

République Algérienne Démocratique et Populaire
Ministère de l'enseignement Supérieur et de la Recherche Scientifique
Université A/Mira de Béjaïa
Faculté des Sciences Exactes
Département recherche opérationnelle



Mémoire De Fin de cycle

En vue d'obtention du diplôme de master en recherche opérationnelle spécialité : MMTD

Thème

Minimisation de l'énergie dans les réseaux de capteurs sans fil

Réalisé par :

- Melle OUALI Feyrouz
- Melle MEZIANE Sarah

Soutenu le 19/10/2020 devant le jury composé de :

<i>Président</i>	Mr TAOUINET Smail	MAA à U.Béjaia.
<i>Examinatrice</i>	M ^{eme} YOUNSI Leila	MAA à U.Béjaia.
<i>Encadreur</i>	Mr KABYL Kamal	MCB à U.Béjaia.

2019/2020

Remerciements

On remercie dieu le tout puissant de nous avoir donné la santé et la volonté d'entamer et de terminer ce mémoire.

Et on tient à remercier vivement **Mr KABYL Kamal** pour nous avoir honorés par son encadrement, pour sa gentillesse et patience, son orientation et précieux conseils.

Et on tient à exprimer notre reconnaissance à tous les enseignants du département de Recherche Opérationnelle qui nous ont tellement aidés durant notre cursus universitaire.

On remercie aussi les membres de jury d'avoir accepté d'évaluer notre travail.

Enfin, dans le souci de n'oublier personne. On tient à remercier toute personne ayant partagé ou aidé de près ou de loin à la réalisation de ce modeste travail.

Dédicaces

A ma chère maman et mon adorable papa.

A mes sœurs et mon frère.

A tous mes chers amis.

A ma familles sans exception.

A mes professeurs et tous ceux qui me sont chers.

Je dédie ce travail avec reconnaissance et amour.

Sarah.

Dédicaces

Je dédie ce modeste travail

A ma maman qui ma soutenu et encouragé durant ces années d'études,

Qu'elle trouve ici le témoignage de ma profonde reconnaissance.

A mon père pour son amour et ses sacrifices.

A mes adorables sœurs et frères, source d'espoir et de motivation.

A ma binôme et chère amie, à qui je souhaite bonne chance pour son
prochain projet.

A tous mes amis qui m'ont encouragé, et a qui je souhaite plus de succès,
tout particulièrement Hani, Sarah, Mounir, Souad, Kamyliya, Hamza,

Nassima, Sirina, Oumana, Imen, Cylia,...

A tous ceux que j'aime.

Feyrouz.

Table des matières

table des Figures	vi
Introduction générale	1
1 Description des réseaux de capteurs sans fil	2
1.1 Introduction	2
1.2 Motivations et Enjeux	2
1.2.1 Motivations	2
1.2.2 En jeux	3
1.3 Capteurs	3
1.3.1 Définition	3
1.3.2 Classification d'un RCSF	4
Capteurs Actifs	4
Capteurs passifs	4
1.4 Architecture d'un nœud de capteur	5
1.4.1 Noeud Source	6
1.4.2 Noeud puits	6
1.5 Généralité sur les réseaux de capteurs sans fil	6
1.5.1 Définition	7
1.5.2 Caractéristiques d'un RCSF	7
1.5.3 Architecture d'un réseau de capteurs sans fil	8
1.5.4 Domaine d'application des RCSF	9
1.5.5 Contraintes de conception des RCSF	11
1.6 Points clés pour la consommation des RCSF	13
1.7 Classification des réseaux de capteurs sans fil	13

1.7.1	Selon le mode de communication	13
1.8	Le routage dans le RCSF	15
1.8.1	Définition de routage	15
1.8.2	Classification des protocoles de routage pour les RCSF	15
a)	Protocoles de routage basés sur la structure du réseau	16
b)	Classification selon l'initiateur de communication	18
c)	Classification selon l'établissement de la route	18
1.9	Conclusion	19
2	Quelques rappels sur la théorie des graphes	20
2.1	Introduction	20
2.2	Concepts de base et notations	21
2.2.1	Définitions générales	21
2.2.2	Graphe orienté	21
2.2.3	Graphe non orienté	22
2.2.4	Sous-graphe engendré	22
2.2.5	Graphe partiel	23
2.2.6	Sous graphe partiel	23
2.2.7	Graphe valué	23
2.2.8	Extrémité initiale et terminale (successeur et prédécesseur)	23
2.2.9	Arcs adjacents, arêtes adjacentes	24
2.2.10	Graphe simple	24
2.3	Connexité dans les graphes	24
2.3.1	Boucle	24
2.3.2	Sommets adjacents	24
2.3.3	Les voisins d'un sommet	24
2.3.4	Chaîne	25
2.3.5	Chaîne simple	25
2.3.6	Chaîne élémentaire	25
2.3.7	Chemin	25
2.3.8	Chemin simple	26

2.3.9	Chemin élémentaire	26
2.3.10	Cycle	26
2.3.11	Circuit	27
2.3.12	Graphe connexe	27
2.3.13	Composante connexe	28
2.3.14	Graphe fortement connexe	28
2.3.15	Composante fortement connexe	28
2.3.16	Arbres et forêts	29
	Caractérisation des arbres	29
	Arbres couvrants d'un graphe	29
2.4	Quelques types de graphes	29
2.4.1	Graphe Complet	29
2.4.2	graphe planaire	30
2.4.3	Graphe biparti	30
2.5	Plus courts chemins	31
2.5.1	Problème du plus court chemin	31
	Définition	31
2.6	Algorithme de recherches de Plus courts chemins	32
2.6.1	Algorithme de Bellman	32
2.6.2	Algorithme de Ford	32
	Algorithme de Dijkstra	33
2.7	Conclusion	33
3	Protocole de routage basée sur l'algorithme de Dijkstra dans un RCSF	34
3.1	introduction	34
3.2	Modèle du réseau	34
3.3	Solution proposée	35
3.4	Hypothèses	35
3.5	Détail de la solution proposée	36
3.5.1	Objectif de notre modélisation	36
3.6	Application sur le C++	44

3.6.1	Présentation de langage C++	44
3.6.2	Application numérique	45
	Résultat	48
3.7	Conclusion	49
	Conclusion générale	50
	Bibliographie	51

Table des figures

1.1	schéma d'un capteur sans fil.	4
1.2	Architecture d'un capteur.	5
1.3	Nœuds capteurs dispersés dans un champ de capteur	7
1.4	Architecture d'un réseau de capteur sans fil	9
1.5	Quelques exemples d'illustrations d'applications des RCSF	11
1.6	les types de communication dans les réseaux de capteurs sans fil	14
1.7	Classification des protocoles de routage	16
1.8	Protocoles de routage à plat.	17
1.9	Topologie hiérarchique.	17
2.1	les sept ponts de Königsberg	20
2.2	Graphe orienté	21
2.3	Graphe non orienté	22
2.4	Sous graphe	22
2.5	Graphe simple	24
2.6	Les voisins d'un sommet	25
2.7	Graphe G possède une chaîne	25
2.8	Graphe G possède un chemin	26
2.9	Graphe ayant trois cycle	26
2.10	Circuit	27
2.11	Graphe connexe	27
2.12	Graphe à deux Composante fortement connexe	28
2.13	Graphe complet	30
2.14	Graphe planaire	30

2.15	Graphe biparti	31
3.1	Réseau associé après la modélisation	37
3.2	Réseau obtenu après l'initialisation	38
3.3	Réseau obtenu à la première itération	39
3.4	Réseau obtenu à la deuxième itération	40
3.5	Réseau obtenu à la troisième itération	40
3.6	Réseau obtenu à la quatrième itération	41
3.7	Réseau obtenu à la cinquième itération	42
3.8	Réseau obtenu à la sixième itération	43
3.9	Réseau obtenu à la septième itération	43
3.10	Résultat de plus court chemin	44
3.11	l'interface de code blocks	45
3.12	Introduire les données	46
3.13	Introduire les données	46
3.14	Introduire les données	47
3.15	Introduire les données	47
3.16	Résultat	48
3.17	Résultat	48
3.18	Résultat	49

Introduction générale

Durant ces dernières décennies, les réseaux de capteurs ont bouleversé le monde. Le besoin d'un suivi continu des phénomènes naturels et aussi la surveillance dans différents domaines, ont renforcé l'intérêt pour cette nouvelle ère de l'informatique embarquée. En revanche, les réseaux de capteurs souffrent de leurs fragilités et de leurs énergies limitées. Les noeuds capteurs sont alimentés par des batteries limités en énergie. Par ailleurs, le remplacement des batteries n'est pas une solution envisageable pour ces derniers, soit à cause du déploiement aléatoire des capteurs, ou à cause de l'hostilité de l'environnement où ils sont placés. Toutefois, la mort d'un ou plusieurs noeuds capteurs interrompt partiellement la communication dans le réseau. De ce fait, une partie des données collectées sera perdue, ce qui en résulte à la mort partielle du réseau[1].

Dans ce mémoire, nous intéressons à résoudre un problème réel en utilisant la théorie des graphes et plus particulièrement le problème de plus court chemin dans les réseaux,

L'objectif principal c'est l'utilisation de l'algorithme de Dijkstra pour trouver des techniques de routage efficaces en termes d'énergie afin que la durée de vie du réseau soit maximale.

Ce mémoire est organisé en trois chapitres comme suit :

* Le premier chapitre nous a permis d'avoir une vue globale sur les réseaux de capteurs sans fil, leurs architectures, leurs caractéristiques, leurs domaines d'application.

* Le deuxième chapitre est consacré à la présentation de quelques définitions de base sur la théorie des graphes et Algorithme de recherches de Plus courts chemins qui seront utilisés dans la suite de notre travail.

* Dans le chapitre 3 nous présentons la modélisation d'un réseau de capteur sans fil, on propose un protocole de routage basé sur l'algorithme dijkstra et la programmation de ce dernier sur le langage C++.

Chapitre 1

Description des réseaux de capteurs sans fil

1.1 Introduction

Les réseaux de capteurs sans fil (RCSF) sont des réseaux ad hoc généralement constitués d'entité autonome miniaturisés appelés nœuds capteurs pouvant communiquer par liaison radio. Les RCSF ont suscités beaucoup d'engouements dans la recherche scientifique en raison notamment des nouveaux problèmes de routage sous forte contrainte de durée de vie du réseau et de faible capacité des nœuds.[2]

1.2 Motivations et Enjeux

1.2.1 Motivations

Les avancées technologiques en matière de production de circuits électroniques, et dans une moindre mesure, les progrès dans le domaine des batteries pour le stockage de l'énergie électrique, ont permis le développement de capteurs autonomes, de petite taille et dont le Coût de fabrication est de plus en plus faible. Les capteurs sont des appareils capables de Récolter différentes sortes d'informations dans leur rayon d'action et de communiquer avec les autres capteurs. Leur faible coût permet ainsi de déployer un grand nombre de capteurs afin de surveiller de Vastes zones accidentées ou inaccessibles. Grâce au grand nombre de capteurs qui le constitue, Un réseau de capteurs sans fil résiste plus facilement aux attaques d'un ennemi ou aux Dysfonctionnements spontanés [3].

La technologie des RCSF est un domaine de recherche nouveau, par rapport à la technologie d'internet. Cette technologie des RCSF a bénéficié d'une position centrale dans l'espace de recherche des réseaux émergents futures ces dernières années. Il y'a de plus en plus de travaux de recherche intéressants sur plusieurs aspects des RCSF : énergie, localisation, synchronisation, mobilité, et changement de topologie, qualité de service, sécurité, traitement dans le réseau...etc.[3]

1.2.2 En jeux

Le routage permet l'acheminement des informations vers une destination donnée à travers un réseau de connexion. En effet, le rôle des techniques de routage consiste à déterminer un acheminement optimal des paquets à travers le réseau au sens d'un certain critère de performance comme la consommation énergétique. Le but est de trouver l'investissement de moindre coût qui assure le routage du trafic nominal et garantit la qualité de service. Le problème qui se pose dans le contexte des réseaux de capteurs est l'adaptation de la méthode d'acheminement utilisée avec le grand nombre de nœuds existant dans un environnement caractérisé par de changements de topologies, de modestes capacités de calcul, de sauvegarde, et d'énergie. Toute conception de protocole de routage implique l'étude des problèmes suivants :

1.3 Capteurs

1.3.1 Définition

Un capteur sans fil est un petit dispositif électronique, peu coûteux, doté de ressources limitées en énergie (batterie), en puissance de calcul et en capacité de stockage. Un capteur possède la capacité de mesurer une valeur physique environnementale (température, lumière, pression, etc.) et de la transmettre en utilisant les communications sans fil à un centre de contrôle appelé une station de base. Les nœuds capteurs peuvent être déployés dans toute application dont l'objectif est de surveiller l'environnement. [4] La figure 1.1 montre un schéma de capteur sans fil

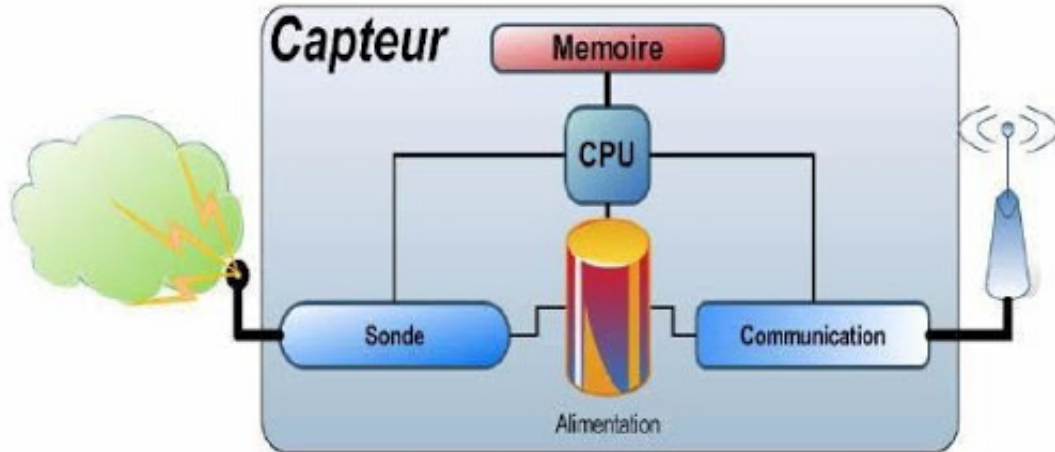


FIGURE 1.1 – schéma d'un capteur sans fil.[5]

1.3.2 Classification d'un RCSF

Capteurs Actifs

Généralement, un capteur actif est un système de mesure qui nécessite une source d'énergie embarquée, la plupart du temps assurée par une batterie, et ce pour la réalisation de la phase de traitement au cours de laquelle le signal est filtré (nettoyé), amplifié et converti dans un format compatible et exploitable. Dans ce cas, le capteur doit non seulement mesurer des propriétés physiques mais doit également effectuer des tâches additionnelles au travers de circuits de traitement et de communication intégrés. Ce type de capteur est surtout utilisé pour assurer des mesures continues en temps réel. [5]

Capteurs passifs

Les capteurs passifs sont des dispositifs qui ne possèdent pas de source d'énergie embarquée et présentent l'avantage d'être facilement intégrables. Ce type de capteur est utilisé dans des applications spécifiques (surveillance environnementale, des instruments de suivis spatial et aéronautique, des applications liées à la santé) qui nécessitent des unités de mesure miniatures, passives, de grande précision et fiables. L'objectif est d'assurer des mesures à distance des grandeurs physiques. Dans ce cas, deux différentes technologies peuvent être utilisées pour la transmission sans-fil de données : la transmission inductive et la transmission radio basée sur la réflexion (transpondeur passif).[5]

1.4 Architecture d'un nœud de capteur

Un capteur est composé principalement d'une unité de : captage, de traitement, de transmission, de stockage et d'énergie. Des composants additionnels peuvent être ajoutés selon le domaine d'application, comme par exemple un système de localisation tels qu'un GPS (Global Positioning-System), un générateur d'énergie (exemple : cellules solaire) ou un mobilisateur lui permettant de se déplacer [6] La figure 1.2 montre une architecture d'un noeud de capteur.

