

La première section de ce chapitre a été consacrée à la description des méthodes formelles de conception assistées qui aident le concepteur à vérifier sa conception avant de la mettre en œuvre, à savoir :

(1) la technique de preuve de théorèmes, étant donné qu'elle est manuelle, elle dépend fortement de l'expérience de l'utilisateur et de ses compétences mathématiques. Par ailleurs, il arrive souvent que la correction manuelle d'une faute engendre une ou plusieurs autres fautes, provoquant un cercle vicieux difficile à maîtriser.

(2) la vérification par model checking est très utile pour la vérification de systèmes de moyenne à grande taille, car il s'agit d'un outil entièrement automatique dont les résultats sont fiables, à condition qu'il n'explose pas. Quand il fournit un contre exemple, il peut inspirer le concepteur comment corriger l'erreur. Toutes ces méthodes de vérification formelles, qu'elles soient manuelles ou automatiques, sont limitées à la recherche des erreurs de conception.

Nous avons décrit par la suite les formalismes de modélisation basés sur des automates : STATECHARTS et langages synchrones. Ces formalismes conviennent très bien à la modélisation des comportements dynamiques qui caractérisent les RCSFs comme expliqué en 2.2, ainsi qu'elles offrent un cadre adéquat pour appliquer les techniques de vérification et/ou de synthèse.

Pour concevoir un contrôleur sûr pour la gestion de la reconfiguration des réseaux de capteurs, nous avons présenté la technique de la synthèse de contrôleurs discrets (DCS) issue de la théorie de contrôle par supervision. Contrairement au Model checking, qui fournit des indications ou un diagnostic sur la conception d'origine lorsqu'un bug est détecté et demande au concepteur de revenir à la conception et de le modifier avant de procéder à la vérification, DCS est plus constructif puisqu'elle est capable de synthétiser directement un contrôleur sûr de manière automatique. En effet, DCS produit un système restreint (système non contrôlé + contrôleur) respectant la spécification de contrôle associée (une exigence comportementale).

Enfin, nous avons présenté quelques travaux existants qui appliquent la synthèse de contrôleur discrets pour les systèmes informatiques. On observe que le contrôle discret est rarement appliqué dans le domaine des RCSFs, c'est pour cela que notre contribution peut être considérée comme une prise de risque, par son positionnement en rupture vis-à-vis des approches largement adoptées par la grande communauté de chercheurs dans ce domaine. Cette rupture est la conséquence d'un souhait d'apporter des garanties de correction multi-échelle

(locale et globale), lors de la conception d'un réseau de capteurs, et avant son déploiement. En se basant sur les modèles et les méthodes présentés dans ce chapitre, Les chapitres suivants présentent nos contributions.

Deuxième partie

Contributions

Chapitre 3

Génération automatique d'un ordonnancement des activités de noeuds

3.1 Introduction

Le comportement distribué des réseaux de capteurs reste toujours centré sur les deux principaux objectifs : la surveillance et l'acheminement des informations détectées. Dans ce chapitre, nous introduisons un nouvel aspect intéressant sur le plan de la conception : en partant d'une exigence spécifiée initialement, mettons en œuvre une synergie entre un ensemble des nœuds (cluster) basée sur une interaction adéquate. Ceci est réalisé par la génération automatique de code (correct par construction) et sa distribution par la suite en s'appuyant sur la théorie de contrôle par supervision. La technique de synthèse de contrôleurs discrets (DCS) est une application de ce cadre théorique. Dans ce chapitre, nous montrons comment la DCS peut être utilisée pour l'ordonnancement intra-cluster (groupe redondant de noeuds) avec des spécifications exprimant l'exclusion mutuelle lors de l'activation d'un nœud au sein d'un cluster, essentielle pour économiser l'énergie.

Ce chapitre est organisé comme suit : après l'introduction, nous présentons dans la section 2 une synthèse des approches existantes d'ordonnancement d'ac-

tivité des noeuds dans les réseaux de capteurs. Une discussion concernant les insuffisances de ces travaux justifie la méthode proposée. La section 3 détaille notre première contribution dans cette thèse où, par application de la DCS, on génère un ordonnanceur global qui garantit une condition d'exclusion mutuelle lors de l'activation de noeuds d'un même cluster : un seul capteur actif à un moment donné au sein d'un cluster. L'ordonnanceur global obtenu est incompatible avec le fonctionnement distribué, inhérent à un réseau de capteurs. En effet, les ordonnanceurs globaux ne disposent pas d'entrées/sorties. Ils sont représentés par une équation Booléenne encodant toutes les solutions de contrôle acceptables. Ceci nous amène à proposer une manière appropriée de distribution locale de cet ordonnancement global. Nous montrons, dans la section 4, comment les limitations de complexité théoriques de la DCS peuvent être traitées efficacement en bénéficiant des similitudes structurelles des noeuds composant le réseau. Nous déduisons ainsi un contrôleur distribué générique pour n nœuds avec ($n \geq 2$). Pour valider notre approche, les résultats de la simulation par les outils MathWork (Simulink et Stateflow) d'un réseau de capteurs sont présentés dans la section 5. Enfin, la section 6 conclut le chapitre.

3.2 Approches existantes d'ordonnancement d'activité des noeuds

La majorité des travaux qui s'intéressent à l'économie d'énergie dans les réseaux de capteurs montrent qu'une radio inexploitée consomme autant d'énergie que la réception. En outre, ils ont montré aussi que ce n'est pas le nombre de paquets transmis, mais plus tôt la durée de la période de transmission qui affecte le plus les réserves énergétiques des noeuds. Ainsi, il vaut mieux éteindre complètement les noeuds pour préserver leurs batteries. Cependant, cette pratique ne doit pas, par conséquent, affecter le bon fonctionnement du réseau.

L'ordonnancement d'activité des nœuds dans les réseaux de capteurs tout en préservant la couverture de la surface cible peut se faire de diverses façons. Nous distinguons ici les approches centralisées, des approches distribuées.

3.2.1 Les approches centralisées

Ces approches bénéficient d'une entité centrale ayant vue sur tout le réseau. Comme le nombre de capteurs déployés est, généralement, supérieur à l'optimum requis pour effectuer la tâche de surveillance, il est judicieux de diviser les nœuds en ensembles disjoints de sorte que chaque ensemble puisse effectuer individuellement les tâches de surveillance de zone. Ces ensembles sont ensuite activés successivement et, alors que l'ensemble actuel est actif, tous les autres noeuds sont dans un mode veille à basse consommation énergétique. Le but des ces approches centralisées est de déterminer un nombre maximal d'ensembles disjoints car cela a un impact direct sur sur la conservation des ressources énergétiques des capteurs, ainsi que sur la prolongation de la durée de vie du réseau. Ces algorithmes nécessitent des informations globales (telles que les zones de perceptions de capteurs, leurs emplacements, ainsi que leurs résidus énergétiques). Quelques contributions centralisées existantes traitant le problème de couverture et d'ordonnancement de réseaux de capteurs sont présentées.

Dans [144], Les nœuds envoient leurs positions à l'unité centrale, qui à son tour divise la zone cible en domaines (fields), et ordonne les nœuds dans des ensembles (covers), mutuellement exclusifs. Chaque domaine est définit comme un ensemble de points tel que, chaque deux points appartenant au même domaine doivent être couverts par le même ensemble de noeuds. Le but de cette approche est de maximiser le nombre de «covers». Les auteurs annoncent que c'est un problème NP-complet, et ont proposé une heuristique qui sélectionne des ensembles mutuellement exclusifs de nœuds, où les membres de chacun de ces ensembles couvrent complètement la zone surveillée. Les intervalles d'activité sont les mêmes pour tous les ensembles, et un seul des ensembles est active à tout moment. Cette heuristique indique une complexité de l'ordre de $O(n^2)$, où n est le nombre total des nœuds déployés.

Utilisant une approche similaire, Cardei et al [33] proposent un autre algorithme d'ordonnancement centralisé. Les auteurs modélisent le problème de couverture comme étant un nombre maximal d'ensembles disjoints dominants dans un graphe non-orienté G (V, E) où V est l'ensemble de sommets repré-

sentant les noeuds du réseau et E est l'ensemble des arcs reliant ces noeuds. $uv \in E$ si et seulement si u est dans la portée de détection de v et vice-versa. Un mécanisme de graphe coloré est proposé pour calculer le nombre maximal des ensembles disjoints dominants. Cet algorithme d'approximation comporte 2 phases. Au cours de la 1ère phase, tous les sommets sont coloriés à l'aide de l'algorithme de coloriage séquentiel. Le coloriage de sommets est réalisé un par un, en utilisant le minimum de couleurs différentes possibles à celles de voisins. Au cours de la 2éme phase, Après le coloriage de tous les sommets, une heuristique de complexité égale à $O(n^3)$ est utilisée pour construire les ensembles disjoints dominants.

Un autre travail d'ordonnancement centralisé (ESSM) plus récent est proposé par Wan et al dans [153], basé sur la similitude des données mesurées par les noeuds déployés (i.e on suppose que les nœuds très proches détectent les mêmes données). Selon la similitude des données collectées, la station de base classe les différents nœuds du réseau et les nœuds redondants peuvent alors être sélectionnés en définissant une fonction de corrélation issue de la théorie Floue (FuZZY Theory). Par conséquent, ESSM peut activer un nombre minimum de nœuds dans le but de prolonger la durée de vie du réseau. Cependant, pour les applications avec une distribution clairsemée de nœuds ou une faible corrélation spatio-temporelle des données, il sera très difficile pour ESSM d'évaluer les différences entre les nœuds par leur similitude de données.

Les données détectées des nœuds sont généralement corrélées dans des réseaux de capteurs sans fil denses, et la problématique de sélection de nœuds actifs vise généralement à sélectionner un nombre minimum de nœuds pour fournir les services de données requis ainsi qu'augmenter efficacement la durée de vie du réseau. Dans cette optique, les auteurs de [35] proposent trois algorithmes : l'algorithme CSB (Cover Set Balance) pour choisir un ensemble de nœuds actifs avec le tuple partiellement ordonné (plage de couverture des données, énergie résiduelle), l'algorithme CNSC (Correlated Node Set Computing) pour trouver l'ensemble de nœuds corrélé pour un nœud donné ainsi que l'algorithme HREF (High Residual Energy First) pour réduire davantage le nombre de noeuds actifs.

Une méthode d'ordonnancement appelée CPCSS est proposée dans [138], basée sur un modèle de similarité pour calculer le degré de corrélation entre

les données captés par les noeuds et partitionner ainsi ces nœuds en différentes classes. Tout d'abord, le degré de redondance d'un nœud est calculé en fonction de sa zone de détection couverte par les noeuds voisins. Ensuite, une méthode centralisée basée sur la partition de la zone cible est conçue pour obtenir l'ensemble minimum approximatif de nœuds actifs, qui peut conserver la couverture suffisante de la région cible et assurer la connectivité du réseau en même temps.

Bien qu'elle est facile à concevoir et à mettre en œuvre, l'un des problèmes majeurs de l'approche centralisée est qu'elle présente un seul point de défaillance. Tout le réseau deviendra paralysé si un problème survient au nœud central : Le paquet de données ne peut pas être transmis ni traité lorsqu'un nouvel événement est détecté. C'est pour cette raison que plusieurs solutions d'ordonnancement distribuées sont apparues permettant le passage à l'échelle. Ainsi, au lieu d'accumuler la complexité exponentielle dans une seule entité, on la répartit sur tous les noeuds du réseau.

3.2.2 Les approches distribuées

Obtenir un comportement global cohérent à partir de décisions prises localement est l'objectif principal d'une approche distribuée. Chaque nœud décide de sa propre activité en se basant sur l'observation de ses propres voisins. Il n'existe aucune hiérarchie ni infrastructure. Le comportement de chaque nœud est devenu influencé uniquement que par ceux de ses voisins directs. Ce qui nous permet de passer facilement à l'échelle et obtenir en conséquences des solutions plus robustes. Dans cette partie, nous présentons quelques exemples d'algorithmes distribués.

Un algorithme simple d'ordonnancement distribué appelé "Peas" a été proposé dans [160]. Initialement, tous les nœuds sont en mode inactif. Périodiquement, un nœud quitte ce mode et diffuse un message de sondage dans une zone de communication de rayon R. Si aucune réponse n'est reçue durant une période de temps définie, il reste actif jusqu'à ce qu'il épuise sa batterie. Sinon, un voisin qui se trouve à une distance inférieure à P répond à l'émetteur lui permettant de repasser en mode veille. La valeur P est choisie en fonction de la densité des nœuds désirée (nombre de capteurs par unité de surface), alors que la période

pour qu'un nœud s'active et émet le message dépend de la période de détection. Cette approche distribuée est tolérante aux fautes (perte de messages, panne d'un nœud, etc..). Par contre, il ne garantit pas la couverture totale de la zone d'intérêt.

Un autre mécanisme de couverture distribué est proposé par Tian et Georganas dans [147]. Le fonctionnement de cet algorithme est divisé en cycles, où chaque cycle commence par une phase d'auto-ordonnancement suivie d'une phase de détection. Au cours de la première phase, les nœuds vérifient la règle d'éligibilité d'être inactif. Un nœud est dit éligible si ses nœuds voisins (sponsors) surveillent sa région de détection. Pour obtenir les informations de ses voisins, chaque nœud diffuse un message d'avertissement au début de chaque cycle. Ce message contient l'identité et la localisation du nœud. Si plusieurs nœuds voisins prennent la décision d'être en veille simultanément alors il se peut qu'il y ait occurrence de points "noirs" (zones non couvertes). Pour éviter ce problème, un back-off est utilisé où chaque nœud commence l'évaluation de la règle après un temps aléatoire, puis il diffuse un message ADV pour annoncer s'il est prêt à se désactiver. Avant de se désactiver, le nœud attend encore une période TW pour écouter les mises à jour de ses voisins. Les résultats des simulations ont montré que cet algorithme permet de diminuer considérablement le nombre de nœuds actifs et d'augmenter la durée de vie d'une moyenne de 1,7 fois, tout en conservant une couverture totale du réseau.

Dans [166], Zairi et al proposent l'algorithme ERGS comme une amélioration du protocole [147]. L'algorithme ERGS repose sur l'idée d'exploiter l'énergie restante pour décider quel nœud doit entrer dans l'état de veille. La première caractéristique principale de l'algorithme ERGS consiste à appliquer un principe d'équité en équilibrant l'énergie restante des nœuds. La deuxième caractéristique consiste à éviter les phases de négociation, car la décision d'entrer dans l'état de veille utilise une priorité calculée basée sur une connaissance de voisinage à un saut. Cela contribue à prolonger la durée de vie du réseau, par la réduction des échanges de messages d'une part, et par l'évitement des points noirs (points non couverts) d'autres part.

CKN [108] est un algorithme distribué qui permet de désactiver les nœuds redondants tout en satisfaisant la K-connectivité. Si un nœud a moins que k-voisins,

aucun de ses voisins ne se désactive pas, et s'il a plus que K-voisins, au moins K d'entre eux doivent rester actifs. Le paramètre k utilisé par cet algorithme désigne le nombre minimal exigé de voisins actif pour chaque nœud. Le choix de nœuds actifs est complètement basé sur les rangs qui sont aléatoirement obtenus au début de l'exécution de l'algorithme (CKN) dans chaque période, et l'ensemble des nœuds actifs change d'une période à une autre.

CKN ne peut pas assurer une consommation énergétique uniforme, ce pour cela, un nouvel algorithme d'ordonnancement (EC-CKN), basé sur CKN, est développé, "Energy Consumed Uniformly Connected K-Neighborhood algorithm" [165]. Dans (EC-CKN) l'information de résidus énergétiques des nœuds est un paramètre important pour déterminer les nœuds qui doivent être actifs. Un nœud Su diffuse son résidu énergétique courant $Eranku$ puis il calcule un ensemble Eu des voisins ayant Erank > Eranku. Avant qu'il choisisse de se désactiver, Su doit être sûr que tous les nœuds de l'ensemble Eu sont connectés, Erank > Eranku et chacun de ses voisins a au moins k-voisins de l'ensemble Eu. Ces exigences assurent que si un nœud a moins que k voisins, aucun de ses voisins ne se désactive pas, et s'il a plus que k voisins, au moins k d'entre eux doivent rester actifs. Ces exigences sont faciles à garder en calculant localement les informations des voisins à 2 sauts. Cependant, ce protocole exige un niveau élevé de synchronisation et de communication entre les noeuds voisins, sans prendre en considération les limites qui caractérisent les réseaux de capteurs tels que la faible capacité de stockage et de calcul.

Un autre algorithme d'ordonnancement distribué proposé récemment dans [76], appelé DiLCO (Distributed lifetime coverage optimization protocol in wireless sensor networks), qui maintient la couverture et améliore la durée de vie d'un réseau de capteurs sans fil. Tout d'abord, la zone d'intérêt est divisée en sous-régions en utilisant la méthode classique de diviser pour mieux régner. Dans une deuxième étape, une méthode distribuée est appliquée dans chaque sous-région pour optimiser la couverture et les performances de la durée de vie du réseau. Dans une sous-région, DiLCO consiste à choisir un nœud leader qui va ensuite effectuer un ordonnancement d'activité des autres noeuds de sa sous-région. Les défis incluent comment sélectionner le leader le plus efficace dans chaque sous-région et le meilleur ensemble représentatif de nœuds actifs pour assurer

un niveau élevé de couverture. L'impact du nombre de sous-régions choisies pour subdiviser la zone d'intérêt est aussi étudié dans ce travail, en considérant différentes tailles du réseau. Les expériences montrent que l'augmentation du nombre de sous-régions améliore la durée de vie. Plus il y a de sous-régions, plus le réseau est robuste contre la déconnexion aléatoire résultante de nœuds "morts". Cependant, les auteurs n'ont pas trouvé le nombre optimal de sous-régions.

Dans [106] Mostafaei et al. proposent l'algorithme PCLA qui se concentre sur le problème de la couverture partielle, en ciblant des scénarios dans lesquels la surveillance continue d'une partie limitée de la zone d'intérêt est suffisante. PCLA repose sur les automates d'apprentissage "Learning Automata" pour ordonnancer les noeuds (mode actif ou veille). Il vise à minimiser le nombre de noeuds à activer de sorte qu'une partie désirée de la zone d'intérêt soit couverte et que la connectivité entre les capteurs soit préservée.

3.3 Discussion et objectifs visés

Différents points se dégagent à partir de la synthèse de travaux de couverture et d'ordonnancement qui ont été présentés à la section précédente :

— les algorithmes centralisés ne sont pas adaptés aux réseaux de capteurs puisqu'ils souffrent généralement des problèmes de passage l'échelle, de point de défaillance unique et de manque de robustesse. Cette dernière est particulièrement pertinente dans le contexte des réseaux de capteurs puisque les nœuds sont déployés dans des environnements hostiles et sont sujets à des pannes fréquentes ;

— la minimisation du nombre de noeuds actifs nécessaire pour la couverture totale de la zone cible est difficile à réaliser, et n'est pas implémentée par la plupart des approches existantes, qui cherchent principalement à assurer une couverture totale sans se soucier d'éviter les redondances dans les noeuds actifs ;

— beaucoup de travaux de recherche se sont intéressés à l'ordonnancement distribué de noeuds autonomes. Ces différentes approches ont été proposées et évaluées soit à travers des expérimentations ou des simulations.

Cependant, nous remarquons que toutes ces approches d'ordonnancement distribué requièrent la synchronisation des ordonnanceurs locaux et le contrôle de leurs actions, et ces aspects ont été largement étudiés en théorie de contrôle discret.

Notre approche a été développée pour répondre aux manques de travaux existants. Nos objectifs ont été ainsi définis en fonction de ces besoins :

— contribuer à la couverture totale de la surface de surveillance cible, tout en désactivant les noeuds redondants. Afin d'assurer un fonctionnement correct du réseau, la zone surveillée par les noeuds actifs doit être égale à celle surveillée par les noeuds déployés initialement. Ainsi, l'occurrence de tout phénomène surveillé reste toujours détectable ;

— minimiser le nombre de noeuds actifs. L'objectif est d'activer le nombre minimal des noeuds qui garantissent la couverture de la zone cible. Ce critère de minimisation est garanti sur un sous-ensemble de noeuds du réseau qui peuvent être considérés comme étant redondants, car surveillant la même zone. Ce sous-ensemble est communément connu sous la désignation de "*cluster*". Par conséquent les réserves énergétiques d'un nombre maximum de noeuds redondants sont préservées afin de prolonger la durée de vie du réseau ;

— proposer une méthodologie basée sur des outils formels afin de modéliser, concevoir et générer automatiquement un ordonnanceur local correct pour chaque nœud du réseau en appliquant la technique de la DCS. Nous nous intéressons à la DCS, parmi les diverses méthodes formelles existantes, en raison de sa capacité à construire des conceptions correctes par construction.

La prochaine section présente l'approche de conception de l'ordonnancement visant les objectifs sus-cités à l'échelle d'un *cluster*.

3.4 Méthode de conception pour la génération automatique d'ordonnancement intra-cluster

3.4.1 Hypothèses

Avant la présentation des différentes étapes de notre méthode d'ordonnancement, nous introduisons les hypothèses envisagées pour son développement :

— les nœuds sont déployés de manière aléatoire dans la zone d'intérêt et sont connectés à une station de base unique ;

— les nœuds initialement déployés couvrent entièrement et de manière redondante la zone d'intérêt ;

— tous les nœuds déployés sont homogènes, ayant les mêmes caractéristiques (plages de détection et de communication, niveau d'énergie initial, etc..) ;

— les nœuds sont numérotés avec des indices distincts et chaque nœud connaît son index.

3.4.2 Approche de clustering adoptée

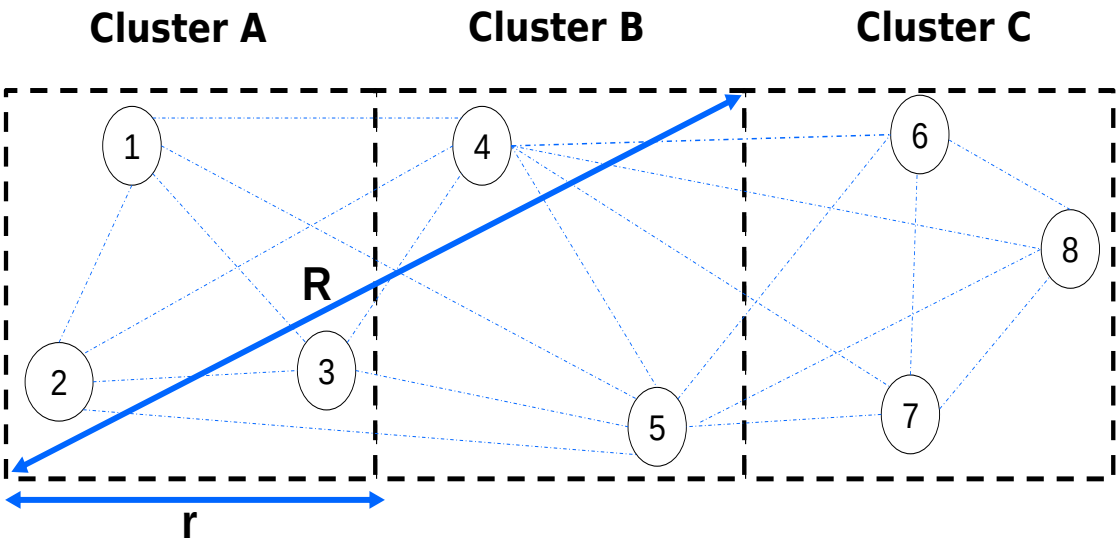

Figure 3.1 – Approche du clustering adoptée

Le regroupement des nœuds en *clusters* est l'une des techniques les plus adoptées dans le domaine des réseaux de capteurs pour satisfaire les objectifs de passage à l'échelle et prolonger la durée de vie du réseau. Dans ce sens, en s'appuyant sur les principes de l'algorithme GAF [158], on propose de diviser la surface à surveiller en petites zones virtuelles. Chaque nœud doit appartenir à une zone unique en fonction de ses informations de localisation. Comme illustré sur la figure 3.1, les nœuds équivalents (i.e. situés dans la même petite zone virtuelle) forment ainsi un cluster. Chaque nœud est capable de communiquer avec ses voisins de même cluster ainsi que tous les autres nœuds du cluster adjacent. En d'autres termes, Pour deux clusters A et B adjacents, dans le but de préserver la connectivité de l'ensemble du réseau, tous les nœuds de A doivent pouvoir communiquer avec tous les nœuds de B et vice-versa. On désigne par R la portée radio ainsi que par r la plage de transmission des nœuds. La taille d'un cluster est r * r. Afin de garantir la connectivité du réseau, r devrait satisfaire l'équation suivante :

$$r \leq R\sqrt{5} \tag{1}$$

Les nœuds d'un même cluster sont connectés les uns aux autres via des liaisons point à point et ces liens sont bidirectionnels.

La maîtrise de la réduction de la consommation d'énergie s'appuie sur l'alternance, au niveau de chaque nœud entre les modes veille et actif. Cette alternance est loin d'être arbitraire, car pour garantir la satisfaction de l'objectif global du réseau, une synergie doit exister entre les différents nœuds. En d'autre termes, il est nécessaire d'être localement en veille le plus longtemps possible, mais actif assez souvent pour ne pas perdre une mesure importante, ou pour relayer la transmission d'une mesure importante effectuée par un autre nœud. Ce principe est appliqué ainsi à notre structure du réseau tel que pour chaque cluster composé par un ensemble des noeuds équivalents et redondants, on garde un seul nœud en mode actif, alors que ses voisins du même cluster doivent entrer obligatoirement dans leurs états de veille réduisant ainsi l'utilisation redondante des ressources énergétiques.

3.4.3 Étapes suivies pour la génération d'un ordonnanceur du réseau de capteurs

Dans cette section, nous fournissons les étapes à suivre pour obtenir un ordonnancement automatique d'activités d'un nombre réduit de noeuds appartenant à un même cluster.

