Master 2
Réseaux Informatiques et Systèmes Embarqués

Présenté par
François Clad
francois.clad@etu.unistra.fr

Diffusion efficace dans les réseaux de capteurs économes en énergie

Encadré par
Antoine Gallais et Pascal Merindol
{gallais,merindol}@unistra.fr

Au sein de
Équipe Réseaux et Protocoles
Laboratoire des Sciences de l'Image,
de l'Informatique et de la Télédétection
Université de Strasbourg

François Clad : *Diffusion efficace dans les réseaux de capteurs économes en énergie,*
Master 2 Réseaux Informatiques et Systèmes Embarqués, © Juin 2011

REMERCIEMENTS

Avant toute chose, je souhaite remercier les personnes qui m'ont apporté leur soutien au cours de ce stage et, sans qui, ces six derniers mois, bien que studieux, ne m'auraient pas été si agréables.

Je remercie en particulier Pascal Mérindol et Antoine Gallais pour la confiance qu'ils m'ont accordée, l'expérience qu'ils ont su me faire partager et l'attention qu'ils ont apportée à mes travaux.

Je souhaite également remercier Julien, Damien, Erkan, Vincent, Fabrice, Jean-Jacques, et plus généralement l'ensemble de l'équipe Réseaux et Protocoles pour leur bonne humeur, leur patience face à mes nombreuses questions et leurs conseils avisés.

Pour finir, je remercie mes collègues de Master RISE atterris dans l'équipe Image et Calcul Parallèle Scientifique, Jean-François et Etienne, pour leur soutien lors de mes déboires avec l'administration.

RÉSUMÉ

Dans le contexte des réseaux de capteurs sans fil, les communications radio constituent la principale source de consommation énergétique. Ces communications ont généralement lieu entre un nœud puits et les autres capteurs ($1 \rightarrow n$, collecte d'information) ou entre les nœuds eux-mêmes ($n \rightarrow n$, diffusion d'information). Afin de favoriser l'efficacité de ce type d'échange et ainsi réduire la consommation énergétique, il est primordial d'utiliser une "structure de communication" adaptée. Une telle structure devra minimiser le nombre de retransmissions nécessaires à l'acheminement des messages, et donc le nombre de relais utilisés.

Dans ce mémoire, nous commençons par décrire les travaux déjà menés dans le domaine puis proposons une solution distribuée construisant une structure adaptée aux deux types de communications envisagés. Dans ce but, nous commençons par appliquer un algorithme localisé visant à limiter le nombre de relais dans le réseau et favoriser ainsi l'efficacité des communications $n \rightarrow n$. Ensuite, grâce à un mécanisme de routage construit par propagation, nous adaptons les chemins aux communications $1 \rightarrow n$ tout en réduisant encore le nombre de relais. Une variable d'ajustement est introduite afin d'obtenir un compromis paramétrable entre ces deux types de communication.

TABLE DES MATIÈRES

1 INTRODUCTION

L'étude des réseaux de capteurs sans fil[18] a connu une récente envolée, en raison des progrès technologiques réalisés dans les domaines de la microélectronique et des systèmes embarqués. Composé de nombreuses entités autonomes, appelées nœuds-capteurs, un tel réseau est utilisé pour collecter des informations sur l'environnement dans lequel il est déployé. Dans un scénario de déploiement typique, il est admis que ces informations sont ensuite acheminées vers une station de base afin d'y être traitées.

Un nœud-capteur est généralement un petit appareil de taille limitée contenant un ou plusieurs capteurs, un processeur, une mémoire, une interface de communication et une alimentation électrique. Une large variété de capteurs mécaniques, thermiques, biologiques, optiques et magnétiques peut être attachée au nœud-capteur pour mesurer les propriétés de son environnement. En raison de sa faible capacité de stockage et de ses conditions de déploiement, dans des zones souvent difficiles d'accès, chaque nœud dispose d'une interface radio lui permettant de transmettre ses données à une station de base, aux ressources moins limitées, appelée puits. Bien que certains nœuds-capteurs soit capables de puiser de l'énergie dans leur environnement grâce, notamment, à des cellules photovoltaïques, la plupart d'entre eux sont alimentés par une batterie. Cette batterie, généralement difficile voire impossible à recharger et dont les communications radio constituent la principale source de consommation énergétique, fait de l'économie d'énergie une problématique de première importance dans le contexte des réseaux de capteurs.

Les applications des réseaux de capteurs sont aussi diverses que les informations qu'ils peuvent collecter. En médecine, ils peuvent servir à surveiller l'état d'un patient resté à domicile, envoyant des compte-rendus réguliers au personnel soignant et alertant les secours en cas de besoin. En éthologie, les nœuds peuvent être implanté sur des animaux afin de suivre et ainsi comprendre leurs déplacements. Ils peuvent également être utilisés pour étudier les mouvements des glaciers, surveiller le niveau de pollution de l'air ou encore détecter des feux de forêt.

Du fait de leur nombre élevé, pouvant aller de quelques dizaines à plusieurs milliers, de leur déploiement aléatoire et de leur portée radio limitée, il est rarement considéré qu'un nœud est à portée de tous les autres. Certains d'entre eux devront donc servir de *relais* à leurs voisins afin de leur permettre d'atteindre les nœuds plus éloignés. Ce type de communication porte le nom de routage multi-sauts. Contrairement aux réseaux d'opérateur classiques, un nœud est rarement, voire jamais en pratique, amené à échanger des messages avec un seul autre (communication unicast $1 \to 1$). La grande majorité des communications s'opère entre le puits et les nœuds, dans un sens comme dans l'autre ($1 \leftrightarrow n$), ou entre un nœud et tous les autres ($n \to n$). Lorsqu'un nœud

ou le puits envoie un message à tous les autres, on parle de diffusion (ou *broadcast*).

Mécanismes de diffusion

Nous avons précédemment défini les réseaux de capteurs comme un ensemble de nœuds-capteurs accompagné d'un puits. Cependant, la présence de ce dernier peut être transitoire, le temps de collecter les données des nœuds-capteurs (le puits peut donc être absent - éteint ou hors de portée - le reste du temps). Dans une telle situation, les capteurs doivent pouvoir s'échanger efficacement des messages malgré l'absence du puits, mais également être capables, lors de son arrivée/activation, d'optimiser leurs communications avec lui. Notre travail a donc consisté à construire une structure hybride adaptée à la fois aux communications $1 \rightarrow n$ (avec le puits) et $n \rightarrow n$ (entre nœuds). Le principal critère déterminant la qualité d'une structure de communication étant le nombre de retransmissions, nous nous sommes efforcés à réduire le nombre de relais de manière globale et de minimiser le nombre de *sauts* que doit traverser un message pour atteindre le puits.

Un réseau de capteurs peut être représenté par un graphe non orienté ayant pour sommets les nœuds-capteurs et pour arêtes les liens radio actifs entre ceux-ci. Les sommets n'ayant aucune arête incidente sont dit *isolés*, ceux en possédant exactement une sont appelés *feuilles*[1] et les autres portent le nom de nœuds internes, ou *relais*. La structure minimisant la longueur des chemins entre un sommet donné et tous les autres (optimisation $1 \rightarrow n$) se nomme arbre des plus courts chemins (SPT), et celle qui minimise le nombre de relais dans l'ensemble du graphe (optimisation $n \rightarrow n$) équivaut à un arbre couvrant maximisant le nombre de feuilles (MLST). Alors qu'un SPT est relativement aisé à construire, notamment grâce à l'algorithme du gradient, il en va tout autrement pour le MLST. En effet, le problème du MLST appartient à la catégorie des problèmes NP-difficiles : il n'existe pas d'algorithme *efficace* pour leurs résolutions (sauf si $P = NP$). L'ensemble des relais d'un MLST formant un ensemble dominant connecté de taille minimale (MCDS), ces deux problèmes peuvent être considérés comme équivalents (bien que leurs résolutions ne soient pas équivalentes, le résultat optimal est le même).

En raison des difficultés liées à la résolution de ce problème, une méthode exacte ne peut être envisagée dans un environnement aussi contraint, en termes de puissance de calcul comme de communication, qu'un réseau de capteurs. La première partie de notre travail a donc consisté en la recherche d'un algorithme approximant un MLST et satisfaisant les contraintes inhérentes aux réseaux de capteurs, à savoir une faible complexité de calcul et une prise de décision locale basée sur une connaissance partielle du réseau (voisinage à k sauts). Parallèlement, nous avons également cherché à résoudre ce problème de manière optimale afin d'établir une borne supérieure à laquelle comparer notre proposition. Celle-ci a pour objectif la construction d'une structure de communication hybride adaptée aux communications par diffusion.

1 En pratique, une feuille est un nœud ayant un seul voisin relais

Ce mémoire est organisé de la manière suivante. Le chapitre 2 présente une vue d'ensemble des travaux déjà menés dans le domaine et détaille les plus pertinents d'entre eux. Le chapitre 3 expose le fonctionnement et les performances de notre proposition. Enfin, le chapitre 4 conclut ce mémoire, rappelant brièvement les résultats obtenus et indiquant les pistes envisagées pour des recherches futures.

2 CONTEXTE

2.1 PRÉSENTATION GÉNÉRALE

Dans cette section sont présentés brièvement les travaux déjà menés pour la résolution du problème du MLST. Nous abordons tout d'abord les méthodes de résolution exacte, par programmation linéaire, et les algorithmes d'approximation centralisés, que nous utiliserons comme borne supérieure, puis les heuristiques distribuées, adaptées aux réseaux de capteurs.