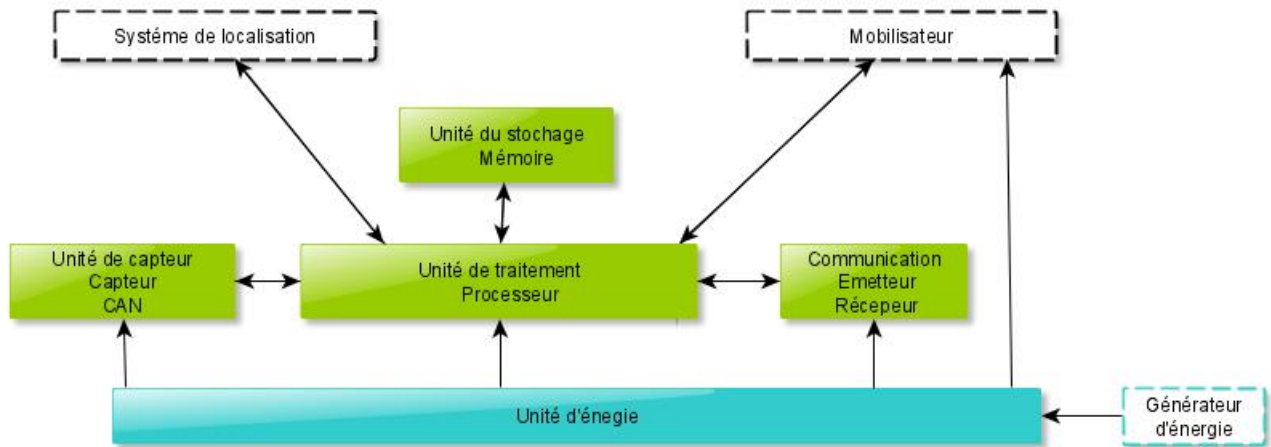


FIGURE 1.2 – Architecture d'un capteur.[6]

- **Unité d'énergie** Un capteur est muni d'une source d'énergie, généralement une batterie, pour alimenter tous ses composants. Les batteries utilisées sont soit rechargeables ou non. Souvent, dans les environnements sensibles, il est impossible de recharger ou changer une batterie. Pour cela, l'énergie est la ressource la plus précieuse puisqu'elle influe directement sur la durée de vie des capteurs et donc d'un réseau de capteurs .
- **Unité de captage** La fonction principale de l'unité de captage est de capturer ou mesurer les données physiques à partir de l'objet cible. Elle est composée de deux sous-unités : le récepteur (reconnaissant la grandeur physique à capter) et le transducteur (convertissant le signal du récepteur en signal électrique). Le capteur fournit des signaux analogiques, basés sur le phénomène observé, au convertisseur Analogique /Numérique (CAN). Ce dernier transforme ces signaux en données numériques et les transmet à l'unité de traitement.
- **Unité de traitement** L'unité de traitement exécute les procédures permettant au nœud de collaborer avec les autres nœuds pour effectuer la tâche assignée au réseau. Cette unité est également composée d'un processeur et d'un système d'exploitation spécifique. Elle

acquiert les informations en provenance de l'unité d'acquisition et les envoie à l'unité de transmission.

- **Unité de transmission** Cette unité est responsable de toutes les émissions et réceptions de données via un support de communication sans fil. Les différents choix de média de transmission incluent la Radiofréquence(RF), le Laser et l'Infrarouge.
- **Unité de stockage(Mémoire)** L'unité de stockage inclut la mémoire de programme (dont les instructions sont exécutées par le processeur) et la mémoire de donnée (pour conserver des données fournies par l'unité de captage et d'autres données locales). La taille de cette mémoire est souvent limitée essentiellement par les considérations économiques et s'améliorera aussi probablement au fil des années.

1.4.1 Noeud Source

son rôle principal est de détecter les phénomènes physiques ou physiologiques se produisant dans son environnement immédiat afin de les transmettre, directement ou via multiples sauts, à un utilisateur final. C'est en fait un noeud capteur [2].

1.4.2 Noeud puits

Est un noeud régulier doté d'un convertisseur série connecté à une seconde unité de communication (GPRS, Wi-Fi, WiMax, etc.). La seconde unité de communication fournit une retransmission transparente des données provenant de noeuds capteurs à un utilisateur final ou d'autres réseaux comme internet [2].

1.5 Généralité sur les réseaux de capteurs sans fil

Les réseaux de capteur sans fil constituent une classe de réseaux ad hoc composés de nœuds capteurs mobile et/ou statique pouvant être déployés dans des environnements connus ou inconnus. Ces capteurs disposent d'une capacité énergétique leurs permettant de fonctionner de manière autonome et intelligente, et de communiquer via des liaisons radio [7].

1.5.1 Définition

Un RCSF est un réseau composé de plusieurs nœuds de capteurs pouvant communiquer entre eux par une liaison sans fil. La distribution de ces nœuds dans un champ d'observation permet d'obtenir des informations concernant un phénomène mesuré à différentes positions et en temps réel. Ces informations sont ensuite transmises à une Station de Base (BS) par liaison directe en mono-saut, ou par l'intermédiaire des autres nœuds de capteur en multi-saut. La BS peut stocker localement les données des capteurs. Elle peut également rendre ces mesures disponibles en ligne à d'autres utilisateurs [8]. La figure 1.3 montre la dispersion des capteurs dans un champ.

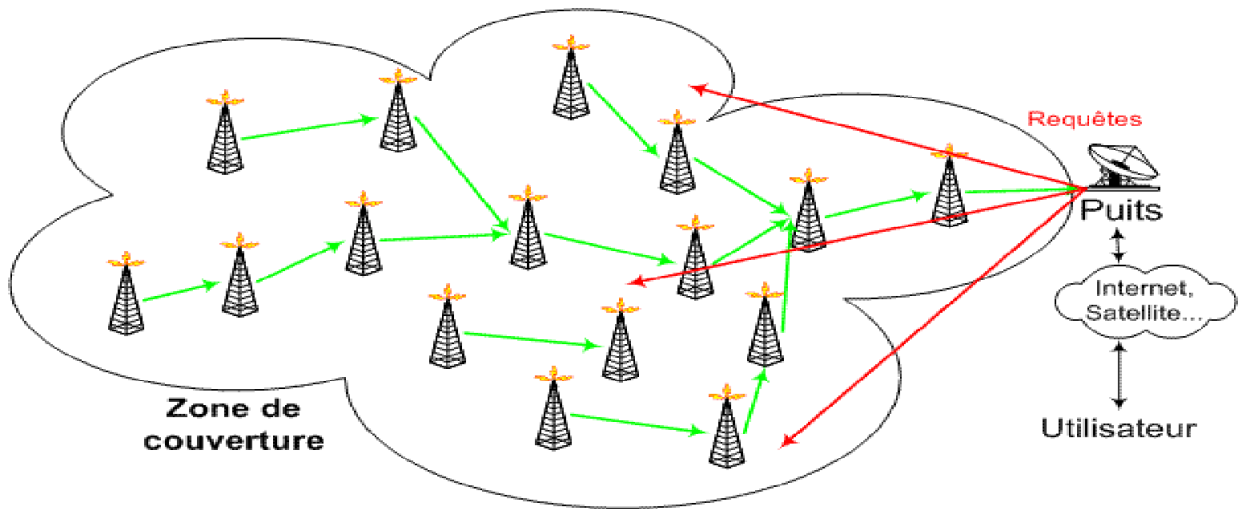


FIGURE 1.3 – Nœuds capteurs dispersés dans un champ [9]

1.5.2 Caractéristiques d'un RCSF

Un RCSF présente les caractéristiques suivantes :

- **Absence d'infrastructure** : Les réseaux ad-hoc en général, et les réseaux de capteurs en particulier se distinguent des autres réseaux par la particularité d'absence d'infrastructure préexistante et de tout genre d'administration centralisée.
- **Taille importante** : Un réseau de capteurs peut contenir des milliers de nœuds.
- **interférences** : Les liens radio ne sont pas isolés, deux transmissions simultanées sur une même fréquence, ou utilisant des fréquences proches, peuvent interférer.

- **Topologie dynamique** : Les capteurs peuvent être attachés à des objets mobiles qui se déplacent d'une façon libre et arbitraire rendant ainsi la topologie du réseau fréquemment changeante.
- **Sécurité physique limitée** : Les réseaux de capteurs sans fil sont plus touchés par le paramètre de sécurité que les réseaux filaires classiques. Cela se justifie par les contraintes et limitations physiques qui font que le contrôle des données transférées doit être minimisé.
- **Bande passante limitée** : Une des caractéristiques primordiales des réseaux basés sur la communication sans fil est l'utilisation d'un médium de communication partagé. Ce partage fait que la bande passante réservée à un nœud est limitée.
- **Contrainte d'énergie** : La caractéristique la plus importante dans les réseaux de capteurs est la ressource énergétique, car chaque capteur du réseau possède de faibles ressources en termes d'énergie (batterie). Afin de prolonger la durée de vie du réseau, une minimisation des dépenses énergétiques est exigée pour chaque nœud du réseau [10]

1.5.3 Architecture d'un réseau de capteurs sans fil

Un réseau de capteur sans fil (RCSF) présenté dans la Figure 1.4, est généralement constitué d'un ensemble de nœuds capteurs, variant de quelques dizaines à plusieurs milliers, déployés dans une zone géographique ou un environnement d'intérêt, communiquant entre eux par des ondes radio pour acheminer les données captées vers un nœud puits appelé station de base, qui peut être un ordinateur, reçoit les données des capteurs et les transmet à l'utilisateur par d'autres réseaux tels que Internet ou par satellite pour analyser ces données et prendre des décisions. Dans le cas le plus simple, les nœuds capteurs seront dans le voisinage direct de la station de base (communication à un saut). Cependant, dans le cas d'un réseau à grande échelle, ils ne sont pas tous dans le voisinage de la station de base et les données seront acheminées d'un nœud source vers la station de base en transitant par plusieurs nœuds, selon un mode de communication multi-sauts comme le montre la figure 1.4.

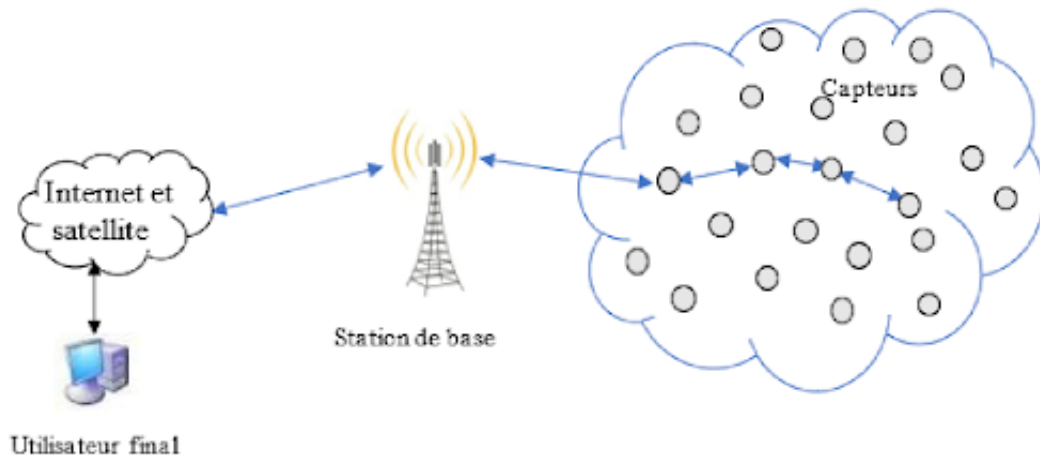


FIGURE 1.4 – Architecture d'un réseau de capteur sans fil [4]

En fonction de la taille et de la topologie du réseau, il pourrait y avoir une ou plusieurs stations de bases qui peuvent être fixes ou mobiles dans la zone du réseau. La station de base est habituellement supposée avoir une grande capacité de calcul et elle n'a pas de contraintes en termes d'énergie et de stockage. La tâche d'un nœud capteur comprend la création des paquets de données captées et la transmission sans fil de ces données à la station de base ou à d'autres nœuds capteurs. Lorsque le nœud capteur consomme toute son énergie (sa batterie est épuisée), il ne peut fournir aucun service. Si cela se produit, le nœud capteur est considéré comme un nœud mort et il sera retiré de la topologie du réseau. [4]

1.5.4 Domaine d'application des RCSF

Grace aux évolutions de la technologie touchant les domaines : électronique, informatique industrielle, instrumentation, réseaux et télécommunication, le champ d'applications des réseaux de capteurs sans fil est de plus en plus en élargissement. Parmi les applications des RCSF nous avons [11] :

- **Les Applications militaires** Comme la plupart des technologies, les applications militaires étaient les premières à intégrer les RCSF. Ils sont déployés dans un secteur stratégique ou difficile d'accès, pour y surveiller tous les mouvements (alliés ou ennemis), pour analyser le champ de bataille avant d'y envoyer du renfort, la détection des attaques biologiques ou chimiques, . . . etc, ainsi, beaucoup de projets dans le domaine des RCSF ont été lancés dans

le but d'aider des unités militaires et pour protéger des villes contre des attaques telles que les menaces terroristes, etc. Dans ce cadre, nous pouvons citer le projet DSN (Distributed Sensor Network) de la DARPA (Defense Advanced Research Projects Agency) qui a été l'un des premiers projets dans les années 80 ayant utilisé les réseaux de capteurs pour le rassemblement de données distribuées en vues d'une exploitation.

