ДИПЛОМНА РОБОТА
БАКАЛАВРА

ДРБ. ІП - 49.00.00.000 ПЗ

Група ІП-18-1

Шпитко Михайло

2022

Міністерство освіти і науки України
Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

УДК 004.4

ДИПЛОМНА РОБОТА БАКАЛАВРА

Тема: "Створення вебзастосунків стеком MERN (Mongo, Express, React, Node)"

(назва згідно з наказом ректора)

Спеціальність – 121 "Інженерія програмного забезпечення"

ПОЯСНЮВАЛЬНА ЗАПИСКА

ДРБ.ІП - 49.00.00.000 ПЗ

(позначення)

Студент

ІП-18-1 /М.М.Шпитко/
(шифр групи) (підпис) (дата) (розшифровка підпису)

Керівник роботи

проф. /М.С.Пасєка/
(посада) (підпис) (дата) (розшифровка підпису)

Консультанти:

з питань нормоконтролю

доц., к.т.н /В.Я. Піх/
(посада) (підпис) (дата) (розшифровка підпису)

Допускається до захисту
Завідувач кафедри

проф., д.т.н. /В.І.Шекета/
(посада) (підпис) (дата) (розшифровка підпису)

2022
(рік)

Івано-Франківський національний технічний університет нафти і газу

Інститут, факультет інформаційних технологій

Кафедра інженерії програмного забезпечення

Ступінь вищої освіти бакалавр

Спеціальність 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:
Зав. кафедрою **ІПЗ**
проф. **В.І. Шекета**
“ ____ ” _____2022 р.

ЗАВДАННЯ

НА ДИПЛОМНУ РОБОТУ БАКАЛАВРА СТУДЕНТОВІ

Шпитко Михайлу Михайловичу

(прізвище, ім’я, по-батькові)

1. Тема проекту (роботи) “Створення вебзастосунків стеком MERN (Mongo, Express, React, Node)”

керівник проекту (роботи) Пасєка Микола Степанович проф.

затверджені наказом закладу вищої освіти від “25” травня 2022 р. № ____

2. Строк подання студентом проекту (роботи) 14.06.2022 р.

3. Вихідні дані до проекту (роботи) Результати і матеріали отримані під час проходження переддипломної практики

4. Зміст розрахунково - пояснювальної записки (перелік питань, які потрібно розробити)

1. Вступ до проблеми
2. Огляд існуючих концепцій, рішень та сервісів в даній області
3. Побудова моделі або алгоритму власного рішення
4. Документування реалізації власного оригінального рішення вибраними засобами

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

1. Сенсорні вузли (рис. 1.1 ст 10)
2. Оптимальна траєкторія мобільності (рис. 1.3 ст. 24)
3. Тривалість життя мережі в мережі з 100 датчиками (рис. 3.3 ст. 49)
4. Мережа як графік (рис. 2.1 ст 30)

6. Консультанти розділів проекту (роботи)

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання _____

Керівник _____
(підпис)

Завдання прийняв до виконання_____ _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту	Примітка
1	Визначення та обгрунтування теми роботи	17.05.2022	виконано
2	Огляд існуючих концепцій, рішень та сервісів в даній області	18.05.2022	виконано
3	Побудова моделі або алгоритму власного рішення	21.05.2022	виконано
4	Документування реалізації власного оригінального рішення вибраними засобами	25.05.2022	виконано
6	Оформлення пояснювальної записки кваліфікаційної роботи	30.05.2022	виконано

Студент _____
(підпис)

Керівник роботи _____
(підпис)

АНОТАЦІЯ

В дипломній роботі представлена стратегія маршрутизації, яка враховує залишкову енергію датчиків під час встановлення маршрутів. Результати експерименту підтверджують, що запропоновані схеми дозволяють значно подовжити термін служби мережі, порівняно з існуючими методиками. Сенсорні вузли в бездротових сенсорних мережах зазвичай живляться від батарейок і залишаються нерухомими після розгортання. Коли сенсорний вузол закінчує енергію, він більше не забезпечуватиме зондування та обробку даних. Це може призвести до значних втрат у мережі через перерозподіл маршрутів і збій виявлення і звітування про події в середовищі. У цій дипломній роботі ми представляємо дві схеми мобільності приймача: переміщення базового поглинача і маршрутизацію залишкової енергії для продовження терміну служби мережі за сценарієм, що керується випадковими подіями і обчислює оптимальне орієнтовне положення приймального вузла, враховуючи як географічну відстань від датчиків до приймача, так і навантаження на передачу датчиків.

КЛЮЧОВІ СЛОВА: ВЕБ-ЗАСТОСУНОК, МОБІЛЬНИЙ ПРИЙМАЧ, МЕРЕЖА, СЦЕНАРІЙ ПОБУДОВИ, ВУЗОЛ, СЕНСОРНИЙ ДАВАЧ, МАРШРУТИЗАЦІЯ.

ANNOTATION

The thesis presents a routing strategy that takes into account the residual energy of sensors during route establishment. The results of the experiment confirm that the proposed schemes make it possible to significantly extend the service life of the network, compared to existing methods. Sensor nodes in wireless sensor networks are usually powered by batteries and remain stationary after deployment. When a sensor node runs out of energy, it will no longer provide sensing and data processing. This can lead to significant network losses due to route redistribution and failure to detect and report events in the environment. In this thesis, we present two receiver mobility schemes: base sink relocation and residual energy routing for network lifetime extension under a random event-driven scenario that calculates the optimal target position of the receiver node, taking into account both the geographical distance from the sensors to the receiver and the load to transmit sensors.

KEY WORDS: WEB APPLICATION, MOBILE RECEIVER, NETWORK, BUILDING SCENARIO, NODE, SENSOR, ROUTING.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ 7
ВСТУП 8

1 ОГЛЯД ІСНУЮЧИХ ПРОГРАМНИХ РІШЕНЬ ТА СЕРВІСІВ В ОБЛАСТІ ПОБУДОВИ БЕЗДРОТОВИХ МЕРЕЖ ТА IoT 10
1.1 Бездротова сенсорна мережа 10
1.2. Модель межі бездротового давача 11
1.3. Опис проблеми дослідження 14
1.4. Обгрунтування вибаного напряму дослідження 15
1.5. Опис очікуваних результатів дипломної роботи 16
1.6. Структура програмної реалізації 17
1.7 Опис моделі споживання енергії 18
1.8 Мобільність приймача в бездротових сенсорних мережах 20
1.9. Рішення для мобільних вузлів 26
1.10 Алгоритми мобільності поглиначів 30

2 АЛГОРИТМІЧНІ ТА МАТЕМАТИЧНІ РІШЕННЯ ДЛЯ ЗАДАЧ ПОБУДОВИ МЕРЕЖІ 34
2.1. Модель сенсорної мережі 34
2.3. Маршрутизація з урахуванням залишкової енергії (REAR) 43
2.4 Переваги середнього стаку для реалізації веб-сервісу IoT 46

3 РЕАЛІЗАЦІЯ АЛГОРИТМІЧНОГО ТА ПРОГРАМНОГО РІШЕННЯ ЗАДАЧІ ПОБУДОВИ ІЕБ-СЕРВІСУ 48
3.1. Проведення експериментального дослідження 48
3.2. Аналіз результатів та оцінка ефективності 51
3.3 Реалізація API REST для пристрою IoT за використанням MEAN 58

3.4 Розширена реалізація сервісу IoT 61

ВИСНОВКИ 68

СПИСОК ПОСИЛАНЬ НА ДЖЕРЕЛА 69

БІБЛІОГРАФІЧНА ДОВІДКА

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ПЗ - Програмне забезпечення

MERN – Mongo Express React Node

WSN – Бездротова сенсорна мережа

IoT – Інтернет речей

ВСТУП

Термін «Інтернет речей» (Internet of Things, IoT) був запропонований в 1999 році Кевіном Ештоном, одним з трьох засновників Центру автоматичної ідентифікації Массачусетського університету (Auto-ID Center). Існує кілька визначень цього терміна, і кожне з них недостатньо точне. Ми будемо використовувати визначення, запропоноване компанією Gartner (тієї самої, яка придумала термін ERP): «Інтернет речей — це мережа фізичних об'єктів, які мають вбудовані технології, що дозволяють здійснювати взаємодію з зовнішнім середовищем, передавати відомості про свій стан і приймати дані ззовні».

Сучасний Інтернет складається з тисяч корпоративних, наукових, урядових та домашніх комп'ютерних мереж. Об'єднання мереж різної архітектури і топології здійснюється за допомогою протоколу IP. Кожному учаснику Мережі (або групі учасників) присвоюється IP-адреса, постійний або тимчасовий (динамічний).

Аналогічним чином Інтернет речей сьогодні складається з безлічі слабо пов'язаних між собою мереж, кожна з яких вирішує свої завдання. Наприклад, в офісній будівлі може бути розгорнуто відразу кілька мереж: для управління кондиціонерами, системою опалення, освітленням, безпекою і т.д. Ці мережі можуть працювати за різними стандартами, і об'єднання їх в одну мережу представляє собою нетривіальну задачу. Крім того, існуюча (четверта) версія протоколу IP (IPv4) дозволяє використовувати всього лише 4,22 мільярда адрес, через що виникла проблема їх вичерпання. І хоча не кожному апарату, який підключається до Мережі, необхідний унікальний IP-адреса (але все одно необхідний унікальний ідентифікатор), в зв'язку з бурхливим зростанням Інтернету речей проблема нестачі адрес може стати обмежуючим фактором.

Фраза Інтернет речей (IoT) описує поєднання електронних апаратних пристроїв і підключення до Інтернету. Хоча в строгому значенні цей термін

відноситься до гетерогенної мережі взаємопов’язаних пристроїв на практиці багато пристроїв IoT не під’єднані безпосередньо один до одного, а скоріше підключаються до інших служб, які потім надають інтерфейси іншим пристроям або користувачам-людям. Такі пристрої все частіше стають звичайним місцем у будинках, офісах та промисловості: наприклад, розумні термостати або контролери освітлення. В цій роботі досліджується використання стеку інструментів веб-розробки MEAN (названого за його складовими частинами: MongoDb, Express.js, Angular.js і Node.js) для розробки внутрішніх служб і інтерфейсних веб-інтерфейсів для користувача даних, отриманих такими пристроями в IoT.

1 ОГЛЯД ІСНУЮЧИХ ПРОГРАМНИХ РІШЕНЬ ТА СЕРВІСІВ В ОБЛАСТІ ПОБУДОВИ БЕЗДРОТОВИХ МЕРЕЖ ТА IoT

1.1 Бездротова сенсорна мережа

Бездротові сенсорні мережі (WSN) зазвичай використовуються для моніторингу діяльності та повідомлення про події, такі як пожежа, перегрів тощо в певній області чи середовищі.

Сенсорна мережа - система розподілених сенсорних вузлів, взаємодіючих між собою, а також з іншими мережами для запитів, обробки, передачі та надання інформації, отриманої від об'єктів реального фізичного світу з метою вироблення відповідної реакції на цю інформацію. Таким чином, сенсорна мережа включає в себе як мінімум сенсори, актуатори і комунікаційні вузли. Основною областю застосування сенсорної мережі є контроль і моніторинг реальних показників фізичних середовищ і об'єктів та в деяких випадках - управління цими об'єктами (активація в них певних процесів). Приклади сенсорних мереж: всепроникні сенсорні мережі (USN - Ubiquitous Sensor Network), мережі для транспортних засобів (VANET - Vehicular Ad Hoc Network), муніципальні мережі (HANET - Home Ad hoc Network), медичні мережі (MBAN (S) - Medicine Body Area Network (services)) і ін.

Область покриття сенсорної мережі може становити від декількох метрів до декількох кілометрів за рахунок здатності ретрансляції повідомлень від одного елемента мережі до іншого. Сенсорна мережу має здатність до ретрансляції повідомлень по ланцюжку від одного вузла до іншого, що дозволяє в разі виходу з ладу одного з вузлів організувати передачу інформації через сусідні вузли без втрати якості.

У недавньому минулому бездротові сенсорні мережі знайшли свій шлях у широкий спектр застосувань і систем із дуже різними вимогами та

характеристиками [1]. Фактична реалізація бездротової сенсорної мережі широко використовується в багатьох сферах, особливо у військових, біологічних і медичних застосуваннях, екологічних додатках та деяких комерційних застосуваннях. У таких програмах WSN може виконувати зондування в багатьох умовах навколишнього середовища, включаючи наступні [2]:

- температура,
- вологість,
- Автомобільний рух,
- Стан блискавки,
- тиск,
- Грунтовий склад,
- Рівні шуму,
- Наявність або відсутність
- певні види об'єктів,
- механічного напруження прикріплені об’єкти, і Поточні характеристики, такі як швидкість, напрямок і розмір об'єкта

1.2. Модель межі бездротового давача

В основному в інфраструктурі бездротової сенсорної мережі є два компоненти: вузли-приймачі та вузли датчиків. Вузли-приймачі розглядаються як базові станції в мережі, які бездротовим способом отримують і збирають пакети даних, згенеровані з усіх сенсорних вузлів мережі, і надають їх користувачам. З базової станції користувачі можуть отримати доступ до даних, можливо через Інтернет, для подальшої обробки даних та отримання корисної інформації [3]. Залежно від розміру мережі та топології мережі може бути один або декілька вузлів-приймачів, і вузли-приймачі можуть бути стаціонарними в одній позиції або патрулювати в зоні мережі [6]. Вузол-приймач із функціональними можливостями базової станції зазвичай забезпечений великим запасом енергії та великою обчислювальною потужністю , оскільки він працює як стержень у системі сенсорної мережі.

Останні досягнення в області мікроелектромеханічної системи дозволяють сенсорним вузлам бути меншими за виробничою собівартістю, меншими за розміром і багатофункціональними технічно та економічно

доцільними [4]. Сенсорні вузли — це електронні пристрої, які широко розгорнуті по всій мережі, щоб повністю охопити навколишнє середовище, і оснащені сенсорними пристроями, які можуть контролювати різноманітні умови навколишнього середовища.