2.1.1 *Résolution exacte*

Dans [3], Fujie présente un algorithme par séparation et évaluation pour la résolution du problème du MLST en programmation linéaire. Celui-ci commence par exécuter différentes heuristiques, notamment [10] et [12], lesquelles peuvent, dans certains cas, fournir un résultat optimal avec un temps de calcul minime. Si le résultat de l'une des heuristiques contient $(n-1)$ feuilles, où n est le nombre de nœuds dans le graphe, il est nécessairement optimal (cas d'une étoile) et l'algorithme peut donc s'arrêter. Sinon, le problème est subdivisé en sous-problèmes pour être résolu par programmation linéaire.

Nous nous sommes basés sur la formulation par contraintes décrite dans son travail pour établir un modèle de programmation linéaire, lequel a ensuite pu être implémenté dans un solveur générique.

Dans un second article ([4]), l'auteur propose deux autres formulations du problème du MLST, toujours en utilisant une approche par programmation linéaire. La première, nommée *Edge-Vertex Formulation* (Formulation Sommet-Arête), est très proche de celle utilisée dans l'article précédent. Elle utilise deux variables (x, y) : x indiquant pour chaque arête si elle fait ou non partie de l'arbre et y servant à déterminer si un sommet est feuille. Dans la seconde approche, appelée *Vertex Formulation* (Formulation Sommet), les contraintes sont fusionnées de manière à ne plus utiliser que la seule variable y.

La première formulation présentée dans cet article nous a permis d'adapter notre modèle afin d'en réduire la complexité. Ainsi que nous le détaillons par la suite, cela n'était toutefois pas suffisant pour rendre cette approche applicable à des topologies réalistes. Il a donc été nécessaire de recourir à des algorithmes d'approximations.

2.1.2 *Approximations centralisées*

Un premier algorithme d'approximation centralisée pour le MLST est présenté par Lu et Ravi[10]. Celui-ci permet de construire en temps polynomial une 3-approximation d'un MLST, c'est-à-dire un graphe dont le nombre de feuille est au moins le tiers de celui d'un MLST. Il consiste à créer de manière naïve une forêt *feuillue*, composée d'arbres contenant un nombre important de feuilles, puis à relier ces arbres, for-

mant ainsi une unique arbre couvrant. La complexité de cet algorithme est en $O(m+n)$, où m représente le nombre d'arêtes dans le graphe initial et n le nombre de sommets.

Dans [12], Solis-Oba propose un autre algorithme centralisé pour l'approximation du MLST en temps polynomial. Le principe est toujours de construire une forêt *feuillue* dont les arbres seront ensuite fusionnés, mais des contraintes supplémentaires sont appliquées ici afin de préserver un nombre plus important de feuilles.

En pratique, il s'agit de (i) sélectionner dans le graphe de départ un sommet de degré supérieur ou égal à 3. Celui-ci deviendra racine d'un nouvel arbre et ses voisins en seront les feuilles. On applique ensuite (ii) à chaque feuille un ensemble de règles de croissance permettant d'étendre l'arbre tout en maintenant une forte proportion de feuilles. Lorsque toutes les feuilles ont été traitées, on reprend à l'étape 1 en cherchant une nouvelle racine. On continue ainsi jusqu'à ce qu'il n'y ait plus de sommet éligible. On connecte alors de manière arbitraire les arbres créés précédemment et les sommets isolés.

La complexité de cet algorithme est en $O(m)$, pour un ratio d'approximation de 2. Cette seconde méthode est donc à la fois plus performante et moins coûteuse que [10]. Ainsi que nous l'avons annoncé en début de section, ces solutions centralisées ne sont pas envisageables en déploiement réel, mais serviront de référence pour évaluer notre proposition.

2.1.3 *Heuristiques distribuées*

Cette section porte sur les heuristiques distribués approximant un MLST. A la différence des précédentes, les techniques présentées ici sont des algorithmes distribués ou localisés et donc potentiellement applicables à un réseau de capteurs sans fil.

Dans [11], Rovedakis propose une solution reprenant les principes de l'algorithme précédent, cependant elle se différencie par une construction à la fois distribuée et auto-stabilisante, plus en phase avec les contraintes des réseaux *ad-hoc*. Cet algorithme convergera donc toujours vers un état stable, quel que soit l'état initial.

La première étape consiste à créer un ensemble d'arbres *feuillus* à partir des sommets de degré supérieur ou égal à 3, mais aucune règle de croissance ne leur est ensuite appliquée. On obtient donc un ensemble d'étoiles ayant au moins trois branches. Lors de la seconde étape ces étoiles sont reliées entres elles aux sommets isolés. Ceci fait intervenir un mécanisme d'élection visant à limiter le nombre de feuilles perdues dans le processus. Le graphe retourné est une 2-approximation d'un MLST obtenu avec une complexité en $O(n^2)$, où n est le nombre de sommets dans le graphe initial.

Dans [9] est présentée une vue d'ensemble des algorithmes, distribués et localisés, permettant de diminuer le nombre d'arêtes dans un graphe. Parmi ces techniques figurent le graphe de Gabriel[5], le graphe de Yao[17] et le graphe de voisinage relatif (RNG)[14], mais aussi l'arbre couvrant de poids minimal pour lequel de nouvelles méthodes de construction sont proposées. Ces approches, appelées arbres couvrants de poids minimaux *k*-localisés, consistent tout d'abord à

calculer un MST local avec une connaissance à k sauts, puis à échanger des informations entre voisins afin de créer, selon le modèle choisi, l'union ou l'intersection des MSTs locaux.

Wu et Li[15], décrivent une solution localisée pour la construction d'une approximation de MCDS. Se basant sur une connaissance de son voisinage à deux sauts, chaque nœud détermine son état, dominant ou dominé, et fait part de sa décision à ses voisins. L'ensemble dominant connecté (CDS) ainsi formé est ensuite *élagué*, tout en maintenant la connexité de l'ensemble, grâce à des règles de réduction. L'algorithme permet d'obtenir un CDS de taille réduite pour une complexité en $O(\Delta^3)$, où Δ est le degré maximum d'un sommet dans le graphe, et de 1 à 2 messages par nœud (si l'on considère connu le voisinage direct).

Adjih et al.[1] proposent une méthode alternative pour approximer un MCDS, basée sur des relais multipoints (MPRs) et appelée MPR-CDS. Dans un premier temps, deux techniques pour le calcul de l'ensemble des MPRs, formant un CDS du 2-voisinage de chaque nœud, sont présentées, l'une naïve et l'autre utilisant les identifiants des nœuds. Cette dernière est décrite comme moins efficace que la solution naïve, mais a pour avantage qu'un nœud peut détecter par lui-même s'il est dans l'ensemble des MPRs d'un de ses voisins, permettant ainsi une économie en terme de communication.

Dans un second temps est exposé l'algorithme de construction du MPR-CDS, lequel repose sur deux règles. Un nœud se considère dans le CDS 1) s'il possède l'identifiant le plus petit parmi tous ses voisins ou 2) s'il appartient à l'ensemble des MPRs de son voisin de plus petit identifiant. Les performances de cet algorithme sont ensuite évaluées comparativement à la solution généralisée de Wu-Li[2], pour des résultats très proches.

2.2 ALGORITHMES POUR LE CALCUL DU MLST

Cette section détaille certaines méthodes de calcul et d'approximation d'un MLST évoquées dans la section précédente. Dans une première partie sont décrites les solutions centralisées qui nous serviront de références puis, dans une seconde partie l'algorithme distribué à la base de notre proposition.

2.2.1 *Solutions centralisées*

Ce travail a débuté par la recherche d'une méthode pour la résolution du problème du MLST de manière optimale. Du fait de sa complexité et de sa centralisation, une telle résolution ne serait bien évidemment pas envisageable dans un réseau de capteurs. En revanche, elle fournira une borne supérieure pour une évaluation objective des solutions proposées. Là où les heuristiques sont d'ordinaire comparées entre elles, s'annonçant "meilleures" que d'autres, nous nous efforcerons de proposer une comparaison, absolue, à un optimum.

Optimisation linéaire

Le problème du MLST étant NP-difficile, il n'existe pas d'algorithme efficace pour le résoudre de manière exacte. La seule solution est donc

d'explorer toutes les possibilités. Nous utiliserons à cette fin la programmation linéaire, une méthode de recherche opérationnelle très utilisée pour la résolution de problèmes d'optimisation tels que le nôtre.

Cette méthode requiert une formalisation du problème sous forme de fonctions linéaires afin de représenter l'objectif et les contraintes inhérentes au problème. La **fonction-objectif** a pour but la minimisation ou maximisation d'un paramètre, tandis que les **contraintes** définiront le polygone des solutions admissibles (*feasible region*).

MODÈLE Soit $G = (V, E)$ un graphe connexe non orienté, où V représente l'ensemble des sommets et E celui des arêtes. Soit $T = (V, E_T)$, avec $E_T \subseteq E$, un arbre couvrant dans G. Pour tout sommet $i \in V$, on note $\delta(i)$ l'ensemble des arêtes adjacentes à i dans G. Pour toute arête $(i,j) \in E$, $x_{(i,j)} = 1$ si $(i,j) \in E_T$ et $x_{(i,j)} = 0$ sinon. Enfin, pour tout arc $(i,j) \in E$, $f_{(i,j)}$ représente le flux traversant cet arc.

$$\text{Maximiser} \sum_{i \in V} y_i \tag{1}$$

$$\text{Tel que} \sum_{(i,j) \in \delta(i)} x_{(i,j)} + (|\delta(i)| - 1) \times y_i \leqslant |\delta(i)| \ (i \in V) \tag{2}$$

$$f_{(i,j)} \leqslant (n-1) \times x_{(i,j)} \ ((i,j) \in E) \tag{3}$$

$$\sum_{(j,i) \in E} f(j,i) = \sum_{(i,j) \in E} f(i,j) + 1 \ (i \in V \backslash \{s\}) \tag{4}$$

$$\sum_{(s,i) \in E} f(s,i) = n \tag{5}$$

La première équation (1) constitue la fonction-objectif du modèle. Elle vise à maximiser la somme sur V des y_i ; or, d'après (2), $y_i = 1$ si et seulement si i est une feuille dans T. L'équation peut donc être traduite par "maximiser le nombre de nœuds feuilles". Les autres contraintes (3 à 5) servent à garantir la connexité, l'absence de cycle et la couverture du graphe final. En d'autres termes, à s'assurer que l'on obtient bien un arbre couvrant. Nous utilisons pour ce faire le mécanisme d'un graphe de flux. Ce flux, représenté par des jetons, est initialisé à $|V|$ sur un sommet choisi arbitrairement dans V (5). Puis, pour chaque autre sommet, est appliquée la règle (4), forçant le flux (somme des jetons sur les liens entrants) entrant à être égal au flux sortant plus un. Enfin, la règle (3) interdit toute circulation de flux sur les arêtes n'appartenant pas à T.