- **Les Applications médicales** Le suivi des personnes dépendantes est un véritable défi. Les RCSF propose un système de suivi pour les personnes dépendantes vivant seules à domicile ou dans un établissement 24 h /24. Ce système est basé sur la présence d'un réseau multi-capteurs déployé dans le cadre de vie de la personne surveillée couplé à un système d'identification sans fil et un algorithme d'apprentissage, les applications médicales basées sur les RCSF existent également aujourd'hui, citons la surveillance de la glycémie, la détection précoce de cancers.
- **Les Applications environnementales** Ces types de RCSF peuvent servir à l'observation d'un site susceptible de subir les effets d'une pollution, ou bien d'éventuel incendie, inondation, volcan ou tsunami. La construction en temps réel d'une cartographie grâce à des capteurs, disséminés sur le site capable de relever des informations sur la pollution, la température, le niveau d'eau et d'oxygène. . . , etc. Ces informations doivent être relayées vers les services spécialisés, le déploiement des capteurs dans une forêt peut aider à détecter un éventuel début de feu, et par la suite peut faciliter les moyens de lutte en temps réel.
- **Les Applications industrielles** Dans ce domaine, les réseaux de capteurs sans fil offrent une grande flexibilité d'emploi puisqu'ils permettent de s'affranchir des contraintes liées aux câblages. Il est alors possible de satisfaire des contraintes de poids, de mobilité, de facilité de déploiement, . . . etc .

Parmi les applications des réseaux de capteurs sans fil dans le milieu industriel, nous citons la surveillance de l'état de santé d'un ouvrier ou du risque de le voir exposé à des conditions de travail dangereuses (exposition à la radioactivité), la gestion des stocks, contrôle des machines à distances, etc.

la figure 1.5 montre Quelques exemples d'applications des RCSF



FIGURE 1.5 – Quelques exemples d’illustrations d’applications des RCSF

1.5.5 Contraintes de conception des RCSF

Les principaux facteurs et contraintes influençant l’architecture des réseaux de Capteurs peuvent être résumés comme suit [12] :

- **La tolérance de fautes** : Certain nœuds peuvent générer des erreurs ou ne Plus fonctionner à cause d’un manque d’énergie, un problème physique ou une interférence. Ces problèmes n’affectent pas le reste du réseau, c’est le principe de la tolérance de fautes. La tolérance de fautes est la capacité de maintenir les fonctionnalités du réseau sans interruptions dues à une erreur intervenue sur un ou plusieurs capteurs.
- **L’échelle** : Le nombre de nœuds déployés pour un projet peut atteindre le million. Un nombre aussi important de nœuds engendre beaucoup de transmissions inter nodales et nécessite que le puits ”sink ” soit équipé de beaucoup de mémoire pour stocker les informations reçues.
- **Les coûts de production** : Souvent, les réseaux de capteurs sont composés d’un très grand nombre de nœuds. Le prix d’un nœud est critique afin de pouvoir concurrencer un réseau de surveillance traditionnel. Actuellement un nœud ne coûte souvent pas beaucoup plus que 1 dollar. A titre de comparaison, un nœud Bluetooth, pourtant déjà connu pour être

un système low-cost, revient environ à 10 dollar .

- **L'environnement** : Les capteurs sont souvent déployés en masse dans des endroits tels que des champs de bataille au-delà des lignes ennemies, à l'intérieur de grandes machines, au fond d'un océan, dans des champs biologiquement ou chimiquement souillés, ... Par conséquent, ils doivent pouvoir fonctionner sans surveillance dans des régions géographiques éloignées.
- **La topologie de réseau** : Le déploiement d'un grand nombre de nœuds nécessite une maintenance de la topologie. Cette maintenance consiste en trois phases : Déploiement, Post-déploiement (les capteurs peuvent bouger, ne plus fonctionner,...), Redéploiement de nœuds additionnels.
- **Les contraintes matérielles** : La principale contrainte matérielle est la taille du capteur. Les autres contraintes sont que la consommation d'énergie doit être moindre pour que le réseau survive le plus longtemps possible, qu'il s'adapte aux différents environnements (fortes chaleurs, eau,..), qu'il soit autonome et très résistant vu qu'il est souvent déployé dans des environnements hostiles.
- **Les médias de transmission** : Dans un réseau de capteurs, les nœuds sont reliés par une architecture sans-fil. Pour permettre des opérations sur ces réseaux dans le monde entier, le média de transmission doit être normé. On utilise le plus souvent l'infrarouge (qui est license-free, robuste aux interférences, et peu onéreux), le bluetooth et les communications radioZigBee.
- **La consommation d'énergie** : Un capteur, de par sa taille, est limité en énergie ($< 1.2V$). Dans la plupart des cas le remplacement de la batterie est impossible. Ce qui veut dire que la durée de vie d'un capteur dépend grandement de la durée de vie de la batterie. Dans un réseau de capteurs (multi-sauts) chaque nœud collecte des données et envoie/transmet des valeurs. Le dysfonctionnement de quelques nœuds nécessite un changement de la topologie du réseau et un re-routage des paquets. Toutes ces opérations sont gourmandes en énergie, c'est pour cette raison que les recherches actuelles se concentrent principalement sur les moyens de réduire cette consommation.

1.6 Points clés pour la consommation des RCSF

Comme nous venons de le mentionner, la source d'alimentation des noeuds des RCSF présente un handicap devant l'évolution de ces systèmes. A l'épuisement de la source d'alimentation, le noeud se déconnecte du réseau. En plus, dans le cas de communication multi-saut, la déconnexion d'un noeud engendre la déconnexion de tous les noeuds qui communiquent grâce à lui. Pour ce, nous avons pensé à étudier la gestion de la consommation dans ces systèmes afin de l'optimiser et par suite, augmenter la durée de vie du noeud et du réseau tout entier. Dans la littérature, il existe plusieurs méthodes qui permettent d'obtenir un système à faible consommation. Généralement elles sont classées en trois catégories suivant qu'elles touchent le matériel et/ou le logiciel. La première consiste à concevoir des composants spécifiques conçus pour consommer le minimum d'énergie : ceci est réalisé à base des techniques matérielles. Alors que la seconde méthode est basée sur des techniques logicielles qui consistent à optimiser le code afin de réduire sa consommation. Enfin la dernière méthode consiste à réaliser une synergie entre le logiciel et le matériel afin d'optimiser la consommation totale du système. Elle s'appuie sur des mécanismes matériels pour diminuer la consommation mais utilise aussi le logiciel pour réaliser une adaptation dynamique de la consommation en fonction de la charge courante du système.[13]

1.7 Classification des réseaux de capteurs sans fil

1.7.1 Selon le mode de communication

Le mode de communication utilisé dans le réseau de capteurs dépend du type d'application et des techniques utilisées pour faire acheminer l'information des capteurs à la station de base. On distingue dans cette classification trois types de communication : les réseaux de capteurs à un seul-saut (single-hop WSN), les réseaux de capteurs multi-sauts (multi-hop WSN) et les réseaux de capteurs hiérarchiques.

Dans les réseaux de capteurs à un seul-saut, les noeuds capteurs envoient les données collectées directement à la station de base sans passer par des noeuds intermédiaires. Dans ce type de réseau les noeuds envoient leurs données en utilisant une forte puissance comme illustré dans la figure 1.6 (a).

Dans les réseaux de capteurs multi-sauts, un noeud capteur envoie ses données à la station de

base par l'intermédiaire de ses noeuds voisins en utilisant une petite puissance de transmission, la figure 1.6 (b) illustre un exemple sur ce type de communication. Ce type de réseau est appliqué dans plusieurs domaines d'application, mais il reste difficile à mettre en oeuvre [Shio K. S et al, 2011].

Dans les réseaux de capteurs hiérarchisés, la zone d'observation est divisée en clusters. Un clusterhead est élu pour chaque cluster. Ce dernier s'occupe de récupérer les informations auprès des capteurs dans son cluster et de les transmettre directement à la station de base figure 1.6 (c) ou via un mode multi-saut entre les clusters head figure 1.6 (d) [1].

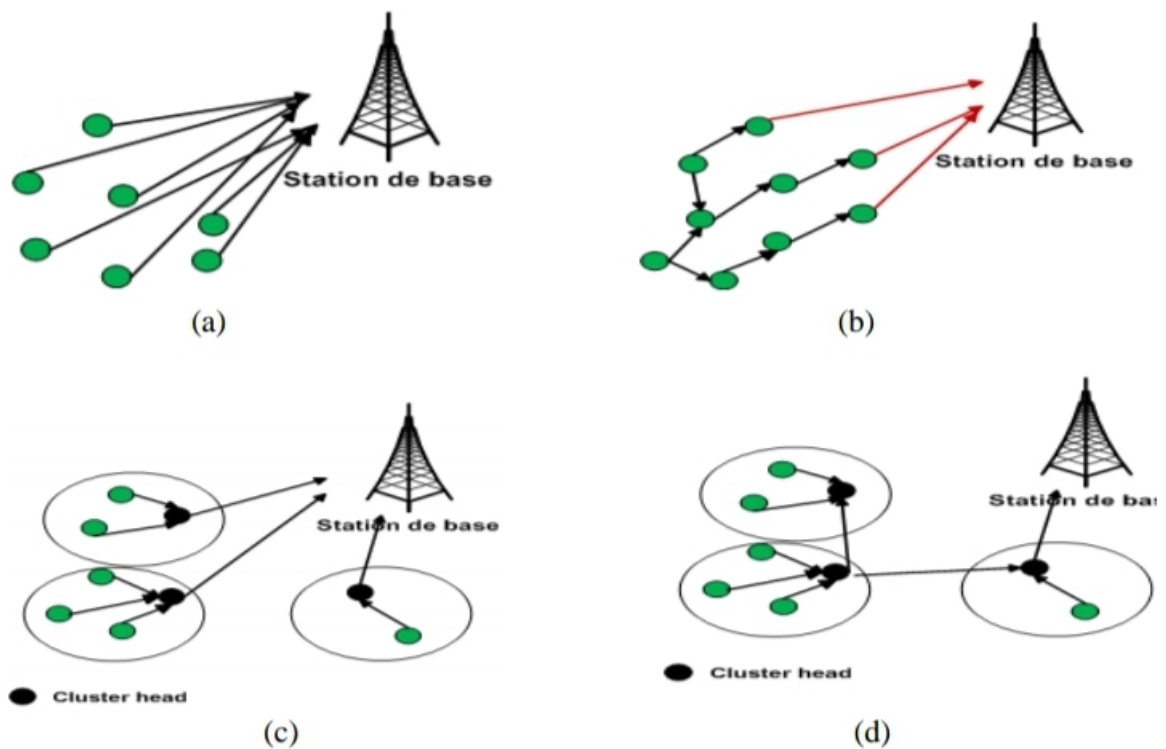


FIGURE 1.6 – les types de communication dans les réseaux de capteurs sans fil

1.8 Le routage dans le RCSF

1.8.1 Définition de routage

Le routage est un processus qui permet de sélectionner des chemins dans un réseau pour transmettre des données depuis un expéditeur jusqu'à un ou plusieurs destinataires. On parle de routage dans différents domaines : réseaux téléphoniques, réseaux électroniques (comme Internet), réseaux de transports, Le routage est le mécanisme par lequel des chemins sont sélectionnés dans un réseau pour acheminer les données d'un expéditeur jusqu'à un ou plusieurs destinataires. Le routage est une tâche exécutée dans de nombreux réseaux, tels que le réseau téléphonique, les réseaux de données électroniques comme l'Internet, et les réseaux de transports. Sa performance est importante dans les réseaux décentralisés, c'est-à-dire où l'information n'est pas distribuée par une seule source, mais échangée entre des agents indépendants. Le terme routage désigne l'ensemble des mécanismes mis en œuvre dans un réseau pour déterminer les routes qui vont acheminer les paquets d'un terminal émetteur à un terminal récepteur. On distingue généralement deux entités :

- * L'algorithme de routage.
- * Le protocole de routage[14].

1.8.2 Classification des protocoles de routage pour les RCSF

Les protocoles de routage pour les RCSF peuvent être classés selon plusieurs critères. La Figure 1.7 illustre une classification qui se base sur quatre critères : la structure du réseau, fonction du protocole, l'établissement de la route et l'initiateur de la communication[15].

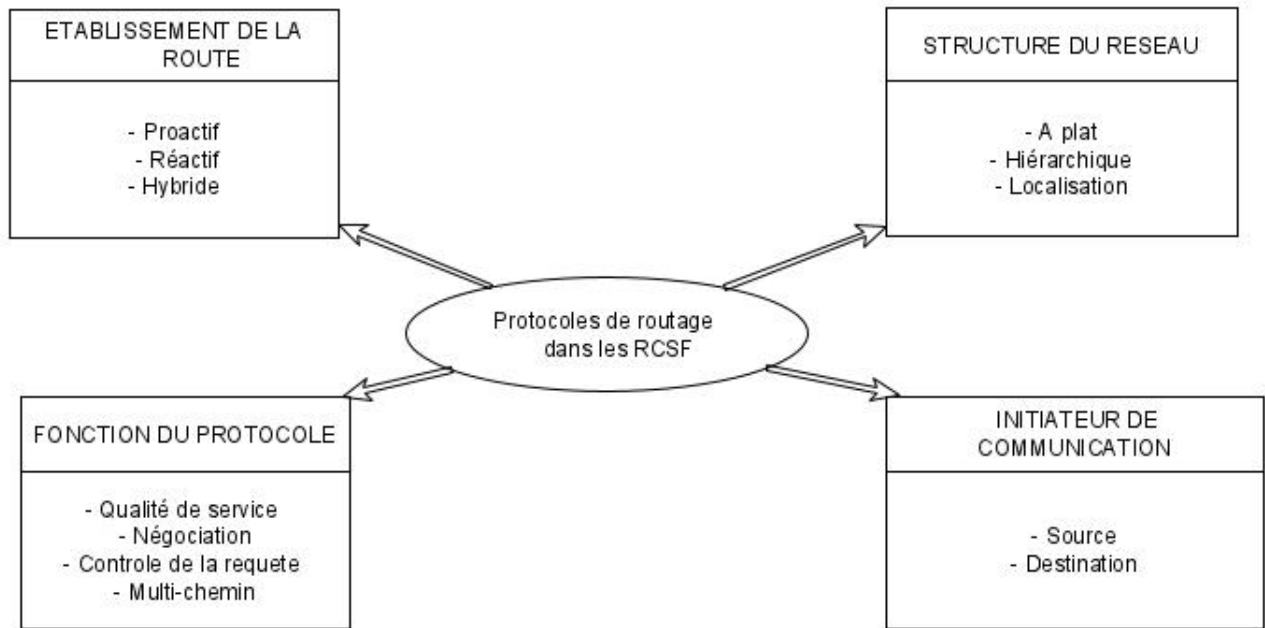


FIGURE 1.7 – Classification des protocoles de routage.

a) Protocoles de routage basés sur la structure du réseau

Les protocoles de routage basés sur la structure du réseau peuvent être classés en trois catégories : protocoles à plat (Flat based routing), protocoles hiérarchiques (Hierarchic based routing/Clustering based routing) et protocoles basés sur la localisation géographique (Location based routing) [15].

1. Routage à plat Comme illustré sur la Figure 1.8, dans cette catégorie de protocoles, les noeuds ont le même rôle et ils collaborent entre eux pour accomplir la tâche de routage. En raison du grand nombre de tels noeuds, il n'est pas faisable d'affecter un identificateur à chaque noeud.