На додаток до компонентів датчиків, сенсорні вузли також здатні обробляти дані та передавати дані. Робочий процес сенсорних вузлів включає генерацію пакетів даних, які містять інформацію в зоні зондування, і бездротову передачу їх на базову станцію або інші вузли датчика. Через обмеження максимального діапазону передачі пакети даних із сенсорного вузла можуть бути не в змозі дістатися безпосередньо до вузла-приймача. У цьому випадку для пересилання даних до місця призначення потрібні інші вузли датчика. Таким чином, передача даних може залучати кілька сенсорних вузлів, щоб отримати пакет даних і направити їх назад до вузла-приймача.

У цьому сценарії кожному сенсорному вузлу можна призначити подвійні ролі [5] як генератора даних і маршрутизатора даних (іноді його називають вузлом ретрансляції). Сенсорні вузли, які знаходяться ближче до приймача, зазвичай потрібні для пересилання пакетів даних з інших вузлів датчиків, які знаходяться далеко від приймача в топології мережі (як показано на рисунку 1.1).

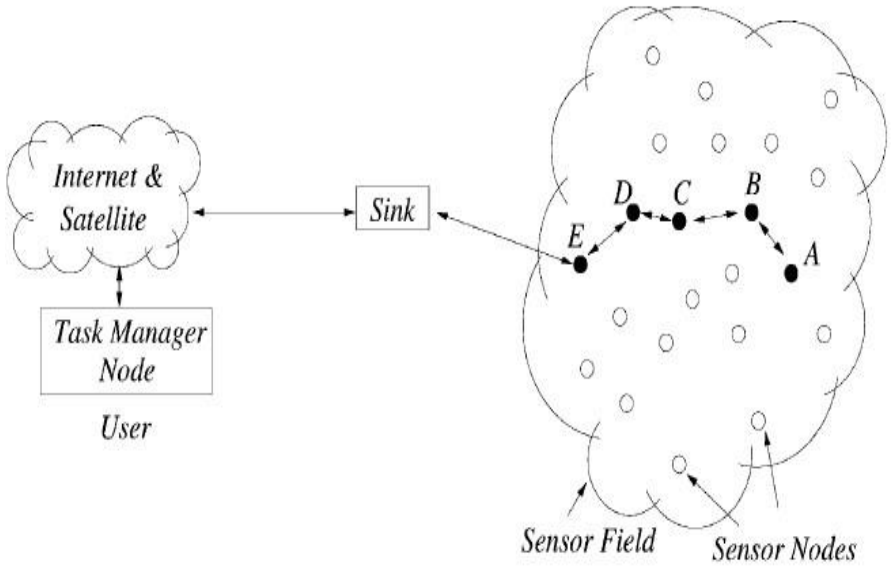


Рисунок 1.1 - Сенсорні вузли, розкидані в сенсорному полі [6]

У бездротовій сенсорній мережі (WSN) датчики розкидані в полі та взаємодіють один з одним бездротовим способом. Проте сенсорні вузли живляться від батареї з обмеженим джерелом енергії. Більше того, порівняно з вузлами стоків, обчислювальна потужність датчика також слабша. Сенсорний вузол споживає енергію від батареї (зазвичай <0,5 Ач, 1,2 В, згідно з [6]), і коли сенсорний вузол закінчує енергію, він більше не може надавати жодних послуг, включаючи зондування, обробку даних або передачу даних. Коли це станеться, датчик вважається « неактивним» і буде видалено з топології мережі. Тривалість життя сенсорної мережі визначається як інтервал часу від її розгортання до моменту, коли загине «критична» кількість сенсорних вузлів, що робить мережу непридатною для використання [7] [8]. Отже, термін служби сенсорного вузла сильно залежить від заряду акумулятора. Невелика частина «неактивних» сенсорних вузлів може безпосередньо вплинути на весь термін служби мережі і, можливо, призвести до величезних втрат у мережі через перерозподіл маршруту та збій виявлення і звітування про події в середовищі. Тому, щоб продовжити термін служби мережі та гарантувати надійність сенсорної мережі, ефективне споживання енергії та енергозбереження мають велике значення в бездротових сенсорних мережах при проектуванні та розгортанні мереж для практичного використання.

Іншим важливим питанням є продуктивність мережі. У деяких середовищах системи сенсорних мереж повинні бути дуже чутливими до зміни деяких умов навколишнього середовища (наприклад, температури реактора на атомній електростанції) і вимагати швидкого реагування на події або явища в навколишньому середовищі. Тому гарантія успішної доставки даних та швидкості обробки та передачі даних відіграє вирішальну роль у наданні надійних послуг зондування. Зазвичай дослідники приймають затримку передачі як вимірювання для оцінки продуктивності та якості обслуговування системи сенсорної мережі, а отже, мінімізації затримки

передачі та максимізації виходу енергоефективним способом також є основною проблемою в дослідницьких роботах.

1.3. Опис проблеми дослідження

Оскільки вузли-приймачі зазвичай забезпечуються більшою енергетичною підтримкою та обчислювальною потужністю, дослідження енергозбереження здебільшого проводяться для мінімізації споживання електроенергії між вузлами датчика та/або реле [6]. За функціями сенсорних вузлів загалом енергоспоживання можна розділити на три області: зондування, зв'язок та обробка даних. З трьох доменів сенсорний вузол витрачає максимум енергії на передачу даних [6]. Це призводить до того, що дослідження в галузі мереж зосереджуються в основному на мінімізації витрат на зв'язок при передачі даних для досягнення оптимальної енергоефективності.

Дані датчика надсилаються з місця виникнення події через проміжні сенсорні вузли до стоку . Згідно з моделлю передачі даних з кількома стрибками (показано на рисунку 1.1), пакети даних надсилаються до вузла-приймача через різні датчики, а датчики, які знаходяться ближче до приймача, повинні отримувати та пересилати дані від інших датчиків, які знаходяться далеко від приймача. Чим ближче до датчик, тим більше даних йому потрібно переслати. Тому для обробки роботи з передачі даних для датчиків, які знаходяться дуже близько до приймача, особливо тих датчиків, які знаходяться лише за один крок від приймача, що означає, що вони можуть передавати дані безпосередньо в пристрій, потрібно багато обчислювальних і комунікаційних ресурсів. вузол . Це призводить до ситуації, коли ці датчики споживають набагато більше енергії і, отже, вичерпують свою енергію набагато швидше, ніж інші. Ці датчики в кінцевому підсумку стають вузьким місцем [1], яке негативно впливає на термін служби мережі. Це

відбувається тому, що велика частина датчиків залежить від цих датчиків «ближче до », і коли вони вмирають, велика кількість даних не може досягти приймача, що призводить до серйозного зниження продуктивності мережі. Ця проблема була визначена та розглянута як проблема «гарячої точки» в ранніх дослідницьких роботах [10], і приклад такої ситуації показано на рисунку 1.2.

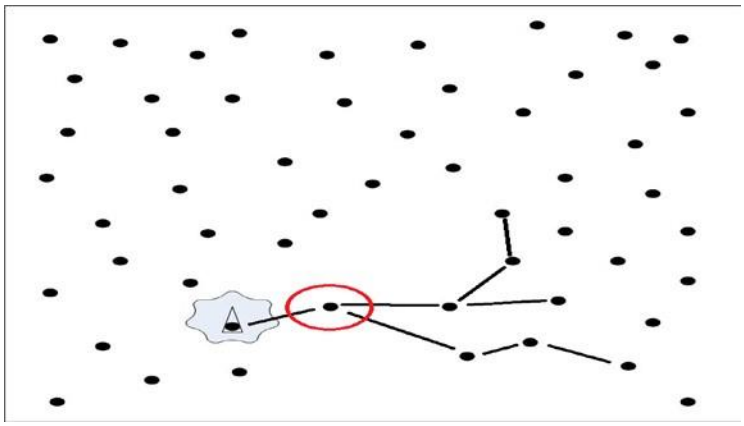


Рисунок 1.2 - Демонстрація «проблеми гарячої точки»

На рисунку 1.2 ми бачимо, що вузол датчика А в колі позначений як сильно навантажений вузол. Відповідно до шляхів передачі даних, які позначені лініями на цьому рисунку, датчик А відповідає за пересилання даних від інших датчиків, і тому розсіювання енергії зосереджено на цьому конкретному датчику.

1.4. Обгрунтування вибаного напряму дослідження

Низка останніх підходів для зниження споживання енергії зосереджується на перенесенні навантаження з датчиків на вузол поглинання [3]. На відміну від класичної моделі системи, коли вузли-приймачі залишаються нерухомими десь у мережі та пасивно отримують дані від датчиків, ми можемо дозволити вузлу-приймачу бути мобільним і переміщатися в зоні мережі, щоб активно шукати датчики, які надсилають

дані та підійти ближче до них. Загальна ідея цього підходу мобільності приймача полягає в тому, щоб перекласти тягар обробки даних і споживання енергії з датчиків на приймач, щоб продовжити термін служби мережі, оскільки вузли-приймачі, як правило, набагато більш продуктивні в обчислювальній потужності та енергозабезпеченні. Крім того, оскільки дальність передачі є важливим параметром для визначення споживання енергії при передачі даних, активні переміщення вузлів-приймачів ближче до активних датчиків призводять до зменшення відстані передачі та меншої кількості проміжних вузлів для передачі даних. Таким чином, споживання енергії має тенденцію до більш рівномірного розподілу в мережі, і проблема «гарячої точки» полегшується, щоб продуктивність мережі могла бути покращена з точки зору терміну служби та якості обслуговування.

Питання максимального терміну служби мережі за допомогою підходу до мобільності приймача включають те, як контролювати переміщення приймача для досягнення найбільш ефективного збору даних, як для гарантування якості обслуговування, так і для зниження споживання енергії. Наприклад, залежно від системних вимог можуть вирішувати підходи щодо мобільності

і) коли перемістити приймач, щоб реагувати на будь-яку подію або зміну в мережі; іі) фактичне положення вузла-приймача та шляхи маршрутизації до приймача; ііі) траєкторія мобільного приймача.

1.5. Опис очікуваних результатів дипломної роботи

У цій роботі представлені дві схеми розміщення вузлів-приймачів у бездротових сенсорних мережах у сценарії, керованому подіями, з одним мобільним приймачем, здатним переміщатися в межах усієї зони зондування. Перша схема зосереджена на виборі оптимального положення для мобільного приймача на основі не тільки розташування випадкових подій, але й навантаження на передачу датчиків. Завантаження датчика відноситься до

загальної кількості даних, що передаються за секунду. Остаточне положення вибирається ближче до найбільш сильно завантажених датчиків, щоб зменшити енергію, що використовується для передачі даних.

Друга схема зосереджена на виборі відповідних шляхів маршрутизації назад до на основі залишкової енергії вузлів датчика. З часом рівень енергії сенсорних вузлів змінюється. Таким чином, запропонована схема зчитує залишкову енергію вузла датчика і має на меті зменшити використання цих датчиків з відносно низькою залишковою енергією, а замість цього вибирає датчики з більш високою залишковою енергією під час обчислення маршрутів. Тому всі датчики можуть жити довше на основі цієї схеми керування маршрутизацією, щоб збільшити термін служби мережі. Метою обох запропонованих підходів є рівномірний розподіл споживання енергії між датчиками та подальше покращення продуктивності мережі з точки зору терміну служби.

1.6. Структура програмної реалізації

Для реалізації ідеї мобільного приймача в системі сенсорної мережі було запропоновано широкий спектр стратегій, протоколів і алгоритмів, більшість з яких демонструють помітні покращення в зоні бездротової сенсорної мережі. Щоб збалансувати розсіювання енергії в мережі, в кількох попередніх роботах було введено багаторівневі сенсорні вузли та ретрансляційні вузли, щоб покращити продуктивність мережі з точки зору включення терміну служби мережі, збалансованої маршрутизації та відмовостійкості [2] [3]. Більше того, низка робіт показала, що використання деяких мобільних вузлів або мобільних колекторів даних (MDC) може значно покращити продуктивність мережі з точки зору терміну служби, покриття та підключення. Розглядаються різні підходи, які переміщують вузол-приймач у межах мережі або навколо неї для збору даних з активних сенсорних вузлів

як на основі часу, так і на основі подій. У цій розглянуто кілька різних тем, включаючи схеми маршрутизації, алгоритми маршрутизації, схеми збору даних, протоколи мобільності приймачів і внесок використання кількох приймачів на основі підходу до мобільності приймача. Ми бачимо, як ці підходи можуть ще більше покращити мережу з точки зору енергозбереження та затримки передачі.

1.7 Опис моделі споживання енергії

Сенсорний вузол, як правило, невеликий за розміром, і можливості вузла датчика з точки зору потужності обробки, пам'яті, зв'язку та енергопостачання обмежені. Сенсорний вузол зазвичай складається з сенсорної схеми, процесора цифрових сигналів і радіоприймача. Комунікаційні частини датчика відповідають за більшу частину споживання енергії. Для розрахунку розсіювання енергії при бездротовій передачі в цій роботі використовується модель розсіювання радіоенергії, представлена в [3] і [5], як показано на рисунку 1.3.