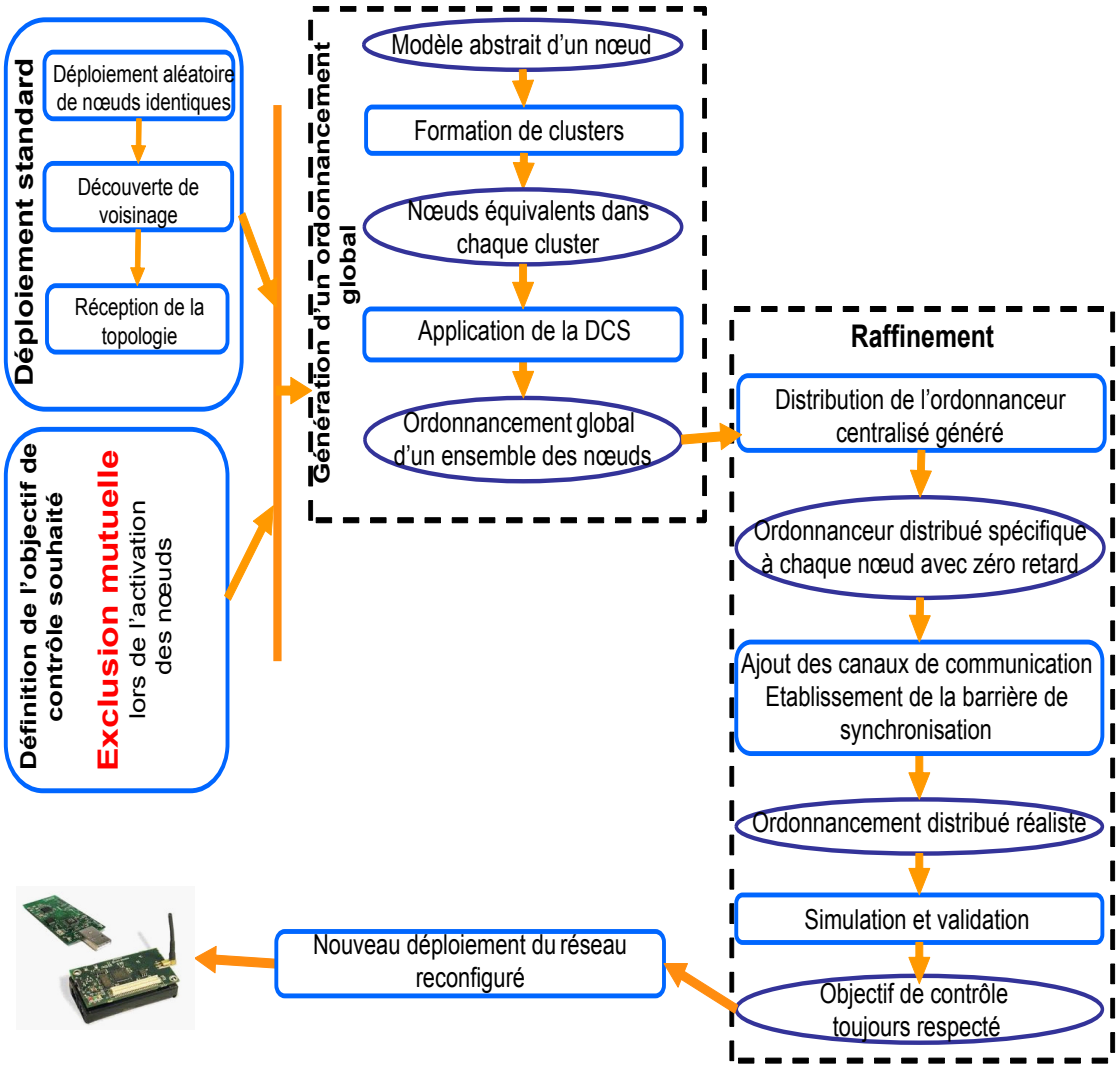

FIGURE 3.2 – Méthode proposée pour la conception et l'implémentation d'un ordonnancement automatique d'un ensemble redondant des noeuds

L'approche est illustrée à la figure 3.2. Après la réception de la topologie du réseau et la répartition des noeuds dans des clusters redondants, on définit une

exigence à respecter (objectif de contrôle global). Cette exigence est l'exclusion mutuelle lors de l'activation de noeuds d'un même cluster. Ensuite, chaque nœud est modélisé par un modèle formel séquentiel, comme illustré dans la figure 3.3, représentant le comportement logique du nœud et faisant abstraction des dispositifs physiques. Cette modélisation formelle favorise ainsi l'application de la technique de la DCS afin de générer automatiquement un ordonnanceur global correct par construction qui garantit le respect de l'exigence d'exclusion mutuelle définie au début.

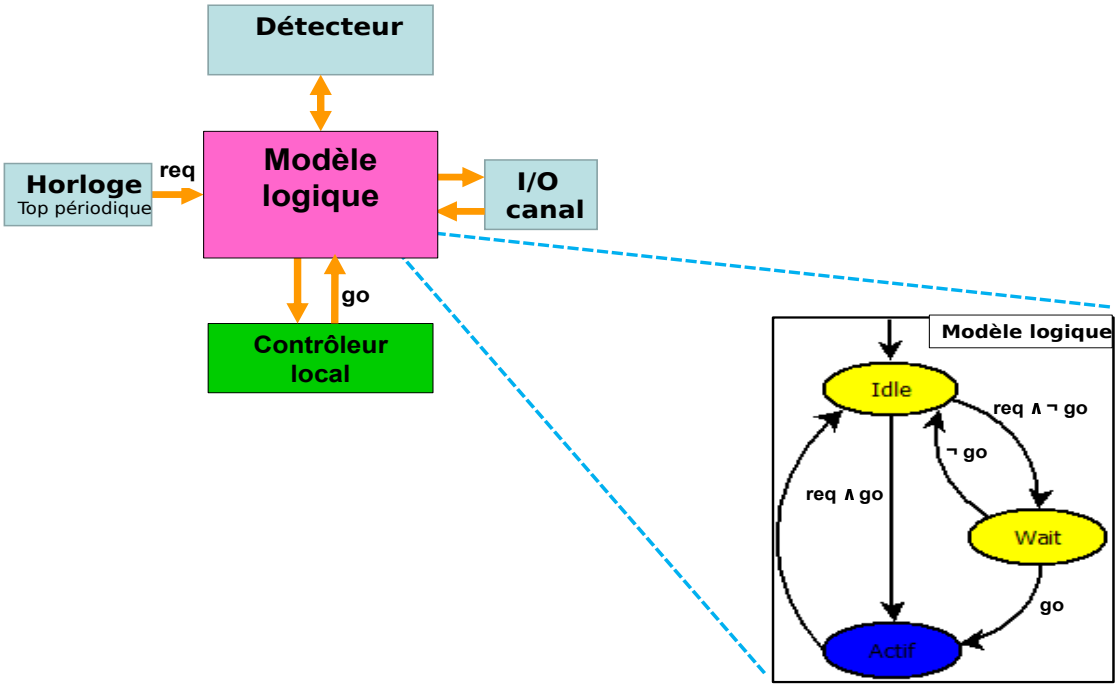

FIGURE 3.3 – Modèle formel abstrait d'un nœud-capteur

Cependant, l'ordonnanceur obtenu à ce stade est centralisé : il apparaît sous la forme d'une contrainte logique exprimée sur l'état global du réseau et encodant toutes les solutions de contrôle acceptables. Mais, ceci est incompatible avec la nature distribuée des réseaux de capteurs puisque cet ordonnanceur ne dispose pas d'entrées/sorties d'où la nécessité de distribuer ce superviseur global sur chaque nœud du réseau. Le raffinement proposé est amené à agir sur la structure de l'ordonnanceur. Celui-ci est transformé d'abord par la décomposition structurelle en transformant la contrainte de contrôle générée automatique-

ment en un vecteur de fonctions de contrôle, puis par l'ajout d'un mécanisme de communication explicite garantissant la synchronisation entre les noeuds communicants. Ces opérations qui visent à raffiner le comportement de l'ordonnanceur, modifient donc le code de celui-ci, alors qu'il avait initialement été généré automatiquement. Ceci engendre un besoin de simulation et validation du modèle raffiné obtenu, par rapport à l'ordonnanceur abstrait généré initialement par synthèse. La dernière étape est celle d'implémenter l'ordonnanceur validé obtenu sur les noeuds du réseau.

Le modèle formel abstrait d'un nœud

L'application d'outils formels nécessite la mise en place d'un modèle formel abstrait qui devra capturer en même temps l'aspect fonctionnel du nœud en faisant abstraction de son implémentation physique.

Comme illustré sur la figure 3.3, un nœud-capteur est modélisé par 5 blocs :

— un bloc modèle logique, décrit les trois modes d'activité possibles pour un nœud-capteur ;

— un bloc canal d'entrée/sortie grâce auquel le nœud communique avec ses voisins ;

— un bloc temporisateur qui détermine sa période d'activité ;

— un bloc de détection pour détecter le phénomène physique désiré par l'application tel que la température, l'humidité, etc.. ;

— un bloc contrôleur local qui contient la fonction de contrôle spécifique à chaque nœud implémentant l'ordonnancement.

Le bloc modèle logique est représenté par une FSM comportant trois états (*Idle*, *Wait*, *Active*). Initialement à l'état *Idle*, signifie que le nœud est inactif mais écoute toujours le support même si aucun message n'est transmis. L'occurrence d'une requête "*req*" qui est une variable booléenne incontrôlable, indiquant que le nœud doit quitter l'état *Idle* après l'expiration de sa période d'inactivité. Ce phénomène est modélisé par la tick du temporisateur périodique. La variable "*req*" permet au nœud de transiter soit à l'état *Wait* ou *Active* en fonction de la valeur de la variable contrôlable calculée par l'ordonnanceur. Si c'est "vrai",

l'état *Active* est choisie sinon le nœud passe à l'état *Idle*. Dans *Wait*, le nœud doit revenir à son état de repos si la valeur de "*go*" est "faux". Cependant, il peut faire une transition vers l'état *Active* dès que la valeur de go devient "true". À l'état *Active*, les nœuds consomment beaucoup d'énergie tout en assurant une surveillance continue. Cette représentation est inspirée des travaux d'Eric Rutten et ses collègues [47] [168] [7] où ils modélisent aussi le comportement contrôlable d'une tâche ou un processus par un automate à 3 états transformé en un nœud Heptagon par la suite. Ce nœud peut être réutilisé par instanciation ou par la composition parallèle et/ou hiérarchique.

Génération automatique d'un ordonnanceur global

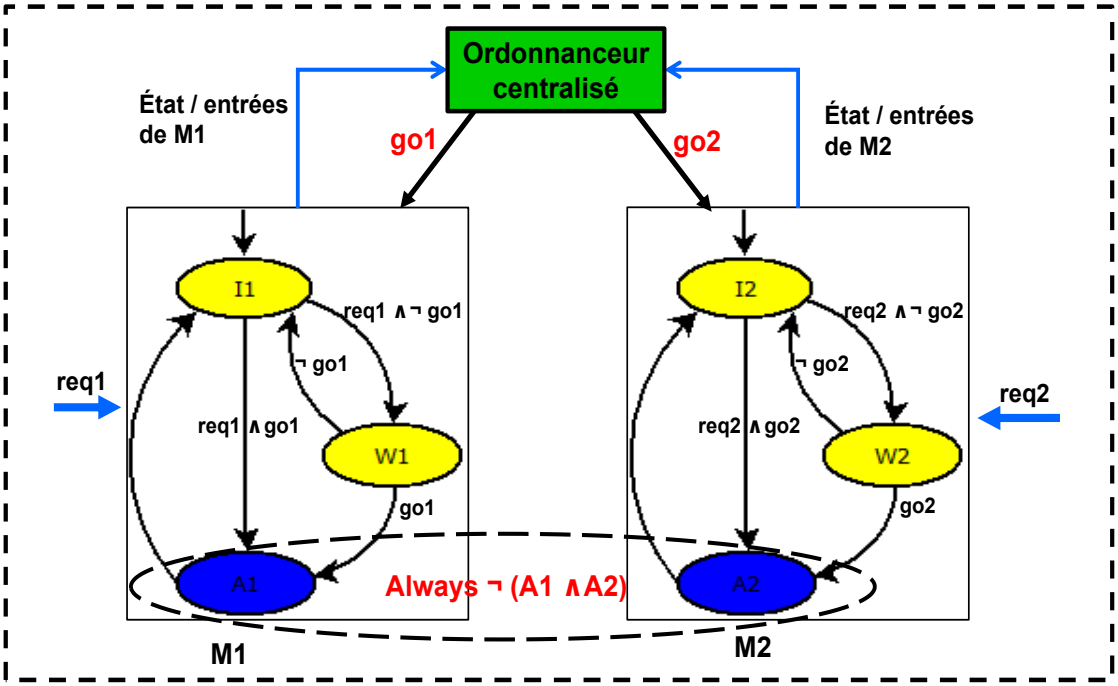

FIGURE 3.4 – Supervision centralisée appliquée à un cluster composé de deux noeuds

Initialement, tous les nœuds composant un cluster sont modélisés selon le modèle formel abstrait introduit ci-dessus. L'exigence d'ordonnancement intra-cluster exprime le besoin d'un accord entre les nœuds, dans le but de ne laisser qu'un seul nœud actif à tout moment. Pour un cluster donné, cela revient à un

problème d'exclusion mutuelle : deux ou plusieurs nœuds ne doivent pas être à l'état *Active* simultanément. En définissant $mutex_{ij}$ comme la configuration dans laquelle les noeuds i et j ne sont pas actifs en même temps :

$$mutex_{ij} = not(Active_i \wedge Active_j)$$

la condition globale d'exclusion mutuelle à l'échelle d'un cluster s'exprime :

$$spec : always(\bigwedge_{i \neq j} mutex_{ij})$$

La DCS est appliquée afin de renforcer la satisfaction de la spécification "*spec*" sur un cluster donné en générant un superviseur qui implémente un ensemble de règles globales pour ordonnancer le cluster. L'ordonnancement est réalisé en affectant des valeurs appropriées aux entrées *go*, conformément à ces règles. Les entrées *go* doivent être désignées comme entrées *contrôlables*.

L'ordonnancement obtenu est un *ordonnancement logique*, puisqu'il nécessite un calcul global sur les nœuds du cluster, mais fait abstraction de la distribution des noeuds.

Example

On considère un cluster composé de deux nœuds concurrents communicants. Comme l'illustre la figure 3.4, chaque nœud est modélisé par une FSM M_i, (i=1,2). Les deux nœuds sont initialement à leurs états *Idle* respectivement I_1 et I_2. Chaque machine M_i reçoit périodiquement une requête req_i de son bloc temporisateur local pour transiter vers l'état *Active*. Dans cet exemple, les variables d'entrée req_1, req_2 sont incontrôlables, alors que go_1, go_2 sont contrôlables. La synchronisation entre les deux machines M_1 et M_2 ne se fait que par la satisfaction de la proposition "*spec*" :

$$\text{spec} : never(A_1 \wedge A_2)$$

"spec" peut être aussi définie par :

$$\text{spec} : always\ mutex_{12} \text{ donc}$$
$$always\ not(Active_1 \wedge Active_2)$$

Un superviseur centralisé renforçant "*spec*" est généré automatiquement à l'aide de la DCS. Pour tout état I_i,W_i,A_i, (i=1,2) et pour toute entrée incontrôlable

req_1 et req_2, ce superviseur calcule la valeur appropriée pour chaque variable contrôlable go_1 et go_2, afin d'imposer la satisfaction de *spec*. Le superviseur généré est représenté par une contrainte booléenne *monolithique* exprimant les configurations autorisées du cluster afin que *spec* soit satisfaite :

$$
\begin{array}{lll}
\text{SUP} = & & \\
\text{(I1,I2)} & \wedge & (req_1 \wedge \overline{go_1} \vee req_2 \wedge \overline{go_2}) \vee \\
\text{(I1,W2)} & \wedge & (req_1 \wedge \overline{go_1} \vee \overline{go_2}) \vee \\
\text{(I1,A2)} & \wedge & (req_1 \wedge \overline{go_1}) \vee \\
\text{(W1,I2)} & \wedge & (\overline{go_1} \vee req_2 \wedge \overline{go_2}) \vee \\
\text{(W1,W2)} & \wedge & (\overline{go_1} \vee \overline{go_2}) \vee \\
\text{(W1,A2)} & \wedge & (\overline{go_1}) \vee \\
\text{(A1,I2)} & \wedge & (req_2 \wedge \overline{go_2}) \vee \\
\text{(A1,W2)} & \wedge & (\overline{go_2})
\end{array}
$$

Par exemple, dans l'état (I_1, I_2), si les valeurs de variables incontrôlables req_1 et req_2 sont tous les deux "vrai", le superviseur autorise soit go_1 = "vrai" soit go_2 = "vrai" mais pas les deux à la fois. Bien que formellement correcte, la représentation du superviseur est structurellement incompatible avec la nature de l'architecture distribuée du réseau de capteurs. Ce problème est abordé dans la section suivante.

Technique de distribution d'un superviseur monolithique centralisé

Les superviseurs générés par DCS ne sont pas structurellement compatibles avec l'opération distribuée inhérente aux réseaux de capteurs, où chaque nœud doit avoir une interface d'entrée/sortie. Ce n'est pas le cas pour la représentation équationnelle d'un superviseur, qui ne code que les valeurs acceptables d'un ensemble de variables, sans produire du résultat, et sans aucune distinction entre les variables d'entrée, d'état et de sortie. Pour résoudre ce problème, une technique systématique est proposée et illustrée pour transformer la contrainte de contrôle générée automatiquement en un vecteur de fonctions de contrôle.

Le superviseur fourni par DCS est implémenté comme une fonction caracté-

ristique [99] :

$$SUP : \mathbb{B}^{|X_c|+|X_{uc}|} \times Q \rightarrow \mathbb{B}$$

définie comme :

$$SUP((\mathbf{x_c}, \mathbf{x_{uc}}), \mathbf{q}) = 1 \text{ ssi } ((\mathbf{x_c}, \mathbf{x_{uc}}), \mathbf{q}) \in SUP$$

Par conséquent, pour contrôler M, on a besoin de résoudre l'équation suivante :

$$SUP((\mathbf{x_c}, \mathbf{x_{uc}}), \mathbf{q}) = 1 \tag{3.1}$$

continuellement, pour chaque transition de M, on considère $\mathbf{x_{uc}}$ comme des variables inconnues. Cependant, les contraintes de distribution spécifiques aux réseaux de capteurs exigent que les variables contrôlables $x_{ci}, i \in \{1 \ldots |X_c|\}$ soient calculées localement pour chaque nœud, en évaluant une expression appropriée. Cela revient à produire une résolution symbolique de l'équation (3.1). La structure de $\mathcal{SUP}$ est impropre à la résolution directe de cette équation.

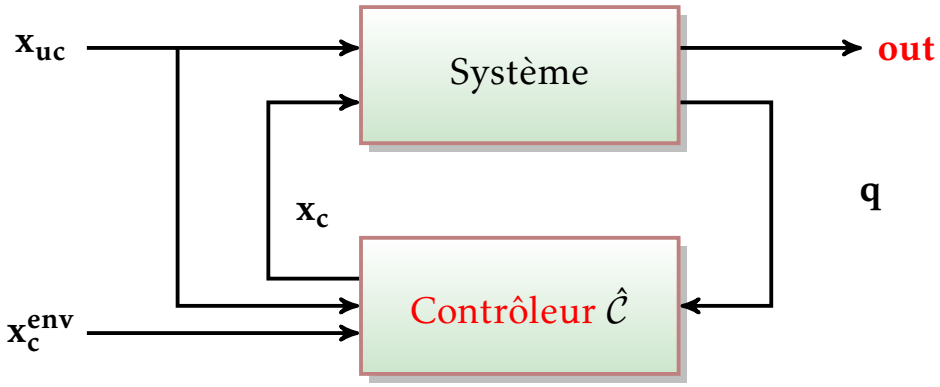

FIGURE 3.5 – architecture de contrôle cible

La technique de décomposition du superviseur présentée en [51] est utilisée pour résoudre l'équation 3.1 symboliquement. Cette approche donne l'architecture de contrôle présentée dans la figure 3.5. Cette technique constitue une décomposition paramétrique d'une fonction caractéristique, défini sur le domaine booléen. Elle est réalisée suivant des opérations, appliquées récursivement, pour chaque variable contrôlable.

Étape 1 : Diviser l'expression SUP afin de séparer la variable inconnue x_{ci} des

sous-expressions de $\mathcal{SUP}$ libre de x_{ci}. Ceci est effectué en réécrivant l'équation 3.1 selon le théorème d'expansion de Boole par rapport à x_{ci}. Soit $SUP_{x_{ci} \leftarrow e}$ l'expression obtenue en substituant l'expression booléenne e à la variable x_{ci} dans SUP. Ces expressions sont également appelées co-facteurs de SUP par rapport à x_{ci}. Ce qui suit est conforme au théorème d'expansion de Boole :

$$SUP = \neg x_{ci} \wedge SUP|_{x_{ci} \leftarrow 0} \vee x_{ci} \wedge SUP|_{x_{ci} \leftarrow 1}$$

Étape 2 : Résoudre symboliquement l'équation de contrôle 3.1 pour la variable x_{ci}. Pour toute valeur de i, les valeurs de x_{ci} qui satisfont cette équation sont détaillées dans le tableau 3.1.

Tableau 3.1 – Valeurs de x_{ci} lorsque $\mathcal{SUP}$ est satisfaisant

$\mathcal{SUP}\|_{x_{ci} \leftarrow 1}$	$\mathcal{SUP}\|_{x_{ci} \leftarrow 0}$	x_{ci}
0	1	**0**
1	0	**1**
1	1	**0 or 1** (x_{ci}^{env})

Comme souligné par ce tableau, la satisfaction simultanée des deux co-facteurs $\mathcal{SUP}|_{x_{ci} \leftarrow 0}$ et $\mathcal{SUP}|_{x_{ci} \leftarrow 1}$ a une signification particulière : Quelle que soit la valeur de x_{ci}, l'équation 3.1 est satisfaite. Cela signifie qu'algébriquement il peut exister un état dans Q où toutes les transitions permises par $\mathcal{SUP}$ conduisent à IUC et donc, x_{ci} peut prendre n'importe quelle valeur. Ceci est modélisé par l'ajout d'une variable paramétrique x_{ci}^{env} dans l'expression de x_{ci}. Cette situation caractérise le indéterminisme de contrôle par rapport à la variable contrôlable x_{ci} : A chaque fois que x_{ci} n'a pas d'influence sur la satisfaisabilité de $\mathcal{SUP}$, sa valeur est déterminée par x_{ci}^{env}. Structurellement, les variables x_{ci}^{env} sont des variables auxiliaires d'environnement (entrée) comme montré par la figure 2.2. D'après la table 3.1, l'expression de x_{ci}, dénotée f_i, est calculée comme suit :

$$f_i = \neg \mathcal{SUP}|_{x_{ci} \leftarrow 0} \wedge \mathcal{SUP}|_{x_{ci} \leftarrow 1} \vee$$
$$x_{ci}^{env} \wedge \mathcal{SUP}|_{x_{ci} \leftarrow 0} \wedge \mathcal{SUP}|_{x_{ci} \leftarrow 1}$$

Notant que l'expression f_i est libre de la variable x_{ci}.

Étape 3 :Substituer f_i par x_{ci} dans $\mathcal{SUP}$. L'expression résultante $\mathcal{SUP}|_{x_{ci}\leftarrow f_i}$ est le point de départ d'une application ultérieure des étapes Diviser / Résoudre/ Substituer définies ci-dessus.

Soit $\tau : \{1 \dots |X_c|\} \rightarrow \{1 \dots |X_c|\}$ une permutation définissant un ordre de résolution défini par l'utilisateur pour les variables contrôlables $\mathbf{x_c}$. La résolution complète de ce système d'équation est effectuée en calculant les expressions symboliques suivantes.

Pour tout $i = \tau(1) \dots \tau(|X_c|)$:

$$f_i = \neg \mathcal{SUP}_i|_{x_{ci}\leftarrow 0} \wedge \mathcal{SUP}_i|_{x_{ci}\leftarrow 1} \vee$$
$$x_{ci}^{env} \wedge \mathcal{SUP}_i|_{x_{ci}\leftarrow 0} \wedge \mathcal{SUP}_i|_{x_{ci}\leftarrow 1}$$

où $\mathcal{SUP}_1 = \mathcal{SUP}$ et $\mathcal{SUP}_i = \mathcal{SUP}_{i-1}|_{x_{ci-1}\leftarrow f_{i-1}}$.

Le contrôleur résultant est un vecteur $\hat{\mathcal{C}}$ de m fonctions booléennes, où m est le nombre de variables contrôlables :

$$\hat{\mathcal{C}} = \begin{pmatrix} f_1(q, x_{uc}, x_{c1}^{env}, f_2, \dots, f_m) \\ f_2(q, x_{uc}, x_{c2}^{env}, f_3, \dots, f_m) \\ \dots\dots \\ f_m(q, x_{uc}, x_{cm}^{env}) \end{pmatrix}$$

Ce processus de décomposition associe chaque variable contrôlable x_{ci} à une variable auxiliaire x_{ci}^{env}. L'action de $\hat{\mathcal{C}}$ est similaire au filtrage : à chaque instant, en fonction de l'état actuel q et de $\mathbf{x_{uc}}$, x_{ci} est affectée à la valeur x_{ci}^{env} ou à la valeur $\neg x_{ci}^{env}$, de sorte que la propriété de contrôle global reste vraie. Le contrôleur est un type particulier de "FSM" booléen, comportant seulement des entrées et des sorties, aucun état et aucune fonction de transition.

Le modèle du système contrôlé est construit en mappant les sorties de $M^{\hat{\mathcal{C}}}$ aux variables contrôlables X_c de M.

Il est important de souligner que le choix de τ peut avoir un impact sur le contrôleur résultant. Il établit un ordre d'évaluation, afin de gérer l'affectation des variables contrôlables en cas de non déterminisme du contrôle, en prenant

en compte les choix faits pour les autres. Ainsi, la variable $x_{c\ \tau(|X_c|)}$ est toujours évaluée en premier, car l'expression de son contrôleur est calculée en dernier et ne dépend pas des autres variables contrôlables.

Tous les choix de τ sont valides du point de vue de la satisfaction de *spec*, mais pas toujours équivalents. Trouver un ordre convenant revient à un choix, fait par le concepteur, avant la décomposition du superviseur. La seule ligne directrice disponible pour faire ce choix est d'exprimer une priorité d'évaluation entre les variables contrôlables.