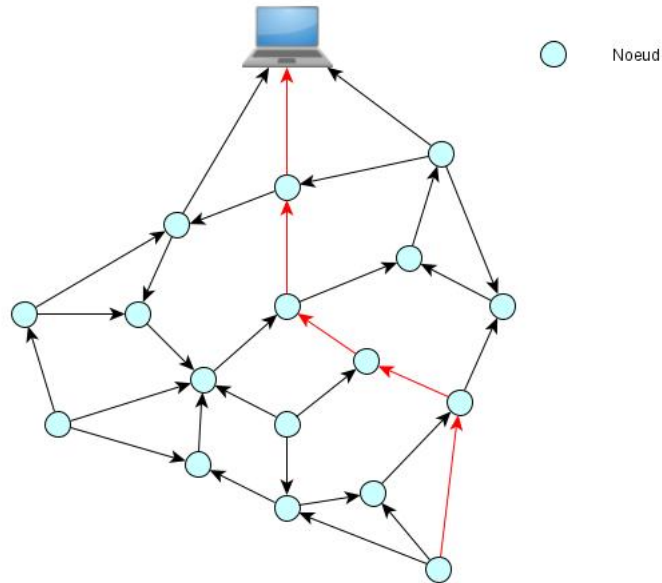


FIGURE 1.8 – Protocoles de routage à plat.

2. Routage hiérarchique Comme illustré sur la Figure 1.9, dans une architecture hiérarchique, les noeuds sont regroupés en cluster et chaque cluster est orchestré par un cluster head ou chef de cluster. C'est ce dernier qui communique les données reçues des noeuds de son cluster à la station de base. Un protocole de routage doit être conçu de telle sorte qu'il prend en considération ce découpage hiérarchique pour assurer la bonne délivrance des paquets au destinataire.

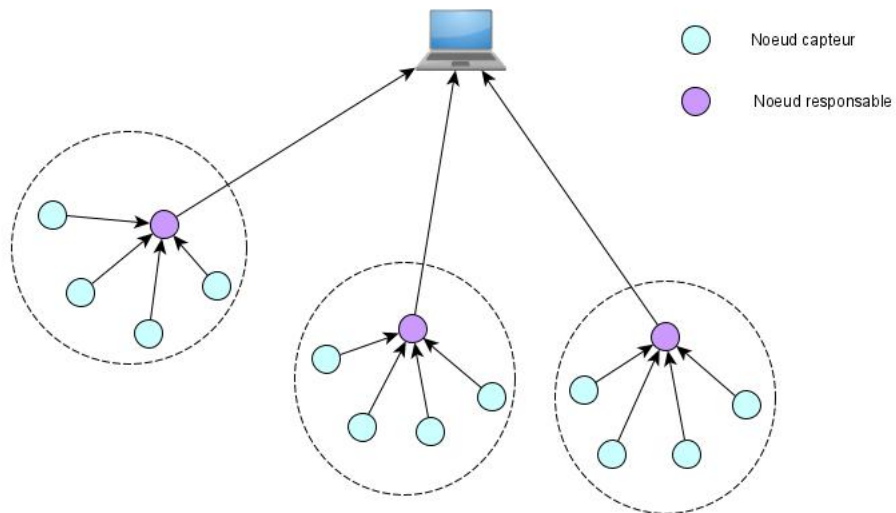


FIGURE 1.9 – Topologie hiérarchique.

3. Routage géographique La plupart des protocoles de routage dans les réseaux de capteurs nécessitent la localisation des noeuds capteurs. En général, ces informations sont nécessaires pour calculer la distance entre deux noeuds particuliers de sorte que la consommation d'énergie puisse être estimée. Puisque il n'y a aucun système d'adressage pour les noeuds dans les réseaux de capteurs (comme les adresses IP) et comme ils sont déployés dans une région d'une manière aléatoire, l'information de localisation de ces noeuds peut être utilisée dans le routage des données d'une manière efficace en termes d'énergie.

b) Classification selon l'initiateur de communication

La communication dans un réseau de capteurs peut être initiée par les noeuds sources ou par les noeuds destinataires[15].

1. Communication lancée par la source Dans les protocoles de communication lancée par la source, les noeuds envoient des données capturées à la destination. Ces protocoles utilisent les données rapportées avec time-driven ou avec event-driven. Ceci signifie que les données sont envoyées à certains intervalles ou quand les noeuds capturent certains évènements.

2. Communication lancée par la destination Les protocoles de communication lancée par la destination utilisent les données rapportées avec query-driven, et dans ce cas, les noeuds répondent aux requêtes envoyées par la destination ou un autre noeud différent. C'est-à-dire propager les requêtes à tous les noeuds d'une région topologique et attendre la réception des données du noeud capteur concerné dans cette région.

c) Classification selon l'établissement de la route

Suivant la manière de création et de maintenance des routes lors de l'acheminement des données, les protocoles de routage peuvent être séparés en trois catégories : les protocoles proactifs, les protocoles réactifs et les protocoles hybrides[15].

1. Protocoles proactifs Les protocoles de routage proactifs essaient de maintenir les meilleurs chemins existants vers toutes les destinations possibles (qui peuvent représenter l'ensemble de tous les noeuds du réseau) au niveau de chaque noeud du réseau.

2. Protocoles réactifs Les protocoles de routage réactifs (dit aussi : les protocoles de routage à la demande) créent et maintiennent des routes selon les besoins. Lorsque le réseau a besoin d'une route, une procédure de découverte de route est lancée.

3. Protocoles hybrides Les protocoles hybrides combinent les deux idées des protocoles proactifs et réactifs. Ils utilisent un protocole proactif pour apprendre le proche voisinage (par exemple le voisinage à deux ou à trois sauts), ainsi, ils disposent des routes immédiatement dans le voisinage. Au de là de la zone de voisinage, le protocole hybride fait appel à un protocole réactif pour chercher des routes.

1.9 Conclusion

Ce chapitre nous a permis d'avoir une vue globale sur les réseaux de capteurs sans fil notamment sur les routages et ses protocoles qui présentent un intérêt considérable vu a l'évolution connue dans ce domaine.

Pour répondre à notre objectif primordial qui est l'optimisation de la consommation de l'énergie dans un RCSF nous allons concevoir un algorithme optimal qui fera l'objet du chapitre qui suit.

Chapitre 2

Quelques rappels sur la théorie des graphes

2.1 Introduction

La théorie des graphes est un outil puissant de modélisation et de résolution de problèmes concrets. A l'origine, la théorie des graphes était présentée comme une curiosité mathématique ; Euler lors d'une de ses promenades nocturnes a voulu tracer un itinéraire circulaire dans la ville de Königsberg. Partant d'un point donné, il voulut visiter les sept ponts de cette ville (disposés selon le schéma ci-dessous) une seule fois seulement, puis retourner à son point de départ.

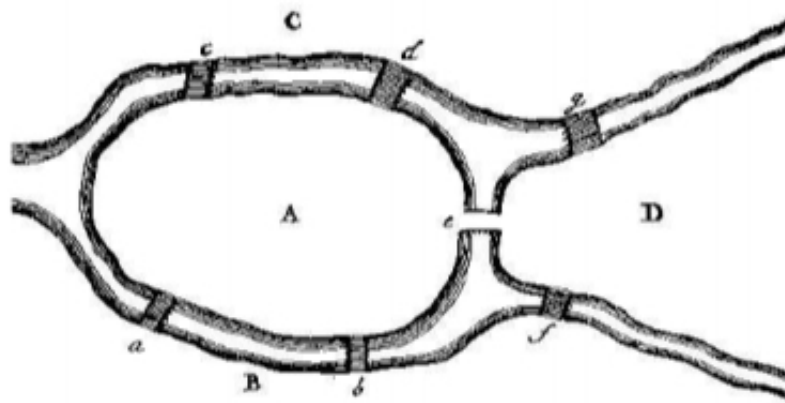


FIGURE 2.1 – les sept ponts de Königsberg

Les points A, B, C et D sont des rives. Ensuite la théorie des graphes a été utilisée pour modéliser des circuits électriques (Kirch-hoff), puis de nombreuses applications dans différents domaines tels : la chimie, la psychologie, etc [16].

2.2 Concepts de base et notations

2.2.1 Définitions générales

Les graphes sont des concepts mathématiques utilisés pour modéliser des relations binaires entre des objets d'un même ensemble. Ils sont fréquemment utilisés pour modéliser des systèmes qui se présentent sous la forme d'un réseau. Il existe deux types de graphes : les graphes orientés et les graphes non orientés [16].

2.2.2 Graphe orienté

Un graphe orienté est un couple $(X;U)$, où X est l'ensemble des sommets du graphe et U , l'ensemble de ses arcs. X et U sont finis. L'arc est une relation entre deux sommets, dotée d'une orientation :

Si $u = (x;y)$ est un arc de U alors avec $x, y \in X$, la relation est orientée de x vers y . Le graphe G est noté $G=(X;U)$ [16].

La figure 2.2 montre un graphe orienté à 5 sommets et 9 arcs.

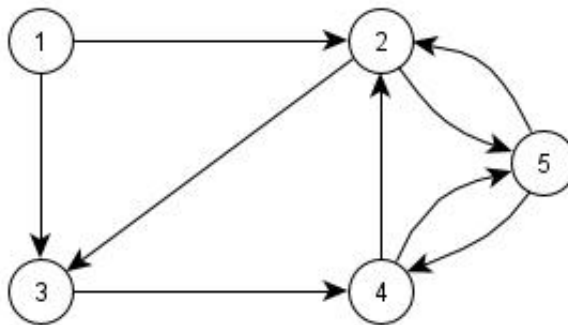


FIGURE 2.2 – Graphe orienté

2.2.3 Graphe non orienté

Un graphe non orienté G est un couple $(X;E)$, où X est un ensemble dont les éléments sont appelés sommets et E un sous-ensemble de parties de X contenant chacune au plus 2 éléments et dont les éléments sont appelés arêtes.

La figure 2.3 illustre un graphe non orienté à 5 sommets et 7 arêtes.

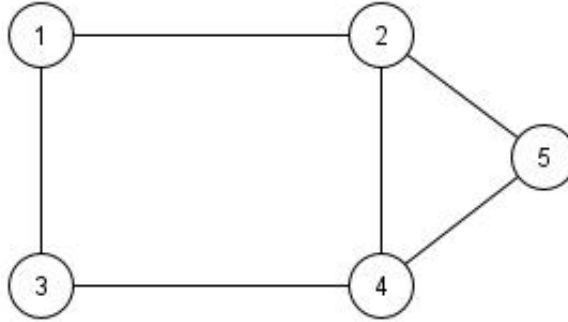


FIGURE 2.3 – Graphe non orienté

2.2.4 Sous-graphe engendré

Un sous graphe est un graphe contenu dans un autre graphe, L'ensemble des sommets du sous-graphe H est un sous-ensemble de l'ensemble des sommets de G et l'ensemble des arêtes de H est un sous-ensemble de l'ensemble des arêtes de G , la figure 2.4 montre un sous graphe de graphe non orienté.

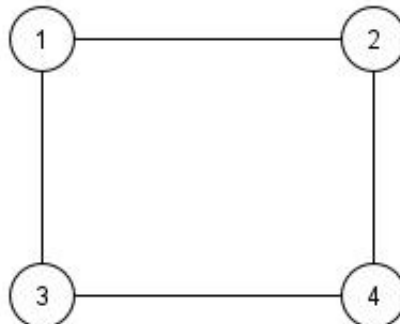


FIGURE 2.4 – Sous graphe

2.2.5 Graphe partiel

Soit $G=(V,E)$ un graphe. Le graphe $G'=(V,E')$ est un graphe partiel de G , si $E' \subset E$. Autrement dit, on obtient G' en enlevant une ou plusieurs arêtes au graphe G .

2.2.6 Sous graphe partiel

Pour un sous-ensemble de sommets A inclus dans V , le sous-graphe de G induit par A est le graphe $G=(A,E(A))$ dont l'ensemble des sommets est A et l'ensemble des arêtes $E(A)$ est formé de toutes les arêtes de G ayant leurs deux extrémités dans A . Autrement dit, On obtient G' en enlevant un ou plusieurs sommets au graphe G , ainsi que toutes les arêtes incidentes à ces sommets.

2.2.7 Graphe valué

Un graphe valué $G=(X;U;v)$ est un graphe $(X;U)$ (orienté ou non orienté) muni d'une application $v : U \rightarrow \mathfrak{R}$. L'application v est appelée valuation du graphe. On peut étendre cette valuation en posant $\forall (x;y) \in X^2, v(x;y)=+\infty$ si $(x;y) \notin U$.

2.2.8 Extrémité initiale et terminale (successeur et prédécesseur)

Soit un arc (i ,j) ; i est dit extrémité initiale de (i ,j) et j extrémité terminale de (i ,j) . On dit aussi que j est un successeur de i , et i un prédécesseur de j . L'arc (i ,j) est dit incident vers l'extérieur en i et vers l'intérieur en j .

Définition Un arc de graphe orienté G de la forme (i ,i) est appelé une boucle. Pour un arc $u=(i ,j)$, le point i est son extrémité initiale, et le point j son extrémité terminale.

On dit que j est un successeur de i s'il existe un arc ayant son extrémité initiale en i et son extrémité terminale en j . l'ensemble des successeurs de i se note $\Gamma_G^+(i)$

De même, on dit j est un prédécesseur de i s'il existe un arc de la forme (j,i) . L'ensemble des prédécesseurs de i se note $\Gamma_G^-(i)$

L'ensemble des sommets voisins de i se note $\Gamma_G(i)=\Gamma_G^+(i) \cup \Gamma_G^-(i)$

On note $\Gamma^+(i)$ l'ensemble des successeurs de i , $\Gamma^-(i)$ l'ensemble des prédécesseurs de i .

2.2.9 Arcs adjacents, arêtes adjacentes

Deux arcs (resp.arêtes) sont adjacents (resp.adjacentes) en un sommet x , si x est une extrémité commune des deux arcs (resp.arêtes).

2.2.10 Graphe simple

Un graph simple est un graphe sans boucles ni arcs (arêtes) multiple.
 la figure 2.5 montre un exemple d'un graphe simple.

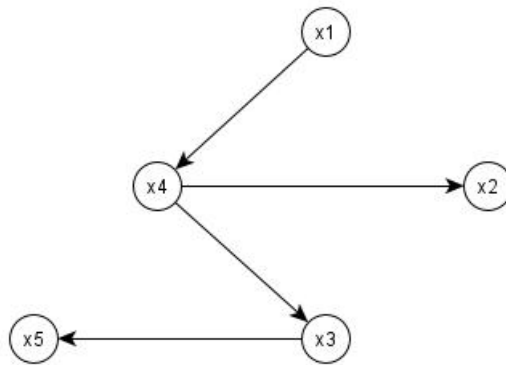


FIGURE 2.5 – Graphe simple

2.3 Connexité dans les graphes

2.3.1 Boucle

Une boucle est une arête dont son extrémité initiale et son extrémité terminal coïncident.

2.3.2 Sommets adjacents

On dit que deux sommets, x et y , sont adjacents si et seulement s'il existe une arête le reliant.

2.3.3 Les voisins d'un sommet

Les voisins d'un sommet x d'un graphe G sont les sommets qui admettent une arête avec x ,
 La figure 2.6 montre un exemple ayant des sommets voisins.