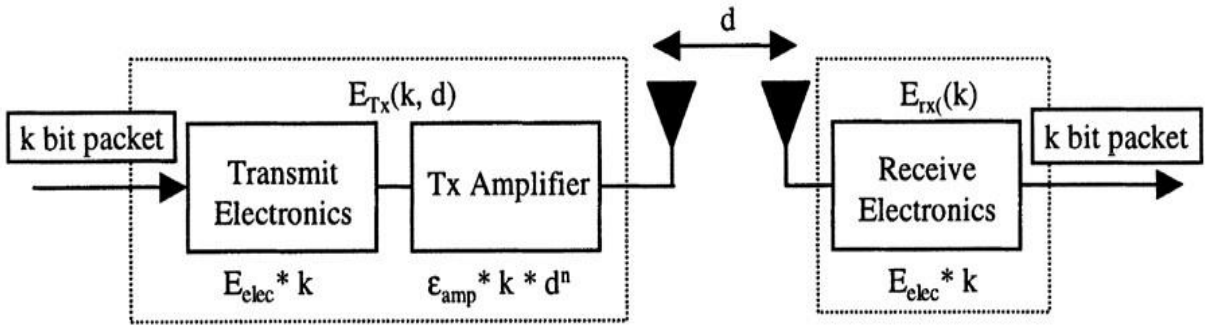


Рисунок 1.3 - Модель розсіювання радіоенергії [3]

У цій моделі енергія використовується схемою передачі/приймання та підсилювачем потужності. Енергія, що витрачається на отримання k біт даних, становить:

$$Приймаюча_енергія(k) = k \cdot E\ ел \qquad (1)$$

Де E elec - це константа, що представляє енергію, яка використовується в схемі для передачі або отримання одного біта даних.

Енергія, що витрачається на передачу k біт даних від датчика u до v

$$Енергія\ передачі\ (k) = k \cdot [E\ elec + E\ amp \cdot d\ q\ (u,v)] \qquad (2)$$

Де E amp є константою представляє енергію, використану в підсилювачі потужності для передачі одного біта даних, а d(u,v) позначає відстань між вузлом u та v .

У цій дипломній роботі ми припускаємо, що відповідні протоколи на фізичному рівні та рівні MAC використовуються для забезпечення надійної передачі даних.

Для оцінки продуктивності WSN в експериментальному моделюванні можна використовувати кілька показників продуктивності. Одним із важливих показників є термін експлуатації мережі . Термін життя сенсорної мережі зазвичай визначається як проміжок часу від початку роботи мережі до часу, коли кількість критичних вузлів «вмирає» (тобто джерела живлення критичних вузлів вичерпуються до такого рівня). ступінь, що мережа більше не залишається корисною) [7] [8]. У літературі використовується ряд різних показників для вимірювання тривалості життя сенсорної мережі [5] [6]. У [5] час життя сенсорна мережа визначається як мінімум

я) час, коли відсоток живих вузлів (тобто вузлів, батареї яких не розряджені) падає нижче визначеного порогу, ii) час, коли розмір найбільшого підключеного компонента мережі падає нижче визначеного порогу, і iii) час, коли покриття падає нижче визначеного порогу.

У роботі [2] автори зосереджуються на охопленні та розглядають час життя мережі як період, протягом якого може бути охоплений весь регіон.

тобто час, коли деякі сенсорні вузли більше не можуть направляти свої дані до стоків.

Деякі дослідники використовували нормалізовані накладні витрати на пересилання та коефіцієнт доставки пакетів [5], щоб оцінити їхній внесок у збір даних. Нормалізовані накладні витрати на пересилання використовуються для перегляду накладних витрат даних, створених для пошуку маршрутизації. Коефіцієнт доставки пакетів являє собою загальну кількість отриманих даних, поділену на загальну кількість даних, переданих датчиками. Цей коефіцієнт показує швидкість доставки даних.

У роботі [1] автори використовують середню енергію на пакет для вимірювання енергоефективності. Також пропускна здатність використовується для визначення масштабованості мережі. У сценаріях, заснованих на часі, середня затримка доставки вводиться для оцінки продуктивності їхніх чотирьох протоколів і вивчення переваги недоліків реалізації. У сценаріях на основі подій, таких як [6], можна виміряти тривалість життя мережі на основі навантаження мережі. Ідея полягає в тому, що чим більше навантаження на вузол, тим менше часу він зможе працювати. Для мережевої затримки можна використовувати середню кількість стрибків для вимірювання загальної затримки передачі в мережі.

1.8 Мобільність приймача в бездротових сенсорних мережах

У [7] автори вперше представили концепцію використання мобільних для збалансування споживання енергії. У цій галузі виконано ряд робіт, наприклад [20]. Існують 3 основні частини, залучені до впровадження мобільності приймача до бездротових сенсорних мереж для підвищення продуктивності мережі: переміщення вузла-приймача, маршрутизація пакетів даних і збір даних.

Рух приймального вузла

У [2] автори запропонували метод обчислення розташування базової станції з використанням цілочисельного програмування для продовження терміну служби мережі та пропускної спроможності даних з урахуванням базових станцій, розташованих на межі зони мережі. Також у [2] оптимальні положення вузлів ретрансляції обчислюються за допомогою ILP для продовження терміну служби. Для економії енергії та збору даних датчиків у розріджених сенсорних мережах була запропонована модель збору даних з використанням Data MULE [3] у поєднанні зі стратегією Random Walk [4]. Ця модель може досягти зменшеного діапазону передачі для датчиків для економії енергії передачі. Однак, оскільки ця модель не може передбачити, коли вузол-приймач рухається до тих датчиків, готових надіслати дані, пакети даних можуть значно затримуватися.

У [5] та [6] автори пропонують модель мобільності приймача на основі протоколу. За словами авторів, їхня робота є однією з перших спроб запровадити мобільний приймач для ефективної доставки даних і підвищення надійності мережі.

Мобільність приймача здебільшого вказує на те, що система ставить більше навантаження на вузол приймача замість того, щоб зосередитися на вузлах датчиків, оскільки вважається, що вузол приймача має необмежене джерело живлення та набагато більші можливості обробки, ніж вузли датчика. Однак те, як обійти всю мережу, також є важливим питанням, оскільки невідвідування деяких областей потенційно призведе до втрати даних. Крім того, необхідно ефективно використовувати енергію при переміщенні вузла .

На основі підходу випадкового блукання в [5] представлено 4 підходи, які можуть вписуватися в різні типи сценаріїв застосування:

Випадкова прогулянка та пасивний збір даних

Часткова випадкова прогулянка з обмеженим поширенням даних із кількома стрибками). Упереджена випадкова прогулянка з пасивним збором даних iv) Детермінована прогулянка з розповсюдженням даних із кількома стрибками

Кожен із вищезазначених підходів має як переваги, так і недоліки, залежно від системних вимог програми. Автори в [5] стверджують, що їхня робота є першим кроком до мобільності стоків у WSN.

Маршрутизація пакетів даних

Як згадувалося в попередніх розділах, енергоефективність є важливою темою при проектуванні WSN. Споживання енергії значною мірою обумовлено передачею даних - як від датчика до датчика, так і від датчика до . Тому ефективний шлях передачі покращить використання енергії в системі та заощадить більше енергії. У [7] автори пропонують підхід до маршрутизації, який приймає рішення щодо маршрутизації локально, а при виборі сусіднього вузла залежить від двох основних факторів: середньої енергії та відстані передачі. У роботі [8] запропонована унікальна схема обробки передачі в системі мобільної мережі-приймача з функцією швидкого відновлення.

У [9] автори представляють свій протокол мобільної маршрутизації для досягнення енергоефективної маршрутизації. Оглядаючи LEACH [7] і LEACH-C [3], вони вказують на те, що ці дві класичні методи маршрутизації на основі кластера не завжди можуть вибрати головку кластера так, щоб гарантії продуктивності були дотримані. На основі LEACH [7] і LEACH-C [3] автори в [9] запропонували свій протокол маршрутизації, який дає змогу мережевій системі розгортати саму головку кластера та покращувати мережу з точки зору енергоефективності. Загальний протокол можна розділити на 2 частини: схема планування попередньої оцінки та схема маршрутизації.

У схемі планування черга розкладу встановлюється на основі середньої залишкової енергії датчиків і відстані від поточної головки кластера до іншої головки. Вся інформація зберігається в таблиці маршрутизації . У схемі маршрутизації протокол встановлює мережу за допомогою LEACH і вперше встановлює чергу планування шляхом випадкового обходу до приймача. На основі середнього споживання енергії протокол обчислює період часу T. Після часу T протокол оновлює таблицю планування і T і знову повторює попередні кроки. Автори в [9] створили самоадаптивний протокол для покращення продуктивності LEACH таким чином, щоб можна було гарантувати енергоефективність, оскільки черга SC heduling базується на середній залишковій енергії датчиків.

У [11] автори пропонують інший протокол маршрутизації на основі кластера, Mobile Sink Based Routing Protocol (MSRP) для вирішення проблеми гарячої точки . Щоб вирішити проблему гарячої точки, MSRP також об'єднує мережу та реєструє головку кластера на основі інформації про залишкову енергію. Динамічна архітектура мобільного приймача в MSRP дозволяє уникнути концентрації енергії на невеликій частині датчиків і подовжує термін служби мережі. Крім того, MSRP також встановлює стратегію руху для . Виходячи з залишкової енергії головок кластерів, приймач завжди буде ближче до головки кластера з вищою залишковою енергією, і ці головки кластера будуть передавати дані з інших головок кластерів.

Збір даних

Проблема збору даних у розріджених сенсорних мережах зустрічається в багатьох сценаріях, таких як моніторинг фізичного середовища, міграції тварин у віддалених районах, погодні умови в національних парках, моніторинг середовища проживання на віддалених островах, моніторинг міського руху тощо [3]. Мета — зібрати дані з датчиків і доставити їх до точки доступу в інфраструктурі [3].

Одна з важливих проблем у реалізації мобільних вузлів-приймачів у Wireless Sensor мережі – це те, як приймач збирає дані від статичних сенсорних вузлів, коли вузол приймача рухається. Оскільки розташування змінюється, сенсорні вузли мають можливість надсилати пакети даних до , коли вузол знаходиться поблизу. Тому традиційні схеми збору та маршрутизації даних у цьому випадку не підходять. У [3] автори представляють аналітичну модель для розуміння ключових показників продуктивності, таких як передача даних, затримка до місця призначення та потужність. У [5] автори підсумували попередні роботи, вказавши, що основним недоліком як Data MULE, так і Random Walk є затримка доставки. Щоб підвищити продуктивність збору даних, автори в [5] представили свої два протоколи збору даних: AVRP і TRAIL і стверджували, що ці два протоколи забезпечують ефективний збір даних, що зменшує затримку доставки в мережах із важкими та невелике навантаження відповідно.

Протокол AVRP є похідним від попереднього протоколу маршрутизації Voronoi-Scoping [3] на основі якоря. Автор розділяє мережу на кілька кластерів. Він ефективний у тому, що датчикам потрібно лише відправити свої дані до найближчих вузлів прив'язки, а прив'язні вузли спілкуються з вузлом-приймачем. Щоб адаптуватися до мобільного стоку, автори вдосконалили Voronoi-Scoping, які пов'язують мобільну з опорними вузлами. AVRP створює структуру доставки та періодично оновлює структуру на основі руху . Тому всім датчикам потрібно зберігати лише інформацію про маршрутизацію найближчого вузла прив'язки, а AVRP усуває потребу в інформації про шлях динамічної маршрутизації, яка є великими витратами на передачу даних. AVRP стабілізує передачу даних і підходить для мереж з великим навантаженням.

Автори в [5] представляють у своїй роботі ще один протокол, TRAIL. Загальна ідея полягає в тому, що протокол фіксує слід руху стоків, і маршрут збору даних буде проходити по цьому сліду. Протокол починається з

випадкового блукання, і як тільки приймач виявляє запит на передачу даних від датчиків, він рухається до них, а сліди записуються та оновлюються для наступного маршруту збору даних. Приклад ілюстрації наведено на рисунку 1.4

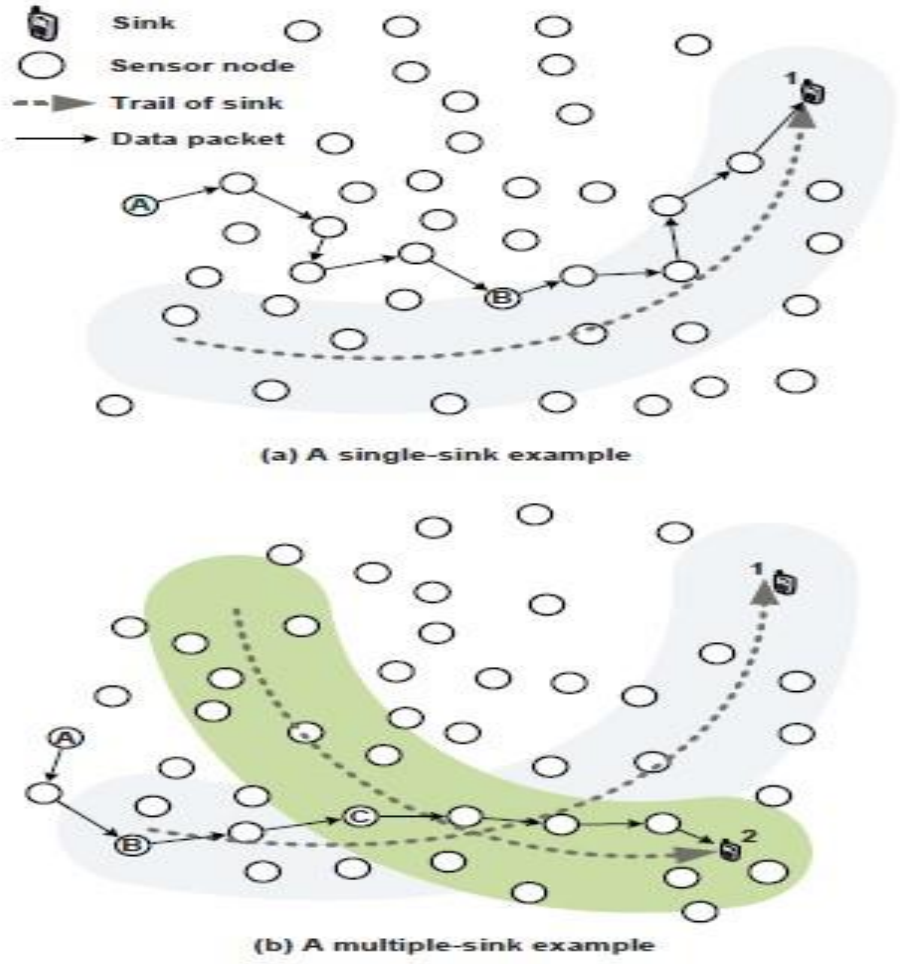


Рисунок 1.4 - Приклад того, як дані пересилаються на прийом у TRAIL

Як показано на рисунку 1.4 (a), вузол датчика A випадковим чином передає дані до датчика B, оскільки не знайдено жодної мобільної стокової доріжки. Коли дані досягають датчика B, який запам’ятовує слід руху стоків, він направляє пакети даних уздовж сліду вузла-приймача, і пакети досягають приймача. На рисунку 1.4 (b) пояснюється, коли інший новий шлях

мобільного приймача перехоплює передачу даних до вузла-приймача 1 і направляє їх до приймача 1.