Application : Distribution de l'ordonnanceur logique global

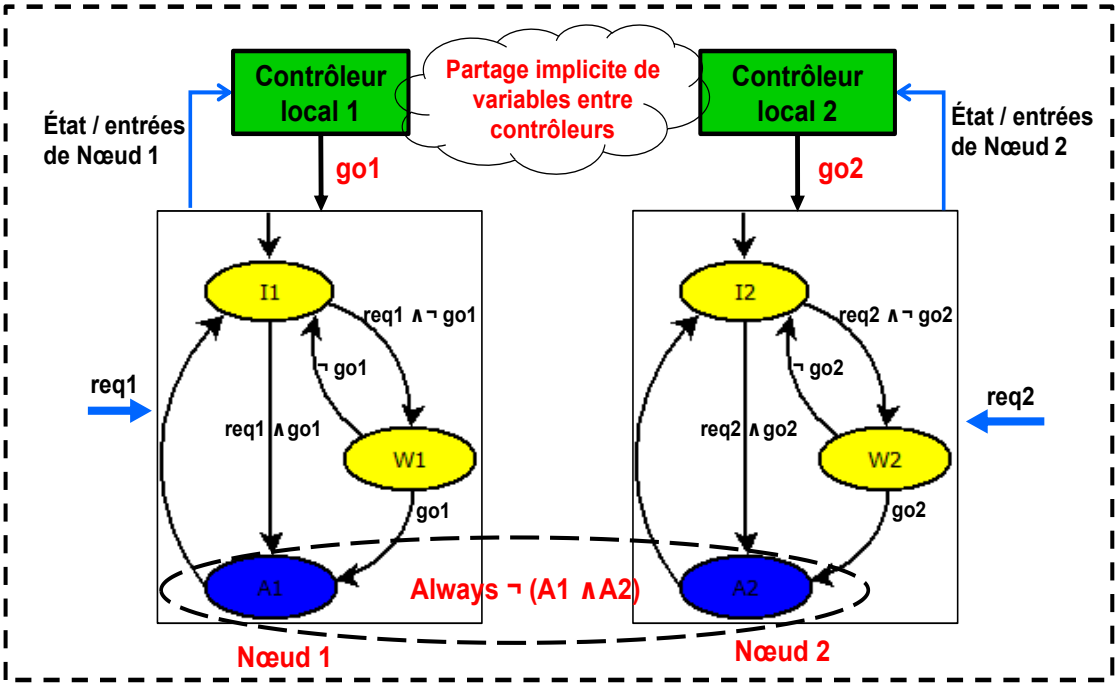

FIGURE 3.6 – Ordonnancement distribué d'un cluster formé de deux noeuds

En appliquant la technique de distribution présentée ci-dessus, chaque nœud devient contrôlé par un contrôleur local, comme illustré sur la figure 3.6. Les valeurs de go_1 et go_2 sont maintenant calculées localement. Ainsi, les fonctions de contrôle associées respectivement aux nœuds 1 et 2 sont présentées comme

suit :

$$go_1 = p_1 \vee (h_1 \wedge f_2)$$
$$go_2 = go_2^{env}$$

where :

$$p_1 = (I_1 \wedge \neg req_1 \wedge go_1^{env}) \vee (\neg I_1 \wedge go_1^{env})$$
$$h_1 = (I_1 \wedge req_1 \wedge go_1^{env}) \vee (\neg I_1 \wedge W_1 \wedge go_1^{env})$$
$$f_2 = (I_2 \wedge \neg req_2) \vee (I_2 \wedge req_2 \wedge \neg go_2^{env}) \vee (\neg I_2 \wedge \neg W_2)$$

Les variables supplémentaires go_i^{env} sont inhérentes à la méthode de distribution. Par construction, leurs valeurs ont un impact sur le calcul de go_i. Elles sont ajoutées au cours du processus de distribution chaque fois que les règles d'ordonnancement expriment une configuration "don't care" par rapport à ce calcul. D'un point de vue comportemental, les variables go_i^{env} modélisent le fait que localement, le nœud i souhaite s'activer. Si go_i^{env} a la valeur "faux", il ne doit pas s'activer. Sinon, il peut s'activer selon l'accord global obtenu en calculant les valeurs go_i.

3.5 Généralisation de l'ordonnancement pour des clusters à taille arbitraire

La taille d'un réseau de capteurs est un problème difficile pour l'application de la DCS. Cette technique est limitée par une complexité spatiale exponentielle en nombre de variables d'état du système exploré, ce qui la rend a priori incompatible avec l'application envisagée. En d'autres termes, construire un contrôleur garantissant l'exclusion mutuelle lors de l'activation des noeuds d'un cluster composé de milliers de nœuds communicants conduit forcément à un problème d'explosion combinatoire.

Il a néanmoins été possible d'identifier une démarche méthodologique étayée par un algorithme, qui permet d'exploiter la symétrie au sein d'un réseau de capteurs et déduire des règles permettant d'étendre la formule d'un contrôleur distribué à un nombre de nœuds arbitrairement grand. Les contrôleurs sont

ainsi synthétisés sur de petits sous-ensembles (des clusters à nombre réduit de noeuds), puis ils sont construits de manière inductive, pour n'importe quelle taille, selon une règle structurelle reposant sur les propriétés de symétrie et de régularité d'un réseau de capteurs.

L'application de la DCS pour un cluster composé de trois nœuds nous permet de dégager les fonctions de contrôle suivantes :

$$
\begin{aligned}
go_1 &= p_1 \vee (h_1 \wedge f_2 \wedge f_3) \\
go_2 &= p_2 \vee (h_2 \wedge f_3) \\
go_3 &= go_3^{env}
\end{aligned}
$$

Alors que pour un cluster de 4 noeuds, on a les fonctions de contrôle suivantes :

$$
\begin{aligned}
go_1 &= p_1 \vee (h_1 \wedge f_2 \wedge f_3 \wedge f_4) \\
go_2 &= p_2 \vee (h_2 \wedge f_3 \wedge f_4) \\
go_3 &= p_3 \vee (h_3 \wedge f_4) \\
go_4 &= go_4^{env}
\end{aligned}
$$

En analysant les expressions ci-dessus, une règle de récurrence peut être trouvée assez facilement. Afin d'obtenir une expression généralisée, cette règle est exprimée de manière algorithmique comme suit :

Algorithm 1 Fonctions de contrôle génériques pour (n ≥ 2)

$a \leftarrow true$
for $i \leftarrow n-1$ to 1 **do**
  $p_i \leftarrow (I_i \wedge \neg req_i \wedge go_i^{env}) \vee (\neg I_i \wedge go_i^{env})$
  $h_i \leftarrow (I_i \wedge req_i \wedge go_i^{env}) \vee (\neg I_i \wedge W_i \wedge go_i^{env})$
  $f_i \leftarrow (I_i \wedge \neg req_i) \vee (I_i \wedge req_i \wedge \neg go_i^{env}) \vee (\neg I_i \wedge \neg W_i)$
  $a \leftarrow (a \wedge f_{i+1})$
  $go_i \leftarrow (p_i \vee (h_i \wedge a))$
end for
$go_n \leftarrow go_n^{env}$

Ce résultat est issu d'une inspection visuelle de l'expression du superviseur distribué. Cette inspection a permis d'identifier des sous-expressions h_i, p_i et f_i, dont la forme évolue de manière régulière avec l'augmentation du nombre des

noeuds du cluster. La "généralisation" porte donc sur l'évolution de la structure de ces expressions.

La correction de cette approche de généralisation peut être vérifiée formellement, pour des tailles n inférieures à 20 noeuds, et par simulation au-delà.

Ce travail de généralisation est en somme laborieux, car il repose sur une intervention manuelle de la part du concepteur. Cependant, il est facilité par la symétrie existant au sein d'un cluster, et aussi par le fait qu'il démarre à partir d'une solution générée par la DCS et donc correcte par construction. Cette piste de conception reste néanmoins très prometteuse, car elle permet, moyennant une intervention manuelle du concepteur, d'aller bien au-delà des possibilités des algorithmes de DCS, permettant d'obtenir un ordonnancement quelle que soit la taille du cluster.

Cependant, ce travail manuel est digne, car il va bien au-delà des capacités de DCS, qui souffre du problème d'explosion combinatoire, et ne peut pas être appliquée pour des systèmes paramétrés, comme un cluster des noeuds de taille n arbitrairement grand.

3.6 Conception des primitives de communication

L'architecture de contrôle distribuée, étudiée dans les sections précédentes, considère que la communication entre les nœuds du réseau est instantanée. Bien que l'abstraction des retards de communication simplifie significativement la conception du contrôle distribué, ceci n'est pas très réaliste dans un contexte des réseaux de capteurs, où les nœuds-capteurs sont reliés par des canaux sans fil sujets aux retards.

Dans cette section, nous explorons l'intégration d'un mécanisme de communication réaliste pour un réseau de capteurs sans toucher le système ni le contrôleur distribué généré initialement avec zéro retard. Afin de garantir une synchronisation entre les nœuds communicants, Un nœud émetteur n'envoie son message que si le destinataire est prêt à recevoir ce message. En d'autres termes, nous ajoutons seulement une «barrière» de synchronisation entre les nœuds communicants, tout en essayant de garantir la cohérence établie initialement par le superviseur centralisé.

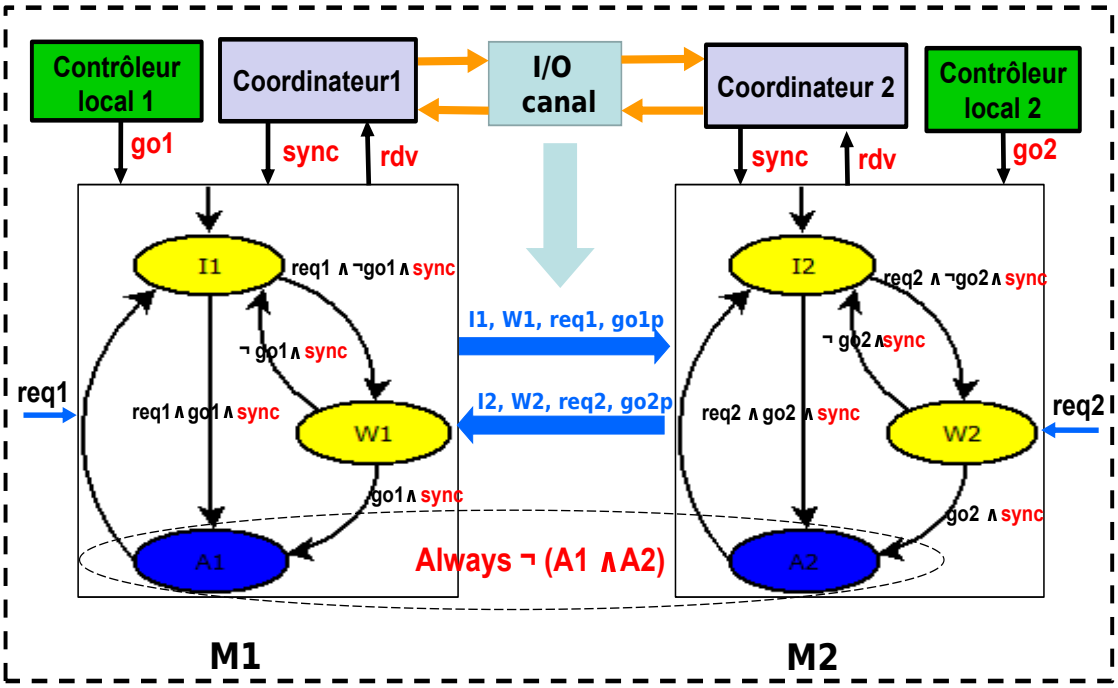

FIGURE 3.7 – établissement de la barrière de synchronisation entre deux noeuds communicants

3.6.1 Implémentation d'une barrière de synchronisation

Implémenter une barrière de synchronisation n'a jamais été une tâche triviale, vu que ces contraintes doivent être satisfaites :

(1) L'exécution d'un protocole de synchronisation basé sur un rendez-vous nécessite un accord mutuel entre les nœuds concernés participant au rendez-vous.

(2) L'implémentation doit être distribuée de telle sorte qu'aucun nœud sache l'état global du réseau.

Afin de garantir une synchronisation entre les nœuds communicants, nous proposons une primitive de communication basée sur un mécanisme de rendez-vous asynchrone ; asynchrone dans le sens où il nécessite un temps arbitraire entre le moment de la demande de communication et son achèvement réel. Par conséquent, nous ne visons pas à toucher l'architecture de contrôle abstraite générée automatiquement. Nous ajoutons seulement une «barrière» de synchronisation entre les nœuds communicants, en essayant de garantir la cohérence

établie initialement par le modèle abstrait. Ainsi, chaque transition dans le modèle initial devient implicitement conditionnée par l'achèvement de cette barrière de synchronisation.

Nous raffinons, ainsi, le modèle abstrait du nœud par l'ajout d'un nouveau module appelé « coordinateur ». Ce module est responsable d'établir la barrière de synchronisation avec les autres noeuds du cluster. L'interaction entre le nœud et son coordinateur est au moyen de signaux logiques rdv et $sync$. Comme illustré sur la figure 3.7, le nœud $M1$, initialement, est à l'état I_1, quand il reçoit un signal req_1 de son bloc temporisateur, il doit transiter soit à l'état W_1 soit à l'état A_1, selon la valeur de sa variable contrôlable go_1 calculée par son contrôleur local. Cependant, le calcul de go_1 ne dépend seulement des variables locales du nœud $M1$ mais aussi des valeurs de certaines variables appelées "distantes" relatives au nœud $M2$. Ceci nécessite la synchronisation avec le nœud $M2$ afin de récupérer les valeurs des ces variables distantes. Ainsi, le nœud $M1$ initialise la communication avec le nœud $M2$ en envoyant seulement le signal RDV à son coordinateur local. Le coordinateur échange, à son tour, avec le coordinateur de nœud $M2$ les informations nécessaires à la synchronisation et répond à la requête de son nœud par le signal $sync$, indiquant le succès du rendez-vous.

3.6.2 Comportement générique du coordinateur

L'architecture "Maître / Esclave" est adoptée pour la modélisation de comportement du coordinateur. Un coordinateur peut être soit en mode "maître" soit en mode "esclave". Le rôle de maître est assuré si un nœud reçoit un souhait local d'être activé par son temporisateur local avant tous ses voisins du cluster. Sinon, il est considéré comme esclave.

Comme l'illustre la figure3.8, le comportement du coordinateur peut être modélisé par un modèle global composé de deux modèles qui s'exécutent en parallèle, appelées respectivement $Master\_task$ et $Slave\_task$. Le modèle $Slave\_task$ à son tour peut être modélisé par un ou plusieurs modèles qui tournent en parallèle relativement au nombre de noeuds voisins dans le même cluster.

A la réception d'un signal RDV du modèle de nœud, le modèle $Master\_task$ du coordinateur est activé et un signal $master$ est envoyé aux différents modèles

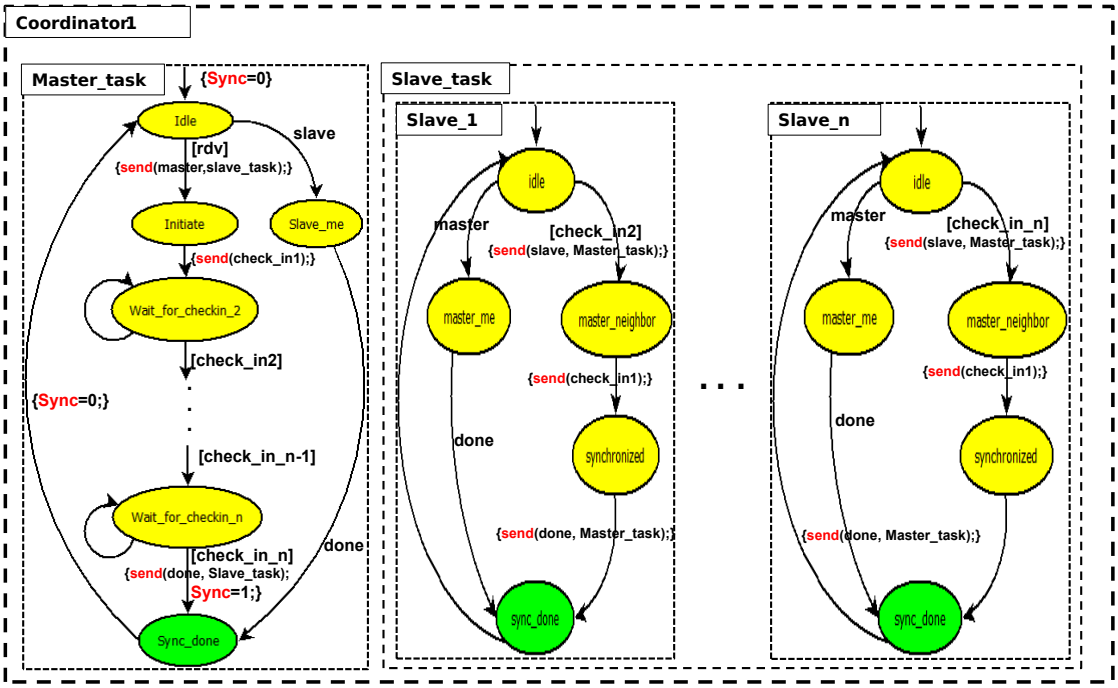

FIGURE 3.8 – établissement de la barrière de synchronisation entre deux noeuds communicants

Slave_task des autres noeuds du cluster déclarant ainsi que son nœud a devenu le maître et qu'il a besoin de se synchroniser avec ses voisins pour calculer la bonne valeur de sa variable contrôlable go_1. Par ailleurs, il entre à l'état *initiate* et émet un signal *check_in*1 initialisant le rendez-vous avec ses voisins du cluster. Après la réception de différents signaux *check_in* de tous ses voisins considérés comme des "esclaves", le coordinateur émet le signal *sync* au modèle de son nœud, indiquant que la synchronisation est réalisée avec succès et qu'il peut maintenant calculer sa fonction de contrôle locale.

Dans le mode "esclave", comme son nom l'indique, le nœud est toujours en attente d'un signal *check_in* d'un autre nœud voisin qui devient maître et voudrait initialiser la communication. Une fois qu'un coordinateur reçoit ce *check_in* du nœud "maître", avant le signal *rdv* de son propre nœud, il envoie tout d'abord le signal *slave* à son modèle *Master_task* pour lui empêcher d'initialiser un nouveau processus de communication et assume ainsi le rôle d'esclave en répondant par un *check_in* envoyé au modèle *Master_task* du coordinateur de

nœud voisin considéré comme maître.

3.7 Simulation et validation

La simulation est la dernière étape de notre méthode. Le but de cette étape est de s'assurer que le réseau de capteurs contrôlé est toujours fonctionnel selon les besoins.

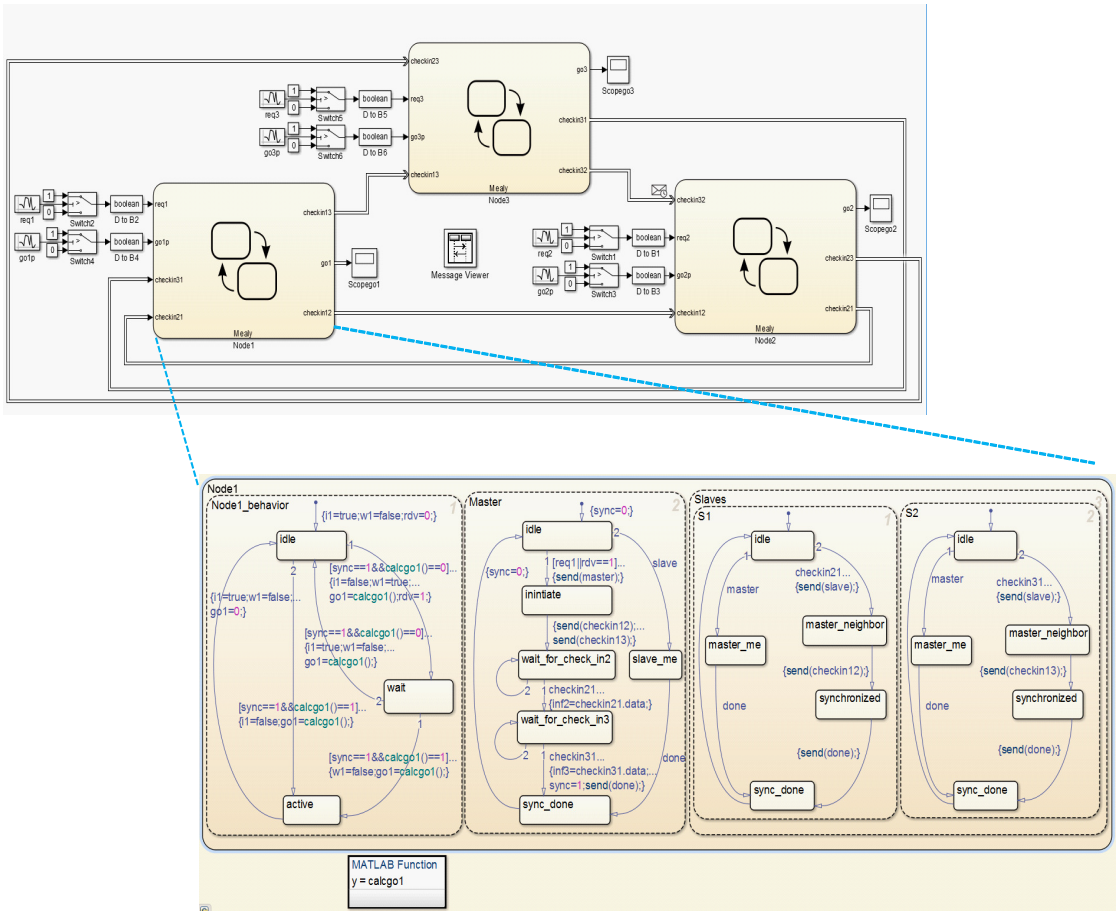

FIGURE 3.9 – Simulation d'un réseau de capteurs composé de 3 noeuds ordonnancé par DCS

Pour valider notre approche, nous simulons un cluster composé de trois nœuds concurrents, à l'aide des outils MathWorks (Simulink et Stateflow). Nous choisissons d'utiliser ces outils, vu leurs riches bibliothèques pour la modélisa-

tion des systèmes embarqués. Ainsi, nous avons utilisé l'outil de DCS symbolique Sigali[99] pour générer automatiquement les fonctions de contrôle locales (une pour chaque nœud du réseau). Ces contrôleurs locaux ont été modélisés sur Simulink [142] par des fonctions Matlab.

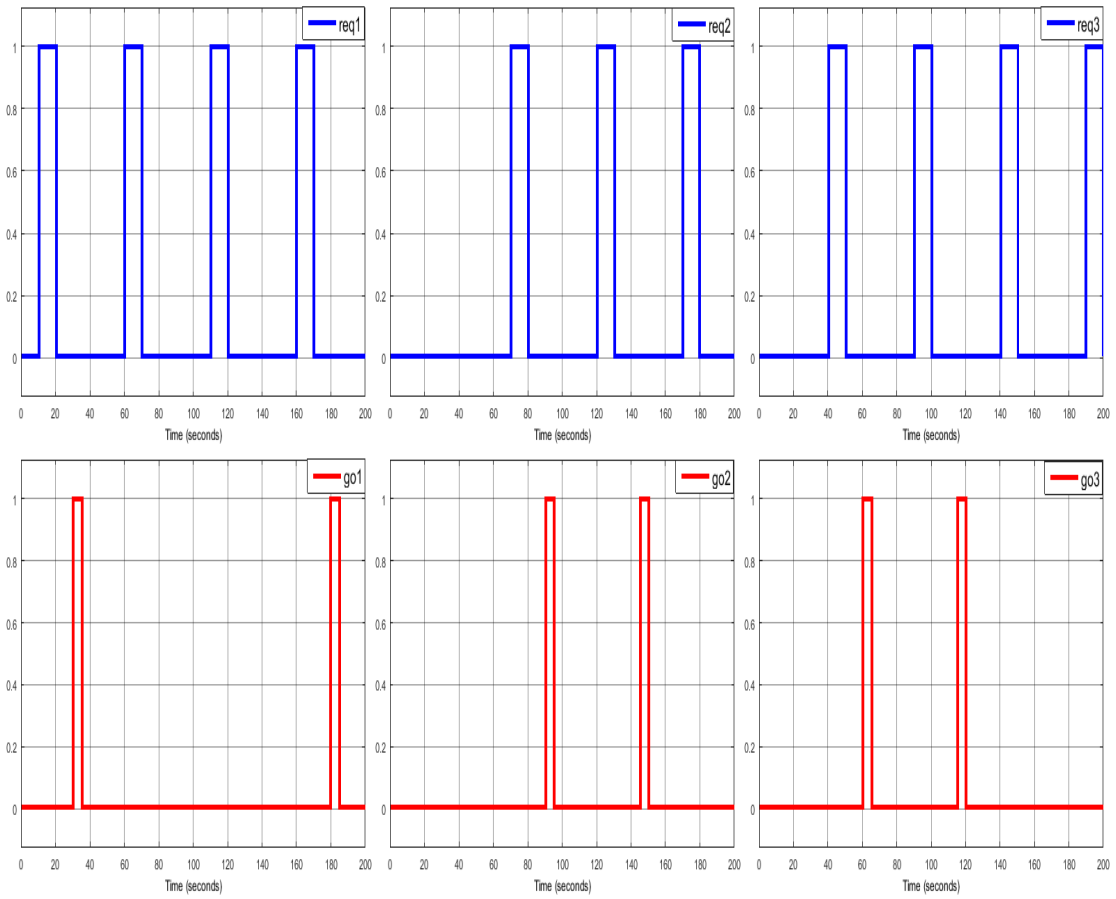

FIGURE 3.10 – Simulation d'un réseau de capteurs composé de 3 noeuds ordonnancé par DCS

Comme illustré par la figure3.9, le comportement de chaque nœud est modélisé par un modèle Stateflow séparé. Les noeuds sont supposés reliés par des canaux de communication point à point. Cette communication est modélisé par l'échange de messages entre les modèles stateflow. Les variables booléennes incontrôlables req_1, $go1_p$, req_2, $go2_p$, req_3 et $go3_p$ sont fournies par Simulink. Les valeurs de ces variables d'entrée sont déterminées à l'aide des fonctions Simulink

permettant de générer des nombres aléatoires qui sont convertis par la suite en booléen avant d'avoir être utilisés par les modèles Stateflow des nœuds.

La trace de simulation obtenue montre le comportement correct du cluster contrôlé. Dans la figure3.10, on peut remarquer qu'à tout moment, une seule des trois variables contrôlables go_1, go_2 et go_3 peut être vraie. En d'autres termes, un seul nœud du cluster contrôlé est dans son état*actif* tandis que les autres nœuds sont toujours dans leurs états *idle*. On constate aussi que à la seconde 60 et la seconde 110, bien que la valeur de la variable incontrôlable req_1 soit vraie, la valeur de go_1 reste fausse. Ceci est expliqué par la valeur de go_3 qui est vrai à ces moments ce qui signifie que le nœud 3 est bien dans son état *actif* assurant la surveillance du cluster.