Cette approche par flux est cependant réservée aux seuls graphes orientés. Pour la rendre utilisable sur des graphes non orientés, il suffit de considérer nos arêtes comme des arcs et de créer, pour chacune, un arc retour.

EXPÉRIMENTATIONS Les expérimentations ont été effectuées avec le logiciel ILOG CPLEX Optimization Studio[1], en version 12.2, sur des machines dont les caractéristiques techniques ont été reportées dans le tableau 1. Ce logiciel intègre un système parallélisant automatiquement la résolution du problème sur tous les cœurs disponibles, profitant ainsi pleinement de la technologie d'hyperthreading du processeur *Intel i7*. Le modèle a été implémenté avec le langage de modélisation

1 http://www-01.ibm.com/software/integration/optimization/cplex-optimization-studio/

Processeur		
Modèle		Intel i7
Fréquence		2.67 GHz
Taille du cache	L1	256 Ko
	L2	1024 Ko (1 Mo)
	L3	8192 Ko (8 Mo)
Nombre de cœurs		4
Nombre de threads		8
Mémoire		
Capacité		8192 Mo (4 × 2048 Mo)
Vitesse		1333 MHz (0.8 ns)
Système		
Distribution		Ubuntu 9.10 “Karmic Koala” 64 bits
Version du noyau		2.6.31-22-generic

TAB. 1: Caractéristiques techniques des machines utilisées pour les expérimentation en programmation linéaire

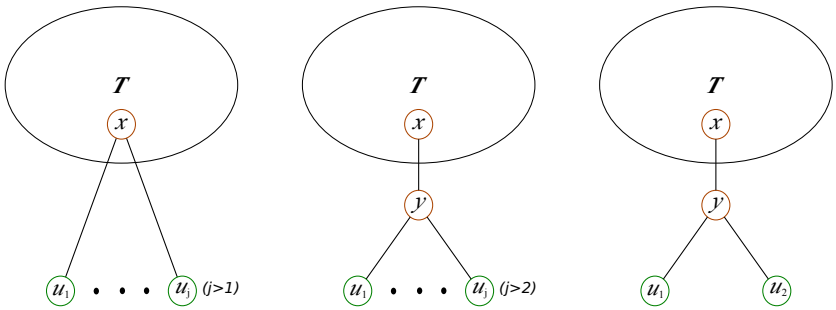
	Degré moyen	Nombre de nœuds				
		40	45	50	55	60
Durée	3	1 s	31 s	75 s	119 s	1207 s
	4	2 s	59 s	92 s	583 s	1876 s
	5	9 s	59 s	538 s	839 s	3691 s
Mémoire	3	10 Ko	92 Ko	2.9 Mo	176.55 Mo	2.06 Go
	4	10 Ko	788 Ko	5.21 Mo	1.02 Go	2.64 Go
	5	10 Ko	10.52 Mo	628.36 Mo	2.25 Go	8.14 Go

TAB. 2: Durée et quantité de mémoire utilisée pour le calcul d’un MLST par la programmation linéaire

GNU MathProg, lequel présente le double avantage d’être à la fois très proche de l’écriture mathématique et de bien séparer le modèle des données. Il a ensuite été traduit au format *CPLEX LP* grâce au *GNU Linear Programming Kit*[2].

Le tableau 2 montre l’évolution de la durée de résolution et de la consommation de mémoire en fonction de la complexité du problème. Cette complexité est ici représentée par deux variables : le nombre de nœuds dans le réseau et leur degré moyen. On y remarque que le coût de la résolution, très faible pour des petites topologies, augmente exponentiellement avec la complexité du problème. De plus, les expérimentations menées sur de plus grandes topologies (65 nœuds et plus) ont toutes échoué, après plusieurs heures de traitement, en raison d’un dépassement de la capacité de la mémoire vive. Bien qu’il soit possible de configurer le logiciel CPLEX pour borner la consommation de mémoire vive, cela aurait un impact sur le temps de traitement, déjà considérable. Nous sommes donc confrontés à une limitation temporelle (durée trop longue) et spatiale (consommation de mémoire

2 http://www.gnu.org/software/glpk/



(a) Règle 1_a (b) Règle 1_b (c) Règle 2

FIG. 1: Règles de croissance dans l'algorithme de Solis-Oba

trop importante) restreignant l'utilisation de cette méthode à de *petites* topologies, dix à vingt fois inférieures à celles que nous souhaiterions étudier.

Approximation

Au vu des limites expérimentales mises en évidence, nous envisageons ici une approche par approximation. L'objectif d'une telle démarche est de réduire la complexité de la résolution tout en restant proche de l'optimum. Galbiati et. al[6] ont démontré que le problème du MLST était MAX SNP-difficile. Il existe donc pour ce problème des α-approximations (avec $\alpha > 1$) s'exécutant en temps polynomial, mais aucun schéma d'approximation en temps polynomial (PTAS). En d'autres termes, cela signifie que l'on peut trouver des algorithmes ayant un ratio d'approximation faible (proche de 1), sans toutefois pouvoir réduire ce ratio "aussi bas que l'on veut". Nous avons choisi d'implémenter l'algorithme [12], celui-ci offrant le meilleur ratio d'approximation parmi ceux présentés dans la littérature.

PRINCIPE Comme énonce dans 2.1.2, l'algorithme se déroule en deux étapes. La première consiste à créer une forêt feuillue, dans laquelle chaque arbre contiendra une importante proportion de feuilles. Ces arbres sont créés de la manière suivante. Dans un graphe de départ, connecté et non orienté, on choisit comme racine de l'arbre un sommet de degré 3 ou plus. Les voisins de ce sommet deviendront alors les feuilles du nouvel arbre. À chacune de ces feuilles sont ensuite appliquées des règles de croissances, comme illustré par la figure 1. Par la suite, on qualifiera d'isolé un nœud n'étant rattaché à aucun arbre.

Dans ces schémas, T représente l'arbre en cours de croissance et x la feuille sur laquelle la règle est appliquée. Les trois règles sont appliquées selon un ordre de priorité, les règles de priorité 1, ou règles 1 (figures 1a et 1b) étant plus prioritaires que la règle de priorité 2, ou règle 2. Ainsi, si x possède au moins deux voisins directs u_i n'appartenant encore à aucun arbre, ceux-ci lui sont rattachés (1_a). Le sommet x perd donc son statut de feuille au profit des u_i, lesquelles deviennent de nouvelles feuilles de T. Si x ne possède qu'un seul voisin isolé y mais que ce dernier à lui-même trois voisins isolés u_i, la règle 1_b peut être appliquée. Des liens sont ajoutés entre les sommets x et y puis entre y et les u_i ; x et y deviennent alors des nœuds internes et les u_i

des feuilles de T. La règle 1_a provoque la *perte* d'une feuille de T pour un gain de deux ou plus, tandis que la règles 1_b lui en fait perdre deux et gagner au moins trois. Dans tous les cas, l'application de l'une de ces règles augmente le nombre de feuilles de T.

Si aucune feuille n'est plus éligible aux règles 1_a ou 1_b, toutes sont à nouveau parcourues à la recherche d'un sommet x ayant un voisin isolé y possédant lui-même exactement deux voisins isolés u_1 et u_2. La règle 2 (figure 1c) est alors appliquée, ajoutant y, u_1 et u_2 à l'arbre de T de la même manière que la règle 1_b. La principale différence entre ces deux règles est qu'il n'y a, avec la règle 2, aucun *gain* de feuille. En effet, deux feuilles potentielles de T, x et y, deviennent nœuds internes tandis que seulement deux nouvelles feuilles, u_1 et u_2 sont créées. Cette règle sert donc essentiellement à étendre l'arbre dans l'espoir de trouver de nouveaux nœuds sur lesquels appliquer les règles de priorité 1.

Lorsqu'aucune règle ne peut plus être appliquée aux feuilles de T, un nouveau sommet isolé de degré supérieur ou égal à 3 est recherché pour former un nouvel arbre. Si le graphe ne contient plus de tels sommets, les arbres sont connectés de manière arbitraire entre eux et aux sommets encore isolés, constituant ainsi un arbre couvrant, 2-approximation d'un MLST.

2.2.2 *Solutions distribuées*

Parmi les heuristiques distribuées présentées dans la section 2.1.3, nous avons sélectionné la méthode de Wu et Li[15], pour sa complexité bien plus faible que [11] et ses performances supérieures à [1]. Dans cette partie, nous décrivons donc les variantes de cet algorithme proposées dans la littérature, puis détaillons le fonctionnement de la version choisie comme base pour le présent travail.

Wu-Li simple

À notre connaissance, [15] est le premier article dans lequel est présentée cette solution. Nous l'appellerons donc, pour des raisons de clarté, *Wu-Li simple*. Cet algorithme introduit notamment le processus de marquage, permettant de différencier les nœuds dominants (marqués) et dominés (non marqués). Initialement, tous les nœuds possédant au moins deux voisins non connectés sont marqués, formant, ainsi que les auteurs l'on démontré, un CDS.