У реальній реалізації TRAIL автори формалізували проблему в задачу оптимізації, яка вирішує оптимальний маршрут уздовж сліду стоку для підвищення продуктивності. Згідно з претензією, цей протокол підходить для мереж з невеликим трафіком через дуже низькі витрати пакетів даних. І AVRP, і TRAIL прагнуть зменшити проблему затримки доставки в попередніх методах.

1.9. Рішення для мобільних вузлів

Мобільний підхід, орієнтований на події

Відповідно до фактичного застосування WSN, велика кількість сценаріїв використовується для моніторингу діяльності в зоні зондування та повідомлення про будь-які події, коли це необхідно. Тому підхід, керований подіями, набагато більше підходить для цих сценаріїв, ніж підхід, керований часом [4], коли датчики періодично надсилають дані в приймач. У [4] автори вводять шаблон, керований подіями, у WSN з мобільним приймачем. А пізніше автори підвищили продуктивність WSN, що керується подіями, дослідивши та визначивши шлях і траєкторію для руху стоку. Робота заснована на Data MULEs [3] з підходом випадкового блукання, де навантаження передачі мережі та середня кількість стрибків використовуються для оцінки продуктивності підходу. Після математичного аналізу автори стверджували, що траєкторія квадрата, повернутого на 45 градусів з центром O, є оптимальною, що показано як Traj A на рисунку 1.3: Оптимальна траєкторія рухливості [4]

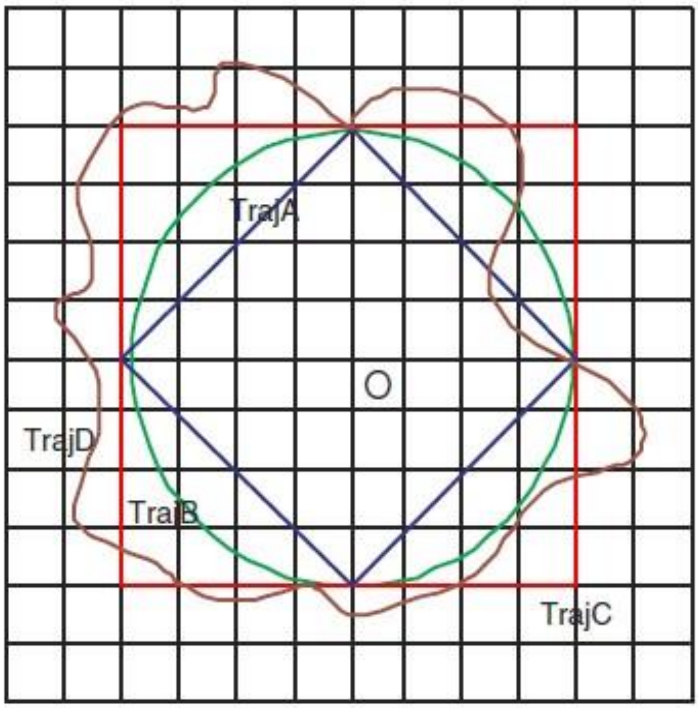


Рисунок 1.3 - Оптимальна траєкторія мобільності

Завдяки переміщенню вузла приймача на довгу оптимальну траєкторію продуктивність системи збільшується з точки зору терміну служби та затримки доставки.

У [6] автори обговорюють внесок розгортання кількох мобільних приймачів в одній мережі за сценарієм, керованим подією. Модель системи та показники продуктивності залишаються незмінними. Автори також використовують математичний аналіз, щоб визначити покращення продуктивності з точки зору навантаження мережі та середньої кількості стрибків. Вони стверджують, що використання більш ніж однієї , яка рухається в зоні сприйняття, може зменшити навантаження, оскільки передача даних розподіляється більш рівномірно. Але середній хміль збільшується, тому що середній радіус збільшується. Загалом, він робить висновок, що зі збільшенням мобільних у системі продуктивність може бути покращена. Однак покращення зменшується, якщо розгорнути занадто багато вузлів.

Енергетично збалансована модель

У роботі [5] автори запропонували свій підхід до вирішення проблеми втрати зв'язку через енергетичні обмеження. Вони зазначають, що фактична продуктивність використання підходу чистого мобільного приймача викликає сумніви, оскільки використання обчислювальних ресурсів, таких як розмір буфера та ресурси передачі, не оптимізовано. Крім того, оскільки місцезнаходження неможливо передбачити заздалегідь, датчики, ймовірно, повинні надсилати дані в , яка на даний момент знаходиться далеко, або чекати, поки вузол піде до датчиків. Це може призвести до додаткового неефективного споживання енергії та значної затримки доставки. Ідея полягає в тому, що система може використовувати як фіксовану, так і мобільну . Датчики можуть надсилати повідомлення або до приймача поблизу, або до наступного вузла ретрансляції та направляти дані до стаціонарного приймача. Згідно з їхнім твердженням, продуктивність гібридного підходу перевищує класичні підходи з фіксованим стоком у [6] та [7] у міру збільшення часу моделювання.

Інтелектуальна схема маршрутизації

Це особливий підхід до реалізації ієрархічних структур у WSN з мобільними приймачами. У роботі [8] автори вказують на те, що коли поглинач рухається в мережі, датчикам потрібно часто оновлювати розташування стока, що призводить до надмірного споживання електроенергії. Автори стверджують, що було проведено дуже мало досліджень щодо встановлення кількох рівнів у системі WSN. Одним з них є дворівневе поширення даних (TTDD) [9]. Автори стверджують, що TTDD добре працює в системах, керованих подіями, але не дуже добре в системах, де розташування стоків змінюється дуже часто, оскільки TTDD потрібно щоразу змінювати структуру.

Щоб вирішити це, новий рівень вузлів агента , відповідальних за збір даних, вводиться між шаром вузла-приймача і шаром вузла датчика. Спочатку приймач надсилає широкомовне повідомлення, щоб визначити місцезнаходження вузлів агента, а потім, шляхом заповнення, усі вузли датчиків отримують інформацію про вузли агента та вирішують наступний стрибок передачі даних. Приймач збирає дані з вузлів агента під час подорожі в мережі. Рисунок 1.4 ілюструє схему передачі даних на основі агента

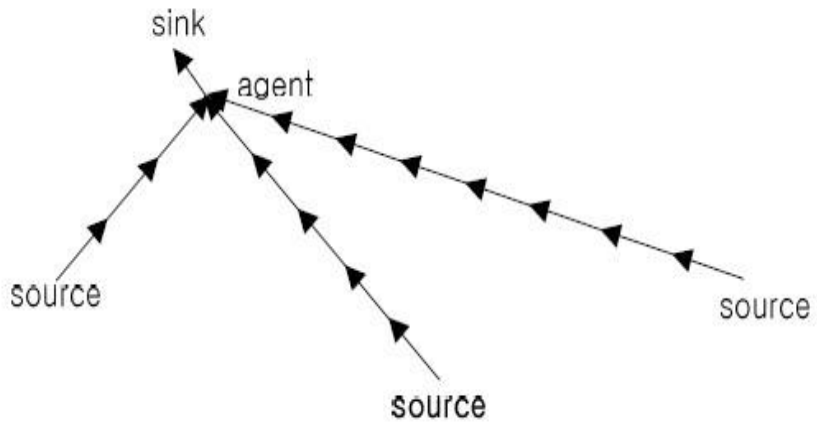


Рисунок 1.4 - Збір даних на основі агента

Проте оптимальність цієї схеми може бути погіршена в міру переміщення поглинача, оскільки приймач може переміщатися ближче до датчиків, ніж відповідні вузли агента. Тому пропонується схема вибору шляху, де вузол датчика зв’язується зі своїми найближчими вузлами, які називаються негайним ретранслятором (IP), а потім ІЧ передають свої відстані один від одного. Це робиться рекурсивно, щоб обчислити загальну відстань до стоку в порівнянні з вихідною довжиною шляху на основі агента. Схема вибере оптимальний шлях на основі загальної відстані, як показано на рисунку 1.5 нижче. Оптимальний шлях – надсилати дані безпосередньо в

приймач замість використання агентів, оскільки загальна відстань менша, ніж шляхи, які використовують агенти для пересилання даних.

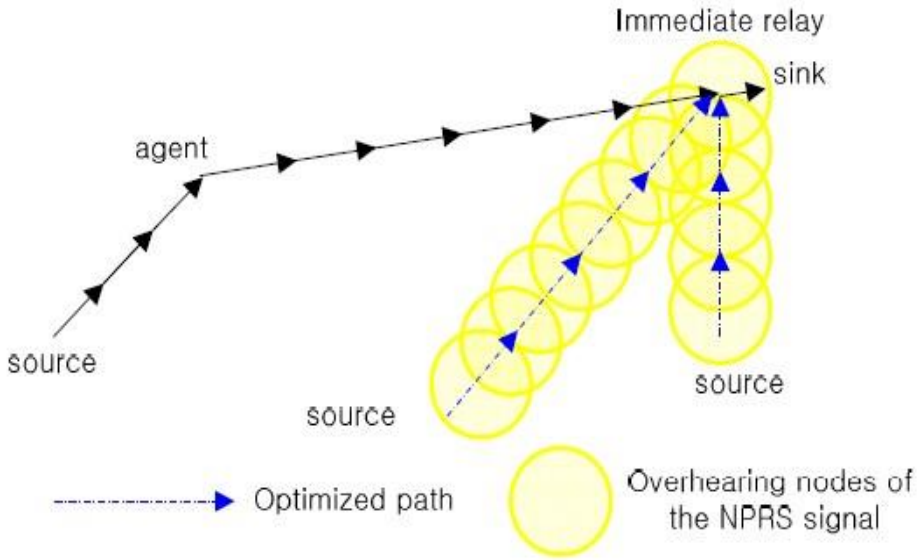


Рисунок 1.5 - Вибір оптимального шляху

1.10 Алгоритми мобільності поглиначів

Розподілений алгоритм у маршрутизації

Досягнення максимального терміну служби WSN можна досягти шляхом встановлення ефективної схеми маршрутизації. Для того, щоб ввести мобільність приймача в WSN, класичні алгоритми маршрутизації [4] більше не підходять, оскільки топологія мережі постійно змінюється через переміщення приймача. У [4] використовується алгоритм балансування навантаження для рівномірного розподілу навантаження між усіма вузлами. У [3] автори пропонують спільну мобільність і алгоритм маршрутизації та показують, що краща стратегія маршрутизації досягається за допомогою комбінації круглих і коротких маршрутів. У [4] автори пропонують

розподілений алгоритм маршрутизації для максимального часу життя в WSN, на відміну від централізованих підходів у [5], [6].

Розподілені алгоритми в допуску затримки.

Допуск до затримок є ще одним важливим питанням серед тем дослідження WSN щодо вимог до застосування. У роботі [7] автори пропонують структуру, яка реалізує підхід мобільного приймача з урахуванням вимог щодо допуску затримки. Ця модель допускає певний рівень затримки доставки даних і гарантує якість обслуговування. Модель називається мобільною поглиначною моделлю з толерантністю до затримок (DTMSM). Пізніше в [8] автори вводять ефективний розподілений алгоритм для реалізації DT-MSM, щоб максимізувати термін служби мережі в системі та підвищити продуктивність датчиків і мобільних приймачів.

Енергоефективна стратегія

У роботі [5] автори враховують виникнення непередбачуваної події та пропонують стратегію енергоефективності (EES) у системі сенсорної мережі, що керується подіями, з мобільністю приймача. Діяльність датчиків базується на випадкових подіях, які не відомі заздалегідь. Коли відбувається нова подія, вона запускає датчики в області події, щоб почати надсилати дані для повідомлення про подію. Інші датчики, які знаходяться за межами зони події, залишаються на утриманні та не надсилають жодних датчиків (для цієї події). EES — це самоадаптивний алгоритм для вибору попереднього положення для вузла-приймача. Так вирішує EES куди перемістити вузол приймача коли перемістити вузол.

Загальна ідея EES полягає в тому, що вузол-приймач рухається до активних датчиків, щоб зменшити загальну відстань передачі для всіх датчиків у відповідь на початок нової події або закінчення існуючої події. Показано, що цей підхід значно покращує термін служби мережі в порівнянні

з мережами зі стаціонарним вузлом-приймачем, а також мережами з мобільним приймачем, використовуючи підхід випадкового блукання. Однак EES враховує лише розташування уражених сенсорних вузлів і не враховує їх навантаження чи залишкову енергію.