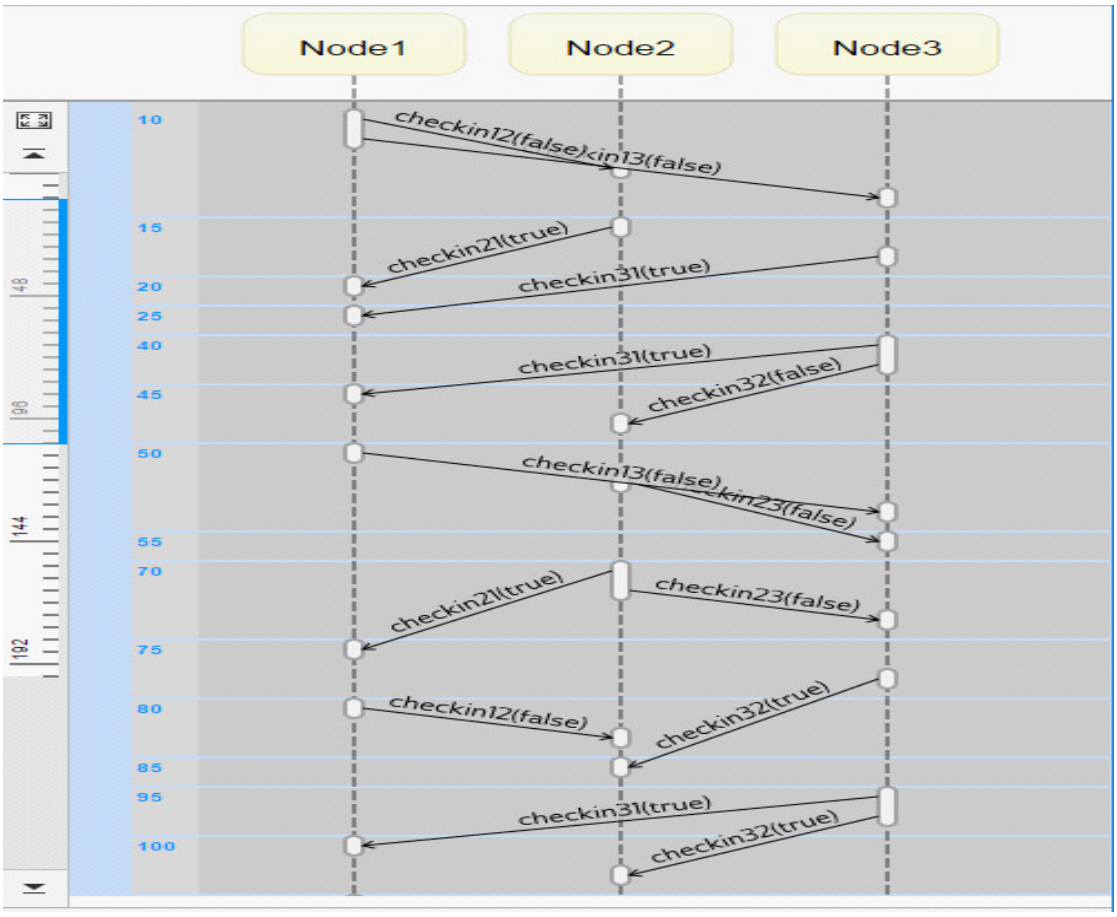

FIGURE 3.11 – Échange des messages au cours de temps entre les noeuds d'un cluster

On peut également observer sur la figure3.10, qu'à la seconde 10, la variable incontrôlable req_1 prend la valeur "vraie" qui permet au nœud 1 d'initialiser la communication avec ses voisins du cluster afin de récupérer les valeurs des variables nécessaires au calcul de sa variable contrôlable go_1. Une telle communication nécessite un certain temps jusqu'à ce que la valeur de go_1 devienne vraie à la seconde 30 où le contrôleur permet au nœud 1 de transiter vers l'état *actif*. Ces résultats de simulation confirment que l'exigence *spec* de notre architecture de contrôle distribué abstraite est toujours respectée (un nœud actif au sein de chaque cluster) même en présence de retard, mais avec un comportement temporel éventuellement dégradé.

Comme les nœuds d'un réseau de capteurs utilisent des messages pour échanger les données nécessaires au calcul de valeurs de leur variables locales, nous ajoutons, pendant la simulation, un bloc "Message Viewer" afin de visualiser l'échange de messages, en fonction du temps, entre les différents modèles Stateflow modélisant les comportements de noeuds. Ceci est illustré sur la figure3.11 où on observe qu'à la seconde 10, le nœud 1 initialise la communication avec leurs voisins en envoyant son message $check\_in$ aux nœuds 2 et 3. Cependant, il reçoit le $check\_in$ du nœud 3 à l'instant 25 de simulation, ce qui illustre la durée nécessaire à l'établissement de la communication et à l'échange des données entre les trois nœuds communicants du cluster.

Les résultats de simulation se montrent satisfaisants permettent de confirmer, à l'échelle des scénarios exécutés, la satisfaction de *spec* après raffinement des actions de communication.

3.8 Conclusion

Ce travail s'est appuyé sur la technique DCS, et a proposé une approche de distribution automatique du contrôleur, qui, combinée avec des mécanismes de communication ordinaires, assure le contrôle global. Cependant, les résultats présentés dans ce chapitre ne s'appliquent pas au réseau entier, mais aux unités redondantes constituées de sous-ensembles de nœuds considérés comme équivalents, appelées Clusters. Afin d'améliorer la mise à l'échelle à la taille réelle d'un réseau de capteurs, les contrôleurs synthétisés sur de petits sous-ensembles trai-

tables, sont construits de manière inductive, pour n'importe quelle taille, selon une règle structurelle reposant sur les propriétés de symétrie et de régularité du réseau, comme indiqué dans la section 5.

L'accent a été mis, dans ce chapitre, sur l'ordonnancement intra-cluster des noeuds à l'aide de la DCS. Par conséquent, Le chapitre suivant étend les éléments précédemment présentés en intégrant un schéma de routage optimal, qui détermine les chemins optimaux permettant à chaque nœud actif d'être connecté au puits.

Chapitre 4

Génération automatique d'un routage optimal multi-critères pour les RCSFs

4.1 Introduction

Compte tenu des capacités limitées de capteurs tant en termes d'énergie que de portée de communication, il s'avère nécessaire pour les noeuds de coopérer pour acheminer un paquet de données à la station de base. Le routage de données dans un RCSF représente alors la fonctionnalité la plus importante du réseau, qui permet la bonne gestion de ressources limitées de ces noeuds. Une telle station de base a généralement plus de ressources en termes de puissance, de calcul, de mémoire et de bande passante que les nœuds-capteurs individuels. C'est pour cette raison qu'un protocole de routage qui effectue les calculs à la station de base tout en minimisant le rôle des nœuds dans la construction des tables de routage s'est avéré être une technique intéressante pour réduire la consommation d'énergie des RCSFs.

Dans ce chapitre, nous présentons d'abord une taxonomie des protocoles de routage pour les réseaux de capteurs. Nous décrivons leurs principales caractéristiques et fonctionnalités qui permettent d'assurer l'acheminement des données vers une station de base. Ensuite, un algorithme de recherche des chemins optimaux adapté aux caractéristiques des réseaux de capteurs (faible capacité de stockage, énergie limitée, etc ..) est proposé. Cet algorithme est généré automati-

quement offrant une méthode pour déterminer les chemins optimaux suivant deux métriques indépendantes : le minimum de nombre de sauts intermédiaires pour atteindre la station de base et le maximum de réserves énergétiques des noeuds choisis tout au long des chemins menant à la station de base.

4.2 Taxonomie des solutions de routage dans les RCSFs

A travers cette taxonomie, nous nous intéressons aux différents mécanismes qui permettent aux données collectées par les nœuds du réseau d'être reçues par la station de base. Ces mécanismes doivent prendre en compte les spécificités de ces réseaux, notamment les aspects d'auto-organisation, de consommation d'énergie et de faible capacité du médium.

Les protocoles de routage de réseaux de capteurs sont classés selon deux critères :

— la structure du réseau ;

— la politique suivie pour recalculer le routage.

On peut distinguer selon la structure du réseau les protocoles de routage suivants :

— les protocoles plats ;

— les protocoles hiérarchiques ;

— les protocoles géographiques.

Selon la politique de calcul des routes pour l'acheminement des données, les protocoles de routage peuvent être aussi séparés en deux catégories :

— les protocoles proctifs ;

— les protocoles réactifs.

Dans les protocoles proactifs les routes sont établies à l'avance en se basant sur l'échange périodique des tables de routage, alors que les protocoles réactifs cherchent les routes à la demande.

4.2.1 Le routage plat

La première catégorie à présenter dans cet état de l'art est les protocoles de routage plats multi-sauts. Dans ce type de réseau, tous les nœuds sont semblables en terme de ressources et possèdent le même rôle. Ils collaborent ensemble afin de mettre en œuvre l'opération du réseau.

Comme ils sont déployés en un grand nombre, il est cependant impossible d'associer un identifiant unique globale à chaque nœud. Ceci a mené à l'apparition d'un nouveau type de protocole de routage orienté données, où la station de base envoie des requêtes à certaines régions et attend des données des capteurs situés dans les régions choisies. SPIN [86] [69] et Directed Diffusion [78] sont les premiers travaux présentant des protocoles de routage orienté données permettant d'économiser l'énergie à travers la négociation et l'élimination des données redondantes.

SPIN

Dans SPIN [86], tous les nœuds dans le réseau sont considérés comme des stations de base potentielles. Ainsi, toutes les informations collectées par chaque nœud sont disséminées par tous les autres nœuds de réseau. Ceci permet à l'utilisateur d'interroger n'importe quel nœud et obtenir les informations exigées immédiatement. SPIN a été conçu pour pallier aux problèmes de l'inondation classique tels que l'implosion et le chevauchement lié au déploiement dense des capteurs. Ces problèmes affectent grandement la durée de vie et les performances du réseau c'est pourquoi la famille de protocoles SPIN adopte des procédures de négociation et d'adaptation aux ressources. SPIN utilise essentiellement trois types de message ADV/REQ/DATA pour communiquer entre les nœuds. Lorsqu'un nœud veut émettre une donnée, il émet d'abord un message ADV qui contient une description de la donnée à émettre. Dés la reception d'un message ADV, le nœud consulte sa base d'intérêt pour vérifier si cette donnée lui intèresse. Si c'est le cas, il émet un message REQ vers son voisin. Sinon il l'ignore. La réception du message REQ par le voisin permet au noeud émetteur de transmettre ainsi la donnée sous forme d'un message DATA. Ce mécanisme assure qu'il n'y a pas d'envoi redondant de messages dans le réseau. Cependant, ce type de protocole n'est pas adapté aux applications de remonte d'alertes.

Direct Diffusion

C. Intanagonwiwat et. al. proposent dans [78] un paradigme populaire d'agrégation de données pour les réseaux de capteurs appelé Direct Diffusion. L'idée principale de Direct Diffusion est de combiner les données provenant de différentes sources au cours de leur transmission en éliminant les redondances et minimisant le nombre de transmissions. Ce qui permet d'économiser l'énergie du réseau et prolonger sa durée de vie. Direct Diffusion repose sur quatre éléments :

— nomination des données, où toutes les données générées par des nœuds sont nommées par des paires "attribut-valeur" afin de décrire les intérêts ;

— propagation des intérêts et établissement des gradients où la station de base diffuse un intérêt sous forme de requête afin d'interroger le réseau sur une donnée particulière.

— propagation des données : La réception de l'intérêt par tous les sources ciblées permet aux noeuds de commencer la collecte d'informations ;

— renforcement des chemins : Après avoir reçu les premières données, le puit renforce le chemin vers le voisin émetteur, en augmentant le débit de captage.

Contrairement à SPIN, Direct Diffusion n'exige pas de maintenir une topologie globale du réseau. Cependant, il n'est pas adapté aux applications, telle que la surveillance de l'environnement, qui exigent une transmission continue de données vers la station de base. En effet, le modèle adopté de collecte de bonnes données répondant aux requêtes peut engendrer un surcoût supplémentaire en terme d'énergie.

Rumour routing

Comme les protocoles mentionnés précédemment utilisent une forme d'inondation contraignante de point de vue consommation énergétique pour la propagation des intérêts ou des données. Le protocole Rumour Routing [29] essaie de trouver un compromis entre l'inondation des intérêts et la propagation des données. Au lieu d'innonder le réseau, ce protocole identifie uniquement les chemins menant aux noeuds ayant observé un événement afin de leur envoyer des rêquetes. L'idée de base du Rumour routing est d'utiliser des agents. Ces agents sont en réalité des messages de longue durée traversant le réseau pour

afin d'établir des tables des voisins. Les requêtes peuvent être routées ultérieurement le long de ces chemins générés par l'agent. Chaque nœud du réseau maintient une liste des voisins et une table d'événements avec des informations de transmission à tous les événements détectés. La liste de voisins est créée et maintenue activement en diffusant une requête ou de manière passive en écoutant les diffusions d'autres nœuds.

De point de vue économie d'énergie, l'utilisation d'un seul chemin entre la source et la destination permet de surclasser le protocole Rumour routing par rapport au celui basé sur l'innondation où les informations sont souvent acheminées par des routes multiples. Cependant, le routage par rumeur trouve ses limites quand le nombre des évenements devient plus grand ce qui provoque un coût significatif du maintien des agents et des tables d'événements de chaque nœud.

Minimum Cost Forwarding Algorithm

MCFA [159] est proposé pour déterminer un chemin minimal entre la source et le puits en se basant sur une variable de coût suivant l'application voulue : taux de consommation énergétique, nombre de sauts, etc.

MCFA s'est déroulé suivant le phasage suivant :

— *calcul des coûts* : La station de base initialise cette phase de calcul en émettant un message ADV qui contient un coût nul alors que les coûts des autres noeuds sont initialisés à une valeur infinie. La réception d'un message ADV permet au noeud de comparer sa valeur locale par rapport à la valeur reçue plus le coût du lien. Si cette nouvelle valeur reçue est plus petite par rapport à sa valeur locale, il met à jour son propre coût et envoie un nouveau message ADV à ses voisins ;

— relais des paquets : MCFA n'utilise aucune identification des nœuds et aucune table de routage. Un paquet émis par une source vers le puits contient necessairement le coût minimal local du nœud. A la réception d'un paquet, le nœud retranche le côut de lien de réception et compare ce coût reçu à son coût local. S'ils sont égaux, le nœud remplace la valeur du coût par son coût local et relaie ainsi le paquet.

4.2.2 Le routage hiérarchique

Le routage hiérarchique permet de partitionner le réseau en sous ensembles de nœuds appelés clusters pour faciliter la gestion des réseaux à grande échelles et préserver par la suite l'énergie. Chaque cluster a un super nœud appelé "cluster-head" chargé de la transmission des messages générés par son cluster aux autres cluster-heads pour atteindre la destination finale (la station de base). le choix de cluster-head diffère d'un protocole de routage hiérarchique à un autre. Il peut être sélectionner à tour de rôle ou selon le nombre de voisins en considérant comme cluster-head le nœud avec le plus de voisins ou bien selon l'énergie résiduelle des noeuds. Nous présentons ici quelques protocoles de routage hiérarchiques :

LEACH

LEACH (Low Energy Adaptive Clustering Hierarchical) [68] est l'une de premières approches de routage hiérarchique les plus populaires pour les réseaux de capteurs. L'idée est de diviser les nœuds de manière distribuée en clusters. Dans LEACH, un nombre réduit de noeuds sont choisis aléatoirement pour être des cluster-heads (CH). Afin de réduire la quantité d'informations transmises à la station de base, les CHs agrègent les données capturées par les noeuds membres appartenant à leur propre cluster, et envoient un paquet agrégé à la station de base.

PEGASIS et Hierarchical-PEGASIS

Au lieu des clusters, PEGASIS (Power-Efficient GAthering in Sensor Information Systems) [90] regroupe les noeuds voisins en une chaîne afin d'éviter le nombre important des messages échangés, necessaires à la formation des clusters. D'une manière périodique, ce protocole choisit un seul noeud de la chaîne pour la transmission des données à la station de base. Les données seront alors agrégées puis transmis de nœud à nœud jusqu'à ce qu'elles atteignent tous la station de base. Bien qu'il évite la méthode de clustering contraignante en terme d'énergie, PEGASIS exige l'ajustement dynamique de la topologie au niveau de chaque nœud ce qui engendre une consommation énergétique importante.

Une extension de PEGASIS a été introduite dans [133] appelée PEGASIS hiérarchique dans le but de diminuer le retard survenu lors de l'acheminement

des paquets à la station de base. Dans cet optique, Certains noeuds sont interdits de transmettre simultanément pour éviter la collision qui est la source majeure du retard. Le protocole PEGASIS hiérarchique construit une chaîne de noeuds, qui se forment un arbre comme la hiérarchie et chaque nœud choisi dans un niveau particulier transmet des données au nœud dans le niveau supérieur de la hiérarchie. Cette méthode assure des transmissions de données en parallèle et réduit le retard significativement.

TEEN

Un protocoles de routage hiérarchique appelé TEEN ((Threshold-sensitive Energy Efficient sensor Network protocol) a été proposé dans [93] pour des applications critiques. L'architecture du réseau est basée sur un groupement hiérarchique à plusieurs niveaux où les noeuds forment des clusters et ce processus va se répéter jusqu'à ce que la station de base soit atteinte. Dans TEEN, Les noeuds sondent le canal continuellement, mais ils transmettent moins fréquemment. Le cluster-head envoie aux membres de leur cluster deux seuils. Le premier seuil permet aux noeuds de transmettre seulement quand la valeur de la grandeur détectée est lui supérieure. Ensuite, un nœud ne retransmet les valeurs relatives aux même donnée que si leur variation par rapport à la valeur transmise précédemment au cluster-head est supérieure au deuxième seuil appelé seuil de variation, fixé aussi par le cluster-head. L'inconvénient principal de ce protocole est que, si les seuils ne sont pas reçus, les noeuds ne communiqueront jamais et l'utilisateur n'obtiendra pas de données du réseau du tout.

4.2.3 Le routage géographique

Les protocoles de routage géographiques n'ont pas besoin de maintenir des tables de routages ou calculer des routes. Ils utilisent seulement les informations de localisation des noeuds pour effectuer le routage . Ils supposent que chaque nœud a la connaissance exacte de sa position, soit grâce à la technologie GPS (Global Positioning System), soit grâce à des méthodes de localisation en estimant par exemple la distance séparant deux noeuds, en fonction des propriétés du signal reçu (temps de propagation, atténuation du signal. . .). Les méthodes de localisation utilisent ces connaissances afin d'en déduire la position estimée des

nœuds du réseau.

GAF

Geographic Adaptive Fidelity [158] est un protocole de routage basé sur la localisation des noeuds. Il est conçu principalement pour les réseaux mobiles ad hoc, mais peut aussi être applicable aux réseaux de capteurs. Dans GAF, la zone à surveiller est divisée en des petites grilles virtuelles de telle sorte que pour deux petites grilles adjacentes A et B, tous les noeuds déployés dans A peuvent communiquer avec tous les noeuds déployés dans B. À l'intérieur de chaque grille, les noeuds se collaborent pour élire un seul nœud. Ce nœud doit rester actif tandis que les autres noeuds de la même grille ferment leurs radios afin d'économiser leurs réserves énergétiques. Le nœud actif assure alors la tâche de surveillance ainsi que l'acheminement de données à la station de base. Ainsi, ce système de partitionnement assure la fidélité du routage tout en préservant les ressources énergétiques de noeuds car il existe au moins un chemin entre un nœud et la station de base.

GEAR

Yu et al. [164] ont proposé l'utilisation d'informations géographiques lors de la diffusion de requêtes aux régions appropriées, car les requêtes de données incluent souvent des informations géographiques. Ce protocole, appelé Geographic and Energy Aware Routing (GEAR), utilise une heuristique de sélection de voisins économe en énergie et basée sur l'information géographique pour acheminer un paquet vers la région destinataire. L'idée principale est de limiter le nombre d'intérêts dans la diffusion dirigée en ne prenant en compte qu'une région donnée plutôt qu'en envoyant les intérêts à l'ensemble du réseau. En faisant cela, GEAR peut économiser plus d'énergie que la diffusion dirigée. Avec GEAR, chaque nœud maintient un coût estimé et un coût d'apprentissage pour atteindre la destination par l'intermédiaire de ses voisins. Le coût estimé est une combinaison d'énergie résiduelle et de distance jusqu'à la destination. Le coût d'apprentissage est un raffinement du coût estimé qui tient compte du routage autour des trous du réseau. Un trou se produit quand un nœud n'a pas de voisin proche par lequel il peut atteindre la région cible autre que lui même. En absence de trous, le coût estimé est égal au coût d'apprentissage. Le coût d'apprentissage se propage d'un saut en arrière à chaque fois qu'un paquet atteint la destination

pour que la configuration de route pour le paquet suivant soit ajustée.

4.2.4 Le routage proactif

Les protocoles proactifs définissent les routes à l'avance indépendamment du modèle de trafic dans le réseau, en se basant sur l'état des liens (distances, nombre de sauts, énergies restantes des noeuds destinataires, etc..) entre un nœud émetteur et les autres nœuds du réseau. Chaque nœud doit mettre à jour régulièrement sa propre table de routage en échangeant des paquets de contrôle avec ses voisins. Par conséquent, un nœud émetteur peut construire à tout moment une route optimale vers n'importe quel autre nœud du réseau en consultant localement sa table de routage. Parmi les protocoles de routage proactifs les plus connus, on peut citer l'OLSR [43] et le DSDV [67].

OLSR (Optimized Link State Routing)

Comme son nom l'indique, ce protocole est une optimisation de l'algorithme classique à état de lien adapté aux besoins d'un réseau local sans fil. L'idée principale d'OLSR est l'emploi des relais multipoints (MPR). Les MPR sont des noeuds sélectionnés qui transmettent des messages de diffusion pendant le processus d'inondation. Cette technique réduit considérablement la surcharge du message par rapport à un mécanisme classique d'inondation. Dans OLSR, les informations sur l'état des liens sont générées uniquement par les nœuds élus en tant que MPR. Ainsi, une deuxième optimisation est obtenue en minimisant le nombre de messages de contrôle inondés dans le réseau. En tant que troisième optimisation, un nœud MPR peut choisir de ne signaler que les liens entre lui et ses sélecteurs MPR. Ainsi, contrairement à l'algorithme classique d'état des liens, des informations partielles sur l'état des liens sont distribuées dans le réseau. Cette information est ensuite utilisée pour le calcul des routes. OLSR fournit des chemins optimaux (en termes de nombre de sauts). Le protocole convient particulièrement aux grands réseaux denses car la technique des MPR fonctionne bien dans ce contexte. . Dans OLSR, chaque nœud diffuse périodiquement deux types de messages : "Hello" et "TC" (Topology Control). Un message Hello contient des informations sur le voisinage du nœud ainsi que son état des liens. Un message TC diffusé sur tout le réseau, contient la liste ds voisins du noeud

qui les a choisi comme MPR. Afin d'éviter l'inondation, Seuls les voisins MPR rediffusent le paquet TC.

DSDV (Destination Sequenced Distance Vector)

DSDV [67] est un protocole proactif de routage à vecteur de distance basé sur l'algorithme de Bellman-Ford [14] et dans lequel les noeuds sont déployés aléatoirement. Dans DSDV, chaque nœud possède une table de routage contenant le saut suivant et le nombre de sauts vers une destination bien précise. La table de routage est actualisée périodiquement. En outre, pour éviter les bouclages, DSDV attribue un numéro de séquence pour chaque route. Ainsi, la route qui dispose d'un numéro de séquence plus grand, sera favorisée.

4.2.5 Le routage réactif

Contrairement aux protocole proactifs, les protocoles réactifs ne calculent les chemins que s'il y' a une demande. Ainsi, le réseau lance une procédure de découverte globale des routes, dans le but d'obtenir une information spécifique, inconnue au préalable. Les routes sont détruites lorsqu'elles ne sont plus utilisées. On dénombre essentiellement deux algorithmes :

AODV (Ad hoc On demand Distance Vector Routing)

Est un protocole de type vecteur de distance basé sur la demande, conçu spécialement pour les réseaux mobiles. Pour créer et découvrir les liaisons entre la source et la destination, AODV [118] utilise un mécanisme des requêtes de type (Route Request/Route Reply). Avec AODV, chaque nœud possède une table de routage comprenant les informations de ses voisins, la table joue un rôle dans le choix d'un voisin qui va transmettre les paquets de la source vers la destination. Quand un noeud veut transmettre un paquet vers une nouvelle destination, il inonnde le réseau par une requête de type Route Request (RREQ). Lorsque la destination ou un noeud ayant un chemin menant à la destination reçoit RREQ, il répond au noeud demandeur par un message de type Route Reply (RREP) en utilisant le chemin inverse. La reception de la réponse (RREP), permet au noeud source d'envoyer ses données. Chaque nœud a un numéro de séquence qui permet de choisir la route la plus récente et de maintenir la consistance des informations de routage.

DSR (Dynamic Source Routing)

DSR [83] est un protocole de routage réactif composé des deux mécanismes Route Discovery et Route Maintenance, qui fonctionnent ensemble permettant aux nœuds de découvrir et de maintenir le routage de la source vers une destination arbitraire dans le réseau. Bien qu'il est similaire à AODV dans le mesure où il forme une route sur demande, DSR utilise le routage source au lieu de compter sur la table de routage de chaque noeud intermédiare. L'utilisation du routage source permet au routage de paquets d'être dépourvu de boucles de manière triviale, évite le besoin d'informations de routage mises à jour dans les nœuds intermédiaires par lesquels les paquets sont transférés. De plus, en incorporant cette route source dans l'en-tête de chaque paquet de données, d'autres nœuds transférant ou entendant l'un de ces paquets peuvent également cacher cette information de routage pour une utilisation future. Le fonctionnement de ce protocole est basé entièrement sur la demande, ce qui permet à la surcharge de paquets de routage de DSR de s'adapter automatiquement à celle requise uniquement pour réagir aux modifications des itinéraires actuellement utilisés. Cependant, DSR ne dispose pas de mécanisme pour savoir quelle route dans le cache est périmée et si le paquet de données est déjà transféré au lien brisé.

4.2.6 Solutions de routage pour la tolérance aux fautes

Comme ils sont, généralement, déployés dans des milieux hostiles, les capteurs sont sujets à des pannes à cause de l'épuisement de leurs batteries, ou l'écrasement par des animaux, etc. Il résulte de l'occurrence des pannes, une difficulté pour acheminer les données collectées à la station de base. Plusieurs protocoles de routage tolérants aux pannes ont été proposés dans la littérature pour remédier à cette problématique dont on peut citer :

Algorithme PEQ (Periodic, Event-driven, Query-based)

PEQ [24] cherche à satisfaire les contraintes majeurs d'un réseau de capteurs : faible latence, fiabilité, recouvrement rapide en cas de panne et conservation d'énergie. La combinaison de routage multi-saut avec la conservation d'énergie permet à PEQ de choisir parmi toutes les routes disponibles, celles qui consomment moins d'énergie. En plus de ce mécanisme préventif qui permet un

routage fiable avant l'occurrence de panne, un mécanisme de recouvrement de pannes est implémenté. Avec ce mécanisme, la route en panne sera remplacée par une autre route qui soit de liens fiables et consomme moins d'énergie. Le paradigme "Publish/Subscribe" est introduit par le protocole PEQ afin d'assurer l'interaction entre le puits et l'ensemble des noeuds déployés. En effet, ces noeuds envoient des notifications d'événements au puits, qui va souscrire son intérêt pour certaines de ces informations. L'information désirée sera publiée par la suite par les noeuds concernés.