Un mécanisme d'élagage est alors utilisé pour réduire la taille de l'ensemble, et ainsi se rapprocher d'un MCDS. Celui-ci est constitué de deux règles (appelées règle 1 et règle 2) qui sont appliquées consécutivement sur chacun des nœuds marqués. La première consiste à retirer le marqueur d'un nœud si tous ses voisins sont couverts par un autre voisin marqué d'identifiant supérieur. La seconde permet d'ôter ce marqueur si tous les voisins du nœud sont couverts par deux autres voisins marqués, d'identifiants supérieurs et connectés entre eux.

Wu-Li avec "wait-and-see"

Stojmenovic et al.[13] se basent sur la solution précédente pour proposer un algorithme de diffusion efficace dans un réseau sans-fil multi-

saut. Celui-ci vise à réduire le surcoût de communication lors d'une diffusion, limitant ainsi les problèmes de collision et de congestion lors de l'opération. L'algorithme se déroule en deux étapes, la première étant effectuée préalablement à tout envoi de message puis lors de changements topologiques, et la seconde à chaque diffusion.

La première étape consiste à réduire la proportion de nœuds relayant les messages. Celle-ci utilise une version légèrement modifiée de l'algorithme de Wu et Li[15], dans laquelle la clé servant à déterminer si le marqueur d'un nœud peut être retiré ou non n'est plus constituée uniquement de l'identifiant du nœud mais d'un couple (degré, position) ou (degré, identifiant).

La seconde étape repose sur un mécanisme connu sous le nom de *wait and see* (attendre et observer). Il consiste ici à faire maintenir par chaque nœud interne une liste des voisins n'ayant pas encore reçu le message. Lors de la première réception d'un message, un nœud interne initialise un *timeout* et cette liste en y plaçant ceux de ses voisins qui n'étaient pas à portée de l'émetteur du message. La liste est ensuite mise à jour à chaque réception supplémentaire du même message. Ce dernier n'est finalement réémis que si la liste n'est pas vide à l'expiration du *timeout*.

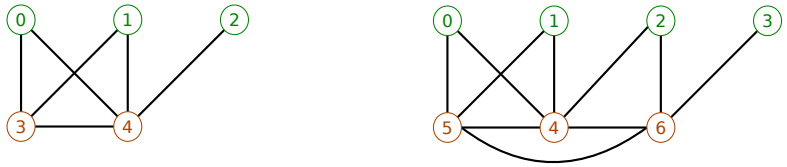
Wu-Li avec économie l'énergie

Dans [16], Wu et al. proposent une extension pour l'algorithme de Wu-Li afin d'augmenter son efficacité énergétique. Le principe appliqué est le même que dans [13], à savoir une modification des règles pour prendre en compte des paramètres supplémentaires. La clé contiendra donc non seulement l'identifiant et le degré du nœud, mais aussi son niveau de batterie, lequel sera prioritaire sur les deux autres lors du choix des relais. Un nœud avec un plus faible niveau de batterie aura donc moins de chances de devenir relais. Cela permet de ne pas sélectionner systématiquement les mêmes relais et donc de répartir la charge sur davantage de nœuds, ce qui aura pour effet d'augmenter la durée de vie du réseau.

Wu-Li généralisé

Dans la première version de leur algorithme, Wu et Li proposaient deux règles pour réduire le nombre de relais après la phase de marquage, retirant le marqueur des nœuds couverts par un ou deux de leurs voisins, respectivement. Même si cela permet de supprimer une forte proportion de relais inutiles, ce traitement devrait être extensible aux nœuds couverts par trois voisins ou plus. Dans [2], Dai et Wu proposent donc une généralisation de ces règles, appelée règle k, permettant de retirer le marqueur d'un nœud couvert par k voisins.

Les auteurs ont montré qu'en plus de retourner un CDS plus petit que l'algorithme initial, cette généralisation rend aussi l'opération moins complexe, en $O(\Delta^2)$ contre $O(\Delta^3)$ avec les règles *1* et *2*, pour le même coût de communication.



(a) Règle 1 (b) Règle 2

FIG. 2: Règles d'élagage de l'algorithme de Wu-Li

Wu-Li hybride

Cette dernière variante est celle que nous utiliserons dans la suite de ce travail. Il s'agit d'une compilation des extensions présentées dans le début de cette section et dont le fonctionnement est le suivant. Chaque nœud commence par apprendre ses voisins directs (1-connaissance) ainsi que les liens entre ceux-ci. Plus formellement, cela se traduit par : soit $N(u) = \{w | \{u, w\} \in E\}$ l'ensemble des voisins directs de $u \in V$, la connaissance requise par u est $N(w) \cap N(u), \forall w \in N(u)$. Une fois cette connaissance acquise, un nœud décide d'être *marqué* si au moins deux de ses voisins ne sont pas connectés entre eux. La taille du CDS ainsi formé est ensuite réduite par l'application successives des deux règles d'élagage.

La première règle consiste à retirer la marque d'un sommet si tous ses voisins sont couverts par un voisin marqué d'identifiant supérieur. Sur la figure 2a, les sommets d'identifiants 3 et 4 sont marqués. Le sommet 3 couvre 0 et 1, tandis que 4 couvre 0, 1 et 2. Dans ce cas là, le nœud 3 a pour voisin 4, lequel couvre tous ses voisins (0 et 1) et possède un identifiant supérieur. Le sommet 3 peut donc retirer son marqueur, laissant 4 seul relais dans ce graphe. La seconde règle retire la marque des sommets dont tous les voisins sont couverts par deux voisins connectés, marqués et d'identifiants supérieurs. Dans l'exemple proposé sur la figure 2b, les nœuds 4, 5 et 6 sont marqués. Les voisins du sommet 4 sont tous couverts par 5 et 6. Ces derniers étant connectés, entres eux et à 4, et ayant un identifiant supérieur, le marqueur du nœud 4 peut être retiré.

En pratique, l'identifiant utilisé par les règles d'élagage est en fait une clé constituée non seulement de l'identifiant (supposé unique) du nœud, mais aussi de son degré. Ainsi, un nœud ayant un degré plus élevé aura plus de chance de rester relais, ce qui permet de réduire encore la taille du CDS. L'identifiant du nœud ne servira plus alors qu'à départager les nœuds de degrés identiques. De même, la règle 2 est légèrement modifiée selon la formulation proposée dans [16] :

Soit u et w deux voisins marqués de v, le marqueur de v est retiré si l'une des conditions suivantes est satisfaite.

- $N(v) \subseteq N(u) \cup N(w)$, $N(u) \not\subseteq N(v) \cup N(w)$ et $N(w) \not\subseteq N(u) \cup N(v)$
- $N(v) \subseteq N(u) \cup N(w)$, $N(w) \not\subseteq N(u) \cup N(v)$ et $\text{clé}(v) < \text{clé}(u)$
- $N(v) \subseteq N(u) \cup N(w)$, $\text{clé}(v) < \text{clé}(u)$ et $\text{clé}(v) < \text{clé}(w)$

En d'autres termes, la comparaison de clé n'a plus lieu que pour résoudre un conflit entre nœuds se couvrants mutuellement. Si l'un d'eux couvre moins de nœuds que les autres, il pourra retirer son marqueur, quelle que soit sa clé. Cette formulation permet de limiter l'utilisation

des clés aux réelles égalités, et produit de meilleurs résultats que la règle k [2].

Le coût de l'algorithme en termes de messages envoyés par nœud est, au plus, de trois. Les deux premiers messages servent à récupérer la connaissance des voisins à deux sauts, tandis que le troisième sera utilisé par un nœud modifiant son état (marqué ou non) pour en tenir ses voisins informés. Chaque nœud commence par envoyer un message *HELLO* afin de se faire connaître de son entourage. À la réception d'un tel message, un nœud en ajoute simplement la source à sa liste de voisins. Un second message, de type *NEIGHBORS* et contenant cette liste de voisins complétée, est ensuite transmis. Ainsi, chaque nœud apprendra ses voisins directs et les voisins de ses voisins, réunissant une connaissance de son voisinage à deux sauts. Le dernier message n'est envoyé que par les nœuds *non marqués*, au moment de leur passage dans cet état. Cela concerne donc les nœuds dont tous les voisins sont connectés entre eux, de même que ceux impactés par l'une des deux règles d'élagage.

3 PROPOSITION ET ÉVALUATION

Ce chapitre présente notre contribution et son évaluation dans un environnement de simulation réaliste. Nous y décrivons dans un premier temps les principes de notre heuristique distribuée, puis nous évaluons ses performances et les comparons à celles d'une 2-approximation centralisée. Le terme de distance, utilisé à de multiples reprises dans ce chapitre, est toujours exprimé en nombre de sauts (les liens du graphe ne sont pas valués). De même, dans ce type d'environnement, les capteurs communiquant par ondes radio, nous considérons qu'un message envoyé par un nœud du réseau est reçu (et donc interprétable) par tous ses voisins.

3.1 HEURISTIQUE

Cette partie expose le fonctionnement de notre heuristique. Dans un premier temps, nous y présentons brièvement une explication de la méthode du gradient[8] sur laquelle est basée une partie de notre solution distribuée. Puis, nous décrivons en détail notre proposition dans un second paragraphe.

3.1.1 *Algorithme du gradient*

Le gradient est l'un des algorithmes les plus utilisés pour le routage dans les réseaux de capteurs, du fait de sa simplicité et de sa capacité à créer rapidement un SPT enraciné sur un nœud donné. En effet, ce type de graphe (un arbre ou un graphe orienté acyclique (DAG) selon les variantes) permet d'optimiser les communications entre l'ensemble des nœuds du réseau et le puits ($1 \rightarrow n$), facilitant ainsi la collecte des données dans un mode de communication appelé multi-sauts (le routage stricto sensu n'est pas sollicité dans un mode uni-saut).