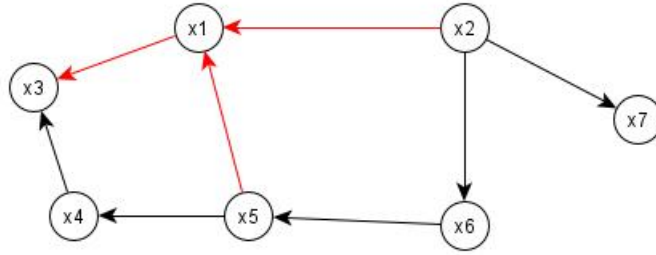


FIGURE 2.6 – Les voisins d'un sommet

2.3.4 Chaîne

Une chaîne est une suite finie de sommets reliés entre eux par une arête, la figure 2.7 montre un graphe possède une chaîne.

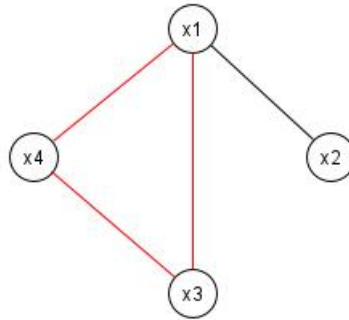


FIGURE 2.7 – Graphe G possède une chaîne

2.3.5 Chaîne simple

Une chaîne simple est une chaîne qui n'utilise pas deux fois la même arête.

2.3.6 Chaîne élémentaire

Une chaîne élémentaire est une chaîne qui utilise pas deux fois le même sommets.

2.3.7 Chemin

Dans le graphe G , un chemin menant x_0 à x_k est une suite de sommets reliés successivement par des arcs orientés dans le même sens, que l'on note (x_0, x_1, \dots, x_k) , la figure 2.8 montre un

chemin de x_1 vers x_5 .

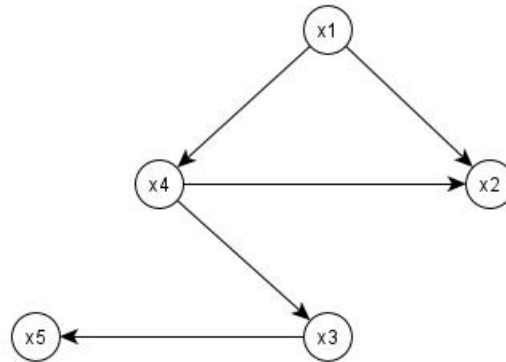


FIGURE 2.8 – Graphe G possède un chemin

2.3.8 Chemin simple

Un chemin simple est un chemin qui passe une seule fois par ses arcs.

2.3.9 Chemin élémentaire

Un chemin élémentaire est un chemin qui passe une et une seule fois par ses sommets.

2.3.10 Cycle

Un cycle est une chaîne dont les extrémités initiales et finales coïncident. la figure 2.9 montre un exemple d'un graphe ayant trois cycle qui sont :

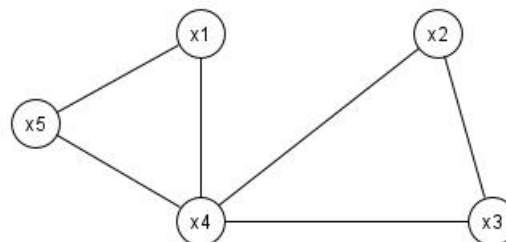


FIGURE 2.9 – Graphe ayant trois cycle

2.3.11 Circuit

Un circuit est un chemin dont les extrémités sont confondues. On le note par $(x_0, x_1, x_2, \dots, x_k = x_0)$ comme illustré dans la figure 2.10

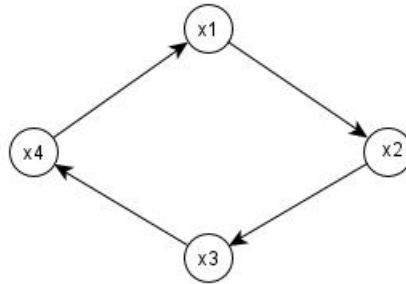


FIGURE 2.10 – Circuit

2.3.12 Graphe connexe

Soit le graphe $G=(X,U)$. On dit que G est connexe si et seulement si $\forall x,y \in X$, il existe une chaîne reliant x à y la figure 2.11 montre un graphe connexe.

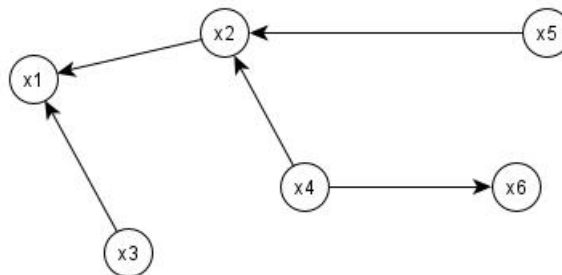


FIGURE 2.11 – Graphe connexe

2.3.13 Composante connexe

Une composante connexe est l'ensemble de sommets qui ont deux à deux une relation de connexité, et tout sommet en dehors de la composante n'a pas de relation de connexité avec les sommets de cette composante noté CC.

2.3.14 Graphe fortement connexe

Un graphe orienté $G=(X,U)$ est dit fortement connexe si $\forall \{ x,y \} \in X$, il existe un chemin de x à y et un autre chemin de y à x .

2.3.15 Composante fortement connexe

On appelle composante fortement connexe, l'ensemble de sommets qui ont deux à deux la relation de forte connexité, de plus tout sommet en dehors de la composante n'a pas de relation de forte connexité avec aucun élément de cette composante notée c.f.c

Remarque

- Un graphe connexe n'admet qu'une seule composante connexe.
 - Un graphe fortement connexe n'admet qu'une seule composante fortement connexe.
- la figure 2.12 montre un exemple d'un graphe à deux composante fortement connexe.

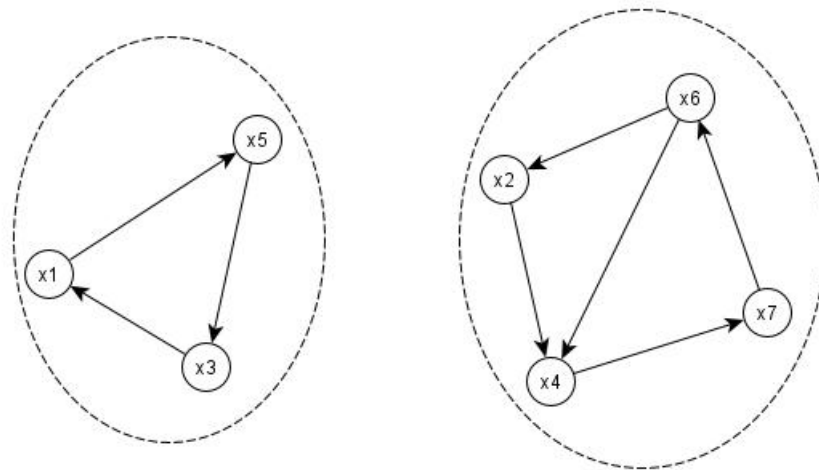


FIGURE 2.12 – Graphe à deux Composante fortement connexe

2.3.16 Arbres et forêts

Un **arbre** est un graphe connexe et acyclique (c'est-à-dire un graphe sans cycle) , Ainsi un arbre est nécessairement simple, une chaîne élémentaire est en particulier un arbre. Les arbres ont des propriétés, on distingue toujours par n_G et m_G ou n le nombre de sommets d'un arbre et m son nombre d'arêtes [17].

Caractérisation des arbres

soit $G = (X;E)$ un arbre, alors :

1. G est acyclique et possède $|X| - 1$ arêtes
2. G est un graphe connexe minimal (chaque arête est un isthme [un isthme est une arête dont la suppression rend le graphe non connexe])
3. G est un graphe maximal sans cycles (l'addition d'une arête quelconque crée un cycle),
4. Toute paire de sommets de G est connectée par une chaîne unique [17].

Arbres couvrants d'un graphe

un arbre couvrant d'un graphe $G = (X;E)$ est un graphe $A(X';E')$ partiel de G qui est un arbre et dont $X' = X$.

2.4 Quelques types de graphes

2.4.1 Graphe Complet

Un graphe G est dit complet si tous les sommets de G sont deux à deux adjacents la figure 2.13 montre un graphe complet à 5 sommets et 10 arêtes[18].

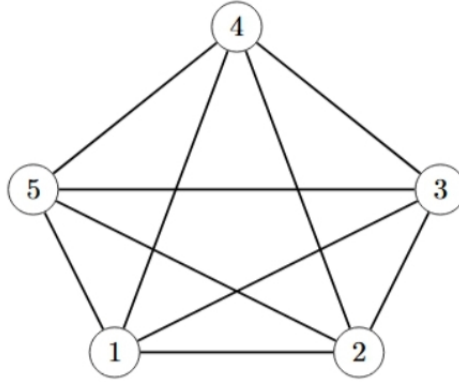


FIGURE 2.13 – Graphe complet

2.4.2 graphe planaire

Un graphe est planaire s'il peut être tracé dans un plan sans qu'aucun croisement entre les arêtes le graphe de la figure 2.14 est planaire[18].

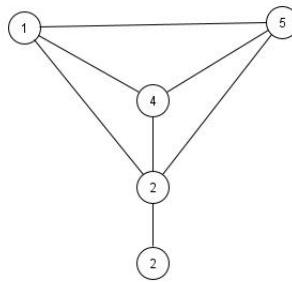


FIGURE 2.14 – Graphe planaire

2.4.3 Graphe biparti

Un graphe $G = (V; E)$ est dit biparti si l'ensemble de ses sommets V peut être partitionner en deux sous-ensembles V_1 et V_2 , de sorte que les sommets de même sous ensemble ne sont pas adjacents[18].

la figure 2.15 montre un exemple d'un graphe biparti.

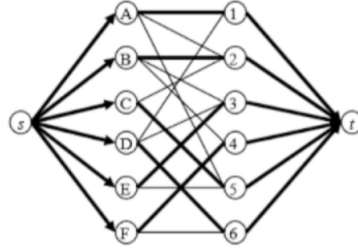


FIGURE 2.15 – Graphe biparti

2.5 Plus courts chemins

2.5.1 Problème du plus court chemin

On se place dans le cas des graphes orientés valués $G = (S; A; V)$. Mais les résultats et algorithmes présentés se généralisent facilement aux cas des graphes non orientés valués. Une autre solution consiste à transformer le graphe non-orienté en un graphe orienté en remplaçant une arête entre deux sommets par deux arcs de sens inverse entre ces sommets[19].

Définition

-Le coût ou poids d'un chemin $c = \langle s_0; s_1; \dots; s_k \rangle$ est égale à la somme des valuations des arcs composant le chemin, c'est à dire,

$$L(c) = \sum_{i=1}^k V(s_{i-1}, s_i)$$

-Le coût d'un plus court chemin entre deux sommets s_i et s_j est noté $\delta(s_i; s_j)$ et est défini par :

$$\delta(s_i; s_j) = \begin{cases} \min \{ L(c) / c = \text{chemin de } s_i \text{ à } s_j \}, & \text{s'il existe au moins un chemin entre } s_i \text{ et } s_j \\ +\infty, & \text{sinon} \end{cases} \quad (2.1)$$

2.6 Algorithme de recherches de Plus courts chemins

2.6.1 Algorithme de Bellman

Données : $R=(S,A,V)$ s :racine

Résultats : π : Plus courtes distances de s à x , B : Arborescence , C : Sous-ensemble de sommets accessibles à partir de s . [20]

a) Initialisation : $\pi (s)=0$, $B=\emptyset$, $C=\{ s \}$.

b) Procédure de calcul :

(.) Chercher un sommet $j \in C^C$ telle que Γ_j^{-1} est contenu dans C

(..) Calculer : $\pi(j)=\min\{ \pi (i) + v_{ij} \} i \in \Gamma_j^{-1}$

(...) Poser : $C=C \cup \{ J \}$

c) Test d'arrêt : Si $| C | = | S |$, terminer

Sinon : retourner en l'étape (b)

2.6.2 Algorithme de Ford

Données :Un graphe value $G=(S,A,V)$, sans circuit absorbant, s_1 une racine.

Résultats : Un potentiel π , Une arborescence de plus courts chemins B . [20]

a) Initialisation : poser $\pi (s_1) =0$ et $\pi (s_j) = +\infty ; \forall j ,j \neq 1$.

b) Tester s'il existe un arcs $(s_i,s_j) \in A$, vérifiant : $\pi (j) > \pi (i) + v_{ij}$ alors poser $\pi (j) = \pi + v_{ij}$

Sinon : Terminer le graphe $G'=(S,B)$ ou $B=\{ (s_i,s_j) \in A \text{ tq : } \pi (j)=\pi (i) + v_{ij} \}$ Contient au moins un chemin C de longueur minimale.

Algorithme de Dijkstra

(Recherche d'un plus court chemin de la racine s à tous les autres sommets dans un réseau ayant toute les évaluations positive)[20].

Données : $R=(S,A,V)$, s :Racine

Résultats : π : Les plus courtes distances, B : Arborescence, C : Sous-ensemble de sommets accessibles a partir de s .

a) Initialisation : $C=\{s\}, \pi(s)=0, B= \emptyset$, $B= \pi(j)=\{ v_{sj}$ si $(s, j) \in A, +\infty$ si non.

b) Procédure de calcul :

(.) chercher un sommet $i \in C^C$ vérifiant : $\pi(i)=\min\{ \pi(j)\} j \in C^C$

(..)Poser : $C=C \cup \{i\}$ et $\pi(j) = \min\{\pi(j), \pi(i) + v_{ij}\} j \in \tau_i \cap C^C$

C) Test d'arrêt : si $|C|=|S|$, Terminer, le Réseau R contient au moins un plus court chemin de s à tout sommet i .

Sinon : retourner en l'étape(b).

2.7 Conclusion

Dans ce chapitre nous avons défini les différentes notions et définitions relatives à la théorie des graphes ainsi, nous avons proposer quelques algorithmes de base que nous allons utiliser dans le chapitre suivant.

Chapitre 3

Protocole de routage basée sur l'algorithme de Dijkstra dans un RCSF

3.1 introduction

Dans un réseau de capteurs, l'énergie est la ressource la plus précieuse qui influe directement sur la durée de vie des capteurs et du réseau en entier. Ceci à cause de la limitation de la capacité énergétique des batteries qui alimentent les capteurs d'une part et la forte possibilité de ne pas remplacer ces batteries d'autre part. Maximiser la durée de vie du réseau revient donc à réduire la consommation énergétique des noeuds. Malgré la diversité des protocoles à basse consommation d'énergie qui ont été proposés, l'énergie dans un réseau de capteurs reste toujours un problème de recherche ouvert, ce qui nécessite d'autres solutions qui viennent renforcer et améliorer les solutions existantes.[15].

Dans ce chapitre, nous allons Implémenter l'algorithme de Dijkstra dans les RCSF pour déterminer un plus court chemin de s à puits (station de base).

3.2 Modèle du réseau

Le réseau considéré est un ensemble de nœuds déployées aléatoirement dans une zone de capture, ou chaque nœud capteur envoie ses données à la station de base par un intermédiaire de ses nœuds voisin, à base de cette structure de réseau on est dans la catégories des protocoles de routage à plat.