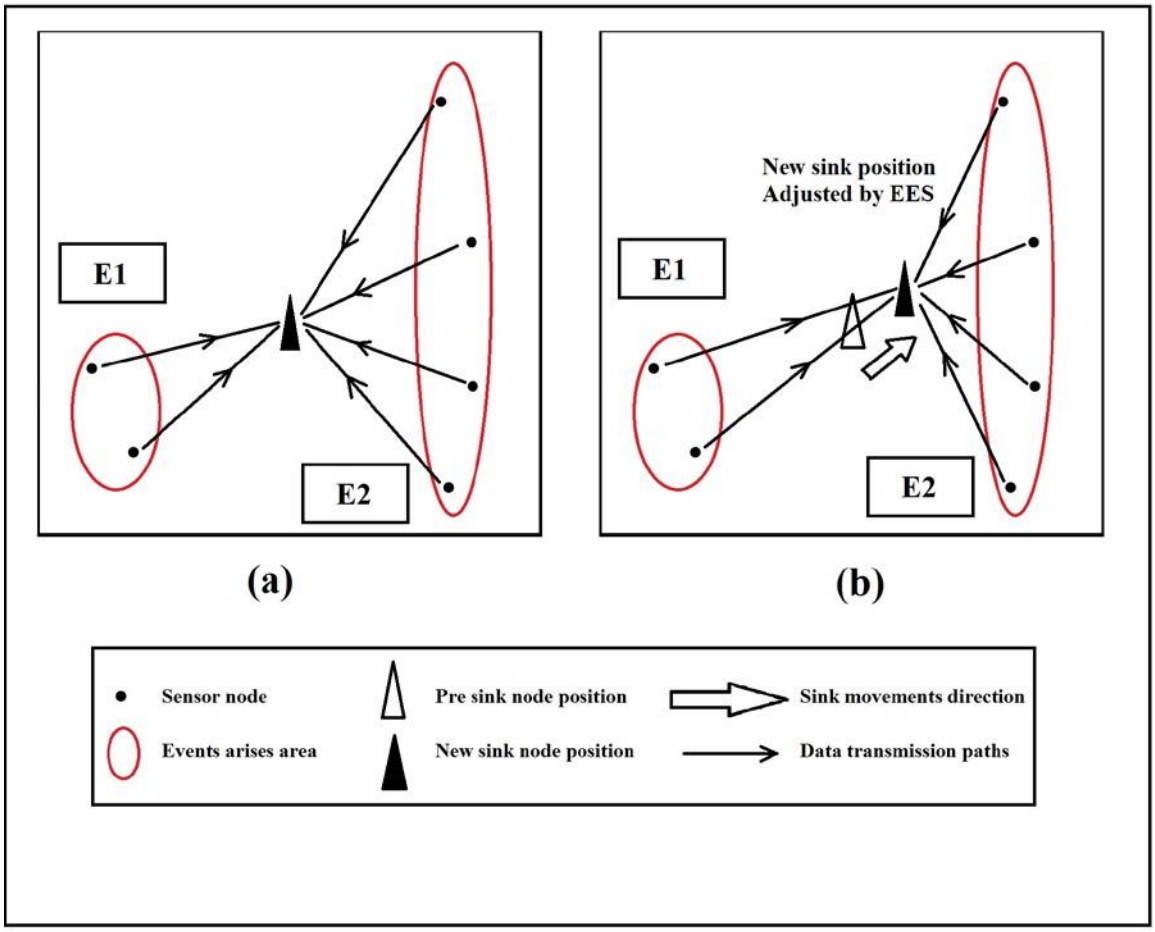


Рисунок 1.8 - Розташування вузла-приймача лише на відстані

Рисунок 1.8 ілюструє цю концепцію, використовуючи дві випадкові події E1 і E2, які виникають у мережі, активуючи два сенсорні вузли і чотири сенсорні вузли відповідно. Оскільки подія E2 охоплює більшу площу та має більше активних датчиків, вона генерує більше даних (за умови, що розмір пакета даних, швидкість передачі тощо для всіх датчиків однакові). За цих обставин вузол повинна наблизитися до події E2, як показано на рисунку 1.8.

У реальній реалізації мережевого процесу одночасно може відбуватися кілька подій. Оптимальне положення приймача має бути ближче до тих подій, які охоплюють більшу площу, оскільки вони запускають більше датчиків для звітування про події і, отже, генерують більше даних для передачі до вузла приймача. Виходячи з наведених вище міркувань, оптимальне положення , р оптимальне = (X s , Y s) визначається такими рівняннями:

$$X\ s = \sum_{vv \in EE} |EE| xx\ vv \qquad (3)$$

$$Y\ s = \sum_{vv \in EE} |EE| yy\ vv \qquad (4)$$

Тут E — множина всіх активних сенсорних вузлів у мережі, а (x v , y v — координати сенсорного вузла $vv \in EE$.

2 АЛГОРИТМІЧНІ ТА МАТЕМАТИЧНІ РІШЕННЯ ДЛЯ ЗАДАЧ ПОБУДОВИ МЕРЕЖІ

У цій ми спочатку опишемо модель нашої системи, включаючи топологію мережі та схеми зв'язку, а потім представимо ряд нових схем розміщення стоків. На відміну від існуючих підходів, запропонований нами підхід враховує як навантаження на вузли датчиків, так і рівні їх залишкової енергії.

2.1. Модель сенсорної мережі

Структура вузла датчика

Кожен окремий сенсорний вузол в нашій архітектурі може сприймати та/або пересилати дані або залишатися неактивним, і фактичний статус вузла зазвичай змінюється кілька разів протягом усього терміну служби мережі. Зміна статусу може відбутися як відповідь на початок/кінець випадкових подій або через рівень енергії у вузлі. Ми визначаємо наступні 4 умови стану для кожного вузла датчика:

Активний : Вузол датчика працює і генерує дані

Ретрансляція : Вузол датчика працює, і він пересилає дані з інших датчиків

Неактивний : Вузол датчика працює, але він не генерує та не пересилає дані

: Вузол датчика вичерпав свою енергію і не працює

Кожен вузол датчика ідентифікується унікальним SensorID. Ми записуємо SensorID кожного вузла датчика, а також інші атрибути, такі як його положення, тобто координати (x , y), статус і залишкова енергія, у таблицю, як показано в таблиці 2.1. Під час процесу роботи мережі всі вузли, включаючи датчики та приймач, можуть отримати доступ до цієї таблиці, щоб отримати інформацію

про стан датчиків і налаштувати підхід до мобільності приймача та динамічні шляхи маршрутизації. Таблиця 2.1 демонструє структуру таблиці стану.

Таблиця 2.1 - Таблиця стану та атрибутів усіх сенсорних вузлів [5]

Ідентифікатор датчика	координата X	Y координата	Активний?	Сенсори?	Залишкова енергія

Положення датчиків фіксуються під час початкового розгортання, але розташування змінюється з часом, залежно від настання подій. Атрибути «Living» і «Sensing» є двійковими значеннями, ініціалізованими відповідно до 1 і 0 для кожного сенсорного вузла під час розгортання, щоб уявити, чи генерує він дані. Якщо датчик не активується подіями, але пересилає дані з датчиків, значення «Sensing» залишиться 0. Розмір даних та споживання енергії будуть записані окремо. Коли відбувається нова подія , певні вузли датчиків будуть активовані і почнуть передавати дані. Коли залишкова енергія вузла досягає 0 або нижче порогового значення, це вказує, що цей вузол датчика вичерпав свою енергію, і значення «Живий» буде встановлено на 0, що означає, що він більше не працює. У цьому випадку вузол не буде братися до уваги для виявлення подій, формування даних або пересилання даних у майбутньому. Поле «Залишкова енергія» записує, скільки заряду акумулятора залишилося у вузлі датчика. Значення в цьому полі безперервно зменшується від початкового значення, оскільки вузол бере участь у різних діях виявлення та пересилання.

Топологія мережі

Наша конфігурація мережі схожа на модель, представлену в [8]. Він складається з набору однорідних сенсорних вузлів, які випадковим чином розподілені в межах мережі. Кожен сенсорний вузол на початку постачається обмеженою кількістю енергії, і є один (мобільний) вузол-приймач, який не

обмежений по енергії. Усі дані, згенеровані вузлами датчика, в кінцевому підсумку надходять до вузла-приймача.

Зона інтересу являє собою двовимірну квадратну область, всередині якої розташовані всі сенсорні вузли. Приймальний вузол може вільно переміщатися в будь-якому місці цієї області. На рисунку 2.1 показана типова топологія мережі, де вузли датчиків представлені чорними точками, а стік — трикутником.

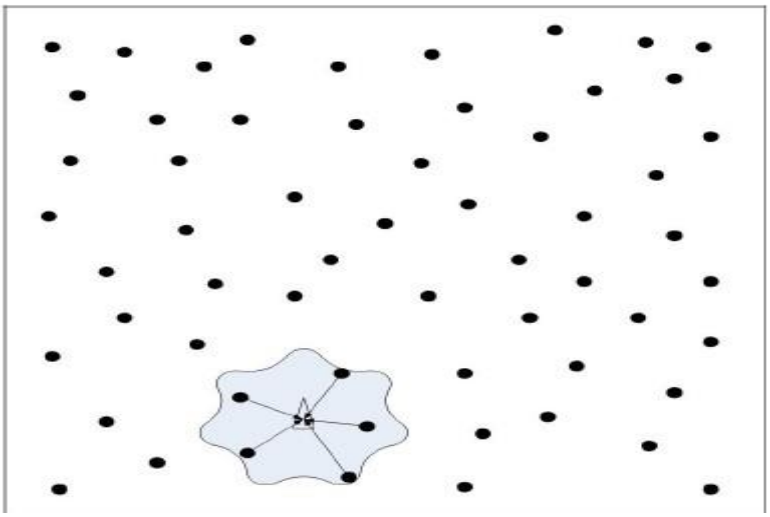


Рисунок 2.1 - Мережа як графік [5]

Маршрутизація

Після визначення положення приймача датчики використовують багатоскапкову передачу даних для бездротової передачі пакетів даних назад до приймача. У літературі показано, що багатострибкова передача зазвичай споживає менше енергії, ніж пряма передача на великі відстані в бездротовому зв'язку з приймачем [6]. Крім того, у випадках, коли приймач знаходиться за межами максимального діапазону передачі сенсорного вузла, єдиною можливою альтернативою є багатострибковий зв'язок.

Щоб налаштувати шляхи маршрутизації до приймача для кожного сенсорного вузла, ми використовуємо стандартний алгоритм найкоротшого

шляху Дейкстри [5], припускаючи, що вузол-приймач є коренем. Вартість кожного краю може бути встановлена відповідним чином, для різних підходів. Наприклад, у нашій першій схемі вартість ребра e (u v) береться за квадрат відстані між вузлами u і v . Чим далі відстань між двома вузлами, тим вище вартість ребра e (u v). У другому підході, з іншого боку, вартість краю залежить як від відстані, так і від залишкової енергії вузла призначення.

Крім ініціалізації, нові шляхи маршрутизації необхідно обчислювати щоразу, коли відбувається зміна топології мережі. Це може виникнути через дві обставини:

і) Вузол приймача переміщується в нове положення; рух призводить до а

зміна положення кореня, тому слід встановити нові шляхи іі) Певні вузли датчиків виснажують свою енергію і їх потрібно видалити з мережі. Це призводить до зміни топології, і шляхи передачі перераховуються.

Модель випадкових подій

У цій дипломній роботі ми пропонуємо адаптивний алгоритм розміщення стоків, який може реагувати на випадково згенеровані події. Подія n представлена такими параметрами:

- час початку події (Start n)
- час закінчення події (End n)
- місце події (X n , Y n)
- радіус події (R n)

Для наших симуляцій ми маємо вказану максимальну кількість подій. Для кожної події значення кожного поля генеруються випадковим чином (у межах відповідного діапазону). Активна тривалість події n визначається як (Кінець n – Початок n). Радіус (R n) події n визначає площу, на яку впливає подія. Для події в місці (X n , Y n) і радіусом R n всі датчики, що лежать в межах кола радіусом R n з центром (X n , Y n) , почнуть генерувати та надсилати пакети даних для повідомлення про цю подію , поки подія активна. На рисунку 2.2 показано дві події n 1 і n 2 з радіусом R 1 > R 2 і датчики, спрацьовані цими подіями.

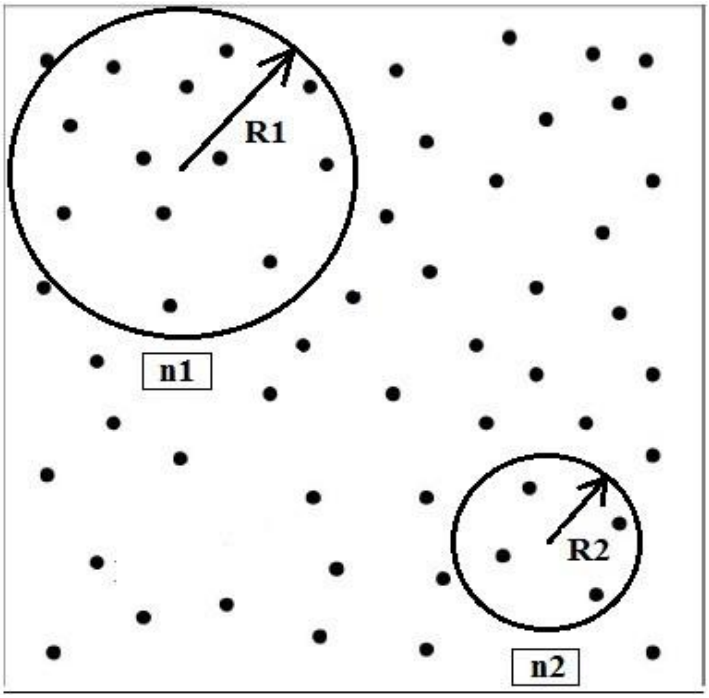


Рисунок 2.2 - Дві випадкові події в зоні зондування

2.2. Технологія руху на основі навантаження (LBM)

Як згадувалося раніше, одна з основних проблем, що впливає на тривалість життя мережі, пов’язана з тим, що розсіювання енергії стає сильно зосередженим на вузлах датчиків, які знаходяться ближче до вузла-приймача, особливо тих, які знаходяться лише на одному переході від вузла-приймача. Метою більшості енергоефективних протоколів маршрутизації та/або збору даних є рівномірний розподіл енергії в мережі, наскільки це можливо. Однак через обмежений діапазон передачі сенсорних вузлів важко повністю усунути цю «проблему гарячої точки». Проблема стає особливо гострою з одним стаціонарним вузлом , оскільки той самий набір вузлів датчиків, близьких до , повинен відповідати за передачу всіх даних. Однак, коли запроваджується мобільність поглиначів, різні вузли датчиків мають більш високі навантаження на дані в різний час, оскільки вузол переміщається з одного місця в інше. На рисунку 2.3(а) показана вузол в її початковому положенні, а також «сильно завантажені» сенсорні вузли

навколо в затіненій області. Якщо вузол нерухома, цей самий набір вузлів завжди залишається сильно завантаженим; але якщо вузліві дозволено переміститися на нове місце, як показано на рисунку 2.3(b), навантаження переноситься на абсолютно новий набір вузлів. З часом, коли вузол рухається, набір сильно завантажених сенсорних вузлів змінюється, і розсіювання енергії більш рівномірно розподіляється між усіма вузлами.