Cette technique fonctionne par propagation d'un message de type *GRADIENT(source, rang)*, où la *source* représente l'identifiant de l'émet-

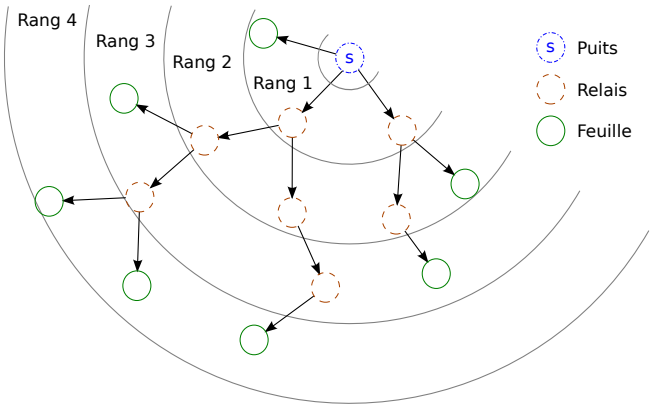


FIG. 3: Algorithme du gradient

teur du message et le *rang* sa distance au puits. Chaque nœud maintient en outre deux variables : son propre rang et l'identifiant du nœud le lui ayant fourni, que l'on appellera son *père*, toutes deux initialement vides.

L'algorithme est initialisé par le puits via l'envoi d'un premier message *GRADIENT(*id_{puits}, 0*)*. À la réception d'un message *GRADIENT(*s, r_s*)*, un nœud n incrémente la valeur de r_s, puis :

- Si n n'avait pas encore de rang r_n ou $r_n > r_s$, r_n devient r_s et s le père de n. Il transmet ensuite le message *GRADIENT(*n, r_n*)* à ses voisins.
- Dans le cas contraire, le message est ignoré.

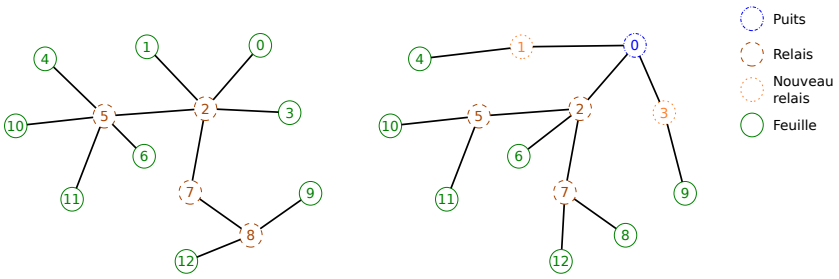
Il est important de noter que l'identifiant du *père* peut également être inclus dans le message : chaque nœud peut savoir s'il possède des fils et connaître, le cas échéant, leurs identifiants. Un nœud sera considéré comme *feuille* s'il n'a aucun fils et comme *relais* dans le cas contraire. Cette fonctionnalité supplémentaire se révèle très utile dans le cadre de notre contribution développée dans la suite. Le processus de propagation du gradient est illustré par la figure 3.

3.1.2 *Contribution*

Dans un réseau de capteurs sans puits (où tous les nœuds sont équivalents en terme de communication), tous les nœuds peuvent être amenés à communiquer avec tous les autres : communication de type $n \to n$. Afin d'optimiser, en termes d'efficacité et de consommation énergétique, ce type de communication par diffusion, il est donc capital de réduire le nombre de retransmissions nécessaires à de tels échanges. Il a été démontré qu'un MCDS était parmi les structures les plus adaptées pour répondre à ce problème.

Cependant, dans le cadre d'un déploiement réel, de nouveaux nœuds peuvent sporadiquement rejoindre le réseau (ou, plus généralement, un nœud peut vouloir changer d'état et devenir un puits). Nous envisageons ici le cas où ce nouveau nœud est un puits et proposons une solution adaptant dynamiquement la structure du réseau aux communications $1 \to n$. Celle-ci permet d'obtenir une structure de graphe hybride, se situant entre l'approximation de MLST originale et un SPT enraciné au puits. Cette structure est donc plus adaptée aux communications $1 \to n$ tout en tirant partie des avantages énergétiques et d'efficacité du modèle $n \to n$. Le compromis entre ces deux types de communications est alors modélisé grâce à un paramètre permettant d'explorer les différentes solutions possibles du modèle le plus $1 \to n$ au modèle le plus $n \to n$. Nous introduisons à cet effet une variable λ (définie dans $[1, \infty[$) permettant de faire tendre le résultat d'un côté ou de l'autre du spectre. Une valeur de λ proche de 1 favorisera les modifications du graphe Wu-Li vers un SPT, alors que les valeurs plus élevées tendront à conserver l'approximation de MLST.

En pratique, nous commençons donc par appliquer l'algorithme de *Wu-Li hybride*, décrit dans la section 2.2.2, afin de construire un CDS de taille réduite. Cette structure sera utilisée pour acheminer les paquets dans le réseau tant qu'aucun puits n'est déclaré.



(a) Graphe obtenu par l'algorithme de Wu et Li

(b) Résultat après application de notre méthode

FIG. 4: Illustration de notre proposition avec $\lambda = 30\%$ (exemple 1)

Lors de son entrée/activation dans le réseau, le puits déclenche un gradient afin d'informer les autres nœuds de sa présence. Le message est ensuite propagé par tous les nœuds, relais comme feuilles. Leur état (relais/feuille) est intégré au message pour être pris en compte pendant la propagation. Plutôt que de considérer uniquement le meilleur rang reçu, un nœud n maintient deux variables supplémentaires :

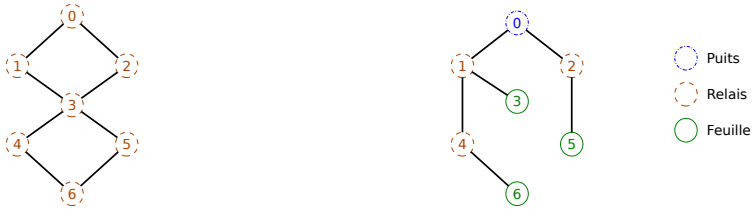
- d_r contenant le meilleur rang obtenu d'un relais r ;
- et d_f celui reçu d'une feuille f.

Si $d_r \leqslant d_f$, n choisit r comme père et transfère le message. Cependant, dans le cas où f offre un meilleur rang, il nous faut déterminer si le gain de distance est *suffisant* pour compenser la *perte* d'une feuille. En effet, la paternité d'une feuille implique sa transformation en relais, et donc une augmentation de la taille du CDS. Le nœud n choisira alors f comme père si et seulement si $\frac{d_r}{d_f} > \lambda$. Ainsi, plus la valeur de λ est élevée et plus le gain de distance, $\frac{d_r}{d_f}$, doit être important pour qu'une feuille devienne relais.

L'identifiant du père étant intégré aux messages du gradient, une feuille observant son propre identifiant dans un message saura qu'elle doit alors devenir relais. Au contraire, un relais n'observant aucun message contenant son identifiant pourra devenir feuille. Cette seconde propriété est très intéressante dans le contexte des communications $n \to n$: elle permet de compenser la perte de feuille induite par des faibles valeurs de λ et de diminuer sensiblement la taille du CDS pour des valeurs de λ plus importantes. En pratique, la partie évaluation démontre expérimentalement que cette propriété fait plus que de compenser la perte de feuille, elle permet d'améliorer le ratio feuilles/relais dans tous les cas de figures ($\forall \lambda \in [1, \infty[$).

Exemple 1

La figure 4 illustre le principe général de notre contribution. Le premier graphe, 4a, représente l'approximation d'un MLST obtenue avec l'algorithme de Wu-Li et constitue le point de départ de notre proposition. Les nœuds d'identifiants 2, 5, 7 et 8 sont des relais, tandis que les autres sont des nœuds feuilles. Aucun puits n'est pris en compte, tous les nœuds sont considérés ici comme des nœuds ordinaires dans un système égalitaire.



(a) Graphe obtenu par l'algorithme de Wu et Li

(b) Résultat après application de notre méthode

FIG. 5: Illustration de la suppression de relais superflus (exemple 2)

Un gradient est ensuite initialisé à partir du nœud d'identifiant 0 : il s'agit de notre puits sur cet exemple. Avec la propagation du gradient le graphe sera alors "réorienté", chaque nœud se connectant au relais de meilleur (plus faible) rang à sa portée. En conséquence, les nœuds 1 et 3 seront reliés directement au puits, le nœud 6 au relais 2 et 12 à 7. Dans cet exemple est utilisée une valeur de λ égale à 1,3. Un nœud pourra donc requérir la transformation en relais de l'un de ses voisins feuilles si le meilleur rang obtenu d'un relais est d'au moins 30% supérieur à celui annoncé par cette feuille. Cette condition est satisfaite par le nœud 4, dont le meilleur rang fourni par un relais, d_r, est égal à 3 pour un d_f à 2. Le nœud 1, à l'origine de d_f, deviendra donc relais à la demande de 4. Il en va de même pour le nœud 3, celui-ci procurant à 9 un rang d_f de 2 pour un d_r à 4.

Au final (figure 4b), deux feuilles (1 et 3) sont perdues mais la distance au puits des nœuds 4 et 9 est réduite de respectivement 33% et 50%.