Avantages

- Les réseaux à plat sont scalables du fait que chaque noeud participe également à la tâche de routage et puisque les noeuds ont besoin seulement des informations sur leurs voisins directs.
- Les réseaux à plat permettent aux protocoles de routage d'être simples, ainsi que nous n'avons aucun besoin d'algorithmes complexes pour faire le choix d'un cluster-head.
- Le fait que les noeuds d'un réseau à plat ont le même rôle et les mêmes propriétés, les noeuds ont besoin juste de connaître seulement leurs voisins.
- La possibilité de réaliser un routage optimal réside de l'une de caractéristiques de routage à plat, celle que tous les noeuds peuvent communiquer entre eux sans avoir appel à un intermédiaire[15].

Inconvénients

- Si les noeuds capteurs sont uniformément distribués dans tout le réseau et il y a un seul noeud puits, Alors, les noeuds au tour de ce dernier épuiseront leurs énergies plus tôt que les autres noeuds Parce que tout le trafic du réseau passe par les noeuds entourant le noeud puits[15].

3.3 Solution proposée

la modélisation de notre problème qui est la minimisation d'énergie dans les RCSF, nous à conduite à l'étude de sommets et d'arêtes, sachant que les nœuds capteurs sont dotés d'une petite batterie.

Dans ce fait, nous avons utilisé l'un des algorithmes de théorie des graphes qui est l'algorithme de dijkstra, qui nous permettra de déterminer les routes à emprunter avec un chemin de plus faible métrique.

3.4 Hypothèses

Notre solution est base sur les hypothèses suivantes :

- Tous les nœuds ont le même niveau énergétique.
- Les nœuds sont déployées aléatoirement dans une zone de capture.
- Le graphe valués connexe ne contient pas de circuit absorbant.
- Chaque noeud peut atteindre la station de base avec communication multi-sauts.

- Les capteurs sont fixe.
- Chaque noeud possède un identifiant unique.
- La mort de chaque capteur n'est causée que par l'épuisement de son énergie.
- La station de base est vue comme une ressource non limitée ni épuisable(l'unité de stockage, l'énergie).

3.5 Détail de la solution proposée

3.5.1 Objectif de notre modélisation

Modélisation de système par un graphe.

Pour cela nous avons modéliser notre problème avec un graphe. l'algorithme de dijkstra permet de déterminer la distance minimale et d'acheminer notre information d'un noeud capteur vers une station de base avec peu d'énergie, dans le but de déterminer le chemin minimal d'acheminement de l'information d'un noeud capteur à une station de base, nous avons fait appel à l'algorithme de dijkstra.

Soit $G=(S,A,V)$; un graphe connexe représentant une relation entre les capteurs du réseau avec :

- S : est un ensemble fini dont les élément sont appelées des noeuds.
- A : est un ensemble fini dont les élément sont appelées des arêtes (liens).
- V : la distance entre deux sommets ou on associe a chaque arête $A=(i,j)$ une longueur π_{ij} .
- C : sous ensemble de sommets accessibles a partir de s.

Dans le cadre de notre modélisation, nous allons considérer qu'on a une seule station de base et plusieurs noeuds de capteurs.

Pour l'application de l'algorithme de dijkstra nous considérons le réseau de capteur donné dans le tableau suivante :

	s_0	s_1	s_2	s_3	s_4	s_5	s_6	s_7
s_0	0	7		3			9	
s_1	7	0	5	2	3		3	
s_2		5	0		9	2	4	
s_3	3	2		0	2			
s_4		3	9	2	0	10		
s_5			2		10	0		3
s_6	9	3	4				0	8
s_7						3	8	0

(3.1)

Ce tableau ci-dessus représente la matrice des distance entre les nœuds de réseau.

la représentation par un graphe est donné comme suite :

pour le graphe si dessous on suppose que on a sept noeud de capteur $C=\{s_0,s_1,s_2,s_3,s_4,s_5,s_6\}$ et une seul station de base qui est le noeud s_7 .

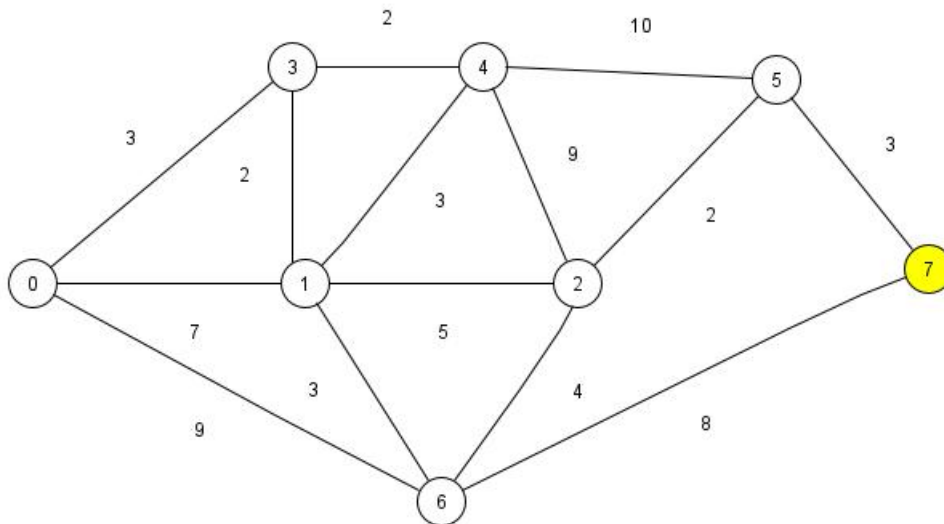


FIGURE 3.1 – Réseau associé après la modélisation.

Résolution on Applique l'algorithme de dijkstra pour déterminer un plus court chemin d'un sommet s_0 à tous les autres sommets j en se basent sur le noeud s_7 .

Initialisation

On pose $C = \{ s_0 \}$; $\pi(s_0) = 0$; $\pi(x) = +\infty \forall x \notin C$.

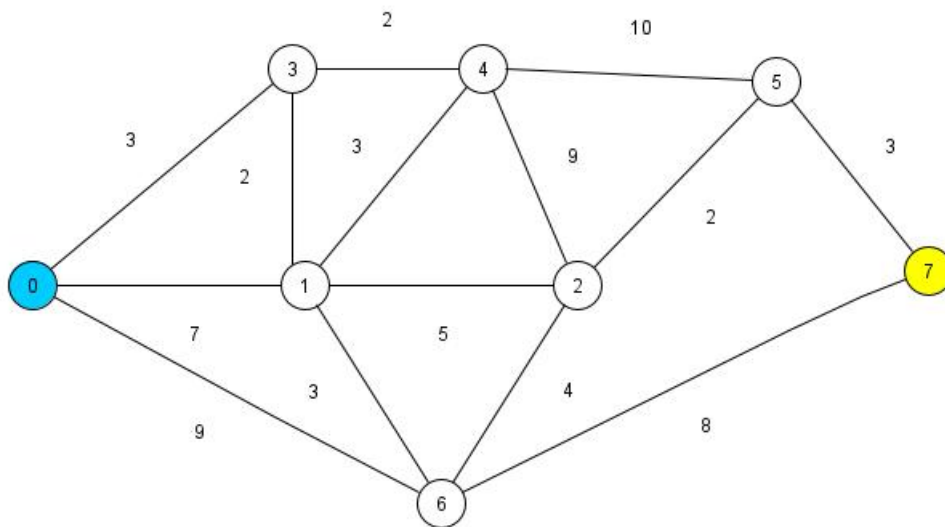


FIGURE 3.2 – Réseau obtenu après l'initialisation.

. Itération 1 :

L'ensemble des sommets est : $C = \{ s_0 \}$.

On examine les sommets voisins de sommet s_0

$$\pi(s_3) = 0 + 3 = 3; \pi(s_1) = 0 + 7 = 7; \pi(s_6) = 0 + 9 = 9$$

$$\min\{ \pi(s_3); \pi(s_1); \pi(s_6) \} = \pi(s_3) = 3.$$

$$C = \{ s_0, s_3 \}$$

$$\pi(s_3) = 3.$$

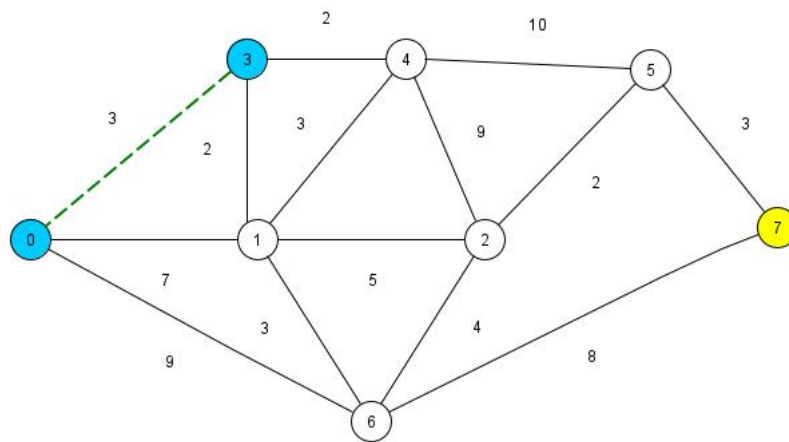


FIGURE 3.3 – Réseau obtenu à la première itération.

. Itération 2 :

L'ensemble des sommets devient : $C = \{ s_0, s_3 \}$

On examine les sommets voisins de sommet s_0 et s_3

$$\pi(s_4) = 3 + 2 = 5; \pi(s_1) = 3 + 2 = 5; \pi(s_6) = 0 + 9 = 9$$

$$\min\{ \pi(s_4); \pi(s_1); \pi(s_6) \} = \pi(s_1) = 5.$$

$$C = \{ s_0, s_3, s_1 \}$$

$$\pi(s_1) = 5.$$

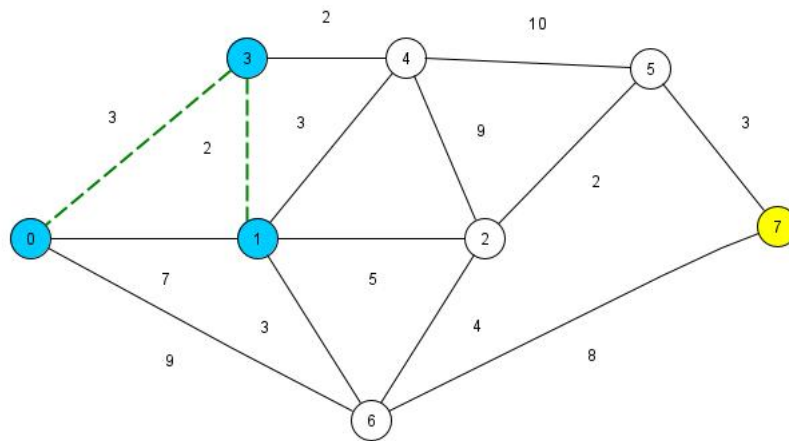


FIGURE 3.4 – Réseau obtenu à la deuxième itération.

. Itération 3 :

L'ensemble des sommets devient : $C = \{ s_0, s_3, s_1 \}$

On examine les sommets voisins de sommet s_0 et s_3 et s_1

$$\pi(s_4) = 3 + 2 = 5; \pi(s_2) = 5 + 5 = 10; \pi(s_6) = 5 + 3 = 8$$

$$\min\{ \pi(s_4); \pi(s_2); \pi(s_6) \} = \pi(s_4) = 5.$$

$$C = \{ s_0, s_3, s_1, s_4 \}$$

$$\pi(s_4) = 5.$$

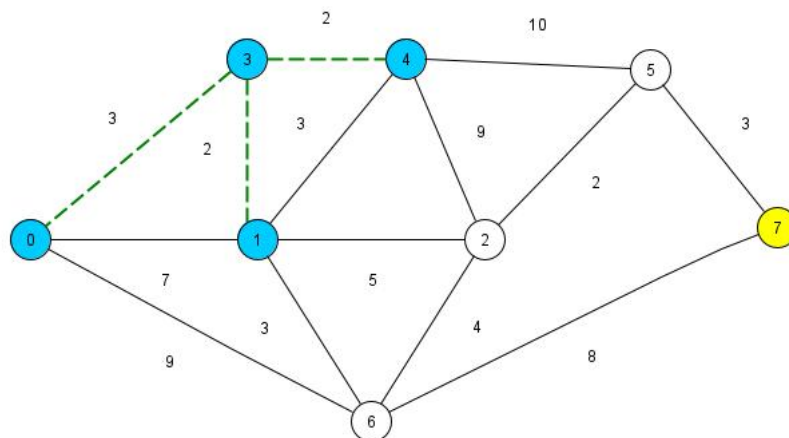


FIGURE 3.5 – Réseau obtenu à la troisième itération

. Itération 4 :

L'ensemble des sommets égale : $C = \{ s_0, s_3, s_1, s_4 \}$

On examine les sommets voisins de sommet s_0 et s_3 et s_1 et s_4

$$\pi(s_5) = 5 + 10 = 15; \pi(s_2) = 5 + 5 = 10; \pi(s_6) = 5 + 3 = 8$$

$$\min\{ \pi(s_5); \pi(s_2); \pi(s_6) \} = \pi(s_6) = 8.$$

$$C = \{ s_0, s_3, s_1, s_4, s_6 \}$$

$$\pi(s_6) = 8.$$

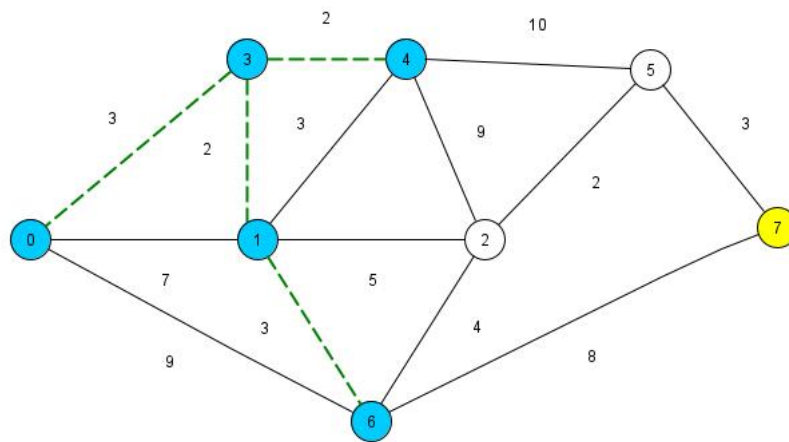


FIGURE 3.6 – Réseau obtenu à la quatrième itération.