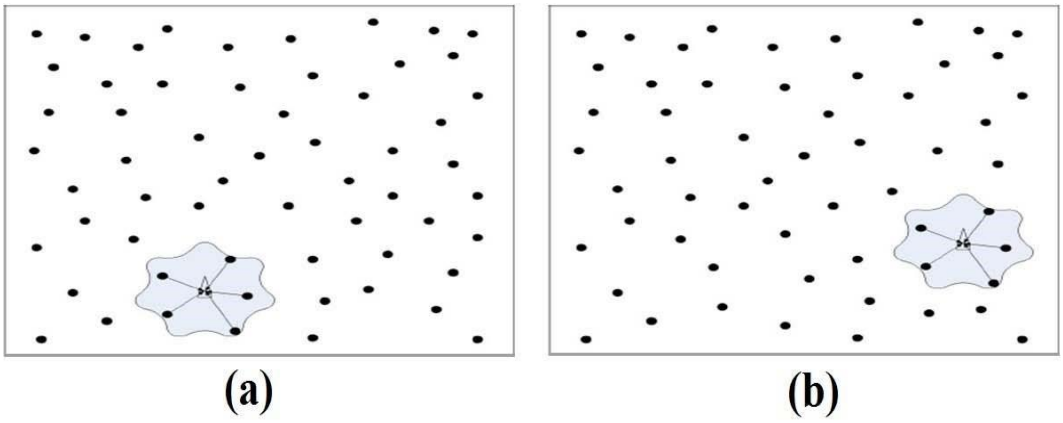


Рисунок 2.3 - Приймальний вузол у (a) початковій позиції (b) кінцевій позиції [5]

У цьому розділі ми представляємо нашу першу адаптивну схему розміщення приймального вузла під назвою Load Based Sink Movement (LBM), яка обчислює нове розташування вузла приймача у відповідь на випадкові події на основі географічне розташування активних датчиків і навантаження передачі на кожен активний вузол.

Відповідно до енергетичної моделі, наведеної загальне споживання енергії вузлом залежить насамперед від двох факторів:

1) відстань передачі до наступного вузла та 2) «навантаження», тобто кількість переданих бітів.

Зі введенням випадкових подій у певних регіонах мережі генерується більше даних, ніж в інших. Крім того, з одного активного субрегіону може бути

більше шляхів , ніж з іншого. Це призведе до нерівномірного розподілу навантаження між різними вузлами датчиків. Таким чином, окрім географічної відстані від активних датчиків до мобільного , кінцеве положення також має враховувати навантаження на вузли датчика. Таким чином, вузол повинна переміщатися ближче до вузлів датчика з великими навантаженнями на передачу, щоб зменшити споживання енергії цими сильно завантаженими сенсорними вузлами.

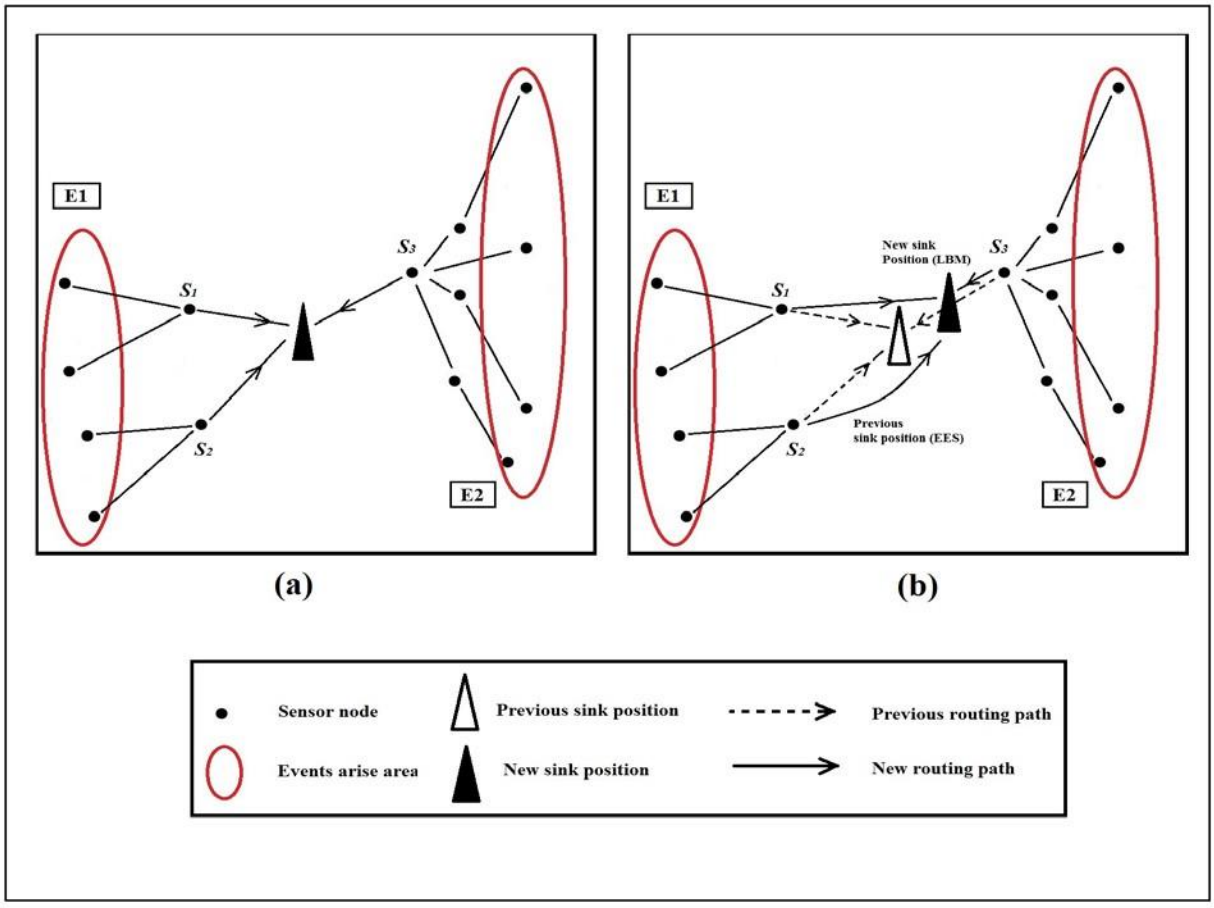


Рисунок 2.4 - Приклади LBM

Наприклад, розглянемо ситуацію на рисунку 2.4(a), з двома подіями E1 і E2, обидві однакового розміру. Згідно зі схемою в [5], вузол стоків повинен бути приблизно посередині між E1 і E2. Однак у запропонованій нами схемі LBM ми враховуємо не тільки розташування датчиків, а й їх навантаження. Крім того, на додаток до датчиків, які генерують дані, тобто всередині регіонів E1 і E2, ми

також намагаємося мінімізувати навантаження на всі датчики, тобто також ті, які пересилають дані. Це означає, що навантаження на датчики s 1 , s 2 і s 3 також враховуються. Зрозуміло, що розміщення вузла-приймача на рисунку 2.4(a) дає набагато більше навантаження на s 3 , і, отже, призводить до вищого розсіювання енергії для вузла s 3 , порівняно з s 1 і s 2 . Якщо вузол-приймач перемістити ближче до s 3 , як показано на рисунку 2.4 (b), відстань передачі для s 3 зменшується і може компенсувати більшу швидкість передачі даних, що призводить до зниження споживання енергії. Звичайно, це досягається ціною збільшення енергоспоживання у вузлах s 1 і s 2 . Однак, розподіляючи розсіювання енергії більш рівномірно по всіх вузлах, ми можемо очікувати загального покращення терміну служби мережі.

Детальний алгоритм

У цьому розділі ми представимо нашу схему переміщення на основі навантаження (LBM).

У нашому обговоренні ми використовуємо наступні позначення. Позначення:

S = набір усіх датчиків у мережі

p final = (X s , Y s) кінцева позиція вузла-приймача

p орієнтовна = (X' s , Y' s) попередня позиція вузла-приймача (для пошуку маршрутів)

p i = (x i , y i) положення сенсорного вузла $ss\ ii \in SS$.

b i = Загальна кількість бітів, переданих сенсорним вузлом $ss\ ii \in SS$.

w i = Загальна кількість бітів, створених вузлом датчика $ss\ ii \in SS$.

d ij = відстань від вузла s i до вузла s j

c ij = вартість ребра e(i □ j)

d max = максимальний діапазон передачі сенсорного вузла

r ij = 1 тоді і тільки тоді, коли вузол s i передає свої дані безпосередньо вузлу s j

Алгоритм 1: рух на основі навантаження (LBM)

Крок 1: Обчисліть орієнтовне положення p орієнтовне = (X' s , Y' s)

На основі EES [51] використовуйте рівняння (3) і (4) для обчислення значення для X 's і Y 's

Крок 2: Розрахувати шляхи

Використовуйте алгоритм Дейкстри [52], щоб обчислити шлях від кожного датчика $ss\ ii \in SS$ до поглинання вузла в (X' s , Y' s)

Для ребра e (i □ j) у топології встановіть r ij = 1, якщо ребро знаходиться на шляху від вузла s і до вузла-приймача.

Крок 3: Визначте навантаження на кожен вузол датчика

Навантаження на вузол $ss\ ii : bb\ ii = \Sigma\ ss\ ii \in SS\ rr\ iiii \cdot bb\ ii + ww\ ii$, $\forall ss\ ii \in SS$

Крок 4: Обчисліть кінцеве положення стоку p final = (X s , Y s)

Рисунок 2.5 - Схема алгоритму LBM

На рисунку 2.5 показано наш алгоритм LBM , який враховує навантаження на кожен вузол датчика. Щоб визначити навантаження, спочатку необхідно знайти орієнтовне розташування вузла . Це робиться на кроці 1 з використанням підходу, представленого в EES [5]. Після отримання орієнтовної позиції вузла-приймача (X 's , Y 's) (на кроці 2), маршрути мережі обчислюються за допомогою добре відомого алгоритму найкоротшого шляху [5].

Для алгоритму LBM вартість $cc\ iiii$ ребра e (i □ j) обчислюється таким чином:

$cc\ iiii = dd\ iiii2$, якщо $dd\ iiii \leq mmmm$

$cc\ iiii = \infty$, інакше

На основі шляхів від кожного вузла датчика до вузла-приймача можна розрахувати загальне навантаження (тобто кількість переданих бітів) для

кожного вузла $ss\ ii \in SS$ (Крок 3). Роблячи це, ми отримуємо навантаження на передачу датчиків для маршрутизації пакетів даних до вузла-приймача. Нарешті, нове розташування визначається на кроці 4, використовуючи рівняння (5) і (6).

$$\sum ssii \in SS\ bb\ ii \cdot xx\ ii$$

$$XX\ ss = \sum ssii \in SS\ bb\ ii \qquad (5)$$

$$YY\ ss = \frac{}{i \in S} \sum \sum ss\ ssii \in SS\ bb\ ii\ bb \cdot bb \cdot S \qquad (6)$$

Кінцеві координати розраховуються як зважена сума координат датчика всіх сенсорних вузлів, які передають дані. Чим вище навантаження на передачу датчика, тим вище вага його координат. Таким чином, кінцева позиція вузла приймача, як правило, буде ближче до цих сильно завантажених датчиків.

LBM розроблено для ефективнішого використання енергії датчиків, що живиться від батарейок, для того, щоб збалансувати розсіювання енергії та вирішити проблему «гарячої точки». Завдяки цьому можна продовжити загальний термін служби мережі, як показано в результатах експерименту в наступній.

2.3. Маршрутизація з урахуванням залишкової енергії (REAR)

Алгоритм LBM, який обговорювався в попередньому розділі, призводить до більш збалансованого розсіювання енергії між вузлами порівняно з існуючими методами, такими як EES [5]. Однак одним з потенційних обмежень підходу є те, що певні вузли можна використовувати знову і знову. В результаті такі вузли будуть мати тенденцію виснажувати свою енергію в порівнянні з іншими вузлами, які використовуються рідше. В інших роботах LBM не має «пам'яті» попередніх передач даних і просто вибирає маршрути на основі відстані, навіть якщо вибрані проміжні вузли можуть мати низький рівень енергії. Якщо одні й

ті самі вузли вибирати багаторазово, вони вичерпують всю свою енергію і можуть призвести до роз'єднання всієї топології.

Щоб усунути це обмеження, ми пропонуємо схему маршрутизації залишкової енергії (REAR). Метою цього алгоритму є уникнення використання вузлів з низьким рівнем енергії при розрахунку маршрутів від кожного датчика до. Низькоенергетичні вузли все ще можна використовувати, якщо немає інших альтернатив; проте використання таких вузлів дорожчає.

Основний внесок алгоритму REAR полягає в тому, що граничні витрати розраховуються по-різному, щоб врахувати рівні енергії в кожному вузлі. Ці оновлені граничні витрати потім використовуються на кроці 2 (Рисунок 2.5), щоб створити маршрути маршрутизації. Крім позначень, наведених ми використовуємо такі нові елементи для REAR :

IE = Початковий рівень енергії для кожного сенсорного вузла, коли мережа вперше розгорнута.