Exemple 2

En raison de sa prise de décision localisée et de sa vision locale du graphe, l'algorithme de Wu-Li ne permet pas de prévenir la formation de cycles. En effet, un nœud n'a connaissance que des liens entre ses voisins directs et ne peut donc prendre en compte les connexions établies au delà. Deux ou plusieurs "branches du graphe" pourront alors être reliées en plusieurs points, créant ainsi des cycles. De par son mécanisme de propagation, l'application d'un gradient permet d'élaguer le graphe généré par l'algorithme de Wu-Li et ainsi supprimer les cycles pour favoriser l'émergence d'un arbre contenu dans le graphe issu de Wu-Li. Lors de ce processus, les relais superflus sont alors mis en évidence et changent d'état pour devenir des feuilles.

Sur la figure 5a les nœuds d'identifiants 3 et 6 ne sont pas à portée de 0. Ne sachant donc pas que ce dernier relie déjà les branches formées d'un côté par 1 et 4, et de l'autre par 2 et 5, ils restent relais. Après l'exécution d'un gradient centré sur 0 (Fig. 5b), les nœuds 3, 5 et 6 constatent cependant leur absence de fils (aucun message contenant leur identifiant dans le champ père) et deviennent alors des feuilles. Ce simple mécanisme est extrêmement efficace comme nous allons le mettre en évidence dans la partie suivante.

Environnement	
Surface	500×500
Répartition des nœuds	Uniforme (PPP)
Couche radio	
Antenne	Omnidirectionnelle
Portée (disque)	40
Couche MAC	
Protocole	IEEE 802.15.4

TAB. 3: Paramètres de l'environnement de simulation WSNet

3.2 ÉVALUATION

Cette section traite des expérimentations par simulation menées afin dévaluer notre proposition. Nous commençons par décrire l'environnement et les outils utilisés pour l'évaluation, puis nous exposons et analysons les résultats obtenus. Comme mentionné précédemment, notre référence pour estimer le résultat optimal est l'approximation centralisée de Solis-Oba dont le ratio d'approximation est de 2.

3.2.1 *Environnement d'évaluation*

Nous avons utilisé pour nos expérimentations le simulateur WSNet[1], en version 9.07. Ce logiciel, développé par l'Institut National de Recherche en Informatique et en Automatique (INRIA), permet de simuler le comportement d'un réseau de capteurs, avec ses échanges de messages, sa consommation énergétique et sa mobilité. Grâce à une API en C et un fichier de configuration XML, il est possible d'implémenter et de paramétrer chacune des couches du réseau. A chaque niveau, WSNet offre aussi la possibilité d'utiliser des protocoles existants, tels que les standards IEEE 802.11 ou 802.15.4, ou encore B-MAC, déjà préimplémentés. De plus, WSNet intègre un module pour la génération de topologies aléatoires selon différents schémas de distribution.

Nos simulations ont été réalisées avec les paramètres reportés dans le tableau 3. Nous disposons d'une quantité variable de nœuds répartis uniformément selon un processus de "point de Poisson" (PPP), sur une surface carrée de 500 unités de côté (les distances sont ici normalisées car seules les portées radio nous préoccupent). Chaque nœud possède une antenne omnidirectionnelle lui permettant d'échanger (envoyer ou recevoir) des messages avec tous les autres nœuds situés dans un rayon de 40 unités autour de lui, et uniquement ceux-ci. Deux nœuds sont donc *à portée* l'un de l'autre si la distance les séparant est inférieure à 40. Nous nous assurons alors que le graphe ainsi obtenu est connexe. L'accès au médium de communication est contrôlé grâce au standard *IEEE 802.15.4*, basé sur la méthode *CSMA/CA*. À l'exception de la méthode de 2-approximation centralisée de Solis-Oba que nous avons implémentée dans un logiciel tierce, les solutions distribuées évaluées ici sont implémentées au niveau la couche de routage du simulateur.

1 http://wsnet.gforge.inria.fr/

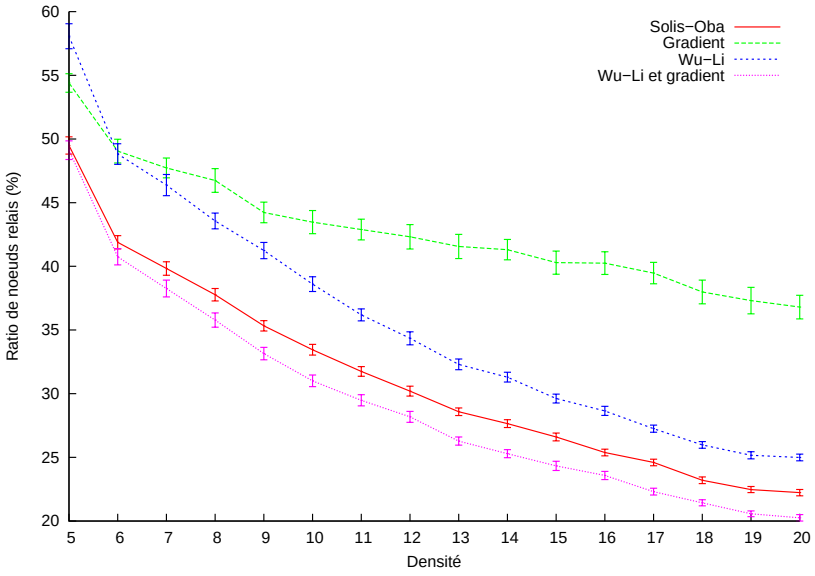


FIG. 6: Proportion de nœuds relais obtenue avec les différents algorithmes

L'algorithme de Solis-Oba, servant de référence par rapport à l'optimal pour nos expérimentations, a été implémenté en langage C. Ce logiciel prend en entrée le graphe connexe obtenu avec WSNet et renvoie en sortie une 2-approximation d'un MLST de ce graphe.

3.2.2 *Résultats*

Dans cette section, nous exposons et analysons les résultats de nos expérimentations. Dans un premier temps, nous comparons les différentes heuristiques à notre 2-approximation de référence, puis nous évaluons l'influence du paramètre de compromis, λ, sur la taille du CDS et la distance moyenne des nœuds au puits. Ces expérimentations sont menées sur des topologies de différentes densités, ici au sens degré moyen plus un des nœuds dans le graphe initial (nombre de nœuds par disque).

Comparaison de la taille des CDSs obtenus avec les différents algorithmes (Fig. 6)

Nous commençons donc par comparer les performances de notre proposition aux algorithmes du gradient et de Wu-Li indépendamment, ainsi qu'à la 2-approximation centralisée de référence. L'objectif étant de minimiser le nombre de relais, nous avons dans un premier temps défini $\lambda = \infty$, de sorte qu'aucune feuille du graphe ne puisse devenir relais. Notre approche ainsi paramétrée, le gradient est utilisé afin d'élaguer le graphe obtenu avec Wu-Li tout en favorisant, dans une moindre mesure, l'optimalité des chemins vers le puits : en pratique, les nœuds feuilles choisissent un "meilleur relais" père et les cycles (contenant de futures feuilles potentielles) présents dans le graphe formé avec Wu-Li sont élagués afin de réduire le graphe, i.e. certains relais deviennent des feuilles.

Les résultats présentés sur la Fig. 6 représentent une moyenne sur 100 simulations, avec un niveau de confiance de 95%. Chaque passe de simulation est effectuée sur une topologie différente, mais, au sein d'une passe donnée, tous les protocoles sont évalués sur la base d'une topologie identique.

Dans un premier temps, il est intéressant de noter que l'algorithme du gradient seul produit des résultats relativement bons, compte tenu de son faible coût (tant en termes de calculs que de messages échangés) et du fait que son objectif initial n'est pas la minimisation d'un CDS mais la construction d'un SPT enraciné au puits : moins de 50% des nœuds sont dans l'état relais lorsque la densité devient supérieure à 6. Bien qu'efficace sur des topologies de faible densité, son comportement est néanmoins plus aléatoire et la taille du CDS décroit logiquement moins vite qu'avec les autres heuristiques.

La méthode de Wu-Li produit des CDSs de plus grande taille à faible densité mais la proportion de relais diminue rapidement lorsque la densité augmente. Nous observons que les résultats du Wu-Li que nous avons retenu pour cette analyse se rapprochent de plus en plus de ceux obtenus avec l'approximation centralisée (sans toutefois l'atteindre pour les densités évaluées).

Les résultats obtenus grâce à notre approche hybride surpassent de manière significative ceux des deux heuristiques utilisées indépendamment. De plus, notre approche produit de meilleurs résultats que la 2-approximation centralisée confirmant ainsi l'efficacité de notre méthode. En effet, déjà légèrement inférieure pour une densité de 5, la taille du CDS est d'environ 8% inférieure à celle de l'algorithme de Solis-Oba avec des densités supérieures à 10. Expérimentalement, nous avons donc montré que notre proposition génère, dans les cas étudiés, des résultats meilleurs qu'une 2-approximation d'un MLST.

Évaluation de l'influence de λ sur la taille du CDS (Fig. 7)

Comme nous l'avons constaté précédemment, notre proposition produit dans des environnements réalistes de meilleurs résultats que la 2-approximation centralisée de référence. Cette analyse n'est cependant pas suffisante selon le critère d'optimalité des chemins que nous nous étions fixé : notre approche paramétrée avec $\lambda = \infty$ ne remplit qu'un seul de nos deux objectifs initiaux, celui de l'approximation du MCDS. Il a en effet fallu, pour obtenir de tels résultats, renoncer à transformer certaines feuilles en relais. La distance entre le puits et les autres nœuds n'est réduite que par la connexion de chaque nœud à son relais de rang minimum.