. Itération 5 :

L'ensemble des sommets devient : $C = \{ s_0, s_3, s_1, s_4, s_6 \}$

On examine les sommets voisins de sommet s_0 et s_3 et s_1 et s_4 et s_6

$$\pi(s_5) = 5 + 10 = 15; \pi(s_2) = 5 + 5 = 10; \pi(s_6) = 5 + 3 = 8$$

$$\min\{ \pi(s_5); \pi(s_2); \pi(s_6) \} = \pi(s_2) = 10.$$

$$C = \{ s_0, s_3, s_1, s_4, s_6, s_2 \}$$

$$\pi(s_2) = 10.$$

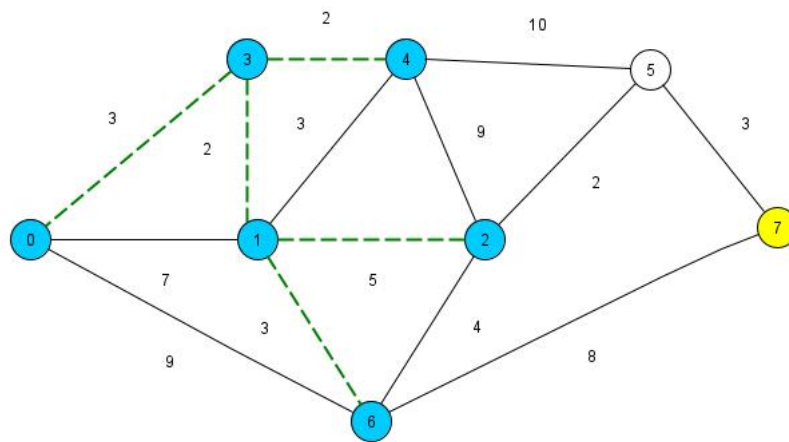


FIGURE 3.7 – Réseau obtenu à la cinquième itération.

. Itération 6 :

L'ensemble des sommets devient : $C = \{ s_0, s_3, s_1, s_4, s_6, s_2 \}$

On examine les sommets voisins de sommet s_0 et s_3 et s_1 et s_4 et s_6 et s_2

$$\pi(s_5) = 10 + 2 = 12; \pi(s_7) = 8 + 8 = 16$$

$$\min\{ \pi(s_5); \pi(s_7) \} = \pi(s_5) = 12.$$

$$C = \{ s_0, s_3, s_1, s_4, s_6, s_2, s_5 \}$$

$$\pi(s_5) = 10.$$

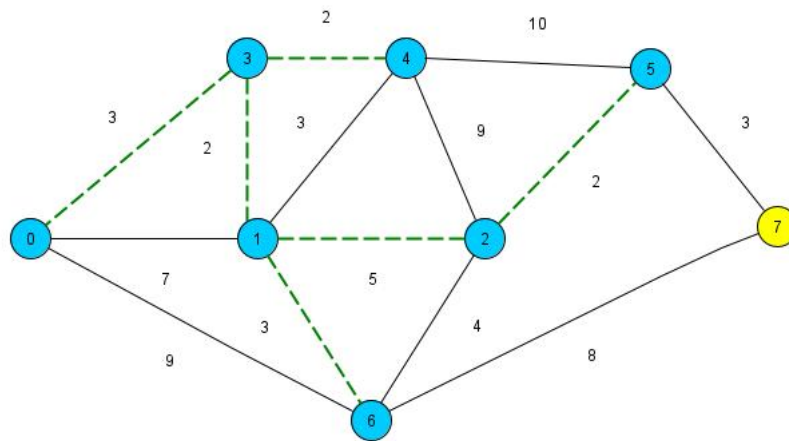


FIGURE 3.8 – Réseau obtenu à la sixième itération.

. Itération 7 :

L'ensemble des sommets devient : $C = \{ s_0, s_3, s_1, s_4, s_6, s_2, s_5 \}$

On examine les sommets voisins de sommet s_0 et s_3 et s_1 et s_4 et s_6 et s_2 et s_5

$$\pi(s_7) = 10 + 2 = 15$$

$$C = \{ s_0, s_3, s_1, s_4, s_6, s_2, s_5, s_7 \}$$

tout les sommet sont marquer, alors on s'arrête, les plus courts chemin sont tels qu'il illustre ce dernier graphe

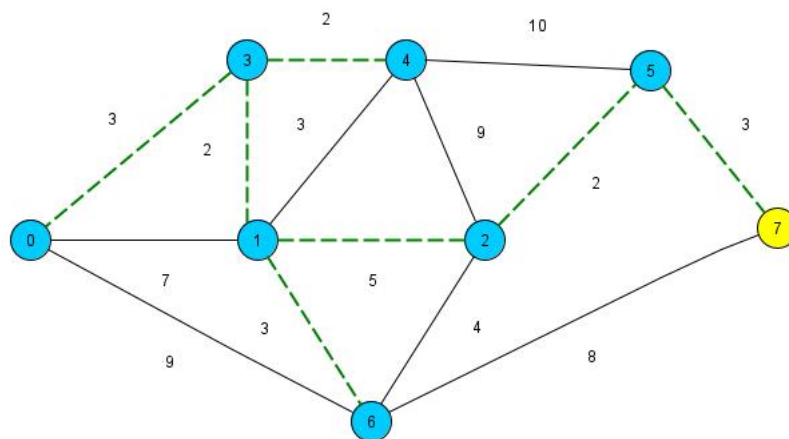


FIGURE 3.9 – Réseau obtenu à la septième itération.

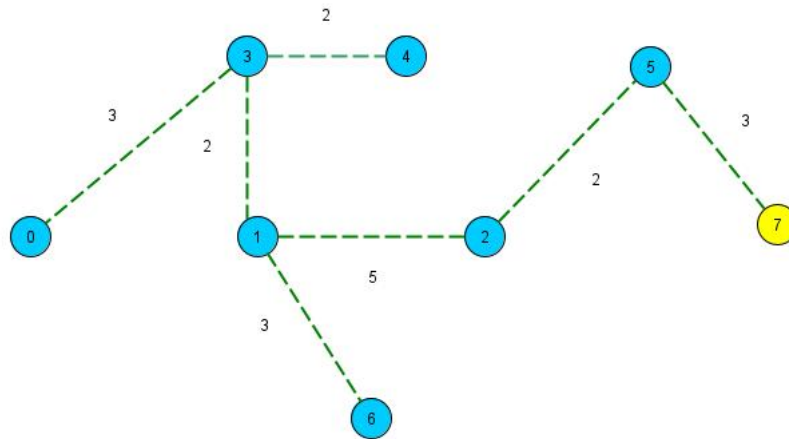


FIGURE 3.10 – Résultat de plus court chemin.

3.6 Application sur le C++

On va programmer l'algorithme de dijkstra pour obtenir un plus court chemin d'un noeud capteur à une station de base d'un réseau donné, et pour le faire nous utiliserons le langage C++.

3.6.1 Présentation de langage C++

Apparu au début des années 90, le langage C++ est actuellement l'un des plus utilisés dans le monde, aussi bien pour les applications scientifiques que pour le développement des logiciels. En tant qu'héritier du langage C, le C++ est d'une grande efficacité. Mais il a en plus des fonctionnalités puissantes, comme par exemple la notion de classe, qui permet d'appliquer les techniques de la programmation-objet.

Programmer pour un ordinateur, c'est lui fournir une série d'instructions qu'il doit exécuter. Ces instructions sont généralement écrites dans un langage dit évolué, puis, avant d'être exécutées, sont traduites en langage machine (qui est le langage du microprocesseur). Cette traduction s'appelle compilation et elle est effectuée automatiquement par un programme appelé compilateur.

Un programme écrit en C++ se compose généralement de plusieurs fichiers-sources. Il y a deux sortes de fichiers-sources :

- ceux qui contiennent effectivement des instructions ; leur nom possède l'extension `.cpp`,
- ceux qui ne contiennent que des déclarations ; leur nom possède l'extension `.h` (signifiant "header" ou en-tête).

Un fichier.h sert à regrouper des déclarations qui sont communes à plusieurs fichiers.cpp, et permet une compilation correcte de ceux-ci. Pour ce faire, dans un fichier.cpp on prévoit l'inclusion automatique des fichiers.h qui lui sont nécessaires, grâce aux directives de compilation include. En supposant que le fichier à inclure s'appelle untel.h, on écrira `include <untel.h>` s'il s'agit d'un fichier de la bibliothèque standard du C++, ou `include "untel.h"` s'il s'agit d'un fichier écrit par nous-mêmes [21].

Aspect pratique

Utiliser Code ::Blocks sous Windows

Lorsque on lance Code ::Blocks, nous voyons apparaitre l'écran ci-contre.

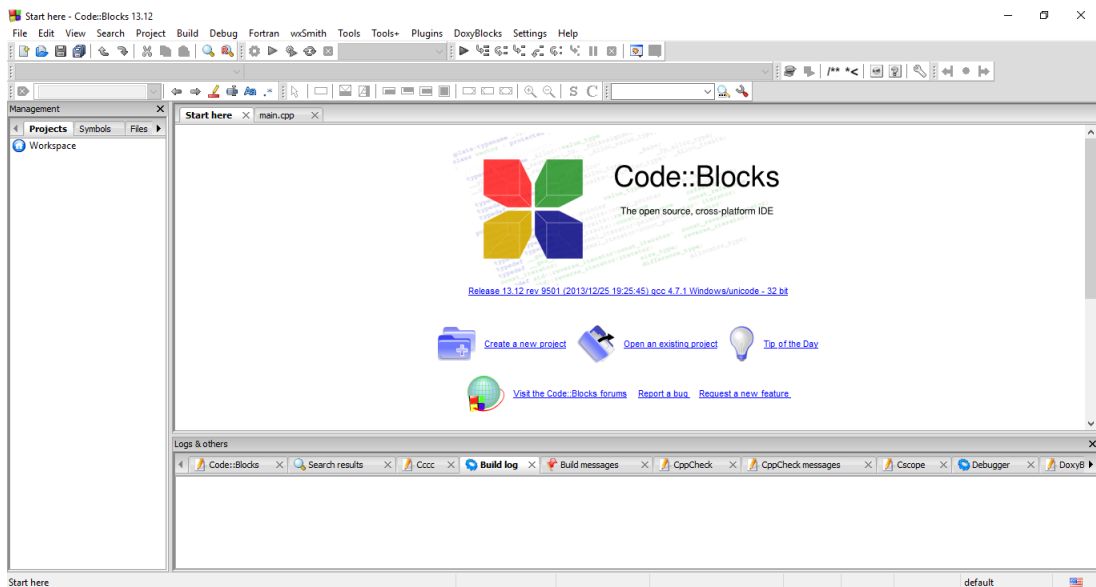


FIGURE 3.11 – l'interface de code blocks.

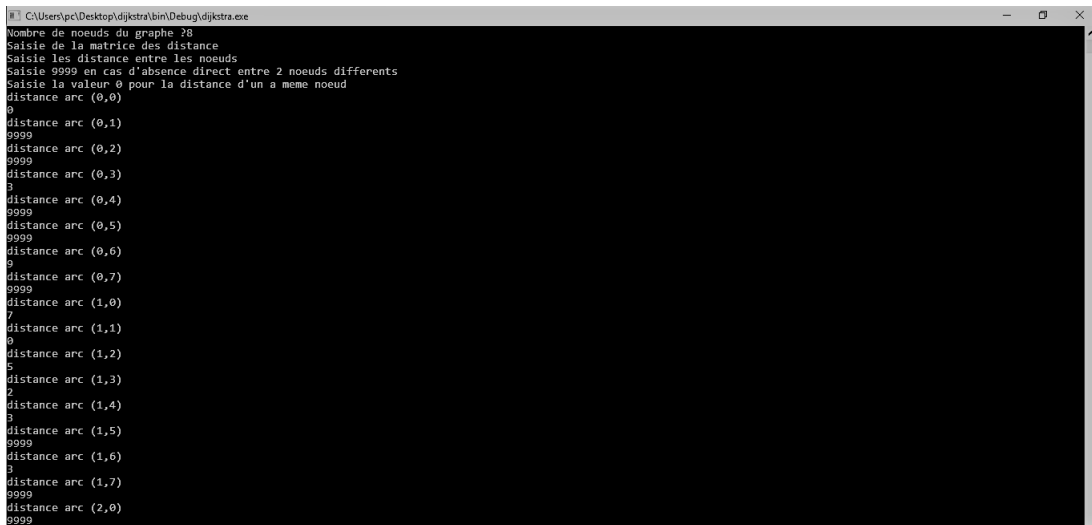
Pour créer un nouveau projet, il faut choisir dans le menu File puis New puis Project.

3.6.2 Application numérique

Nous finissons par restituer les différents résultats de recherche du plus court chemin sur le réseau, obtenus à travers l'exécution de l'algorithme de dijkstra implémenté sous le langage C++.

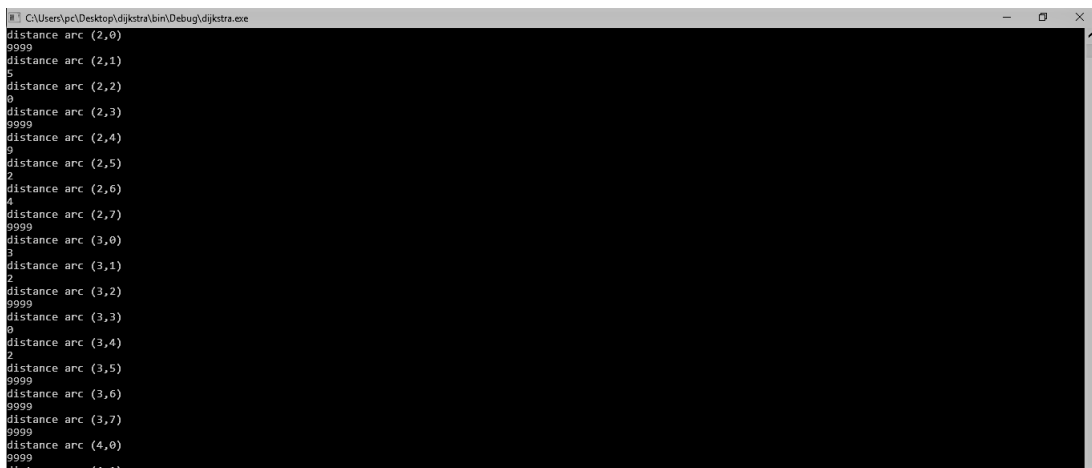
CHAPITRE 3. PROTOCOLE DE ROUTAGE BASÉE SUR L'ALGORITHME DE DIJKSTRA DANS UN RCSF

On introduit la capacité de chaque arc jusqu'à atteindre le dernier.



```
C:\Users\pc\Desktop\dijkstra\bin\Debug\dijkstra.exe
Nombre de noeuds du graphe ?8
Saisie de la matrice des distance
Saisie les distance entre les noeuds
Saisie 9999 en cas d'absence direct entre 2 noeuds differents
Saisie la valeur 0 pour la distance d'un a meme noeud
distance arc (0,0)
0
distance arc (0,1)
9999
distance arc (0,2)
9999
distance arc (0,3)
3
distance arc (0,4)
9999
distance arc (0,5)
9999
distance arc (0,6)
0
distance arc (0,7)
9999
distance arc (1,0)
7
distance arc (1,1)
0
distance arc (1,2)
5
distance arc (1,3)
2
distance arc (1,4)
3
distance arc (1,5)
9999
distance arc (1,6)
3
distance arc (1,7)
9999
distance arc (2,0)
9999
```

FIGURE 3.12 – Introduire les données.



```
C:\Users\pc\Desktop\dijkstra\bin\Debug\dijkstra.exe
distance arc (2,0)
9999
distance arc (2,1)
5
distance arc (2,2)
0
distance arc (2,3)
9999
distance arc (2,4)
0
distance arc (2,5)
2
distance arc (2,6)
4
distance arc (2,7)
9999
distance arc (3,0)
3
distance arc (3,1)
2
distance arc (3,2)
9999
distance arc (3,3)
0
distance arc (3,4)
2
distance arc (3,5)
9999
distance arc (3,6)
9999
distance arc (3,7)
9999
distance arc (4,0)
9999
distance arc (4,1)
```

FIGURE 3.13 – Introduire les données.