Вважається, що це однаково для всіх датчиків.

RE = Рівень залишкової енергії сенсорного вузла s i .

Основне обґрунтування REAR полягає в тому, що вартість використання e (i □ j) повинна відображати не тільки відстань від s i до s j , а й залишкову енергію у вузлі sj . Якщо у s j залишилося дуже мало енергії, то краще не використовувати ребро. Отже, у цьому випадку c ij слід відповідно збільшити. Отже, ми визначаємо вартість краю так:

$$cc\ iiii = dd\ iiii\ 2\ \mathrm{d_ij} + \alpha\alpha \cdot (IIEE - RREE\ ii) \qquad (7)$$

Тут $\alpha\alpha$ — це заздалегідь визначена вага для визначення відносної важливості відстані та компонентів залишкової енергії для визначення загальної вартості краю. Відповідно до рівняння (7), зрозуміло, що чим нижче залишкова енергія вузла (тобто менше значення RE j), тим вищою буде його вартість. Після того як маршрути розраховані, використовуючи призначені граничні витрати,

навантаження на кожен вузол датчика та кінцеве положення поглинача можуть бути визначені способом, подібним до LBM .

Мета REAR полягає в тому, щоб уникнути використання датчиків, які мають низький рівень залишкової енергії. Замість цього він намагається спочатку використовувати датчики з більш високою залишковою енергією для встановлення альтернативних шляхів маршрутизації, щоб можна було збалансувати розсіювання енергії. В результаті кожен окремий сенсорний вузол може досягти більш тривалого терміну служби та продовжити термін служби мережі.

На додаток до LBM і REAR , ми також пропонуємо іншу схему, яка називається гібридною схемою, яка поєднує прийоми як LBM , так і REAR . Цей підхід обчислює маршрути на основі залишкової енергії, як у REAR . Потім розраховується положення стоку за допомогою рівнянь (5) і (6), подібно до LBM , для врахування навантаження на кожен вузол датчика.

Representational State Transfer [2], широко відома як модель REST або RESTful для веб-сервісів, використовує власні операції HTTP: POST, GET, PUT & DELETE [3] для відображення чотирьох основних операцій бази даних — Create, Read, Оновити та видалити. Простий API можна створити, щоб зв’язати ці чотири HTTP дієслова з функціями, які створюють, читають, видаляють або оновлюють записи у веб-сервісі. Цю послугу потім може використовувати будь-який тип автентифікованого клієнтського пристрою.

Клієнтом може бути веб-сторінка або веб-додаток; або послуга може бути використана програмно за допомогою спеціально написаної програмної програми на ПК, смартфоні, планшеті чи іншому пристрої; з простого сценарію (наприклад, на Python); або дуже складний додаток для науки про дані, реалізований на мові низького рівня, або такий інструмент, як MATLAB або R.

Зрозуміло, що RESTful API дуже добре відповідає багатьом класам додатків IoT. Наприклад, пристрій IoT, який не має вбудованого користувальницького інтерфейсу, може діяти як клієнт публікації спеціального призначення:

отримавши доступ до API, він може передати дані своїх датчиків на центральний сервер за допомогою методу POST (Create). Отримані таким чином дані централізовано збираються і доступні для використання за допомогою GET-ing даних із служби.

2.4 Переваги середнього стаку для реалізації веб-сервісу IoT

A. Вузол

Незважаючи на те, що існує безліч способів реалізації REST API, використання стеку MEAN і, зокрема, Nodes & Express, підвищує продуктивність, зменшуючи необхідні зусилля для розробки; забезпечуючи ефективну, ефективну та масштабовану реалізацію. Хоча в даний час ще менш відомий, ніж стек LAMP, MEAN, ймовірно, стане впливовим; особливо враховуючи зростаючу участь Microsoft у Node (включаючи створення власної версії Node для процесорів ARM) [4] як частини Windows 10 IoT Core, яка працює на пристроях ARM, таких як Raspberry Pi 2 [5].

Дослідження показали [4], що сервер Node значно перевершує Apache і Nginx [6] щодо обслуговування динамічного вмісту — і реалізація JavaScript у Node більш ніж у 2,5 рази швидше, ніж більш традиційний підхід PHP, завдяки ефективному використанню доступного обладнання.

B. JSON

REST API повинен споживати та виробляти дані, закодовані узгодженим чином. Хоча існує багато варіантів (XML, YAML тощо), популярним вибором є JSON. JSON добре підходить для даних, які повинні бути читаними як людиною, так і машиною. Дані JSON, можливо, легше зрозуміти читачеві, ніж XML; і також може бути легко розібраний комп'ютером. Інструменти та бібліотеки для аналізу JSON існують у всіх основних мовах програмування та середовищах.

Формат пари ключ-значення в JSON ідеально підходить для використання зі звичайними параметричними даними, наприклад, які можуть бути створені сенсорним пристроєм або передані у вигляді командно-контрольного повідомлення на пристрій або виконавчий механізм.

Дані JSON із Node або іншої реалізації JavaScript уже в рідному форматі і тому не потребують подальшого аналізу, на відміну, наприклад, від XML.

C. MongoDb

Перевага використання документно-орієнтованої бази даних, такої як MongoDb, полягає в тому, що вона пропонує динамічну (а не фіксовану та жорстку) схему [7]. Це означає, що якщо структуру даних бази даних потрібно змінити, в результаті необхідно зберегти додаткові параметри; нові дані з іншою структурою можуть бути розміщені в базі даних, поряд з попередніми даними, без необхідності виконувати масштабні маніпуляції з даними в цілому.

MongoDb також є хорошим вибором для зберігання даних, закодованих JSON. Mongo Db внутрішньо зберігає дані в ефективному двійковому форматі JSON (BSON), що дозволяє швидко та легко імпортувати та експортувати. Він також ідеально підходить для зберігання та обробки великих обсягів даних. Він може масштабуватися до тисяч вузлів і петабайт даних [7].

Mongo Db також демонструє кращу продуктивність під час виконання простих операцій у невеликому масштабі (один вузол), ніж Microsoft SQL Server Express [8].

3 РЕАЛІЗАЦІЯ АЛГОРИТМІЧНОГО ТА ПРОГРАМНОГО РІШЕННЯ ЗАДАЧІ ПОБУДОВИ ІЕБ-СЕРВІСУ

У цій ми оцінюємо ефективність запропонованих нами схем і порівнюємо досягнутий термін служби з існуючими схемами розміщення EES [5]. Спочатку ми представляємо короткий опис експериментальної установки, а потім обговорюємо та аналізуємо результати нашого моделювання. В останньому розділі ми обговоримо результати експерименту та оцінимо продуктивність.

3.1. Проведення експериментального дослідження

Вже було показано, що надання мобільності приймача може значно продовжити термін служби мережі [5]. Енергоефективна стратегія (EES) для розміщення мобільних стоків, представлена в [5], також принесла значне покращення порівняно з підходом випадкового блукання [4]. У цій роботі ми оцінимо запропоновані нами схеми і побачимо, що вони можуть бути використані для досягнення ще більших удосконалень порівняно з EES щодо продовження терміну служби мережі.

У цих експериментах ми вимірюємо, як тривалість життя мережі покращується за набору випадкових подій. Час початку, тривалість, місце розташування та зона охоплення кожної події генеруються випадковим чином. Симуляція запрограмована за допомогою C і виконується як у Dev C++ під Windows 7 [7], і в Xcode в Mac OS X Lion [8]. Ми також використовуємо кілька різних розмірів мережі, і для кожного розміру мережі ми випадковим чином генеруємо кілька розподілів датчиків.

Налаштування топології мережі

Для перевірки запропонованих схем моделювання виконується в 3 мережах різного розміру з випадково розподіленими сенсорними вузлами:

100 датчиків на площі 160 х 160 м

200 датчиків на площі 220 х 220 м

500 датчиків на площі 360 х 360 м

Крім випадкових топологій, ми також розглядаємо топологію однорідної сітки, за якої всі датчики рівномірно розподілені в зоні мережі (рис. 3.1).

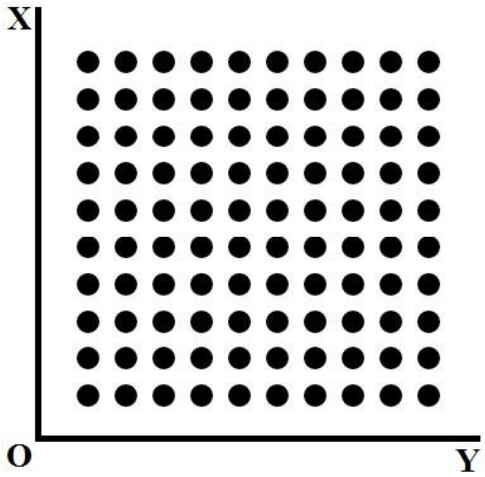


Рисунок 3.1 - Топологія сітки

У наших моделюваннях використовуються такі топології сітки:

- 100 датчиків рівномірно розподілені на площі 160 х 160 м.
- 225 датчиків, рівномірно розподілених на площі 240 х 240 м.
- 400 датчиків рівномірно розподілені на площі 320 х 320 м.

Налаштування випадкових подій

Для кожного набору моделювання ми розглядаємо 500 випадкових подій. Атрибути події, такі як час початку, час закінчення тощо, генеруються випадковим чином і зберігаються у файлі, який надається як вхідні дані в симулятор. У порівнянні з оцінкою продуктивності в EES [5] з 50 подіями в моделюванні, наш експеримент використовує більш високу частоту виникнення подій (500 випадкових подій поміщаються в послідовність запуску під час одного моделювання), щоб оцінити внесок запропонованих нами схем. Для

кожної події радіус покриття варіюється від 6 до 30 метрів в мережі; тривалість події може варіюватися від 1 одиниці часу до максимального часу роботи (3000 одиниць часу) в системі. Фактична тривалість події буде залежати від конкретної програми та може варіюватися від кількох мілісекунд до кількох хвилин або навіть годин.

На кожному проміжку часу певні події можуть початися, деякі події можуть закінчитися, а інші можуть тривати. Симулятор використовує часову шкалу для визначення кожної одиниці часу моделювання, отримує доступ до файлу подій, зчитує Start n і End n кожної події та розміщує їх у послідовності запуску у відповідний час, як показано на рисунку 3.2. Кожен квадрат з лівого боку представляє часовий інтервал, і програма моделювання зчитує початок n і кінець n кожної події та поміщає події у відповідну послідовність запуску на кожному інтервалі часу. Коли програма моделювання досягає цього інтервалу часу, вона виводить події в послідовність і виконує дії початку/закінчення події.

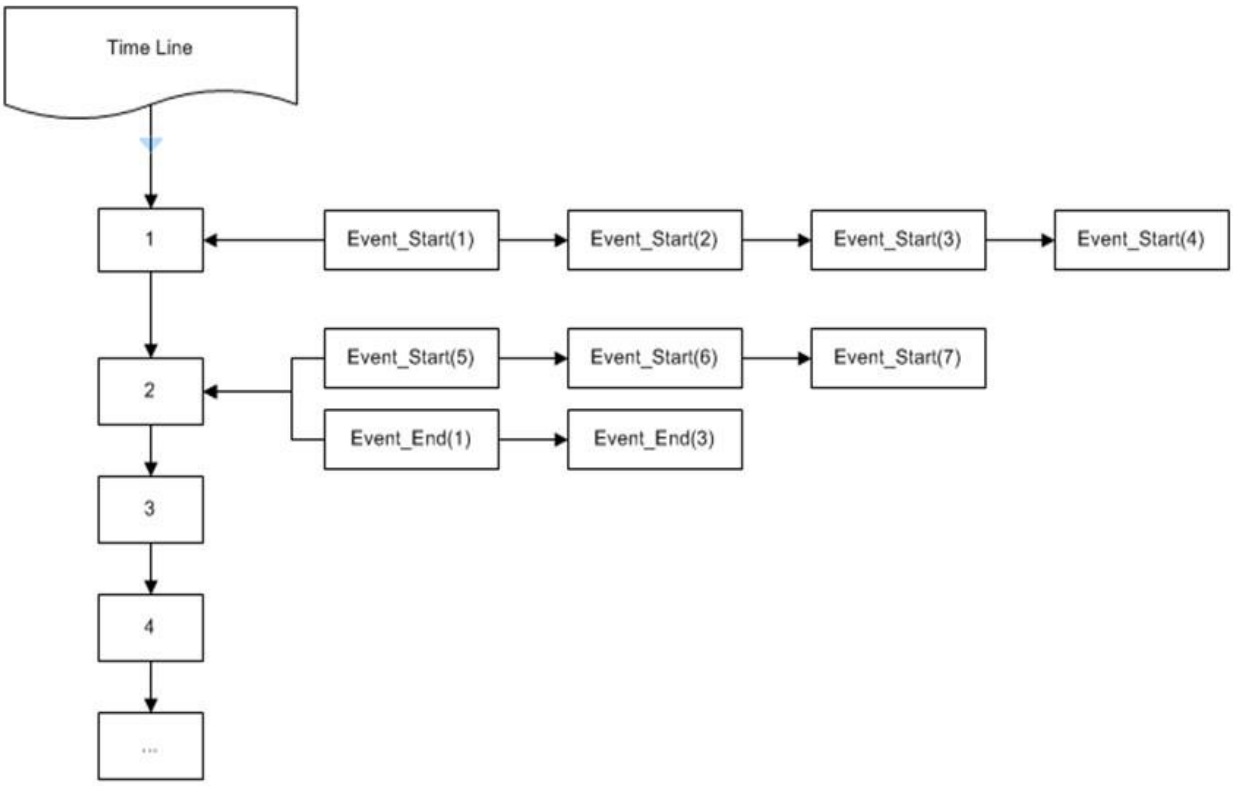


Рисунок 3.2 - Послідовність запуску випадкових подій у експерименті

Інші параметри

Додаткові параметри для нашого моделювання наведені нижче в Таблиці 3.1

Таблиця 3.1 - Перелік параметрів

Параметри	**Значення**
Максимальний діапазон передачі	40-60 метрів
Початкова енергія датчиків	10,0 Дж
Швидкість генерації даних активними датчиками	1 пакет/одиниця часу
Розмір пакета даних	10 біт
Розсіювання енергії для електронної обробки (E elec)	100 нДж/біт
Розсіювання енергії підсилювача (E amp)	100 нДж/біт/м 2
Компонент втрати шляху (q)	2

Моделювання базується на часі і оновлює послідовності запуску події та залишкову енергію з інтервалом в 1 одиницю часу під час роботи мережі.