Dans cette seconde expérience, nous allons donc faire varier la valeur de λ afin d'évaluer son influence sur le nombre de relais. En particulier, comprendre son impact sur le nombre de relais supplémentaire générés par rapport au gain d'optimalité des chemins (compromis #relais - routage $1 \rightarrow n$). D'après la formulation présentée dans la section 3.1.2, plus la valeur de λ augmente et plus le gain de distance doit être important pour qu'un changement d'état soit autorisé. Une valeur plus faible est donc plus *permissive* tandis qu'une valeur élevé sera *restrictive* : λ permet de jouer sur la relaxation de la contrainte d'optimalité des

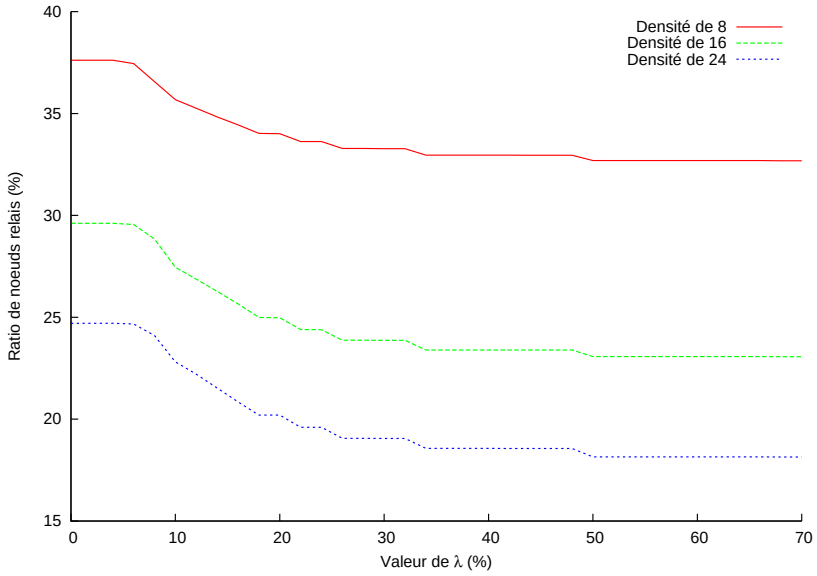


FIG. 7: Influence de λ sur la proportion de nœuds relais

chemins. Afin de faciliter la lecture des courbes, nous avons choisi de nous intéresser à trois densités particulières : 8, 16 et 24, lesquelles correspondent respectivement à des topologies de 500, 1000 et 1500 nœuds.

Il est important de noter que pour $\lambda = 1$ (augmentation de $x = 0$ % sur la figure 7), les résultats que nous obtenons sont meilleurs que ceux du Wu-Li seul. Cela signifie que le nombre de nouveaux relais générés pour améliorer l'optimalité des chemins vers le puits est compensé par le gain des relais supprimé grâce à notre élagage via le SPT du gradient. En effet, notre méthode hybride avec $\lambda = 1$ n'est pas équivalente à un gradient seul, car les nœuds font potentiellement le choix du père couvrant le plus de feuilles (grâce au Wu-Li appliqué précédemment) plutôt que de choisir le "premier" meilleur père découvert (comme dans un gradient classique). Ainsi, notre Wu-Li initial permet de définir une priorité sur le choix du père approprié à la construction d'un CDS minimisé. Ce résultat est très prometteur car il garantit que notre solution est systématiquement gagnante sur les deux tableaux par rapport à un Wu-Li seul : (i) moins de relais et (ii) de meilleurs chemins vers un puits donné.

La figure 7 montre que la proportion de relais, stable au début, commence à décroître rapidement lorsque $(\lambda - 1) \times 100$ atteint 5%, pour finalement se stabiliser vers 50% aux valeurs observées sur la figure précédente. Deux seuils peuvent donc être définis, le premier pour $\lambda < 5\%$ et le second pour $\lambda > 50\%$. La différence entre eux, en terme de taille de CDS, augmente avec la densité, atteignant 5% des nœuds pour une densité de 24.

Évaluation de l'influence de λ sur la distance au puits (Fig. 8)

Après avoir quantifié la perte de feuilles occasionnée par notre proposition pour $\lambda \to 1$ (par rapport à $\lambda = \infty$), il convient maintenant de s'intéresser au gain en terme de distance des chemins vers le puits. La figure 8 représente le nombre de sauts moyen, séparant le puits d'un

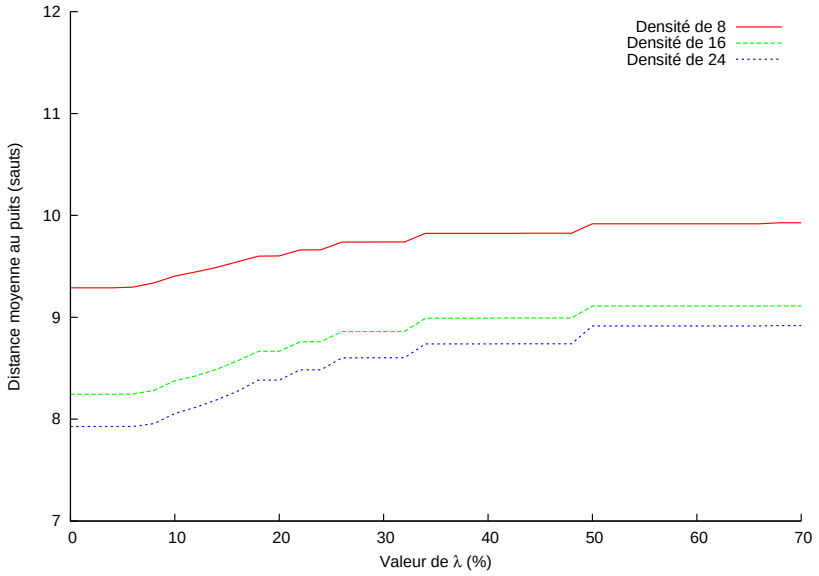


FIG. 8: Influence de λ sur la distance au puits

nœud du réseau, en fonction de λ. Les deux seuils observés dans la figure 7 sont aussi visibles à ce niveau. Nous constatons ainsi qu'une valeur de λ relaxée, sur le premier palier, permet de réduire la distance moyenne au puits de 10 à 12%, ce gain ne dépendant que peu de la densité du réseau.

Notre proposition permet donc de réduire la distance au puits de plus de 10% en utilisant jusqu'à 5% de relais supplémentaires par rapport à $\lambda = \infty$. Le CDS construit est toujours plus petit que l'original, produit par l'algorithme de Wu-Li, tout en étant adapté aux communications $1 \rightarrow n$ depuis et vers le puits.

Analyse générale

Les résultats obtenus montrent que notre solution produit, dans notre environnement d'évaluation réaliste, un CDS de taille inférieure à notre 2-approximation de référence. Cette solution centralisée ayant au moins la moitié du nombre de feuilles d'un MLST, nous pouvons en déduire qu'il en est de même pour notre heuristique. Par dualité feuille-relais, cela met en évidence l'efficacité de notre approche : celle-ci permet une importante réduction du nombre de relais dans le réseau par rapport aux solutions existantes. Ainsi, notre approche permet à la fois d'économiser de l'énergie, augmentant ainsi sa durée de vie, et d'améliorer les communications par diffusion de type $n \rightarrow n$. De plus, lorsque le réseau est configuré dans un mode $1 \rightarrow n$ (un nœud particulier, appelé le puits, est la racine des communications), notre approche permet tout à la fois de réduire le nombre de relais et la distance séparant les nœuds de la racine. Cette réduction du nombre de sauts induit aussi une limitation du nombre de retransmissions nécessaires aux transmissions $1 \rightarrow n$ et des échanges de messages plus efficaces pour un coût énergétique réduit. Il est possible de paramétrer cette distance au puits au prix d'une légère augmentation du nombre de relais (par rapport à notre solution la plus conservatrice en terme de minimisation

de relais). Cette réduction supplémentaire, déterminée par la variable λ, peut représenter jusqu'à 10% de la distance initiale. Deux "asymptotes horizontales" observées via nos simulations en environnement réaliste nous permettent de définir un intervalle, de 5 à 50% sur lequel l'impact de λ est significatif.

4 CONCLUSION

Dans ce mémoire, nous avons présenté un sous ensemble des solutions existantes pour la construction d'un MLST, tant de manière centralisée que distribuée. Notre attention s'est naturellement et tout particulièrement portée sur l'un des algorithmes distribués les plus pertinents — celui dont la complexité nous a semblé satisfaire le plus efficacement les contraintes liées aux réseaux de capteurs sans fils. La proposition introduite dans ce mémoire consiste en la combinaison d'une variante de cette méthode distribuée avec un autre algorithme bien connu, la technique de routage par gradient ayant pour but de construire un SPT enraciné au puits. Notre contribution a pour objectif de créer une structure hybride, située entre un MLST et un SPT, adaptée à la fois aux communications $1 \rightarrow n$ et $n \rightarrow n$. Nous avons également choisi, parmi les algorithmes centralisés présentés, celui présentant le meilleur ratio d'approximation, afin de s'en servir comme référence pour l'évaluation de notre proposition.

Les résultats de nos évaluations par simulation ont dépassé nos attentes. En effet, notre solution produit un graphe plus proche d'un MLST que l'algorithme centralisé de référence. De plus, la variable d'ajustement λ, que nous avons introduite pour définir le compromis structurel entre le MLST et SPT, permet de réduire sensiblement la distance entre les nœuds et le puits en utilisant un minimum de relais supplémentaires.

Perspectives

Les présents travaux pourront bien évidemment être sujets à des développements ultérieurs. Nous envisageons notamment d'évaluer les performances de notre proposition sur des topologies non-uniformes, présentant des irrégularités telles que des trous (zones de faible densité) ou des amas (zones de forte densité). De même, il sera intéressant de nous confronter à un déploiement réaliste dans un environnement tel que la plateforme d'expérimentation SensLAB[1] dont nous disposons à Strasbourg.

Nous prévoyons également d'apporter certaines améliorations à notre heuristique distribuée. Pour cela, nous étudierons dans un premier temps l'impact stratégique du choix du puits sur le nombre de feuilles *gagnées* et la qualité du plan de routage. En effet, un puits situé au centre de la topologie ou ayant un degré élevé pourrait engendrer de meilleurs résultats.