Algorithme CPEQ (Cluster based PEQ)

Le protocole CPEQ [25] vient pour enrichir PEQ en adoptant une approche de clustering afin d'offrir une meilleure gestion de routage. En effet, les noeuds ayant le plus d'énergie résiduelle sont sélectionnés comme noeuds agrégateurs (cluster-heads). Chaque cluster-head établit son cluster, et les noeuds appartenant à ce dernier envoient leurs données au cluster-head qui effectue d'éventuel traitement sur les données avant de les acheminer vers le puits. Chaque nœud du réseau peut devenir cluster-head pendant une certaine période de temps selon son niveau d'énergie. Le but principal de CPEQ est de distribuer d'une manière uniforme la dissipation d'énergie entre les noeuds, et de réduire la latence et le trafic de données dans le réseau.

REAR (Reliable Energy Aware Routing)

Un autre protocole de routage tolérant aux pannes est proposé dans [139] utilisant aussi des mécanismes de conservation d'énergie. REAR sélectionne en avance deux routes : une route principale et une route de secours. Lorsque la route principale tombe en panne, REAR utilise la route de secours pour retransmettre les données immédiatement tout en évitant considérablement le retard de transmission dû à la non disponibilité de la route principale. Cependant, cette solution n'est pas très efficace dans le cas où la route principale reste fonctionnelle puisque les informations de la route de secours seront stockées initulement au niveau des noeuds.

VTRP (Variable Transmission Range Protocole)

VTRP [8] consiste à varier le rayon de transmission pour une meilleure propagation de données. Il cherche à éviter le problème d'obstacles par l'augmentation du rayon de transmission. Ce qui lui permet d'atteindre des noeuds actifs dans

le cas où le rayon actuel utilisé ne couvre aucun nœud à cause de pannes ou d'inactivité des noeuds voisins ou encore dans le cas des réseaux à faible densité. En outre, VTRP offre une meilleure longévité du réseau en évitant l'utilisation fréquente des noeuds critiques (les voisins proches de la station de base) ceci permet d'alléger leur fonction de routage, conserve leurs batteries et augmente ainsi la durée de vie de tout le réseau.

4.3 Discussion et modèle retenu

A travers l'étude présentée, plusieurs points sont à retenir :

— dans la majorité des protocoles de routage hiérarchiques, spécialement LEACH, la sélection des Cluster-heads (CH) à chaque période est réalisée de manière aléatoire. En effet, cette opération sauve un nœud de ne pas épuiser sa batterie mais la rotation aléatoire de rôle de CH reste toujours une opération énergivore puisque une énergie supplémentaire est nécessaire pour échanger les messages de contrôle permettant de choisir le nouveau CH ainsi que la nouvelle formation de cluster ;

— contrairement aux protocoles à topologie hiérarchique, les noeuds exécutant un protocole à topologie plane assurent tous un comportement similaire. En effet, chaque nœud du réseau doit participer au routage de données. Cette approche permet ainsi d'exploiter tous les liaisons de la topologie physique ainsi que garantir une charge de trafic homogène entre les différents noeuds. Cependant, l'implication d'un nombre de noeuds assez important dans le processus de routage de données peut engendrer un trafic de contrôle très élevé.

— les protocoles de routage géographique ne sont pas adaptés à toutes les applications de réseaux de capteurs. Dans ce type de protocoles, les noeuds doivent connaître leurs localisations via GPS ou autre méthode de localisation. Ceci engendre une grande consommation des capacités énergétiques des noeuds. De plus le routage géographique ne cherche pas à établir une équité entre les noeuds de point de vue de l'énergie restante. En effet, un nœud épuisé peut encore être choisi pour l'acheminement du message

quand il est le plus proche de la destination. Ce qui provoque la minimisation de la durée de vie de ces noeuds critiques en particulier et du tout le réseau en général ;

— les protocoles de routage proactifs nécessitent une mise à jour permanente des tables de routages comprenant en outre des informations qui ne seront pas utilisées. De plus, le changement de la topologie suite à une panne de nœud provoque la mise à jour de table de routage toute entière. Ceci ne convient pas aux applications exigeant un court temps de latence. Par contre, l'important avantage de protocoles proactifs est qu'ils n'introduisent aucun délai avant de transmettre un message puisque la route est déjà définie ;

— La politique de recherche des routes suivie par les protocoles de routage à la demande génère une lenteur significative ce qui engendre la dégradation des performances de certaines applications. Ce type de protocole peut être considéré très coûteux en transmission de messages lors de la détermination des routes. Par contre, il n'y a pas une route prédéfinie, la route est maintenue que si nécessaire. Ces protocoles évitent alors d'encombrer les tables de routages de leurs noeuds par des informations inutilisées.

— la majorité des protocoles de routage proposés dans la littérature se concentrent généralement sur l'atteinte d'un seul objectif : garantir une basse consommation énergétique lors de la transmission de données. Le concept derrière cette méthode est de reporter la défaillance de noeuds autant que possible. Mais ceci n'est pas assez satisfaisant puisque l'opération globale du réseau n'est pas garantie suite à l'occurrence de pannes qui sont inévitables. À notre connaissance, la plupart de protocoles de routage tolérant aux pannes proposés dans la littérature utilisent soit la technique de retransmission du paquet sur un chemin alternatif ou la réplication en envoyant plusieurs copies des mêmes données sur des chemins différents. Bien que ces techniques tolèrent la panne d'un certain nombre de noeuds ou de chemins, elles engendrent un surcoût très élevé dû au routage multi-chemin, ce qui augmente considérablement la consommation de l'énergie et de la bande passante.

Cette discussion dégage les points qui justifient le développement du protocole de routage proposé. Nous proposons à travers ces travaux un protocole de routage qui :

— utilise de manière adéquate les ressources sévèrement limitées des noeuds en effectuant tout le calcul nécessaire à la détermination des chemins optimaux, au niveau de la station de base tout en minimisant le rôle des nœuds dans la construction des tables de routage ;

— prend en considération plus d'un seul critère pour optimiser l'établissement des routes, à savoir le coût de transmission de message en terme de distance qui sépare chaque nœud émetteur de la station de base (nombre de sauts intermédiaires), tout en évitant de sélectionner les noeuds critiques (contraints en énergie) ;

— est à la fois proactif et tolérant aux pannes de noeuds, il doit mettre à la disposition de chaque nœud plus d'un chemin. Ce point permet au nœud de reprendre un fonctionnement correct suite à l'occurrence de panne au niveau de l'une de destinations sélectionnées sans nécessiter le lancement immédiat d'une nouvelle phase de détermination de chemin ;

— évite la manipulation des tables de routage enregistrant tout le chemin, afin de s'adapter aux capacités de stockages limitées des noeuds, et ne sauvegarde que des informations locales représentant des destinations à un seul saut. Ainsi le chemin complet sera obtenu par raisonnement récursif, permettant d'élaborer une règle de décision locale à chaque nœud ;

— est correct par construction et généré de façon automatique en se basant sur la technique de la SDC optimale, de telle sorte qu'un contrôleur centralisé est calculé au niveau de la station de base qui peut fournir les chemins les plus courts en terme de distance et les plus efficaces de point de vue énergétique.

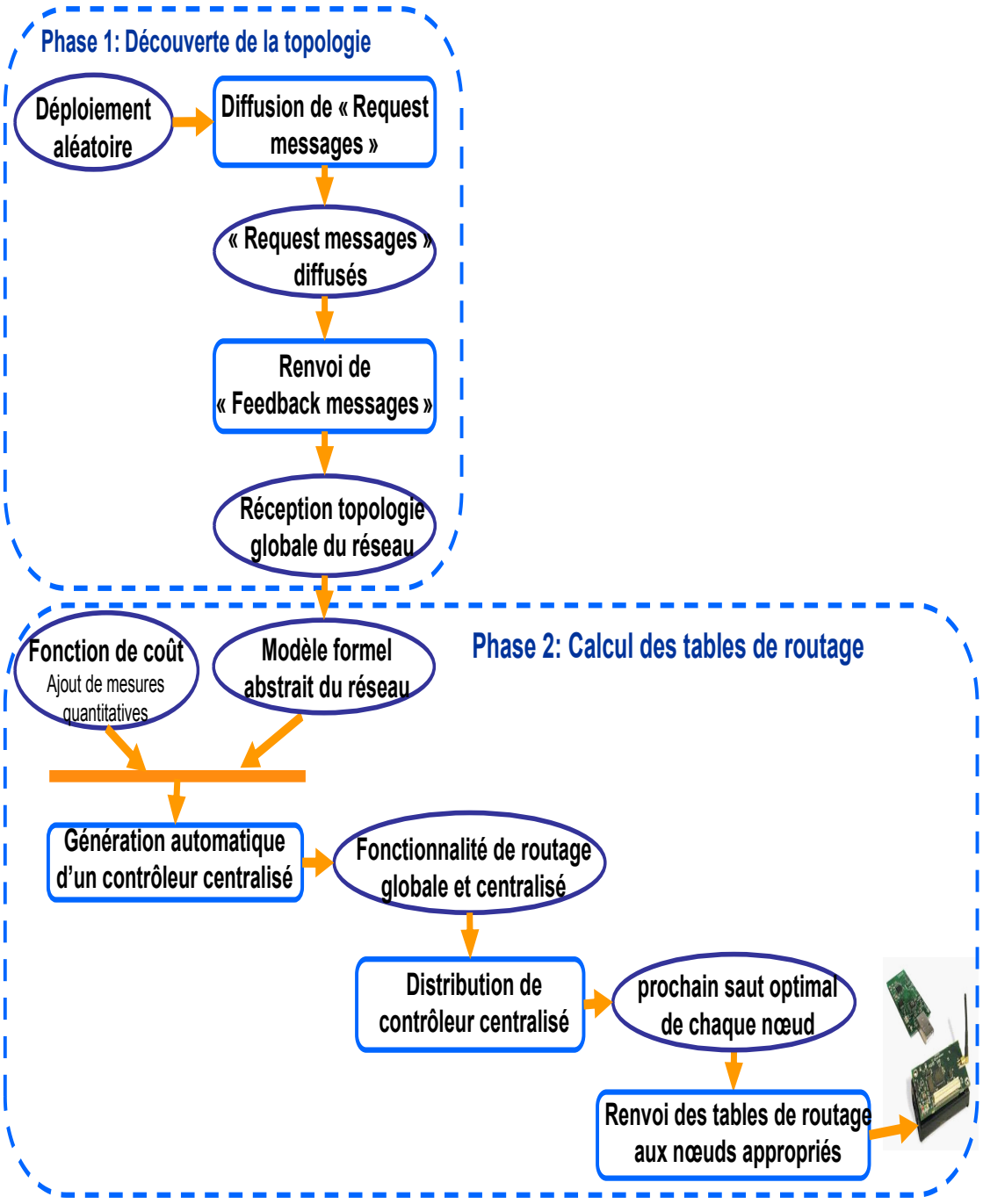

FIGURE 4.1 – une vue d'ensemble de la méthode adoptée pour la génération du protocole de routage optimal

4.4 Routage optimal basé sur deux critères hiérarchisés

4.4.1 Hypothèses

Avant d'introduire les différentes phases de conception du protocole de routage proposé, nous devons mentionner les hypothèses suivantes :

— les nœuds-capteurs sont déployés manuellement ou de manière aléatoire dans une zone cible ;

— tous les nœuds-capteurs sont stationnaires après leur déploiement ;

— nous nous intéressons aux RCSFs avec une seule station de base vers laquelle tous les messages seront acheminés. Cette station de base représente le seul point d'accès au réseau. Ainsi les différents chemins connectant les noeuds à la station de base forment un arbre ayant cette dernière comme racine ;

— toutes les communications sont effectuées à travers des canaux sans fil et une liaison sans fil est établie entre deux nœuds si et seulement s'ils sont à portée de communication les uns des autres ;

— les nœuds du réseau sont numérotés avec des indices distincts et chaque nœud connaît son indice. Pour faciliter la notation, la station de base sera identifiée par 0.

4.4.2 Description du protocole de routage proposé

Le protocole de routage optimal proposé est réalisé en deux phases principales. La figure 4.1 donne une vue d'ensemble de méthode adoptée pour aboutir à ce protocole de routage optimal. Au cours de la première phase, après le déploiement de tous les noeuds dans la zone cible, la station de base procède à collecter la topologie du réseau en diffusant d'abord un message de type "*Request_message*" à tous les noeuds atteignables dans sa portée de communication. Ce message sera inondé par la suite dans tout le réseau. En réponse à ce "*Request_message*", chaque nœud devrait envoyer les informations de sa

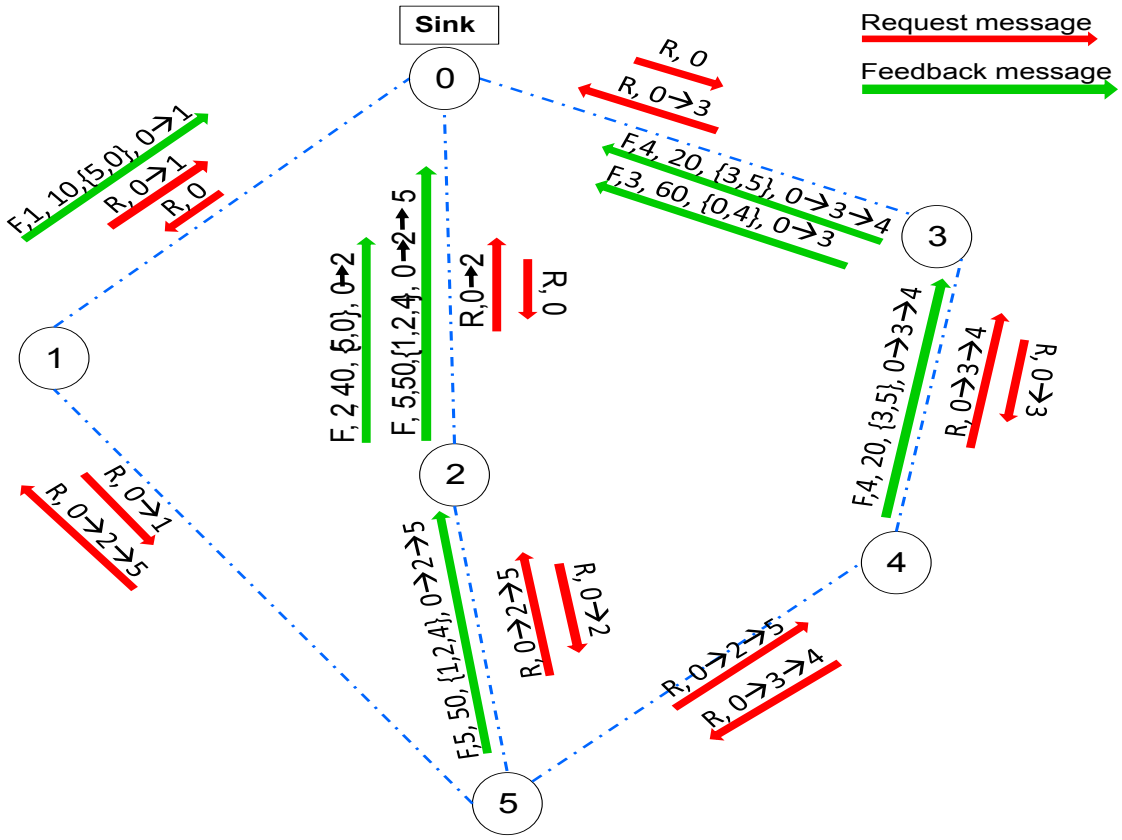

FIGURE 4.2 – Découverte de la topologie du réseau

topologie locale à la station de base en utilisant le "*Feedback_message*". Au cours de la deuxième phase, la station de base calcule les tables de routage de tous les nœuds de réseau en se basant sur les informations reçues durant la première phase.

Le réseau est vu comme une interconnexion de nœuds représentant chacun un cluster unique. Il est modélisé par un graphe de connectivité, exprimé par une une machine d'états finis (FSM). Chaque noeud du réseau est représenté par un état. Les transitions représentent des connexions de point à point entre deux nœuds.

Des poids sont associés à chaque noeud. Il s'agit de quantités positives exprimant soit la distance jusqu'à la station de base, soit le niveau d'énergie résiduel du capteur. Ainsi, il est possible d'associer des mesures de distance et d'efficacité énergétique aux routes suivies pour transmettre un message à la station

de base. Ces mesures peuvent par la suite être optimisées : minimisation de la distance, maximisation de l'efficacité énergétique. Cela est obtenu par la génération automatique d'un contrôleur centralisé, en utilisant la technique de la SCD optimale. Ce contrôleur centralisé est ensuite distribué de manière à fournir des éléments de choix locales à chaque nœud pour sélectionner uniquement son voisin optimal de prochain saut pour atteindre la station de base.

Phase de découverte de la topologie

Inondation de *Request_message* **:**

Afin de découvrir la topologie du réseau considéré, la station de base initie cette phase en diffusant un "*Request_message*" qui sera reçu par tous les noeuds situés à sa proximité. Ceci peut se produire au moment de déploiement des noeuds ou lorsque la topologie du réseau a été changée à cause des pannes des certains nœuds (suite à l'épuisement de leurs énergies) ou à cause de déploiement de nouveaux nœuds. Un "*Request_message*" diffusé par un nœud x inclut un chemin de la station de base à x. Lorsqu'un nœud reçoit un "*Request_message*" pour la première fois, il rediffuse à son tour ce message à tous ses voisins après avoir ajouté son identité au chemin. Il enregistre également l'identité de l'expéditeur de ce message à son ensemble de voisins appelé "$Neighbor\_set$". Lorsqu'un nœud reçoit un "*Request_message* en double, l'identité de l'expéditeur est ajoutée à son ensemble de voisins, mais le message ne sera pas rediffusé. Le format d'un "*Request_message*" est composé de deux champs à savoir le champs "$Type$" et le champs "$Path$". Comme un nœud peut recevoir deux types de messages, soit un "*Request_message*" dénoté par "R" soit un "$Feedback\_message$" dénoté par "F", le premier champs d'un "*Request_message*" est réservé au type afin qu'un nœud puisse différencier les messages reçus. Le deuxième champs "$Path$" contient le chemin (séquence d'identités de nœuds) de la station de base jusqu'à ce nœud destinataire (le nœud qui vient de recevoir le "*Request_message*").

La figure 4.2 illustre ce processus de diffusion qui sera répété de saut en saut jusqu'à ce que tous les nœuds du réseau seront tous notifiés. Prenons l'exemple de nœud 4 qui va recevoir le "*Request_message*" suivant : $(R, 0 \rightarrow 3)$ pour la première fois de la part de nœud 3. Ainsi, il ajoute le nœud 3 à son

ensemble de voisins et rediffuse ce "*Request_message*" en ajoutant son identité au chemin. Le "*Request_message*" diffusé par le nœud 4 a alors le format suivant : $(R, 0 \rightarrow 3 \rightarrow 4)$. Ce "*Request_message*" va être reçu par les noeuds 3 et 5. Ainsi, ces deux noeuds vont enregistrer le nœud 4 dans leurs ensembles de voisins mais ils vont ignorer la rediffusion de "*Request_message*".

La propagation de "*Request_message*" de cette manière a trois objectifs :

— elle informe tous les nœuds que la station de base est entrain de collecter les informations de topologie pour construire les tables de routage ;

— elle aide à construire un chemin depuis chaque nœud jusqu'à la station de base qui va être utilisé par la suite pour transmettre le "*Feedback_message*" à la station de base ;

— un nœud recevant un "*Request_message*" apprend que l'expéditeur de ce message est son voisin.

Renvoi de "*Feedback_message*" à la station de base :

Après qu'un nœud termine la diffusion du "*Request_message*", il doit renvoyer les informations de sa topologie locale à la station de base en utilisant un "*Feedback_message*". Néanmoins, il doit attendre un certain délai, avant de générer ce message. Cette période de temps permet à un nœud d'écouter les émissions de ses voisins, qui transmettent également le même "*Request_message*". Comme l'architecture du réseau considéré se rassemble à un arbre ayant la station de base comme racine, chaque nœud reçoit plusieurs copies du "*Request_message*" de ses voisins situés en amont par rapport à sa position dans l'arbre, de ceux qui sont de même niveau que lui, ainsi que de ceux qui ont situés en aval. Cependant, il enregistre uniquement le chemin à travers le premier voisin situé en amont qui lui est transféré le "*Request_message*". Ce voisin en amont est marqué comme le parent du nœud en question. Par exemple, si le nœud x reçoit le premier "*Request_message*" d'un nœud voisin c, alors c devient le parent de x. Il est donc très simple pour le "*Feedback_message*" de suivre le chemin inverse pris par le premier "*Request_message*" reçu pour atteindre la station de base.

La figure 4.2 illustre la structure d'un "*Feedback_message*" renvoyé par chaque nœud à la station de base. Il contient ainsi les six champs suivants : le premier champs "*Type*" à travers lequel le nœud peut vérifier s'il s'agit bien

d'un "*Feedback_message*" à transférer vers la station de base dénoté par "*F*". Un deuxième champs pour l'identité de l'expéditeur désignée par "*Sender_Id*", un troisième champs réservé pour le niveau d'énergie résiduelle de l'expéditeur dénoté par "*Residual_energy*", un quatrième champs pour enregistrer ses informations de voisinage (un ensemble d'identités de tous ses voisins, dénoté par "*neighbors_set*") ainsi que le chemin vers ce nœud passant par son parent, désigné par le dernier champs "*path_sequence*". Le tableau 4.1 détaille les différents messages de type "*Feedback_message*" renvoyés par les noeuds du réseau de la figure 4.2 à leur station de base. Par exemple, le nœud 5 a 50 comme niveau d'énergie résiduelle, les noeuds 1, 2 et 4 comme voisins, ainsi que le chemin $(0 \rightarrow 2 \rightarrow 5)$ comme "*path_sequence*". Dans cet exemple, 2 est le parent de 5 puisqu'il est le premier nœud qui lui envoyait le "*Request_message*".

Tableau 4.1 – Les différents "*Feedback_messages*" renvoyés à la station de base

Sender_Id	Residual_energy	Neighbors_set	Path_sequence
1	10	5, 0	$0 \rightarrow 1$
2	40	5, 0	$0 \rightarrow 2$
3	60	4, 0	$0 \rightarrow 3$
4	20	3, 5	$0 \rightarrow 3 \rightarrow 4$
5	50	1, 2, 4	$0 \rightarrow 2 \rightarrow 5$

Phase de calcul des tables de routage

Après la réception de toute la topologie du réseau, la station de base peut maintenant calculer la table de routage optimale respective à chaque nœud durant cette deuxième phase. En effet, l'objectif principal du protocole de routage développé est de considérer simultanément deux contraintes distinctes de manière hiérarchique, et ce, dans le but d'augmenter la durée de vie du réseau. Les deux critères retenus sont basés sur le concept d'énergie. Comme la transmission de données dans un réseau de capteurs est généralement effectuée de saut en saut, les chemins retenus doivent ainsi minimiser le coût de transmission d'un message depuis un nœud-capteur jusqu'à la station de base en terme de nombre de sauts intermédiaires necessaires pour atteindre cette station de base. De plus, plusieurs travaux existants ont confirmé que l'équilibrage de charges

énergétiques entre les noeuds permet de prolonger la durée de vie du réseau. Pour ce faire, les chemins retenus doivent aussi éviter les noeuds ayant des réserves énergétiques assez consommées. En d'autres termes, à chaque fois qu'il y a deux ou plusieurs chemins égaux en terme de distance, celui qui offre un maximum de résidus énergétiques sera retenu. Ainsi, les critères retenus pour la détermination d'un chemin optimal sont :

— le nombre de sauts total d'un chemin (nombre de noeuds intermédiaires formant le chemin multi-sauts). Ce critère permet de définir le coût de transmission (délai ou taux d'énergie) nécessaire pour transmettre un message depuis un nœud-capteur jusqu'à la station de base. Ce critère est à minimiser ;

— l'énergie résiduelle tout au long du chemin connectant le nœud à la station de base. Cet objectif représente le minimum des énergies résiduelles rencontrées au niveau des noeuds formant le chemin. Cet objectif est à maximiser dans le but d'éviter les chemins passant par des noeuds ayant des réserves énergétiques épuisées.

Ces deux critères feront l'objet de deux étapes successives d'optimisation, utilisant chacune un algorithme spécifique de synthèse optimale : l'un pour la minimisation des chemins, l'autre pour la maximisation de l'énergie résiduelle. Les tables de routage sont obtenues par distribution du contrôleur obtenu, et sont diffusées vers les nœuds respectifs en utilisant des messages de mise à jour de routage. Le calcul du contrôleur centralisé ainsi que sa distribution sont présentés dans la suite.

4.4.3 Modèle formel abstrait d'un réseau de capteurs

L'approche proposée s'appuie sur une abstraction du réseau de capteurs considéré : un graphe orienté étiqueté, où les sommets modélisent les nœuds du réseau et les arêtes modélisent les canaux de communication point à point. L'avantage de l'abstraction proposée est qu'elle est isomorphe au modèle d'une machine à états finis (FSM), capable de modéliser à la fois la structure du réseau ainsi que l'acheminement d'un message d'un nœud donné à la station de base.

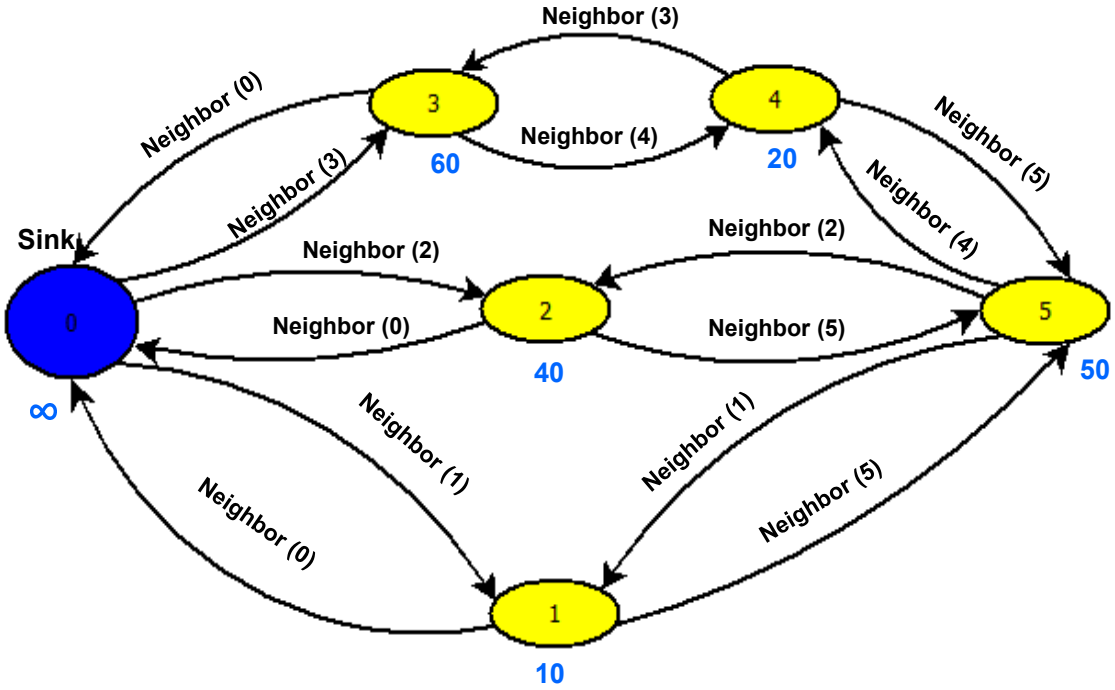

FIGURE 4.3 – Modèle formel abstrait d'un réseau de capteurs

Comme illustré sur la figure 4.3, plusieurs étiquettes sont ajoutées aux sommets et aux arêtes afin de modéliser les fonctionnalités suivantes :

— l'identification de nœud (y compris la station de base) et l'énergie résiduelle sont des étiquettes de sommets, exprimés de façon quantitative ;

— l'identification de canal (et donc, l'identification du voisin) sont des étiquettes sur les arêtes exprimées sous forme de prédicats booléens de décision.