CHAPITRE 3. PROTOCOLE DE ROUTAGE BASÉE SUR L'ALGORITHME DE DIJKSTRA DANS UN RCSF

```
C:\Users\pc\Desktop\dijkstra\bin\Debug\dijkstra.exe
distance arc (4,0)
9999
distance arc (4,1)
3
distance arc (4,2)
9
distance arc (4,3)
2
distance arc (4,4)
0
distance arc (4,5)
10
distance arc (4,6)
9999
distance arc (4,7)
9999
distance arc (5,0)
9999
distance arc (5,1)
9999
distance arc (5,2)
2
distance arc (5,3)
9999
distance arc (5,4)
10
distance arc (5,5)
0
distance arc (5,6)
9999
distance arc (5,7)
3
distance arc (6,0)
9
distance arc (6,1)
3
distance arc (6,2)
```

FIGURE 3.14 – Introduire les données.

```
C:\Users\pc\Desktop\dijkstra\bin\Debug\dijkstra.exe
distance arc (6,0)
9
distance arc (6,1)
3
distance arc (6,2)
1
distance arc (6,3)
9999
distance arc (6,4)
9999
distance arc (6,5)
9999
distance arc (6,6)
0
distance arc (6,7)
8
distance arc (7,0)
9999
distance arc (7,1)
9999
distance arc (7,2)
9999
distance arc (7,3)
9999
distance arc (7,4)
9999
distance arc (7,5)
3
distance arc (7,6)
8
distance arc (7,7)
0
Donnez le noeud source
```

FIGURE 3.15 – Introduire les données.

Résultat

On donne le noeud source est : s_0 , le programme nous donne la distance minimale de s_0 a tous les noeud de graphe en particulier le noeud s_7 qui est la station de base.

```
C:\Users\pc\Desktop\dijkstra\bin\Debug\dijkstra.exe
distance arc (6,7)
0
distance arc (7,0)
9999
distance arc (7,1)
9999
distance arc (7,2)
9999
distance arc (7,3)
9999
distance arc (7,4)
9999
distance arc (7,5)
3
distance arc (7,6)
8
distance arc (7,7)
0
Donnez le noeud source
0
****resultats****
0 0 distance minimale:0
0 3 1 distance minimale:5
0 3 1 2 distance minimale:10
0 3 distance minimale:3
0 3 4 distance minimale:5
0 3 1 2 5 distance minimale:12
0 3 1 6 distance minimale:8
0 3 1 2 5 7 distance minimale:15
continuer(1/0)?
```

FIGURE 3.16 – Résultat.

on tape 1 pour continue, on donne un autre noeud source s_1

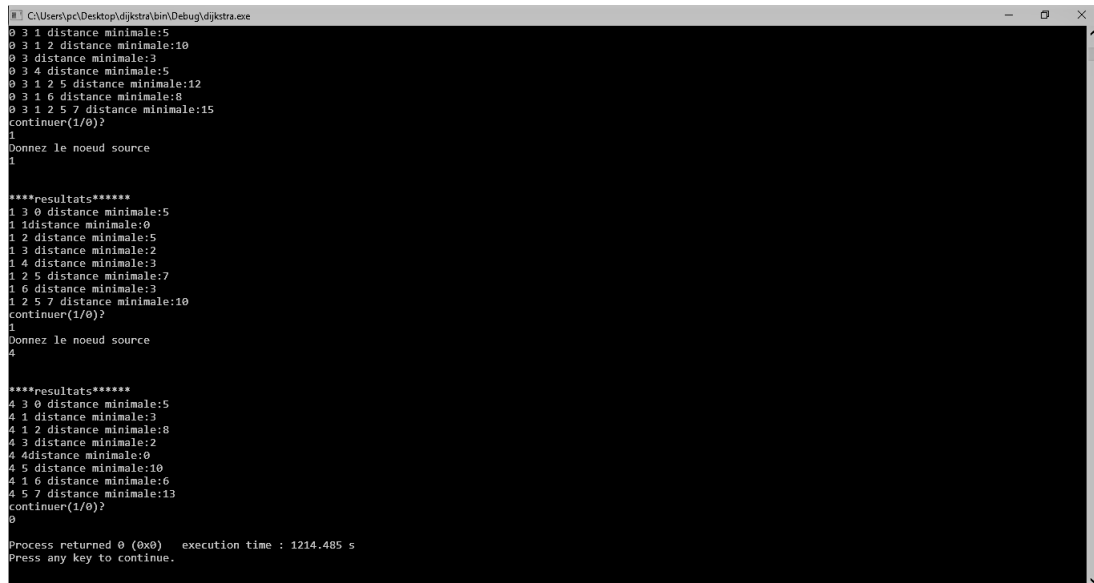
```
C:\Users\pc\Desktop\dijkstra\bin\Debug\dijkstra.exe
****resultats****
0 0 distance minimale:0
0 3 1 distance minimale:5
0 3 1 2 distance minimale:10
0 3 distance minimale:3
0 3 4 distance minimale:5
0 3 1 2 5 distance minimale:12
0 3 1 6 distance minimale:8
0 3 1 2 5 7 distance minimale:15
continuer(1/0)?
1
Donnez le noeud source
1
****resultats****
1 3 0 distance minimale:5
1 1 distance minimale:0
1 2 distance minimale:5
1 3 distance minimale:2
1 4 distance minimale:3
1 2 5 distance minimale:7
1 6 distance minimale:3
1 2 5 7 distance minimale:10
continuer(1/0)?
1
Donnez le noeud source
4
****resultats****
4 3 0 distance minimale:5
4 1 distance minimale:3
4 1 2 distance minimale:8
4 3 distance minimale:2
4 4 distance minimale:0
4 5 distance minimale:10
4 1 6 distance minimale:6
4 5 7 distance minimale:13
continuer(1/0)?
```

FIGURE 3.17 – Résultat.

on tape 1 pour continue

on donne un autre noeud source s_4

si on veut que le processus s'arrête on tape 0.



```
C:\Users\pc\Desktop\dijkstra\bin\Debug\dijkstra.exe
0 3 1 distance minimale:5
0 3 1 2 distance minimale:10
0 3 distance minimale:3
0 3 4 distance minimale:5
0 3 1 2 5 distance minimale:12
0 3 1 6 distance minimale:8
0 3 1 2 5 7 distance minimale:15
continuer(1/0)?
1
donnez le noeud source
1
****resultats****
1 3 0 distance minimale:5
1 1 distance minimale:0
1 2 distance minimale:5
1 3 distance minimale:2
1 4 distance minimale:3
1 2 5 distance minimale:7
1 6 distance minimale:3
1 2 5 7 distance minimale:10
continuer(1/0)?
1
donnez le noeud source
4
****resultats****
4 3 0 distance minimale:5
4 1 distance minimale:3
4 1 2 distance minimale:8
4 3 distance minimale:2
4 4 distance minimale:0
4 5 distance minimale:10
4 1 6 distance minimale:6
4 5 7 distance minimale:13
continuer(1/0)?
0
Process returned 0 (0x0)   execution time : 1214.485 s
Press any key to continue.
```

FIGURE 3.18 – Résultat.

3.7 Conclusion

Ce dernier chapitre est consacré à la modélisation d'un réseau de capteur sans fil par un graphe, sachant que plus le noeud source s'éloigne du noeud puit, plus de consommation d'énergie, et dans le contexte de l'étude de l'énergie et de la façon de maintenir la durée de vie du réseau, on propose une programmation de l'algorithme de Dijkstra sur le langage C++ qui calcule le meilleur chemin ayant un poids minimal, entre la source et la destination comme une solution.

Conclusion générale

Dans ce travail, nous sommes intéressés à la minimisation de l'énergie dans un réseau de capteur sans fil, dans ce dernier les nœuds capteurs sont alimentés par des batteries à faible capacité irremplaçable car ils déploient dans des zones inaccessibles.

Dans un premier lieu, nous avons introduit ce type de réseau, en particulier, nous avons présenté l'architecture, les caractéristiques, les facteurs influant sur la conception des réseaux de capteurs, ainsi que les domaines d'applications de ce genre de réseau. Il a été constaté que la recherche dans les réseaux de capteurs est beaucoup plus orientée vers la conservation de l'énergie afin de prolonger la durée de vie du réseau. A cette fin, le routage est considéré comme l'un des aspects les plus importants à étudier. nous allons mettre l'action sur le routage dans les réseaux de capteurs sans fil.

Notre but est de prolonger la durée de vie du réseau de capteurs en minimisant la consommation d'énergie, sachant que plus le nœud source s'éloigne du nœud puits, plus la consommation de l'énergie est grande dans ce contexte. De nombreux algorithmes, et protocoles ont été proposés dans la littérature pour traiter les problématiques de la minimisation de l'énergie dissipée par les capteurs on a implémenté un protocole de routage basé sur l'algorithme de Dijkstra pour trouver le meilleur chemin ayant un poids minimal entre la source et la destination.

Bibliographie

- [1] A. BENDJEDDOU : Prolongation de la durée de vie des batteries dans les réseaux de capteurs sans fil (RCSF), thèse doctorat 3^{me} cycle LMD, Université Badji Mokhtar-Annaba,(2014/2015).
- [2] Ch.T . KONE : Conception de l'architecture d'un réseau de capteur sans fil de grande dimension, thèse de doctorat Université Henri Poincaré, Nancy I, 2011.
- [3] B. A. KECHAR : Problématique de la consommation d'énergie dans les réseaux de capteur sans fil, mémoire doctorat d'état Université d'Oran, 2010.
- [4] S. MESSAI : Gestion de la mobilité dans les réseaux de capteurs sans fil , thèse de doctorat Université Ferhat Abbas Sétif 1 et l'Université Claude Bernard Lyon 1, 2019.
- [5] Z. KHALILI, M.BOUCHRA : Une technique d'optimisation de la consommation d'énergie dans les réseaux de capteurs sans fil, mémoire master en informatique option : systèmes intelligents université Ahmed Draia,Adrar, 2018/2019.
- [6] S.SAIDANI, S.SOUALMI : Routage à basse consommation énergétique dans les réseaux de capteur sans fil à base de l'algorithme de dijkstra, mémoire de master recherche en informatique, Université de Bejaia, 2014.
- [7] T.YOUCCEF, A. ALI CHERIF, et B.DAACHI : Gestion énergétique dans les réseaux de capteurs sans fil, livre de collection réseaux et télécommunications, 2017.
- [8] J. M. LOBROTON : Systèmes et protocoles de télé-réveil appliqués à l'optimisation énergétique des réseaux de capteurs sans fil, thèse de doctorat, Université de Réunion, 2017.
- [9] I. BOUAZZI : Optimisation d'accès au medium afin d'économiser de l'énergie dans les réseaux de capteurs sans fils, mémoire doctorat Université Monastir, 2018.
- [10] M. BOUALLEGUE : Protocoles de communication et optimisation de l'énergie dans les réseaux de capteurs sans fil, thèse de doctorat, Université Bretagne Loire 2016.

- [11] S. KAISSARI : Conception d'un Réseau de Capteurs Sans Fil, mémoire pour le projet de fin d'études, université Mohammed V école normale supérieur d'enseignement technique, RABAT 2015.
- [12] Y. CHALLAL : <http://creativecommons.org/licenses/by-nc/2.0/fr/>, site.
- [13] L. KHAOULA, M. F. MEGDICHE, C. BELLOUDY, M. ABID, M. AUGUIN : Estimation de la consommation dans les réseaux de capteurs sans fils : étude de cas, article.
- [14] N. MEKKI, K. MOHAMMEDI : Techniques de conservation d'énergie pour les réseaux de capteur sans fil, mémoire de master option : RISR, Université Dr. Tahar Moulay Saida, 2018.
- [15] D. ALLOUACHE, K. AZAMOUM : Optimisation de la consommation d'énergie dans les réseaux de capteurs sans fil, master 2 en informatique, Université de Bejaia, 2014.
- [16] A. SADI : Algorithmes linéaires du nombre de broadcast domination de quelques classes de graphes, mémoire de master 2 en mathématique option : Méthodes et modèles de décision Université Mouloud Mammeri, Tizi-Ouzou 2014/2015.
- [17] I. GUEMARSSA : Reconnaissance des motifs dans des graphes EMF, Mémoire de Fin d'études master filière : informatique 2019 Université de 8 Mai 1945 – Guelma -
- [18] C. BAHLOUL : Appariement de graphes pondérés approche spectrale, mémoire de master Université Dr. Tahar Moulay Saida, 2019.
- [19] IUT Lyon Informatique « Théorie des Graphes » 2011-2012.
- [20] S. TAOUINET : Cours du module, Théorie des Graphes destiné aux étudiantes de licence 2 Recherche Opérationnelle et aide à la décision, Université de Béjaia, (2019/2020).
- [21] O. MARGUIN : Cours d'informatique : « C++ : LES BASES » 2003/2004.

Résumé

Le réseau de capteurs sans fil est un ensemble de noeuds déployés dans une zone de capture pour prélever des grandeurs physiques telles que la température, l'humidité, la pression, etc. Les noeuds capteurs sont alimentés par des batteries généralement irremplaçables et à capacité limitée. Ceci rend l'énergie une ressource critique à conserver pour prolonger la durée de vie du réseau. Dans notre travail, nous avons proposé une solution qui est basée sur le principal défi rencontré par les réseaux de capteurs sans fil est la consommation d'énergie par les noeuds. Par conséquent, une technique d'optimisation de l'énergie doit être en place qui réduit finalement la consommation réelle de puissance des noeuds de capteur. Ici, les algorithmes de plus court chemin sont utilisés dans le routage et il aide à réduire la consommation d'énergie. L'algorithme qui est utilisé pour trouver un chemin optimal c'est l'algorithme de Dijkstra.

Mot clés : Réseau de capteurs sans fils, nœud de capteur, énergie, Protocole de routage, l'algorithme dijkstra.

Abstract

A sensor network is a set of nodes deployed in a capture zone for taking physical values such as temperature, humidity, pressure, etc. The sensor nodes are usually powered by irreplaceable batteries with limited capacity. This makes the energy a critical resource to preserve in order to extend the lifetime of the network. In our work, we have proposed an approach which is based on the main challenge faced by sensor networks is the energy consumption by the nodes. Therefore, an energy optimization technique must be in place that ultimately reduces the actual power consumption of the sensor nodes. Here, shorter path algorithms are used in routing and it helps to reduce power consumption. The algorithm that used to find an optimal path is the Dijkstra algorithm.

Key words : wireless sensor network , Sensor networks, Energy , routing protocol, Dijkstra algorithm.