Dans un second temps, nous pourrions échanger l'ordre d'application des algorithmes. Nous considérerions alors les nœuds relais du gradient comme marqués et leur appliquerions les règles d'élagage de Wu et Li.

1 http://www.senslab.info/

Pour finir, nous envisageons également l'étude de la convergence de nos solutions dans un environnement dynamique (suite à des changements topologiques anticipés ou non) : comment réduire l'impact négatif de ces périodes transitoires sur le plan de commutation ?

BIBLIOGRAPHIE

[1] Cédric Adjih, Philippe Jacquet, and Laurent Viennot. Computing connected dominated sets with multipoint relays. Research Report RR-4597, INRIA, 2002. (Cité à la page 7 et 11.)

[2] F. Dai and J. Wu. An extended localized algorithm for connected dominating set formation in ad hoc wireless networks. *Parallel and Distributed Systems, IEEE Transactions on*, 15(10) :908–920, 2004. ISSN 1045-9219. (Cité à la page 7, 12, et 14.)

[3] Tetsuya Fujie. An exact algorithm for the maximum leaf spanning tree problem. *Computers & Operations Research*, 30(13) :1931 – 1944, 2003. ISSN 0305-0548. (Cité à la page 5.)

[4] Tetsuya Fujie. The maximum-leaf spanning tree problem : Formulations and facets. *Networks*, 43 :212–212, 2004. (Cité à la page 5.)

[5] K. Ruben Gabriel and Robert R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Biology*, 18(3) :259 –278, 1969. (Cité à la page 6 et 31.)

[6] G. Galbiati, F. Maffioli, and A. Morzenti. A short note on the approximability of the maximum leaves spanning tree problem. *Information Processing Letters*, 52(1) :45–49, October 1994. ISSN 0020-0190. (Cité à la page 10.)

[7] Garey and Johnson. *Computers and intractability : a guide to the theory of NP-completeness*. Freeman, New York, NY, 1979. ISBN 9780716710455. (Cité à la page 32.)

[8] Holger Karl and Andreas Willig. *Protocols and Architectures for Wireless Sensor Networks*. Wiley-Interscience, September 2007. ISBN 9780470519233. (Cité à la page 15.)

[9] Xiang-Yang Li, Yu Wang, and Wen-Zhan Song. Applications of k-Local MST for topology control and broadcasting in wireless ad hoc networks. *IEEE Transactions on Parallel and Distributed Systems*, 15 :1057–1069, 2004. ISSN 1045-9219. (Cité à la page 6.)

[10] Hsueh-I Lu and R. Ravi. Approximating maximum leaf spanning trees in almost linear time. *Journal of Algorithms*, 29(1) :132–141, October 1998. ISSN 0196-6774. (Cité à la page 5 et 6.)

[11] Stephane Rovedakis. Construction auto-stabilisante d'un arbre couvrant maximisant le nombre de feuilles. In Maria Gradinariu Potop-Butucaru et Hervé Rivano, editor, *12èmes Rencontres Francophones sur les Aspects Algorithmiques de Télécommunications (AlgoTel)*, Belle Dune France, 2010. (Cité à la page 6 et 11.)

[12] Roberto Solis-Oba. 2-Approximation algorithm for finding a spanning tree with maximum number of leaves. In Gianfranco Bilardi, Giuseppe Italiano, Andrea Pietracaprina, and Geppino Pucci, editors, *Algorithms — ESA' 98*, volume 1461 of *Lecture Notes in Computer Science*, pages 1–1. Springer Berlin / Heidelberg, 1998. 10.1007/3-540-68530-8_37. (Cité à la page 5, 6, et 10.)

[13] I. Stojmenovic, M. Seddigh, and J. Zunic. Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks. *Parallel and Distributed Systems, IEEE Transactions on*, 13 (1) :14 –25, January 2002. ISSN 1045-9219. (Cité à la page 11 et 12.)

[14] Godfried T. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12(4) :261–268, 1980. ISSN 0031-3203. (Cité à la page 6 et 31.)

[15] Jie Wu and Hailan Li. A Dominating-Set-Based routing scheme in ad hoc wireless networks. *Telecommunication Systems*, 18 :13–36, 2001. ISSN 1018-4864. 10.1023/A :1016783217662. (Cité à la page 7, 11, et 12.)

[16] Jie Wu, Fei Dai, Ming Gao, and Ivan Stojmenovic. On calculating power-aware connected dominating sets for efficient routing in ad hoc wireless networks. *Journal of communications and networks*, 4 (1) :59–70, March 2002. (Cité à la page 12 et 13.)

[17] Andrew Chi-Chih Yao. On constructing minimum spanning trees in k-Dimensional spaces and related problems. *SIAM Journal on Computing*, 11(4) :721, 1982. ISSN 00975397. (Cité à la page 6 et 31.)

[18] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. Wireless sensor network survey. *Computer Networks*, 52(12) :2292 – 2330, 2008. ISSN 1389-1286. (Cité à la page 1.)

ACRONYMES

CDS	Connected Dominating Set. 7, 11–13, 16, 17, 20–23, *Glossaire :* ensemble dominant
DAG	Directed Acyclic Graph. 15, *Glossaire :* graphe oriente acyclique
INRIA	Institut National de Recherche en Informatique et en Automatique (`http://www.inria.fr/`). 19
$LMST_k$	*k*-Localized Minimum Spanning Tree. 6, *Glossaire :* arbre couvrant de poids minimal
MCDS	Minimum Connected Dominating Set. 2, 7, 11, 16, 21
MLST	Maximum Leaf Spanning Tree. 2, 5–7, 10, 11, 16, 17, 20, 21, 23, 25
MPR	Multipoint Relay. 7
MST	Minimum Spanning Tree. 6, 7
PPP	Poisson Point Process. 19
PTAS	Polynomial Time Approximation Scheme. 10, 31, 32, *Glossaire :* schema d'approximation
RNG	Relative Neighboring Graph. 6, *Glossaire :* graphe de voisinage relatif
SPT	Shortest Path Tree. 2, 15, 16, 22, 25, *Glossaire :* arbre des plus courts chemins

GLOSSAIRE

P = NP	Le problème de P = NP est l'un des problèmes non-résolus majeurs en informatique théorique. Informellement, il s'agit de savoir si, étant donné un problème dont la solution peut être **vérifiée** *rapidement* (en temps polynomial), cette solution peut aussi être **calculée** *rapidement*. http://www.claymath.org/millennium/P_vs_NP/. 2, 31
Arbre couvrant de poids minimal	Dans un graphe pondéré, un arbre couvrant de poids minimal est un sous-graphe connexe et acyclique, tel que chaque sommet est connecté et que le poids total des arêtes est minimal. 27
Arbre des plus courts chemins	Un arbre des plus courts chemins (ou Shortest Path Tree) est un sous-graphe crée à partir d'un graphe connexe quelconque, tel que la distance, en nombre de sauts, entre la racine et tous les autres nœuds est minimale. 27
Ensemble dominant	Dans un graphe $G = (V, E)$, un ensemble dominant est un sous-ensemble $D \subseteq V$, tel que chaque sommet n'appartenant pas à D est voisin d'un sommet dans D. Les sommets de D sont appelés *dominants*. L'ensemble dominant est dit *connecté* si le sous-graphe induit par D est connexe. 27
Graphe de Gabriel	Le graphe de Gabriel[5] d'un ensemble S de points est le graphe dans lequel deux points distincts A et B de S sont connectés si et seulement si le disque dont le segment [AB] est diamètre ne contient aucun autre élément de S. 6, 31
Graphe de voisinage relatif	Le graphe de voisinage relatif[14] d'un ensemble S de points est le graphe dans lequel deux points distincts A et B de S sont connectés si et seulement si il n'existe aucun autre point C de S qui soit à la fois plus proche de A et de B. Le graphe de voisinage relatif est un sous-graphe du graphe de Gabriel. 27
Graphe de Yao	Le graphe de Yao[17] avec un paramètre entier $k \geqslant 6$ est un graphe orienté défini comme suit. De chaque sommet sont initiés k rayons uniformément espacés divisant le plan en k secteurs. Un arc est ensuite ajouté entre le sommet et son plus proche voisin dans chaque secteur. 6

Graphe oriente acyclique
: Un graphe orienté acyclique est un graphe orienté ne contenant aucun cycle. En d'autres termes, il n'existe aucune séquence non vide d'arcs permettant, à partir d'un sommet quelconque du graphe, de revenir à ce sommet. 27

MAX SNP-difficile
: Classe des problèmes pour lesquels il existe des approximations à ratio fixe s'exécutant en temps polynomial mais aucun PTAS, sauf si P = NP. 10

NP-difficile
: Informellement, un problème NP-difficile[7] est un problème que l'on ne peut pas résoudre en temps polynomial, sauf si P = NP. 2, 7

Séparation et évaluation
: La séparation et évaluation, en anglais *branch and bound*, est une technique communément utilisée pour la résolution des problèmes d'optimisation combinatoire. Sommairement, elle consiste à énumérer les solutions possibles pour un problème tout en les bornant de manière à éviter de larges sous-ensembles de mauvaises solutions. 5

Schema d'approximation
: Un schéma d'approximation pour un problème d'optimisation est un algorithme d'approximation qui prend en entrée non seulement une instance du problème, mais aussi une valeur $\epsilon > 0$ telle que, pour tout ϵ fixé à l'avance, le schéma est un algorithme d'approximation $(1 + \epsilon)$. On dit qu'un schéma d'approximation est un PTAS si, pour tout $\epsilon > 0$ fixé, le schéma s'exécute en temps polynomial par rapport à la taille n de l'entrée. 27