Exemple. Le réseau modélisé par la figure 4.3 est composé ainsi de 5 nœuds stationnaires identifiés respectivement par 1, 2, 3, 4, 5 et un nœud Puits étiqueté "Sink" et identifié par 0. Les coûts statiques sont représentés par des nombres entiers à côté de états correspondants. La valeur bleue au-dessous de chaque état du modèle indique la réserve d'énergie de chaque nœud. Comme le nœud Puits se caractérise par une réserve d'énergie illimitée, nous modélisons son niveau d'énergie résiduel par la valeur ∞. Les différents chemins reliant les noeuds au puits forment un arbre ayant ce dernier comme racine.

Tableau 4.2 – Tous les chemins possibles pour atteindre la station Puits

Nœud	Chemin	nombre de sauts	énergie résiduelle minimale
1	$(1 \rightarrow 0)$	1	∞
	$(1 \rightarrow 5 \rightarrow 2 \rightarrow 0)$	3	40
	$(1 \rightarrow 5 \rightarrow 4 \rightarrow 3 \rightarrow 0)$	4	20
2	$(2 \rightarrow 0)$	1	∞
	$(2 \rightarrow 5 \rightarrow 1 \rightarrow 0)$	3	10
	$(2 \rightarrow 5 \rightarrow 4 \rightarrow 3 \rightarrow 0)$	4	20
3	$(3 \rightarrow 0)$	1	∞
	$(3 \rightarrow 4 \rightarrow 5 \rightarrow 2 \rightarrow 0)$	4	20
	$(3 \rightarrow 4 \rightarrow 5 \rightarrow 1 \rightarrow 0)$	4	10
4	$(4 \rightarrow 3 \rightarrow 0)$	2	60
	$(4 \rightarrow 5 \rightarrow 2 \rightarrow 0)$	3	40
	$(4 \rightarrow 5 \rightarrow 1 \rightarrow 0)$	3	10
5	$(5 \rightarrow 2 \rightarrow 0)$	2	40
	$(5 \rightarrow 1 \rightarrow 0)$	2	10
	$(5 \rightarrow 4 \rightarrow 3 \rightarrow 0)$	3	20

La table 4.2 illustre les différents chemins pouvant être suivis pour atteindre la station puits, le nombre de sauts intermédiaires ainsi que l'énergie résiduelle minimale rencontrée sur chaque chemin. Ainsi, le nœud 5 peut "atteindre" la station de base en 3 sauts à travers le nœud 4 en suivant le chemin $(5 \rightarrow 4 \rightarrow 3 \rightarrow 0)$ ou en 2 sauts seulement en suivant l'un de ces deux chemins $(5 \rightarrow 2 \rightarrow 0)$ ou $(5 \rightarrow 1 \rightarrow 0)$. Le deux derniers chemins sont plus efficaces en termes de distance puisqu'ils offrent un nombre minimal de sauts vers la station de base par rapport au chemin passant par le nœud 4. Cependant, le chemin $(5 \rightarrow 2 \rightarrow 0)$ est plus efficace en termes d'énergie que $(5 \rightarrow 1 \rightarrow 0)$ car il permet d'éviter le nœud 1 qui se caractérise par une réserve énergétique critique. Ainsi, le nœud 5 doit enregistrer dans sa table de routage le nœud 2 comme prochain saut optimal lors de l'acheminement de donnés à la station de base.

Dans la section suivante, nous montrons comment ce choix est déterminé automatiquement en se basant sur la technique de la SCD optimale.

4.4.4 Génération automatique d'un contrôleur centralisé basé sur deux critères

Le principe de cette étape est celui d'une optimisation à un pas, à l'échelle du graphe de connectivité représentant le réseau. On cherche à synthétiser un contrôleur global, puis à le distribuer sur chaque nœud du réseau, de manière à contraindre le choix des nœuds successeurs en se limitant à ceux ayant un poids optimal (minimal ou maximal). Les poids à minimiser sont des mesures agrégées le long des chemins du réseau, lors d'une opération préliminaire. Appliquons ce principe au réseau de capteurs étudié. Un superviseur est automatiquement synthétisé à partir de la topologie du réseau reçue par "*Feedback_message*". La mise en œuvre de ce superviseur garantit l'optimisation d'un critère :

— minimisation du nombre de sauts intermédiaires entre chaque nœud et la station de base, ou

— maximisation des résidus énergétiques de noeuds rencontrés tout au long des différents chemins.

En d'autres termes, on cherche à sélectionner le chemin le plus court et en même temps celui dont le maillon le plus faible en terme d'énergie résiduelle reste le plus fort.

L'ordre d'application de ces critères d'optimisation est important, et n'engendre pas les mêmes résultats. L'approche décrite dans la suite de ce chapitre fait le choix d'acheminer un message vers le puits en utilisant qu'un nombre minimal de nœuds, en créant un routage à longueurs de chemins minimaux. Sur les chemins restants, une optimisation énergétique est appliquée. La priorité de ces deux critères peut être inversée, sans impact sur la description de l'approche.

Le superviseur est généré au niveau de la station de base, suite à la réception de la topologie globale du réseau. Il est donc centralisé. Ce superviseur centralisé est distribué par la suite pour fournir à chaque nœud des éléments de choix locaux pour déterminer son prochain saut optimal vers la station puits où le critère d'optimalité est respecté globalement.

Premier objectif : Minimiser le nombre de sauts intermédiaires

En appliquant l'opération de synthèse, un superviseur est calculé qui n'autorise que les transitions allant aux états suivants ayant la valeur la plus faible du poids approprié : le nombre minimal de sauts vers la station puits.

Synthèse des chemins de longueur minimale

Il prend en entrée les éléments d'une FSM décrivant la topologie du réseau, ainsi qu'une fonction de coût :

— Q : les états du modèle (nœuds du réseau)

— $P \in Q$: l'état final du modèle (la station Puits)

— X : enselble des variables du modèle, variable d'entrée exprimée en prédicats booléens de décision

— f : la fonction de transition conduisant à l'état suivant q' tel que $q' = f(q, x)$

— $C : Q \to \mathbb{R}^+$, la fonction de coût associant à chaque nœud sa "contribution" à la longueur globale d'un chemin. Dans ce contexte, nous avons $C(q) = 1, \forall q \in Q$ car chaque nœud compte pour un saut dans le calcul de la distance.

En sortie, le superviseur $MIN\_PATH \subset Q \times Bool^{|X|}$ est l'ensemble des nuplets état-vecteur d'entrées menant vers P le long d'un chemin de longueur minimale.

Le calcul de cet algorithme est effectué en deux étapes :

1. calculer $D : Q \to \mathbb{R}^+$. Associer ainsi à chaque nœud $q \in Q$ la mesure $D(q)$ de la longueur du plus court chemin séparant q de P. D est définie comme le plus grand point fixe des équations récurrentes suivantes :

$$D_0(q) = \begin{cases} 0 & si\ q = P \\ +\infty & Otherwise \end{cases} \tag{4.1}$$

Répéter :

$$D_{i+1}(q) = min \begin{cases} C(q) + \min\limits_{x}(D_i(f(q,x))) \\ D_i(q) \end{cases} \tag{4.2}$$

jusqu'à ce que : $D_{i+1}(q) = D_i(q), \forall q \in Q$

2. La deuxième étape consiste, à partir de D, à construire le superviseur MIN_PATH qui pilotera le système de manière à générer les meilleures trajectoires possibles atteignant P. En d'autres termes, pour tout état $q \in Q$, la politique de contrôle consiste en le calcul du meilleur successeur immédiat $q' = f(q,x)$ ayant le moindre coût $D(q')$, en jouant sur la variable booléenne de décision x tel que $D(q')$ est le plus petit :

$$MIN\_PATH = \{(q,x) \in Q \times Bool^{|X|} : D(q) > D(f(q,x)) \text{et} \\ D(f(q,x)) = \min_{y \in Bool^{|X|}} (D(f(q,y)))\} \quad (4.3)$$

Il a été prouvé que sur des systèmes finis avec des coûts positifs sur les états, cet algorithme est certain de terminer par une complexité polynomiale en la taille du système [135] [92]. Le plus grand point fixe obtenu est la solution optimale du problème de synthèse optimale.

Exemple. Par application de cet algorithme, sur l'exemple de la figure 4.3, le superviseur MIN_PATH obtenu contient les n-uplets (q, x) suivants :

$$MIN\_PATH = \{(1, Neighbour(0)), (2, Neighbour(0)), (3, Neighbour(0)), \\ (4, Neighbour(3)), (5, Neighbour(2)), (5, Neighbour(1))\} \quad (4.4)$$

Ainsi, il autorise par exemple au nœud 5 de choisir soit le nœud 1, soit le nœud 2 comme prochain saut optimal car les deux noeuds ont un nombre minimal de sauts équivalent vers la station Puits. Cependant, il inhibe la transition conduisant au nœud 4 puisque ce dernier est situé sur un chemin plus long en terme de distance (3 sauts intermédiaires).

Le tableau 4.3 illustre les chemins à distance optimale qui peuvent être empruntés par les nœuds du réseau pour atteindre la station Puits, chemins garantis par l'utilisation du superviseur MIN_PATH.

Tableau 4.3 – Chemins optimaux générés basés sur le critère de nombre minimal de sauts

nœud	chemin	nombre minimal de sauts
1	$(1 \rightarrow 0)$	1
2	$(2 \rightarrow 0)$	1
3	$(3 \rightarrow 0)$	1
4	$(4 \rightarrow 3 \rightarrow 0)$	2
5	$(5 \rightarrow 2 \rightarrow 0)$ $(5 \rightarrow 1 \rightarrow 0)$	2

Deuxième objectif : équilibrer les réserves énergétiques restantes

Si un nœud cesse de fonctionner en raison d'un manque d'énergie, les chemins de routage via ce nœud seront complètement perdus ; ceci dégrade ainsi la performance du réseau. Pour éviter cette situation "catastrophique", l'équilibre énergétique entre les nœuds est une caractéristique souhaitable lors de la conception d'un algorithme de routage. Pour prolonger autant que possible la durée de vie du réseau, les nœuds ayant des faibles réserves énergétiques doivent être évités en tant que voisins de destination, s'il existe d'autres alternatives énergétiques résiduelles plus élevées. En outre, selon le premier critère, il peut y avoir plusieurs successeurs avec un nombre minimal de sauts. Par conséquent, parmi les chemins optimaux, il est important de ne privilégier que ceux le long desquels l'énergie restante dans chaque nœud est la plus élevée. Ce critère s'appuie sur une notion d'efficacité énergétique d'un état q par rapport à un ensemble de chemins menant de q à P. Il s'agit d'une fonction de coût fortement impactée par l'existence de "maillons faibles" le long d'un chemin. Ainsi, l'efficacité énergétique d'un nœud disposant d'importantes réserves d'énergie peut être faible, s'il existe un nœud en aval possédant une efficacité faible, dûe à sa propre réserve d'énergie, ou à l'efficacité mesurée encore plus en aval. L'optimisation consiste ensuite à choisir parmi tous les chemins possibles celui ou ceux dont le maillon faible est le plus fort.

Ce processus est centralisé, et repose donc sur une récupération des informations du réseau au niveau du puits en suivant les étapes suivantes :

1. récupérer les niveaux d'énergies résiduelles de tous les nœuds ;

2. calculer l'efficacité énergétique de chaque nœud ;
3. calculer le superviseur MAX_EN associant à chaque nœud uniquement les successeurs dont l'efficacité énergétique est la meilleure ;
4. distribuer MAX_EN sur chaque nœud.

L'algorithme de calcul du superviseur est détaillé ci-dessous.

Synthèse des chemins à haute efficacité énergétique.

Il prend en entrée les éléments d'une FSM décrivant la topologie du réseau, ainsi qu'une fonction de coût :

— Q : les états du modèle (nœuds du réseau)

— $P \in Q$: l'état final du modèle (la station Puits)

— X : ensemble des variables du modèle, variable d'entrée exprimée en prédicats booléens de décision

— f : la fonction de transition conduisant à l'état suivant q' tel que $q' = f(q, x)$

— $E : Q \rightarrow \mathbb{R}^+$, la fonction de coût associant à chaque nœud son niveau d'énergie résiduelle.

En sortie, le superviseur $MAX\_EN \subset MIN\_PATH$ est l'ensemble des n-uplets état-vecteur d'entrées menant vers P le long d'un chemin optimal de meilleure efficacité énergétique.

Le calcul de cet algorithme est effectué aussi en deux étapes :

1. calculer $R : Q \rightarrow \mathbb{R}^+$. Associer ainsi à chaque nœud $q \in Q$ la mesure $R(q)$ de son efficacité énergétique pour atteindre P. R est définie comme le plus grand point fixe des équations récurrentes suivantes :

$$R_0(s) = \begin{cases} -\infty & si\ q \neq P \\ +\infty & if\ q = P \end{cases} \tag{4.5}$$

Répéter

$$R_{i+1}(q) = max \begin{cases} min(E(q),\ \underset{x}{max}(R_i(f(q,x)))) \\ R_i(q) \end{cases} \tag{4.6}$$

jusqu'à ce que $R_{i+1}(q) = R_i(q) \forall q \in Q$

2. La deuxième étape consiste, à partir de R, à construire le superviseur MAX_EN. Pour tout état s, la politique de contrôle consiste en le calcul du meilleur successeur immédiat $q' = f(s,x)$ ayant le meilleur niveau d'énergie, en jouant sur la variable booléenne de décision x tel que R (q') est le plus grand :

$$MAX\_EN = \{(q,x) \in MIN\_PATH \times Bool^{|X|} : R(f(q,x)) = \max_{y \in Bool^{|X|}} (R(f(q,y)))\} \tag{4.7}$$

Exemple. Appliquons cette seconde étape d'optimisation sur la topologie précédemment optimisée. Parmi les chemins de longueur minimale identifiés, nous ne conservons que ceux qui maximisent le minimum des réserves énergétiques des nœuds rencontrés. Ainsi, le superviseur MAX_EN généré a la forme suivante :

$$MAX\_EN = \{(1, Neighbour(0)), (2, Neighbour(0)), (3, Neighbour(0)), (4, Neighbour(3)), (5, Neighbour(2))\} \tag{4.8}$$

Ainsi, le seul chemin partant du nœud 5 passe par le nœud 2 comme meilleur prochain saut et non le nœud 1 car celui-ci représente un lien faible par rapport au nœud 2 en termes d'énergie.

La table 4.4 illustre les chemins optimaux qui peuvent être obtenus en implémentant le superviseur MAX_EN.

Tableau 4.4 – Chemins optimaux générés en se basant sur le critère de maximisation des minimums des énergies résiduelles rencontrés

nœud	chemin	minimum d'énergie résiduelle rencontrée sur le chemin
1	$(1 \rightarrow 0)$	∞
2	$(2 \rightarrow 0)$	∞
3	$(3 \rightarrow 0)$	∞
4	$(4 \rightarrow 3 \rightarrow 0)$	60
5	$(5 \rightarrow 2 \rightarrow 0)$	40

4.4.5 Distribution du routage

Le superviseur MAX_EN généré précédemment est un ensemble global, contenant la décision de route(s) à suivre pour chaque nœud du réseau. Il est nécessaire, à partir de MAX_EN de produire une règle de routage locale à chaque nœud. A noter que la décision prise localement n'est peut-être pas déterministe, car il peut subsister pour certains noeuds des chemins équivalents selon les deux critères envisagés.

Le principe de cette distribution de MAX_EN consiste à produire la fonction de routage R suivante :

$$R : Q \rightarrow 2^{Bool^{|X|}}$$

telle que :

$$R(q) = \{x \in 2^{Bool^{|X|}} | \forall c \in x : (q, c) \in MAX\_EN\}$$

associant à chaque nœud q la ou les actions de communication c permettant de joindre ses meilleurs successeurs.

4.4.6 Implémentation et outils utilisés

Le réseau étudié a été modélisé en utilisant un langage et un outil aujourd'hui obsolète : le langage Mode Automata et son compilateur Matou [96] ce qui permet de générer un fichier de format z3z représente l'entrée de l'outil symbolique de synthèse de contrôleur Sigali [99]. Ce dernier a été utilisé avec succès pour synthétiser automatiquement le contrôleur optimal centralisé, qui a par la suite été distribuée manuellement en utilisant la méthode présenté ci-dessus. La mise en évidence du résultat attendu a été faite par simulation.

4.4.7 Analyse de performances

Nous présentons dans cette partie des résultats de simulation de l'algorithme de routage proposé ainsi qu'une comparaison à des algorithmes existants, suivant leurs objectifs et leurs caractéristiques, qui permet de valider l'algorithme

proposé.

Comparaison à des algorithmes existants

Le tableau 4.5 présente une comparaison de l'algorithme proposé avec un ensemble de travaux existants. Ces travaux ont été choisis vu qu'ils prennent en compte les mêmes ou un sous ensemble des critères de sélection considérés par notre algorithme.

Les paramètres pris en compte lors de la comparaison de ces travaux sont :

— le(s) critère(s) sur lequel(s) se base la détermination des chemins optimaux ;

— le taux de connaissance de la topologie qu'un nœud doit avoir pour pouvoir déterminer ses chemins optimaux. Rappelons que les algorithmes peuvent être centraux, distribués ou locaux ;

— le coût de l'algorithme du point de vue du nombre de messages de contrôle nécessaires à la récupération de la topologie du réseau ainsi qu'au calcul de chemins optimaux ;

— la convergence qui définit quand les algorithmes doivent être exécutés (algorithme proactif ou réactif).

Les algorithmes évoqués dans le tableau ci-dessus, qu'ils soient distribués ou locaux, supposent que chaque nœud est capable de définir sa fonction de coût en se basant sur celles de ses voisins, mais aucun algorithme distribué, parmi les travaux auxquels l'algorithme proposé a été comparé, ne spécifie comment il récupère une telle information.

L'approche de routage proposée dans ce document s'appuie sur un algorithme de calcul centralisé, requérant donc une fusion d'informations au niveau du nœud puits, permettant de reconstituer la topologie du réseau d'une part, et la distribution des énergies résiduelles d'autre part.

Ainsi, l'avantage de cet algorithme réside dans son exactitude lors de la prise en compte des deux critères (longueur des chemins, et énergie résiduelle le long des chemins). Sa spécificité est portée par la prise en compte de manière exhaustive de la "santé" énergetique de chaque chemin potentiel menant vers le puits. En cherchant systématiquement les chemins comportant le maillon faible

Tableau 4.5 – Comparaison de l'algorithme de routage proposé à des algorithmes existants

Algorithme	Métrique	Type de protocole	Coût de l'algorithme	Convergence
proposé	minimise le nombre de sauts et maximise le minimum des énergies résiduelles	centralisé	modéré	proactif
MMBCR [143]	maximise le minimum des énergies résiduelles	centralisé ou distribué	élevé	réactif
DEBR [91]	balancer les résidus énergétiques de noeuds	Local	élevé	proactif
PSO based routing [10]	établit un compromis entre l'efficacité énergétique et l'équilibrage des énergies résiduelles	centralisé	élevé	proactif
MHRM [36]	minimise le nombre de sauts pour atteindre la station de base	distribué	élevé	proactif

le plus fort, il contribue à un lissage de la consommation et à une augmentation de la durée de vie du réseau, en augmentant en priorité la durée de vie des noeuds les plus affaiblis. Cette approche peut intégrer facilement la notion de panne de nœuds, car elle s'appuie sur la reconstruction régulière de la topologie du réseau.

Il va de soi que la fusion d'informations, nécessaire pour l'application de notre approche, représente un surcoût important en termes de communication. Néanmoins, entre deux phases de recalcul de règles de routage (par synthèse optimale), la politique reste statique au niveau de chacun des nœuds, ce qui a tendance à réduire les échanges des messages de contrôle nécessaires pour récupérer la topologie du réseau d'une part et pour informer les noeuds des chemins optimaux qu'ils peuvent emprunter d'autre part.

La reconfiguration ainsi réalisée est certes moins réactive, vis-à-vis des nécessités locales à chaque nœud, mais aussi moins incontrôlée, car obéissant à une périodicité à définir. Elle offre ainsi davantage de prédictabilité, ce qui constitue un atout important.

L'étude de cette approche de routage reste néanmoins qualitative, à l'échelle du travail mené durant cette thèse. L'impact de la fusion d'informations sur la longevité du réseau n'a pas été chiffré, et reste une perspective de ce travail.

Résultats de simulation de l'algorithme proposé

Nous étudions dans cette partie, l'impact de l'utilisation des chemins optimaux sur la durée de vie des réseaux ainsi que sur l'énergie totale consommée. Pour ces objectifs, la simulation a été réalisée en considérant 300 nœuds déployés aléatoirement dans une zone de 300 x 300 mètres carrés. Initialement, tous les noeuds ont le même niveau d'énergie. Le rayon de communication de chaque nœud est égal à 10m ainsi que son rayon de détection est égal à 5m. Nous avons fait en sorte que chaque nœud soit capable de déterminer son taux d'énergie résiduelle à chaque instant. Pour des raisons de simplification, ce taux est mis à jour uniquement lors de l'utilisation de la radio (réception/transmission de messages). Lorsque ce taux est inférieur à un certain seuil, le nœud est supposé mort et il ne participera plus à aucune activité. La radio est aussi supposée parfaite (tout message émis sur le canal sera reçu correctement par la/les destination(s)).

Tableau 4.6 – Paramètres de simulation

Paramètre	Valeur
nombre des noeuds déployés	300
surface de la zone de déploiement	300 x 300
portée de communication	10m
portée de détection	5m
Énergie initiale d'un nœud	2 J
Puissance de transmission	0,07 mw
Puissance de réception	0,03 mw
Taille de paquet	4000 bits

Les paramètres de simulation pris en compte durant nos simulations sont

résumés dans le tableau 4.6. Ainsi, nous avons choisi d'utiliser le couple OMNeT++/Castalia [157] parmi les autres simulateurs disponibles pour évaluer les performances de notre algorithme. OMNeT++ [151] [150] est un simulateur à événements discrets orientés objet, basé sur C++. Il a été conçu pour la simulation des systèmes de réseaux de communication, des systèmes distribués, etc. Castalia [81] est un framework fonctionnant au-dessus d'OMNeT++ et est spécialement conçu pour les réseaux de capteurs sans fil. Castalia a beaucoup attiré notre attention par le fait que c'est un simulateur de réseaux de capteurs sans fil générique, c'est-à-dire que l'implémentation d'un nœud n'est pas spécifique à un modèle de capteur. Outre cela, Castalia utilise aussi des modèles réalistes, en particulier au niveau de l'utilisation de la radio.

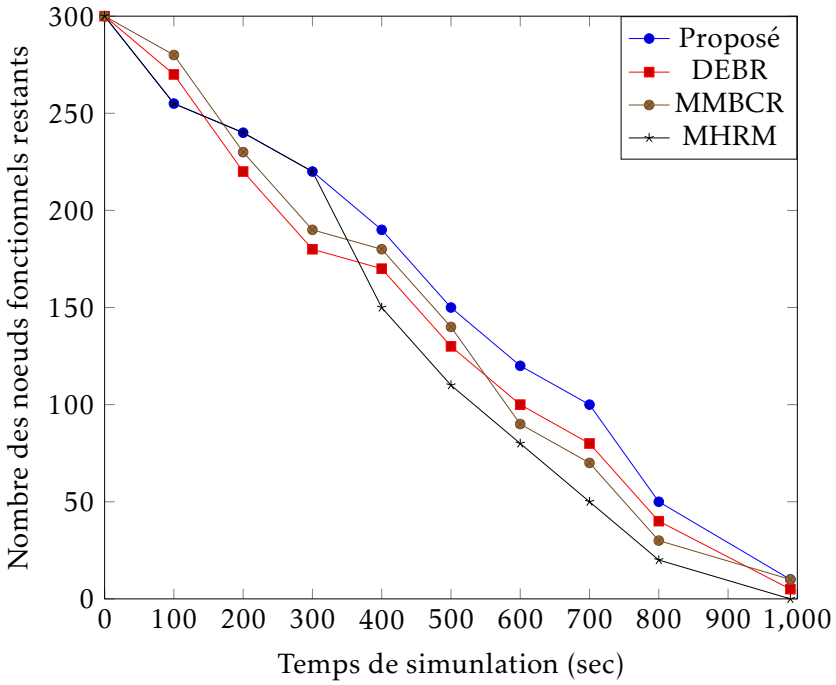

FIGURE 4.4 – Comparaison de l'algorithme proposé aux algorithmes existants en terme des durées de vie des réseaux

Tout d'abord, nous comparons l'algorithme proposé aux algorithmes similaires existants suivants, MHRM [36], MMBCR [143] et DEBR [91], en termes de durée de vie des réseaux par rapport au critère de nombre de noeuds fonctionnels restants. Les résultats ont été représentés sur la figure 4.4.

On constate à travers la figure 4.4 que l'algorithme proposé a une meilleure durée de vie du réseau par rapport aux MHRM, MMBCR et DEBR. Les nœuds dans MHRM sélectionnent le nœud de plus longue distance vers la station de base afin de minimiser le nombre de sauts. En conséquence, les nœuds meurent rapidement suite à la grande quantité d'énergie consommée pour maintenir une communication à long terme. L'algorithme proposé s'est comporté de la même manière aux premières périodes puisque le choix va se faire sur le nombre de sauts. Un tel résultat est attendu, vu qu'au départ tous les noeuds ont le même taux d'énergie. Après utilisation des chemins et surtout épuisement des réserves des noeuds les plus sollicités, le choix se basera plus sur ces réserves d'énergie qui ne seront plus équitables entre les noeuds. Ainsi, les noeuds épuisés peuvent être contournés ce qui explique l'augmentation de la durée de vie du réseau par l'algorithme proposé par rapport à MHRM. D'autre part, pour garantir une consommation énergétique équitable, certains noeuds dans DEBR ou MMBCR transmettent les messages de données à la station de base par d'autres noeuds qui peuvent être dans la direction opposée de la station de base, ce qui conduit à une consommation massive d'énergie. Cependant, l'algorithme proposé n'augmente pas inutilement les coûts de transmission à chaque fois qu'il y a équivalence par rapport à l'énergie résiduelle des noeuds. Il sélectionne ainsi le chemin le plus court tout en évitant des parcours plus longs ce qui n'est pas le cas des algorithmes optimisant seulement le maximum des énergies résiduelles des noeuds.

Nous comparons également les performances de cet algorithme en termes de la quantité énergétique totale consommée du réseau. La figure 4.5 montre que notre algorithme est plus performant par rapport aux MMBCR, DEBR et MHRM. On peut observer à travers la figure 4.5 que l'algorithme proposé consomme moins d'énergie par période par rapport aux autres algorithmes existants. Ceci s'explique par la meilleure utilisation des ressources limitées des nœuds, en terme de distance et énergie, en effectuant tous les calculs nécessaires à la construction des tables de routage, au niveau de la station de base puis transmettre à chaque nœud du réseau un message contenant seulement son meilleur voisin.

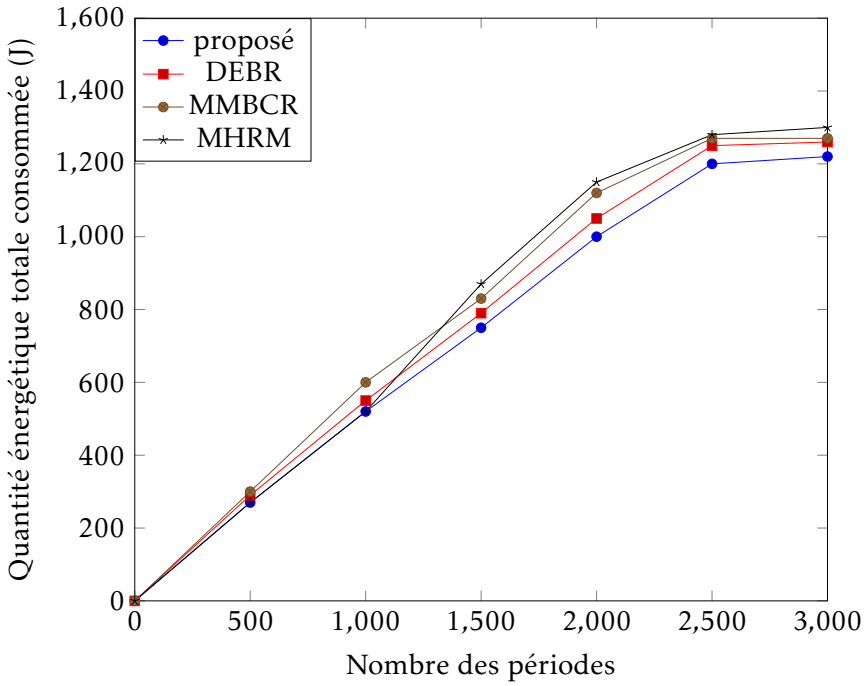

FIGURE 4.5 – Comparaison des algorithmes en termes de la consommation énergétique totale des réseaux

4.5 Conclusion

Dans ce chapitre, nous avons défini une méthode correcte-par-construction qui permet de générer automatiquement une fonctionnalité de routage optimale. Cette approche repose sur la technique DCS (Discrete Controller Synthesis) optimale, telle que mise en œuvre par [97] mais sans prendre en compte l'incontrôlabilité tels que l'échec d'une liaison de communication ou la panne d'un nœud. En effet, ceci est laissé comme une future extension de recherche pour ce travail.

Vu que l'énergie est l'un des critères clés des protocoles dédiés aux réseaux de capteurs, elle a été considérée, dans le développement de cet algorithme, sur deux formes différentes : énergie résiduelle au niveau des chemins et nombre de sauts (coût énergétique de transmission de message). La minimisation de la consommation d'énergie en termes de nombre de sauts représente l'un des plus importants critères considérés par tous les protocoles dédiés aux réseaux de capteurs. Pour cette raison, cet objectif a été pris en compte comme premier critère pour la recherche des chemins. Des travaux existants [143] [91] ont montré que l'équilibrage de charge entre les noeuds permet aussi d'augmenter la durée de vie du réseau. Pour réaliser cet objectif, le taux d'énergie disponible au niveau de chaque nœud a été considéré. Plus particulièrement les chemins maximisant ces réserves énergétiques ont été pris en compte. La considération simultanée de deux critères permet alors de contourner les noeuds critiques sans pour autant augmenter les coûts de transmission.

Les résultats de simulations nous ont réconfortés par rapport aux objectifs. Il a été démontré que l'algorithme proposé surclasse les algorithmes existants, à savoir DEBR, MMBCR et MHRM en termes de diverses mesures de performance telles que la durée de vie du réseau et la consommation énergétique totale. L'impact de l'algorithme proposé sur la durée de vie est assez encourageant et représente l'un de ses avantages. Il faut aussi noter que l'algorithme n'est pas encombrant de point de vue espace mémoire nécessaire pour le stockage de tous les chemins optimaux au niveau de chaque nœud. Les capteurs ne sont pas contraints de mémoriser des tables de routage complexe, mais de simples informations sur les destinations optimales à un seul saut. Vu les différentes

contraintes des ressources des capteurs, cet aspect représente un nouvel avantage de l'algorithme proposé.

Conclusion générale

Les dernières avancées technologiques des systèmes embarqués en terme de miniaturisation et de supports de communication sans fil ont favorisé l'émergence et l'exploitation des réseaux de capteurs dans de nombreux domaines. Cependant, un frein au déploiement de ces réseaux est qu'ils doivent faire face à des contraintes liées aux ressources limitées des leurs noeuds capteurs. Les travaux présentés dans ce manuscrit s'inscrivent dans ce cadre là. Dans ce qui suit, nous faisons un résumé de nos deux contributions, et nous donnons les perspectives pour les travaux futurs.

Résumé des contributions

Nous nous sommes intéressé, dans la première contribution de cette thèse, au problème d'ordonnancement de l'activité des noeuds en vue de réduire la consommation d'énergie. La réduction de la consommation a pour but de maximiser la durée de vie de l'ensemble du réseau. En se basant sur la technique de la SCD, Nous avons proposé une méthode pour modéliser, concevoir et générer automatiquement un ordonnanceur local correct par construction pour chaque nœud de réseau. Le déroulement de la méthode a été comme suit : En s'inspirant de l'algorithme GAF [158], toute la zone à surveiller a été divisée en petites zones virtuelles. Chaque petite zone appelée "cluster" contient un ensemble redondant de nœuds identiques. Tous les noeuds appartenant à deux petites zones adjacentes doivent pouvoir communiquer afin de garantir la connectivité de réseau. Ainsi, par application de la SCD, et pour profiter de la redondance des noeuds déployés dans chaque cluster, un ordonnanceur centralisé a été

généré automatiquement respectant la spécification de contrôle imposée : un seul capteur actif au sein de chaque cluster. Cependant, l'ordonnanceur obtenu, apparu sous la forme d'une contrainte logique exprimée sur l'état global de réseau, n'est pas inhérent à l'architecture distribuée de réseaux de capteurs. Afin de rendre réalisable cette architecture de contrôle, le superviseur est d'abord transformé de manière systématique depuis sa représentation équationnelle en un vecteur de fonctions, puis par l'ajout d'un mécanisme de communication explicite garantissant la synchronisation entre les noeuds communicants. Ces opérations qui visent a raffiner le comportement de l'ordonnanceur, modifient donc le code de celui-ci, alors qu'il avait initialement été généré automatiquement. Ceci engendre un besoin de simulation et validation du modèle raffiné obtenu, par rapport à l'ordonnanceur abstrait généré initialement par synthèse.

Comme la vérification formelle, la SCD est confrontée aussi aux mêmes complexité théorique exponentielle. En effet, les résultats présentés ne s'appliquent pas au réseau entier, mais aux unités redondantes constituées de sous-ensembles de nœuds considérés comme équivalents, appelées Clusters. Néanmoins, il a été possible d'identifier une démarche méthodologique étayée par un algorithme, qui permet d'exploiter la symétrie au sein d'un réseau de capteurs et déduire des règles inductives permettant d'étendre la formule d'un contrôleur distribué à un nombre de nœuds arbitrairement grand.

Une génération automatique hors ligne d'une fonctionnalité de routage optimale est présentée dans la deuxième partie de ces travaux. Spécifiquement, vu que l'aspect énergie représente l'un des critères majeurs des réseaux de capteurs, un chemin optimal devrait avoir à la fois une longueur minimale et traverser des nœuds ayant une énergie résiduelle maximale. La génération du routage optimal repose sur la technique de la SCD optimale, telle que mise en œuvre par [97]. Selon cette approche, un contrôleur centralisé est automatiquement calculé puis et il est ensuite distribué de manière à fournir à chaque nœud de réseau des éléments de choix locales pour sélectionner son prochain saut lors de l'acheminement de données à la station de base. Cela permet à la fois la génération automatique et la distribution du code correct par construction. Ainsi, le chemin choisi devrait être le plus court et le plus efficace sur le plan énergétique. En effet, pour prolonger autant que possible la durée de vie du

réseau, les noeuds ayant des faibles réserves d'énergie ont été évité en tant que voisins de destination, s'il existe d'autres alternatives énergétiques résiduelles plus élevées. Cette proposition nous a permis à la fois de préserver les réserves énergétiques minimales ainsi que d'avoir une réserve d'énergie équitable pou tous les nœuds de réseau. Les résultats de simulations nous ont réconfortés par rapport aux objectifs. Il a été démontré que l'algorithme proposé surclasse les algorithmes existants, à savoir DEBR, MMBCR et MHRM en termes de diverses mesures de performance telles que la durée de vie du réseau et la consommation énergétique totale.

Perspectives

Les perspectives ouvertes par ces travaux sont nombreuses et variées. Dans la continuité de nos travaux, nous envisageons tout d'abord d'étendre l'approche d'ordonnancement automatique d'activité des noeuds afin d'intégrer une notion de panne de capteur, avec l'objectif de produire un ordonnancement réactif, permettant de reconfigurer le réseau suite à d'éventuelles pannes de capteurs.

En ce qui concerne la problématique du routage, nous avons proposé une méthode correcte-par-construction qui permet de générer automatiquement une fonctionnalité de routage optimale. Cette approche repose sur la technique de la SCD optimale afin de fournir une table de routage locale spécifique à chaque nœud lui permettant d'atteindre la station de base en suivant le plus court chemin tout en évitant les noeuds ayant des énergies assez épuisées. Cependant, Nous n'avons pas pris en compte l'incontrôlabilité lors de calcul de contrôleur tels que l'échec d'une liaison de communication ou la panne d'un nœud. Comme un réseau de capteurs dans son ensemble doit répondre à des exigences critiques de tolérance aux fautes et de disponibilité, nous envisageons ainsi d'enrichir l'algorithme de routage proposé, en intégrant une notion de panne locale d'un capteur, et étudier l'impact d'une telle panne sur le fonctionnement global du réseau.

Dans ce travail de thèse, les objectifs de contrôle sont relativement simples. Nous envisageons d'étudier des scénarios de contrôle plus élaborés, par exemple, un contrôle sur des séquences d'événements et/ou d'états.

Nous souhaitons aussi tester notre approche dans le contexte industriel par un outil logiciel permettant de produire la description d'un réseau ordonnancé prêt à être implémenté sous forme matérielle (langage synchrone, VHDL, C, nesC).

Liste des publications

Conférences internationales avec comité de lecture

— Anis Mezni, Emil Dumitrescu, Eric Niel, Samir Ben Ahmed, Multi Criteria Automatic Generation of an Optimal Routing Schemes for WSN In International Wireless Communications and Mobile Computing Conference (IWCMC 2018)

— Anis Mezni, Emil Dumitrescu, Eric Niel, Samir Ben Ahmed, Automatic Generation of Wireless Sensor Networks Scheduling In the 2017 International Conference on High Performance Computing and Simulation (HPCS 2017)

— Anis Mezni, Emil Dumitrescu, Eric Niel, Samir Ben Ahmed, A design method for WSN's automatic scheduling generation In 14th ACM International Symposium on Mobility Management and Wireless Access, (MOBIWAC 2016)

Séminaires et workshop nationaux

— Anis Mezni, Emil Dumitrescu, Eric Niel, Samir Ben Ahmed, Automatic Generation of an Optimal Routing Schemes Based on Hierarchical Criteria for WSN In the First Computer Science University of Tunis El Manar PhD Symposium (CUPS'17)

Bibliographie

[1] Organization Accellera. *Property Specification Language Reference Manual*. Rapp. tech. Version 1.1. Grenoble, France, 2004.

[2] I. F. Akyildiz et al. « Wireless Sensor Networks : A Survey ». In : *Comput. Netw.* 38.4 (mar. 2002), p. 393–422. issn : 1389-1286. doi : 10.1016/S1389-1286(01)00302-4. url : http://dx.doi.org/10.1016/S1389-1286(01)00302-4.

[3] Cecilia Albert et Lisa Brownsword. « Meeting the Challenges of Commercial-Off-The-Shelf COTS Products : The Information Technology Solutions Evolution Process ITSEP ». In : *Proceedings of the first international conference on COTS-based software systems*. ICCBSS '02. London, UK, UK : Springer-Verlag, 2002, p. 10–20. isbn : 3-540-43100-4.

[4] Karine Altisen et al. « Using controller-synthesis techniques to build property-enforcing layers ». In : *European Symposium on Programming*. Springer. 2003, p. 174–188.

[5] Rajeev Alur et David Dill. « Automata for modeling real-time systems ». In : *International Colloquium on Automata, Languages, and Programming*. Springer. 1990, p. 322–335.

[6] Carina Alves et Anthony Finkelsteiin. « Challenges in COTS Decision-making : A Goal-driven Requirements Engineering Perspective ». In : *Proceedings of the 14th international conference on software engineering and knowledge engineering*. SEKE '02. Ischia, Italy : ACM, 2002, p. 789–794. isbn : 1-58113-556-4.

[7] Xin An. « High Level Design and Control of Adaptive Multiprocessor Systems-on-Chip ». Thèse de doct. Université de Grenoble, 2013.

[8] Thanasis Antoniou et al. « A new energy efficient and fault-tolerant protocol for data propagation in smart dust networks using varying transmission range ». In : *Proceedings of the 37th annual symposium on Simulation*. IEEE Computer Society. 2004, p. 43.

[9] Eugene Asarin, Oded Maler et Amir Pnueli. « Symbolic Controller Synthesis for Discrete and Timed Systems ». In : *Hybrid Systems II, LNCS 999*. Springer, 1995, p. 1–20.

[10] Md Azharuddin et Prasanta K Jana. « PSO-based approach for energy-efficient and energy-balanced routing and clustering in wireless sensor networks ». In : *Soft Computing* 21.22 (2017), p. 6825–6839.

[11] Heribert Baldus, Karin Klabunde et Guido Muesch. « Reliable set-up of medical body-sensor networks ». In : *European Workshop on Wireless Sensor Networks*. Springer. 2004, p. 353–363.

[12] Silvano Balemi et al. « Supervisory control of a rapid thermal multiprocessor ». In : *IEEE Transactions on Automatic Control* 38.7 (1993), p. 1040–1059.

[13] R Bellman. *Dynamic programming/by Richard Bellman ; with a new introduction by Stuart Dreyfus*.

[14] Richard Bellman. « On a routing problem ». In : *Quarterly of applied mathematics* 16.1 (1958), p. 87–90.

[15] Yann Ben Maissa et al. « Transactions on Petri Nets and Other Models of Concurrency VIII ». In : sous la dir. de Maciej Koutny, Wil M. P. van der Aalst et Alex Yakovlev. Berlin, Heidelberg : Springer Berlin Heidelberg, 2013. Chap. Modeling and Analyzing Wireless Sensor Networks with VeriSensor : An Integrated Workflow, p. 24–47. isbn : 978-3-642-40465-8. doi : 10.1007/978-3-642-40465-8_2. url : http://dx.doi.org/10.1007/978-3-642-40465-8_2.

[16] Johan Bengtsson et al. « UPPAAL ?a tool suite for automatic verification of real-time systems ». In : *International Hybrid Systems Workshop*. Springer. 1995, p. 232–243.

[17] Albert Benveniste et al. « The Synchronous Languages 12 Years Later ». In : *PROCEEDINGS OF THE IEEE* 91.1 (2003).

[18] Gérard Berry et Georges Gonthier. « The Esterel synchronous programming language : Design, semantics, implementation ». In : *Science of computer programming* 19.2 (1992), p. 87–152.

[19] Nicolas Berthier. « Programmation synchrone de pilotes de périphériques pour un contrôle global de ressources dans les systèmes embarqués ». Thèse de doct. Université de Grenoble, 2012.

[20] Nicolas Berthier et Hervé Marchand. « Discrete controller synthesis for infinite state systems with reax ». In : *IFAC Proceedings Volumes* 47.2 (2014), p. 46–53.

[21] Yves Bertot et Pierre Castéran. *Interactive theorem proving and program development : Coq ?Art : the calculus of inductive constructions*. Springer Science & Business Media, 2013.

[22] Roderick Bloem et al. « Specify, Compile, Run : Hardware from PSL ». In : *Electron. Notes Theor. Comput. Sci.* 190.4 (nov. 2007), p. 3–16. issn : 1571-0661.

[23] Richard Bonichon, David Delahaye et Damien Doligez. « Zenon : An Extensible Automated Theorem Prover Producing Checkable Proofs ». In : *LPAR*. 2007, p. 151–165.

[24] Azzedine Boukerche, Richard Werner Nelem Pazzi et Regina Borges Araujo. « A fast and reliable protocol for wireless sensor networks in critical conditions monitoring applications ». In : *Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*. ACM. 2004, p. 157–164.

[25] Azzedine Boukerche, Richard Werner Nelem Pazzi et Regina Borges Araujo. « Fault-tolerant wireless sensor network routing protocols for the supervision of context-aware physical environments ». In : *Journal of Parallel and Distributed Computing* 66.4 (2006), p. 586–599.

[26] Jean-Louis Boulanger. *Formal Method : Industrial Used from Model to the Code*. Wiley, mai 2012. isbn : 9781848213623.

[27] Marius Bozga et al. « Kronos : A model-checking tool for real-time systems ». In : *International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems*. Springer. 1998, p. 298–302.

[28] Marius Bozga et al. « The IF toolset ». In : *Formal Methods for the Design of Real-Time Systems*. Springer, 2004, p. 237–267.

[29] David Braginsky et Deborah Estrin. « Rumor routing algorthim for sensor networks ». In : *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. ACM. 2002, p. 22–31.

[30] Christian Brunette et al. « A metamodel for the design of polychronous systems ». In : *The Journal of Logic and Algebraic Programming* 78.4 (2009), p. 233–259.

[31] Bryant et Randal E. « Graph-Based Algorithms for Boolean Function Manipulation ». In : *IEEE Trans. Comput.* 35.8 (août 1986), p. 677–691. issn : 0018-9340. doi : 10.1109/TC.1986.1676819. url : http://dx.doi.org/10.1109/TC.1986.1676819.

[32] Kai Cai et W Murray Wonham. « Supervisor localization : a top-down approach to distributed control of discrete-event systems ». In : *IEEE Transactions on Automatic Control* 55.3 (2010), p. 605–618.

[33] Mihaela Cardei et Jie Wu. « Energy-efficient coverage problems in wireless ad-hoc sensor networks ». In : *Computer communications* 29.4 (2006), p. 413–420.

[34] ANDR Charles. « Representation and analysis of reactive behaviors : A synchronous approach ». In : *Computational Engineering in Systems Applications, CESA*. T. 96. 1996, p. 19–29.

[35] Hongju Cheng et al. « Energy-efficient node selection algorithms with correlation optimization in wireless sensor networks ». In : *International Journal of Distributed Sensor Networks* 10.3 (2014), p. 576573.

[36] Shao-Shan Chiang, Chih-Hung Huang et Kuang-Chiung Chang. « A minimum hop routing protocol for home security systems using wireless sensor networks ». In : *IEEE Transactions on Consumer Electronics* 53.4 (2007).

[37] Chee-Yee Chong et Srikanta P Kumar. « Sensor networks : evolution, opportunities, and challenges ». In : *Proceedings of the IEEE* 91.8 (2003), p. 1247–1256.

[38] Edmund M Clarke. « The birth of model checking ». In : *25 Years of Model Checking*. Springer, 2008, p. 1–26.

[39] Edmund M Clarke, E Allen Emerson et A Prasad Sistla. « Automatic verification of finite state concurrent system using temporal logic specifications : a practical approach ». In : *Proceedings of the 10th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*. ACM. 1983, p. 117–126.

[40] Edmund M. Clarke, E Allen Emerson et A Prasad Sistla. « Automatic verification of finite-state concurrent systems using temporal logic specifications ». In : *ACM Transactions on Programming Languages and Systems (TOPLAS)* 8.2 (1986), p. 244–263.

[41] Edmund M Clarke et Orna Grumberg. « Avoiding the state explosion problem in temporal logic model checking ». In : *Proceedings of the sixth annual ACM Symposium on Principles of distributed computing*. ACM. 1987, p. 294–303.

[42] Edmund M Clarke et al. « Model checking and the state explosion problem ». In : *Tools for Practical Software Verification*. Springer, 2012, p. 1–30.

[43] Thomas Clausen et Philippe Jacquet. *Optimized link state routing protocol (OLSR)*. Rapp. tech. 2003.

[44] Jean-Louis Colaço, Bruno Pagano et Marc Pouzet. « A conservative extension of synchronous data-flow with state machines ». In : *Proceedings of the 5th ACM international conference on Embedded software*. ACM. 2005, p. 173–182.

[45] Philippe Darondeau et Laurie Ricker. « Distributed control of discrete-event systems : A first step ». In : *Transactions on Petri Nets and Other Models of Concurrency VI*. Springer, 2012, p. 24–45.

[46] Aditya N Das et al. « Data-logging and supervisory control in wireless sensor networks ». In : *International Journal of Sensor Networks* 6.1 (2009), p. 13–27.

[47] Gwenaël Delaval, Hervé Marchand et Eric Rutten. « Contracts for modular discrete controller synthesis ». In : *ACM Sigplan Notices*. T. 45. 4. ACM. 2010, p. 57–66.

[48] Gwenaël Delaval, Eric Rutten et Hervé Marchand. « Integrating discrete controller synthesis into a reactive programming language compiler ». In : *Discrete Event Dynamic Systems* 23.4 (2013), p. 385–418.

[49] Jing Deng, Richard Han et Shivakant Mishra. « INSENS : Intrusion-tolerant routing for wireless sensor networks ». In : *Computer Communications* 29.2 (2006), p. 216–230.

[50] Doron Drusinsky et David Harel. « Using statecharts for hardware description and synthesis ». In : *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 8.7 (1989), p. 798–807.

[51] Emil Dumitrescu et al. « A supervisor implementation approach in Discrete Controller Synthesis ». In : *Emerging technologies and factory automation*. 2008, p. 1433–1440.

[52] Christian C Enz et al. « WiseNET : an ultralow-power wireless sensor network solution ». In : *Computer* 37.8 (2004), p. 62–70.

[53] D. Estrin et al. « Connecting the physical world with pervasive networks ». In : *IEEE Pervasive Computing* 1.1 (jan. 2002), p. 59–69. issn : 1536-1268. doi : 10.1109/MPRV.2002.993145.

[54] H Foster, H Marschner et Y Wolfsthal. « IEEE 1850 PSL : The Next Generation. 2005. » In : *IEEE Std 1850-2005* (2005). url : #http://www.pslsugar.org/papers/ieee1850psl-the_next_generation.pdf#.

[55] Antoine Gallais. « Ordonnancement d'activité dans les réseaux de capteurs : l'exemple de la couverture de surface ». Thèse de doct. Université des Sciences et Technologie de Lille-Lille I, 2007.

[56] Antoine Gallais et al. « Localized sensor area coverage with low communication overhead ». In : *IEEE Transactions on Mobile Computing* 7.5 (2008).

[57] Vincenzo Giordano et al. « Supervisory control of mobile sensor networks : math formulation, simulation, and implementation ». In : *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 36.4 (2006), p. 806–819.

[58] Alain Girault et Clément Ménier. « Automatic production of globally asynchronous locally synchronous systems ». In : *International Workshop on Embedded Software*. Springer. 2002, p. 266–281.

[59] Alain Girault et Eric Rutten. « Modeling Fault-tolerant Distributed Systems for Discrete Controller Synthesis ». In : *Electron. Notes Theor. Comput. Sci.* 133 (mai 2005). ACM ID : 1706220, p. 81–100. issn : 1571-0661.

[60] B. Gong et al. « Multihop Routing Protocol with Unequal Clustering for Wireless Sensor Networks ». In : *2008 ISECS International Colloquium on Computing, Communication, Control, and Management*. T. 2. Août 2008, p. 552–556. doi : 10.1109/CCCM.2008.99.

[61] Gregor Gössler et Joseph Sifakis. « Composition for Component-based Modeling ». In : *Sci. Comput. Program.* 55.1-3 (mar. 2005), p. 161–183. issn : 0167-6423.

[62] Thilo Hafer et Wolfgang Thomas. « Computation Tree Logic CTL* and Path Quantifiers in the Monadic Theory of the Binary Tree ». In : *ICALP*. 1987, p. 269–279.

[63] Nicholas Halbwachs et al. « The synchronous data flow programming language LUSTRE ». In : *Proceedings of the IEEE* 79.9 (1991), p. 1305–1320.

[64] Nicolas Halbwachs. *Synchronous programming of reactive systems*. T. 215. Springer Science & Business Media, 2013.

[65] M. Hammad et J. Cook. « Compositional Verification of Sensor Software Using Uppall ». In : *2012 IEEE 23rd International Symposium on Software Reliability Engineering*. Nov. 2012, p. 351–360. doi : 10.1109/ISSRE.2012.5.

[66] David Harel. « Statecharts : A visual formalism for complex systems ». In : *Science of computer programming* 8.3 (1987), p. 231–274.

[67] Guoyou He. « Destination-sequenced distance vector (DSDV) protocol ». In : *Networking Laboratory, Helsinki University of Technology* (2002), p. 1–9.

[68] Wendi Rabiner Heinzelman, Anantha Chandrakasan et Hari Balakrishnan. « Energy-efficient communication protocol for wireless microsensor networks ». In : *System sciences, 2000. Proceedings of the 33rd annual Hawaii international conference on*. IEEE. 2000, 10–pp.

[69] Wendi Rabiner Heinzelman, Joanna Kulik et Hari Balakrishnan. « Adaptive protocols for information dissemination in wireless sensor networks ». In : *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*. ACM. 1999, p. 174–185.

[70] Jason Hill et al. « System architecture directions for networked sensors ». In : *ACM SIGOPS operating systems review* 34.5 (2000), p. 93–104.

[71] Charles Antony Richard Hoare. « Communicating sequential processes ». In : *Communications of the ACM* 21.8 (1978), p. 666–677.

[72] Bernard Houssais. « The synchronous programming language signal : A tutorial ». In : (2002).

[73] Danny Hughes et al. « An intelligent and adaptable flood monitoring and warning system ». In : *UK E-Science all hands meeting*. NeSC. 2006.

[74] Jérôme Hugues et al. « On the formal verification of middleware behavioral properties ». In : *Electronic Notes in Theoretical Computer Science* 133 (2005), p. 139–157.

[75] Dieter Hutter et al. *Applied Formal Methods-FM-Trends 98 : International Workshop on Current Trends in Applied Formal Methods, Boppard, Germany, October 7-9, 1998, Proceedings*. Springer, 2007.

[76] Ali Kadhum Idrees et al. « Distributed lifetime coverage optimization protocol in wireless sensor networks ». In : *The Journal of Supercomputing* 71.12 (2015), p. 4578–4593.

[77] IEEE. « IEEE Standard for Property Specification Language (PSL) ». In : *IEEE Std 1850-2005* (2005), p. 1–143.

[78] Chalermek Intanagonwiwat, Ramesh Govindan et Deborah Estrin. « Directed diffusion : A scalable and robust communication paradigm for sensor networks ». In : *Proceedings of the 6th annual international conference on Mobile computing and networking*. ACM. 2000, p. 56–67.

[79] P. Jacquet et al. « Optimized link state routing protocol for ad hoc networks ». In : *Proceedings. IEEE International Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century.* 2001, p. 62–68. doi : 10.1109/INMIC.2001.995315.

[80] Nadeem Javaid et al. « An energy-efficient distributed clustering algorithm for heterogeneous WSNs ». In : *EURASIP Journal on Wireless communications and Networking* 2015.1 (2015), p. 151.

[81] Miloš Jevtić, Nikola Zogović et Goran Dimić. « Evaluation of wireless sensor network simulators ». In : *Proceedings of the 17th telecommunications forum (TELFOR 2009), Belgrade, Serbia*. Citeseer. 2009, p. 1303–1306.

[82] David B Johnson et David A Maltz. « Dynamic source routing in ad hoc wireless networks ». In : *Mobile computing* (1996), p. 153–181.

[83] David B Johnson, David A Maltz, Josh Broch et al. « DSR : The dynamic source routing protocol for multi-hop wireless ad hoc networks ». In : *Ad hoc networking* 5 (2001), p. 139–172.

[84] Gabriel Kalyon et al. « Synthesis of communicating controllers for distributed systems ». In : *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*. IEEE. 2011, p. 1803–1810.

[85] Taeho Kim, David Stringer-Calvert et Sungdeok Cha. « Formal verification of functional properties of a SCR-style software requirements specification using PVS ». In : *Reliability Engineering & System Safety* 87.3 (2005), p. 351–363.

[86] Joanna Kulik, Wendi Heinzelman et Hari Balakrishnan. « Negotiation-based protocols for disseminating information in wireless sensor networks ». In : *Wireless networks* 8.2/3 (2002), p. 169–185.

[87] Fabienne Lagnier, Pascal Raymond et Christian Dubois. « Formal verification of a critical system written in Saga/Lustre ». In : *Workshop on Formal Methods, Modelling and Simulation for System Engineering*. St Quentin en Yvelines (France), fév. 1995.

[88] Peng Li et John Regehr. « T-check : Bug Finding for Sensor Networks ». In : *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*. IPSN '10. Stockholm, Sweden : ACM, 2010, p. 174–185. isbn : 978-1-60558-988-6. doi : 10.1145/1791212.1791234. url : http://doi.acm.org/10.1145/1791212.1791234.

[89] Feng Lin. « Control of networked discrete event systems : dealing with communication delays and losses ». In : *SIAM Journal on Control and Optimization* 52.2 (2014), p. 1276–1298.

[90] Stephanie Lindsey et Cauligi S Raghavendra. « PEGASIS : Power-efficient gathering in sensor information systems ». In : *Aerospace conference proceedings, 2002. IEEE*. T. 3. IEEE. 2002, p. 3–3.

[91] Wen Lu, Hu Zhao et Haixing Zhao. « Distributed energy balancing routing algorithm in wireless sensor networks ». In : *Recent Advances in Computer Science and Information Engineering* (2012), p. 227–232.

[92] Oded Maler. « On optimal and sub-optimal control in the presence of adversaries ». In : *IFAC Proceedings Volumes* 37.18 (2004), p. 1–12.

[93] Arati Manjeshwar et Dharma P Agrawal. « TEEN : a routing protocol for enhanced efficiency in wireless sensor networks ». In : *null*. IEEE. 2001, 30189a.

[94] Reference Manual. « The Accellera PSL Language Reference Manual ». In : (2004). url : www.eda.org/vfv/docs/PSL-v1.1.pdf.

[95] Florence Maraninchi et Yann Rémond. « Argos : an automaton-based synchronous language ». In : *Computer languages* 27.1 (2001), p. 61–92.

[96] Florence Maraninchi et Yann Rémond. « Mode-automata : a new domain-specific construct for the development of safe critical systems ». In : *Science of computer programming* 46.3 (2003), p. 219–254.

[97] Hervé Marchand, Olivier Boivineau et Stéphane Lafortune. « Optimal control of discrete event systems under partial observation ». In : *Decision and Control, 2001. Proceedings of the 40th IEEE Conference on*. T. 3. IEEE. 2001, p. 2335–2340.

[98] Hervé Marchand et Éric Rutten. « Managing multi-mode tasks with time cost and quality levels using optimal discrete control synthesis ». In : *Real-Time Systems, 2002. Proceedings. 14th Euromicro Conference on*. IEEE. 2002, p. 241–248.

[99] Hervé Marchand et al. « Synthesis of discrete-event controllers based on the signal environment ». In : *Discrete Event Dynamic Systems* 10.4 (2000), p. 325–346.

[100] Kenneth L McMillan. *Symbolic Model Checking : An Approach to the State Explosion Problem*. Rapp. tech. CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER SCIENCE, 1992.

[101] Mentor-Graphics. *Advanced FPGA synthesis*. 2010. url : http://www.mentor.com/products/fpga/news/fpga%20-precision-synthesis-2010a.

[102] Anis Mezni et al. « A Design Method for WSN's Automatic Scheduling Generation ». In : *Proceedings of the 14th ACM International Symposium on Mobility Management and Wireless Access*. ACM. 2016, p. 19–26.

[103] Robin Milner. « Calculi for Synchrony and Asynchrony ». In : *Theor. Comput. Sci.* 25 (1983), p. 267–310.

[104] R. Mingming et E. Dumitrescu. *Automatic error correction based on discrete controller synthesis*. Sept. 2007.

[105] M. Morisio et al. « COTS-based Software Development : Processes and Open Issues ». In : *J. Syst. Softw.* 61.3 (avr. 2002), p. 189–189. issn : 0164-1212.

[106] Habib Mostafaei et al. « A sleep scheduling approach based on learning automata for WSN partialcoverage ». In : *Journal of Network and Computer Applications* 80 (2017), p. 67–78.

[107] Laurent Mounier, Ludovic Samper et Wassim Znaidi. « Worst-case lifetime computation of a wireless sensor network by model-checking ». In : *Proceedings of the 4th ACM workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*. ACM. 2007, p. 1–8.

[108] Suman Nath et Phillip B Gibbons. « Communicating via fireflies : geographic routing on duty-cycled sensors ». In : *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*. IEEE. 2007, p. 440–449.

[109] Y. Oddos, K. Morin-Allory et D. Borrione. « SyntHorus : highly efficient automatic synthesis from PSL to HDL ». In : *Very large scale integration (VLSI-SoC), 2009 17th IFIP*. Oct. 2009, p. 83–88.

[110] Yann Oddos. « Vérification semi-formelle et synthèse automatique de circuits à partir de spécifications temporelles écrite en PSL ». Thèse de doct. Grenoble, France, 2009.

[111] Peter Csaba Ölveczky et José Meseguer. « Semantics and pragmatics of real-time maude ». In : *Higher-order and symbolic computation* 20.1-2 (2007), p. 161–196.

[112] Peter Csaba Ölveczky et Stian Thorvaldsen. « Formal modeling and analysis of the OGDC wireless sensor network algorithm in Real-Time Maude ». In : *International conference on Formal methods for open object-based distributed systems*. Springer. 2007, p. 122–140.

[113] Susan Owicki et David Gries. « Verifying properties of parallel programs : An axiomatic approach ». In : *Communications of the ACM* 19.5 (1976), p. 279–285.

[114] Susan Owicki et Leslie Lamport. « Proving liveness properties of concurrent programs ». In : *ACM Transactions on Programming Languages and Systems (TOPLAS)* 4.3 (1982), p. 455–495.

[115] Sam Owre et al. « PVS : Combining specification, proof checking, and model checking ». In : *International Conference on Computer Aided Verification*. Springer. 1996, p. 411–414.

[116] KM Passino et PJ Antsaklis. « On the optimal control of discrete event systems ». In : *Decision and Control, 1989., Proceedings of the 28th IEEE Conference on*. IEEE. 1989, p. 2713–2718.

[117] Rodolfo Pellizzoni et al. « A predictable execution model for cots-based embedded systems ». In : *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2011 17th IEEE*. IEEE. 2011, p. 269–279.

[118] Charles Perkins, Elizabeth Belding-Royer et Samir Das. *Ad hoc on-demand distance vector (AODV) routing*. Rapp. tech. 2003.

[119] Carl Adam Petri. « Introduction to general net theory ». In : *Net theory and applications*. Springer, 1980, p. 1–19.

[120] Amir Pnueli. « The Temporal Semantics of Concurrent Programs ». In : *Proceedings of the international symposium on semantics of concurrent computation*. London, UK : Springer-Verlag, 1979, p. 1–20. isbn : 3-540-09511-X.

[121] Amir Pnueli. « The temporal semantics of concurrent programs ». In : *Theoretical computer science* 13.1 (1981), p. 45–60.

[122] Joseph Polastre, Jason Hill et David Culler. « Versatile Low Power Media Access for Wireless Sensor Networks ». In : *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*. SenSys '04. Baltimore, MD, USA : ACM, 2004, p. 95–107. isbn : 1-58113-879-2. doi : 10.1145/1031495.1031508. url : http://doi.acm.org/10.1145/1031495.1031508.

[123] Dumitru Potop-Butucaru et Benoit Caillaud. « Correct-by-construction asynchronous implementation of modular synchronous specifications ». In : *Fundamenta Informaticae* 78.1 (2007), p. 131–159.

[124] P. Ramadge et W. Wonham. « The control of discrete event systems ». In : *Proceedings of the IEEE*. T. 77. 1. Jan. 1989, p. 81–98.

[125] S Ramesh. « Communicating reactive state machines : Design, model and implementation ». In : *IFAC Workshop on Distributed Computer Control Systems*. 1998.

[126] S Ramesh. « Implementation of communicating reactive processes ». In : *Parallel Computing* 25.6 (1999), p. 703–727.

[127] S Ramesh et Chandrashekhar M Shetty. « Impossibility of synchronization in the presence of preemption ». In : *Parallel processing letters* 8.01 (1998), p. 111–120.

[128] S Ramesh et al. « A toolset for modelling and verification of GALS systems ». In : *International Conference on Computer Aided Verification*. Springer. 2004, p. 506–509.

[129] C Rolland. « Requirements engineering for COTS based systems ». In : *Information and software technology* 41.14 (1999), p. 985–990. issn : 0950-5849.

[130] Elizabeth M Royer et Chai-Keong Toh. « A review of current routing protocols for ad hoc mobile wireless networks ». In : *IEEE personal communications* 6.2 (1999), p. 46–55.

[131] Eric Rutten. *"Exercises with Mode Automata, Sigali and SigalSimu"*. popart.inrialpes.fr/ girault/Projets/Synthese/exercices-synthese-controlleurs.ps. Mis en ligne le 16th October 2003.

[132] Takashi Sakairi et al. « Model based control system design using SysML, Simulink, and computer algebra system ». In : *Journal of Control Science and Engineering* 2013 (2013), p. 9.

[133] Andreas Savvides, Chih-Chieh Han et Mani B Srivastava. « Dynamic fine-grained localization in ad-hoc networks of sensors ». In : *Proceedings of the 7th annual international conference on Mobile computing and networking*. ACM. 2001, p. 166–179.

[134] W. K. G. Seah, Z. A. Eu et H. P. Tan. « Wireless sensor networks powered by ambient energy harvesting (WSN-HEAP) - Survey and challenges ». In : *2009 1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace Electronic Systems Technology*. Mai 2009, p. 1–5. doi : 10.1109/WIRELESSVITAE.2009.5172411.

[135] Helmut Seidl. « Least solutions of equations over N ». In : *Automata, Languages and Programming* (1994), p. 400–411.

[136] Raja Sengupta et Stéphane Lafortune. « An optimal control theory for discrete event systems ». In : *SIAM Journal on control and Optimization* 36.2 (1998), p. 488–541.

[137] Rahul C Shah et Jan M Rabaey. « Energy aware routing for low energy ad hoc sensor networks ». In : *Wireless Communications and Networking Conference, 2002. WCNC2002. 2002 IEEE*. T. 1. IEEE. 2002, p. 350–355.

[138] Binbin Shi et al. « A novel energy efficient topology control scheme based on a coverage-preserving and sleep scheduling model for sensor networks ». In : *Sensors* 16.10 (2016), p. 1702.

[139] Kee-Young Shin et al. « REAR : reliable energy aware routing protocol for wireless sensor networks ». In : *Advanced Communication Technology, The 9th International Conference on*. T. 1. IEEE. 2007, p. 525–530.

[140] Victor Shnayder et al. « Simulating the Power Consumption of Large-scale Sensor Network Applications ». In : *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*. SenSys '04. Baltimore, MD, USA : ACM, 2004, p. 188–200. isbn : 1-58113-879-2. doi : 10.1145/1031495.1031518. url : http://doi.acm.org/10.1145/1031495.1031518.

[141] Shaolong Shu et Feng Lin. « Supervisor synthesis for networked discrete event systems with communication delays ». In : *IEEE Transactions on Automatic Control* 60.8 (2015), p. 2183–2188.

[142] *Simulink : Simulation and Model-Based Design http ://uk.mathworks.com/products/simulink/.*

[143] Suresh Singh, Mike Woo et C. S. Raghavendra. « Power-aware Routing in Mobile Ad Hoc Networks ». In : *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking*. MobiCom '98. Dallas, Texas, USA : ACM, 1998, p. 181–190. isbn : 1-58113-035-X. doi : 10.1145/288235.288286. url : http://doi.acm.org/10.1145/288235.288286.

[144] Sasa Slijepcevic et Miodrag Potkonjak. « Power Efficient Organization of Wireless Sensor Networks ». In : 2001, p. 472–476.

[145] *Stateflow : Model and simulate decision logic using state machines and flow charts. http ://uk.mathworks.com/products/stateflow/.*

[146] Rong Su et J. G. Thistle. « A Distributed Supervisor Synthesis Approach Based on Weak Bisimulation ». In : *2006 8th International Workshop on Discrete Event Systems*. Juil. 2006, p. 64–69. doi : 10.1109/WODES.2006.1678409.

[147] Di Tian et Nicolas D Georganas. « A coverage-preserving node scheduling scheme for large wireless sensor networks ». In : *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. ACM. 2002, p. 32–41.

[148] Simon Tschirner, Liang Xuedong et Wang Yi. « Model-based validation of QoS properties of biomedical sensor networks ». In : *Proceedings of the 8th ACM international conference on Embedded software*. ACM. 2008, p. 69–78.

[149] *Uppaal home page. http ://www.uppaal.com/.*

[150] András Varga. « Discrete event simulation system ». In : *Proc. of the European Simulation Multiconference (ESM'2001)*. 2001.

[151] András Varga et Rudolf Hornig. « An overview of the OMNeT++ simulation environment ». In : *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*. ICST (Institute for Computer Sciences, Social-Informatics et Telecommunications Engineering). 2008, p. 60.

[152] Jeffrey M. Voas. « The Challenges of Using COTS Software in Component-Based Development ». In : *Computer* 31.6 (juin 1998), p. 44–45. issn : 0018-9162.

[153] Runze Wan, Naixue Xiong et al. « An energy-efficient sleep scheduling mechanism with similarity measure for wireless sensor networks ». In : *Human-centric Computing and Information Sciences* 8.1 (2018), p. 18.

[154] Thomas Watteyne, Isabelle Augé-Blum et Stéphane Ubéda. « Dual-mode real-time mac protocol for wireless sensor networks : a validation/simulation approach ». In : *Proceedings of the first international conference on Integrated internet ad hoc and sensor networks*. ACM. 2006, p. 2.

[155] Geoffrey Werner-Allen et al. « Monitoring volcanic eruptions with a wireless sensor network ». In : *Wireless Sensor Networks, 2005. Proceeedings of the Second European Workshop on*. IEEE. 2005, p. 108–120.

[156] W.M. Wonham. *Supervisory Control of Discrete-Event Systems*. ECE 1636F/1637S 2009-10. 2010.

[157] Xiaodong Xian, Weiren Shi et He Huang. « Comparison of OMNET++ and other simulator for WSN simulation ». In : *Industrial Electronics and Applications, 2008. ICIEA 2008. 3rd IEEE Conference on*. IEEE. 2008, p. 1439–1443.

[158] Ya Xu, John Heidemann et Deborah Estrin. « Geography-informed Energy Conservation for Ad Hoc Routing ». In : *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*. MobiCom '01. New York, NY, USA : ACM, 2001, p. 70–84. doi : 10.1145/381677.381685. url : http://doi.acm.org/10.1145/381677.381685.

[159] Fan Ye et al. « A scalable solution to minimum cost forwarding in large sensor networks ». In : *Computer Communications and Networks, 2001. Proceedings. Tenth International Conference on*. IEEE. 2001, p. 304–309.

[160] Fan Ye et al. « PEAS : A robust energy conserving protocol for long-lived sensor networks ». In : *Distributed computing systems, 2003. Proceedings. 23rd international conference on*. IEEE. 2003, p. 28–37.

[161] Wei Ye, John Heidemann et Deborah Estrin. « Medium access control with coordinated adaptive sleeping for wireless sensor networks ». In : *IEEE/ACM Transactions on Networking (ToN)* 12.3 (2004), p. 493–506.

[162] Jane Yang Yu et Peter Han Joo Chong. « A survey of clustering schemes for mobile ad hoc networks ». In : *IEEE Communications Surveys & Tutorials* 7.1 (2005), p. 32–48.

[163] Liyang Yu, Neng Wang et Xiaoqiao Meng. « Real-time forest fire detection with wireless sensor networks ». In : *Wireless Communications, Networking and Mobile Computing, 2005. Proceedings. 2005 International Conference on*. T. 2. IEEE. 2005, p. 1214–1217.

[164] Yan Yu, Ramesh Govindan et Deborah Estrin. « Geographical and energy aware routing : A recursive data dissemination protocol for wireless sensor networks ». In : (2001).

[165] Zhuxiu Yuan et al. « A balanced energy consumption sleep scheduling algorithm in wireless sensor networks ». In : *Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International*. IEEE. 2011, p. 831–835.

[166] Sajeh Zairi et al. « Nodes self-scheduling approach for maximising wireless sensor network lifetime based on remaining energy ». In : *IET Wireless Sensor Systems* 2.1 (2012), p. 52–62.

[167] Renyuan Zhang et al. « Distributed supervisory control of discrete-event systems with communication delay ». In : *Discrete Event Dynamic Systems* 26.2 (2016), p. 263–293.

[168] Mengxuan Zhao et al. « Discrete control for smart environments through a generic finite-state-models-based infrastructure ». In : *European Conference on Ambient Intelligence*. Springer. 2014, p. 174–190.

Table des matières

Ordonnancement des réseaux de capteurs sans fil embarqués

Résumé

Les réseaux de capteurs sans fil ont attiré beaucoup d'activités de recherche et développement au cours de la dernière décennie. Pourtant, leur utilisation est restreinte, à ce jour, à la surveillance et l'acheminement des informations détectées. Cette thèse vise à introduire un nouvel aspect intéressant de point de vue fonctionnel. Partant d'une exigence spécifiée initialement, mettre en œuvre une synergie à plusieurs niveaux entre un ensemble des nœuds, tout en se basant sur une interaction adéquate. Ceci est réalisé par la génération automatique de code (correct par construction) et sa distribution par la suite en s'appuyant sur la théorie de contrôle par supervision. La synthèse de contrôleurs discrets SCD est une application de ce cadre théorique. Dans ce manuscrit, nous montrons comment la technique de la SCD peut être utilisée dans le domaine de réseaux de capteurs sans fil. Ainsi son potentiel se décline à deux niveaux. L'ordonnancement intra-cluster (groupe redondant de capteurs) avec des spécifications exprimant l'exclusion mutuelle lors de l'activation d'un capteur au sein d'un cluster, essentielle pour économiser l'énergie du réseau ainsi que la génération automatique d'un algorithme de routage optimal et multicritères pour les réseaux de capteurs. Spécifiquement, un chemin optimal devrait avoir à la fois une longueur minimale en termes de distance tout en évitant des chemins composés de nœuds dits « maillons faibles » (ayant un niveau d'énergie plus bas que la majorité).

Les outils formels cités s'appuient sur une démarche de modélisation basée sur des machines à états finis communicantes. Les verrous scientifiques sont liés à la nature d'un réseau de capteurs ainsi qu'à sa taille. La SCD génère des contrôleurs monobloc, alors qu'au sein d'un réseau de capteurs le traitement est essentiellement distribué. La problématique est celle de la distribution d'un contrôleur global, qui se présente sous forme d'une contrainte logique exprimée sur l'état global du réseau, sur chacun des nœuds du réseau, en ajoutant la synchronisation nécessaire pour garantir un fonctionnement distribué équivalent au contrôleur initialement généré.

Mots clés : réseaux de capteurs, synthèse de contrôleurs discrets, ordonnancement, génération automatique, correct par construction

---

: Laboratoire Ampère
25 Avenue Jean Capelle, Villeurbanne 69621 Cedex – France

Automatic Generation of Wireless Sensor Networks Scheduling

Abstract

Wireless Sensor networks are attracted many activities of research and development during the last decade. Yet, the distributed behavior of a WSN remains centered on two main objectives: sensing and routing. This thesis advocates the introduction of an additional feature, which can be considered interesting from a functional point of view and potentially from the power consumption one: starting from a designer-specified requirement, implement a multiple level synergy between (groups of) nodes, based on adequate interaction. This is achieved by automatic generation and distribution of correct-by-construction code, relying on the Supervisory Control Theory. The Discrete Controller Synthesis (DCS) technique is an application of this theoretic framework. In this thesis, we show how DCS can be used for WSN. Thus, its potential is at two levels. The intra-cluster scheduling of a redundant group of sensors with specifications expressing the mutual exclusion during the activation of a sensor within a cluster, essential to save the energy within the network and then a multicriteria automatic generation of an optimal routing functionality. Specifically, an optimal path should have both a minimal length and go through nodes having maximal residual energy.
The cited formal tools lean on a modelling approach based on communicating finite state machines (CFSM). The scientific challenges are generally related to the nature of the WSN as well as to its size. The DCS can only generates a monoblock controllers, while the WSN's behavior is essentially distributed. The issue is how to distribute a global controller, who appears in the form of a logical constraint expressed on the global state of the network, into local controllers while adding the necessary synchronization to guarantee a distributed functioning equivalent to the initially generated controller.

Keywords: wsn, discret control synthesis, scheduling, automatic generation, correct by construction

FOLIO ADMINISTRATIF

THESE DE L'UNIVERSITE DE LYON OPEREE AU SEIN DE L'INSA LYON

NOM : MEZNI
(Avec précision du nom de jeune fille, le cas échéant)

DATE de SOUTENANCE : 14 Mars 2019

Prénoms : Anis

TITRE : Ordonnancement des réseaux de capteurs sans fil embarqués

NATURE : Doctorat

Numéro d'ordre : 2019LYSEIX030

Ecole doctorale : Electronique, Electrotechnique et Automatique

Spécialité : Automatique

RESUME : Les réseaux de capteurs sans fil ont attiré beaucoup d'activités de recherche et développement au cours de la dernière décennie. Pourtant, leur utilisation est restreinte, à ce jour, à la surveillance et l'acheminement des informations détectées. Cette thèse vise à introduire un nouvel aspect intéressant de point de vue fonctionnel. Partant d'une exigence spécifiée initialement, mettre en œuvre une synergie à plusieurs niveaux entre un ensemble des nœuds, tout en se basant sur une interaction adéquate. Ceci est réalisé par la génération automatique de code (correct par construction) et sa distribution par la suite en s'appuyant sur la théorie de contrôle par supervision. La synthèse de contrôleurs discrets SCD est une application de ce cadre théorique. Dans ce manuscrit, nous montrons comment la technique de la SCD peut être utilisée dans le domaine de réseaux de capteurs sans fil. Ainsi son potentiel se décline à deux niveaux. L'ordonnancement intra-cluster (groupe redondant de capteurs) avec des spécifications exprimant l'exclusion mutuelle lors de l'activation d'un capteur au sein d'un cluster, essentielle pour économiser l'énergie du réseau ainsi que la génération automatique d'un algorithme de routage optimal et multicritères pour les réseaux de capteurs. Spécifiquement, un chemin optimal devrait avoir à la fois une longueur minimale en termes de distance tout en évitant des chemins composés de nœuds dits « maillons faibles » (ayant un niveau d'énergie plus bas que la majorité). Les outils formels cités s'appuient sur une démarche de modélisation basée sur des machines à états finis communicantes. Les verrous scientifiques sont liés à la nature d'un réseau de capteurs ainsi qu'à sa taille. La SCD génère des contrôleurs monobloc, alors qu'au sein d'un réseau de capteurs le traitement est essentiellement distribué. La problématique est celle de la distribution d'un contrôleur global, qui se présente sous forme d'une contrainte logique exprimée sur l'état global du réseau, sur chacun des nœuds du réseau, en ajoutant la synchronisation nécessaire pour garantir un fonctionnement distribué équivalent au contrôleur initialement généré.

MOTS-CLÉS : réseaux de capteurs, Synthèse de contrôleurs discrets, Ordonnancement, Génération automatique, correct par construction

Laboratoire (s) de recherche : Laboratoire Ampère, 25 Avenue Jean Capelle, Villeurbanne 69621 Cedex, France

Directeur de thèse: Eric Niel

Président de jury : Néjib Ben Hadj Alouane

Composition du jury :

Rutten, éric	Chargé de recherche HDR	INRIA Grenoble Rhône-Alpes	. Rapporteur
Robbana, riadh	Professeur	INSAT- Université de Carthage	Rapporteur
Ben Ayed Jemni, leila	Professeur	ENSI-Université de la Manouba	Examinatrice
Niel, éric	Professeur	Ampère-INSA de Lyon	Directeur de thèse
Ben Ahmed, samir	Professeur	FST-Université Tunis Elmanar	Co-directeur de thèse
Dumitrescu, emil	Maître des conférences	Ampère-INSA de Lyon	Invité