3.2. Аналіз результатів та оцінка ефективності

У наших моделюваннях ми порівнюємо наступні 4 різні стратегії:

- Розташування на основі розташування (EES) [5]
- Розташування на основі навантаження (LBM)
- Маршрутизація залишкової енергії (REAR)
- Гібридний підхід, який поєднує як LBM, так і REAR (Hybrid).

Для кожного розміру мережі ми створили 5 різних випадкових топологій і протестували кожну топологію з 5 наборами послідовностей випадкових подій. Отже, результати, повідомлені для випадкових мереж, є середніми за 25 запусків.

Як згадувалося раніше, різні дослідники використовували різні показники для вимірювання терміну служби бездротової сенсорної мережі. У наших

моделюваннях ми взяли такі часто використовувані показники для вимірювання тривалості життя мережі:

Показник 1 : перший вузол датчика виснажує енергію

Показник 2 : 10% загальної кількості датчиків виснажують енергію

Показник 3 : мережа відключається, тобто принаймні один вузол датчика не може знайти дійсний шлях для маршрутизації даних назад до приймача.

Порівняння тривалості життя для випадкових мереж

На рисунках 3.3 – 3.5 показані результати для різних випадкових мереж.

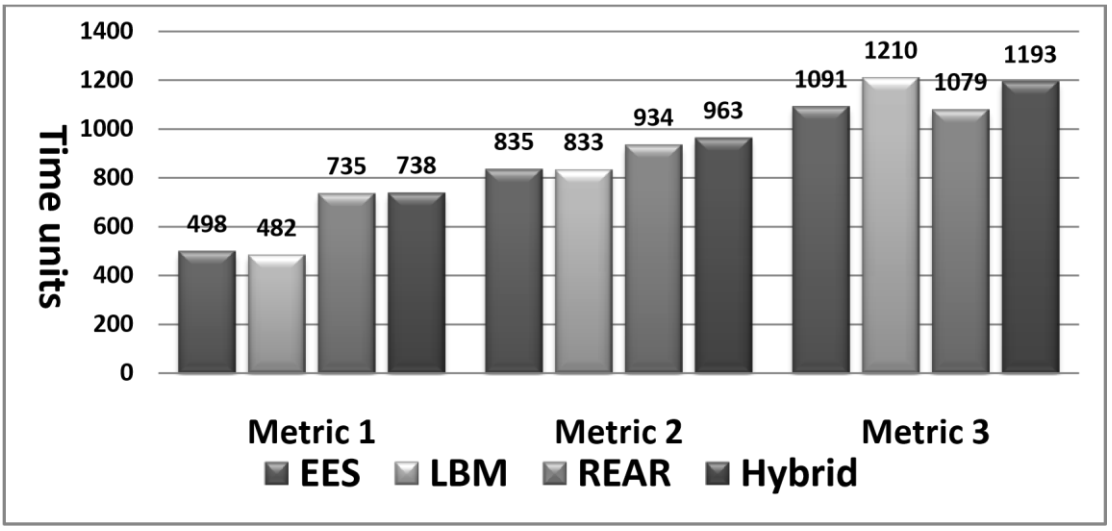


Рисунок 3.3 - Тривалість життя мережі в мережі з 100 датчиками

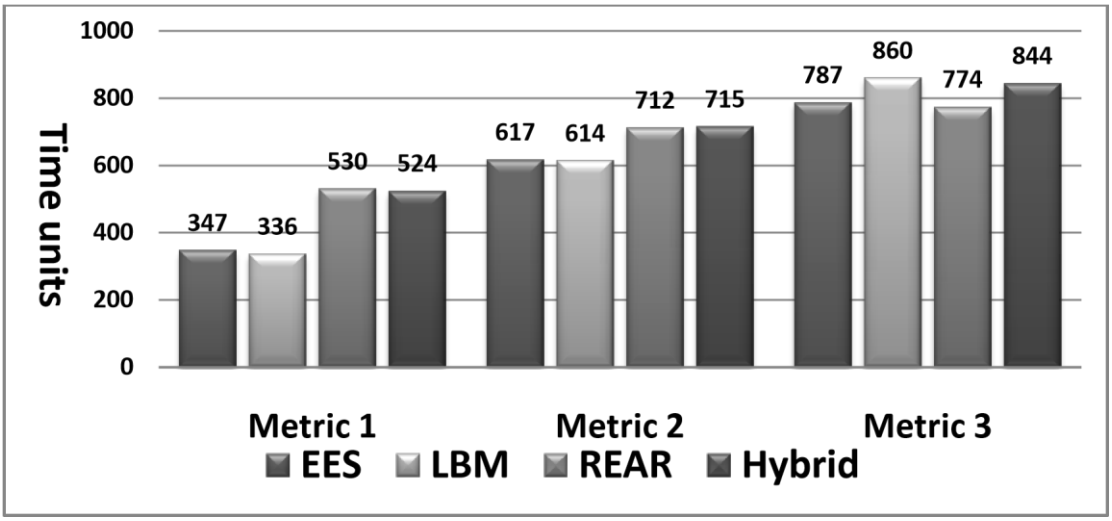


Рисунок 3.4 - Тривалість життя мережі в мережі з 200 датчиками

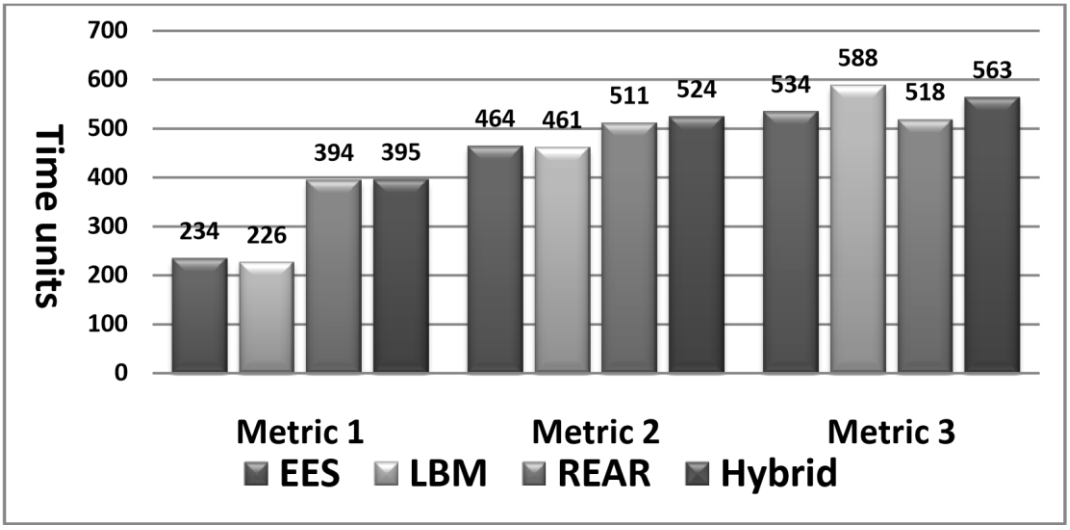


Рисунок 3.5 - Тривалість життя мережі в мережі з 500 датчиками

Згідно з результатами моделювання на рисунках 3.3, 3.4 та 3.5, ми бачимо, що LBM покращує загальний термін служби мережі (Показник 3) порівняно з EES на 12-20 %. Це пояснюється тим, що LBM впроваджує збалансований метод використання енергії з сильно завантаженими вузлами і дозволяє цим вузлам жити довше, ніж EES, щоб надавати послуги пересилання даних. Однак LBM не враховує залишкову енергію датчиків.

Таким чином, він може продовжувати використовувати низькоенергетичні датчики для налаштування маршрутів. Тому LBM не показує покращення збереження енергії для кожного сенсорного вузла. REAR перевершує EES з точки зору продовження терміну служби кожного сенсорного вузла. REAR використовує альтернативні датчики для встановлення маршрутів з більш високою залишковою енергією.

Таким чином, кожен датчик може прожити довше за рахунок зменшення споживання енергії, особливо коли його залишкова енергія низька. Результати показують, що REAR може продовжити термін служби мережі на 50-60%, якщо використовувати метрику 1. Гібридний підхід, який реалізує схеми LBM і REAR в роботу мережі, покращує термін служби кожного датчика і загальний термін служби мережі, як очікувалося.

Тривалість життя мережі (з використанням будь-якої метрики) постійно зменшується зі збільшенням розміру мережі. Це пов’язано з тим, що у великих мережах працює більше датчиків, і більше даних буде створено та передане в приймач. Тому для передачі даних потрібно більше енергії. Однак початкова енергія, що подається на кожен датчик, залишається незмінною, що призводить до загального скорочення терміну служби.

Порівняння тривалості життя для топологій сітки.

Результати для топологій сітки узгоджуються з результатами для випадкових мереж. Результати на рисунках 3.6, 3.7 та 3.8 показують, що LBM подовжує термін служби мережі (метрика 3), а REAR — подовжує термін служби кожного датчика (метрика 1). Загалом, топологія сітки, здається, призводить до дещо вищого часу життя, особливо з використанням Метрики 1 , порівняно з випадковими мережами. Це, ймовірно, пов’язано з кращою загальною зв’язністю, що призводить до більшого вибору шляхів маршрутизації серед вузлів топології сітки.

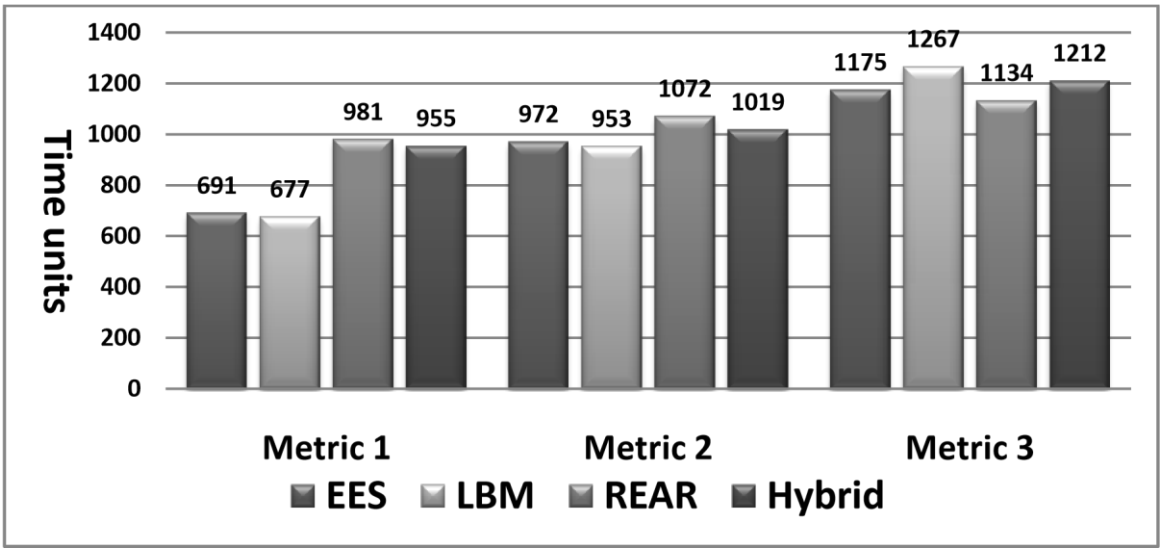


Рисунок 3.6 - Тривалість життя мережі в мережевій мережі з 100 датчиками

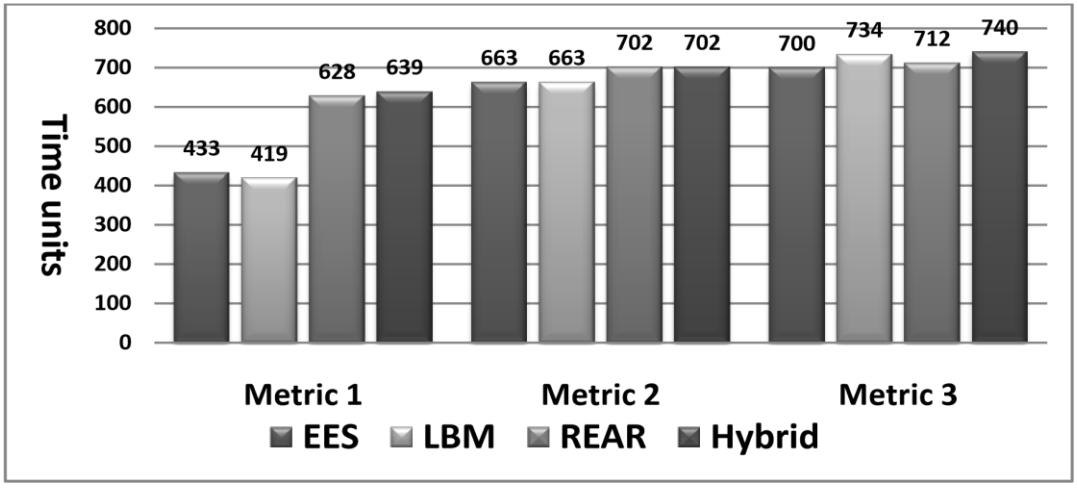


Рисунок 3.7 - Тривалість життя мережі в мережевій мережі з 225 датчиками

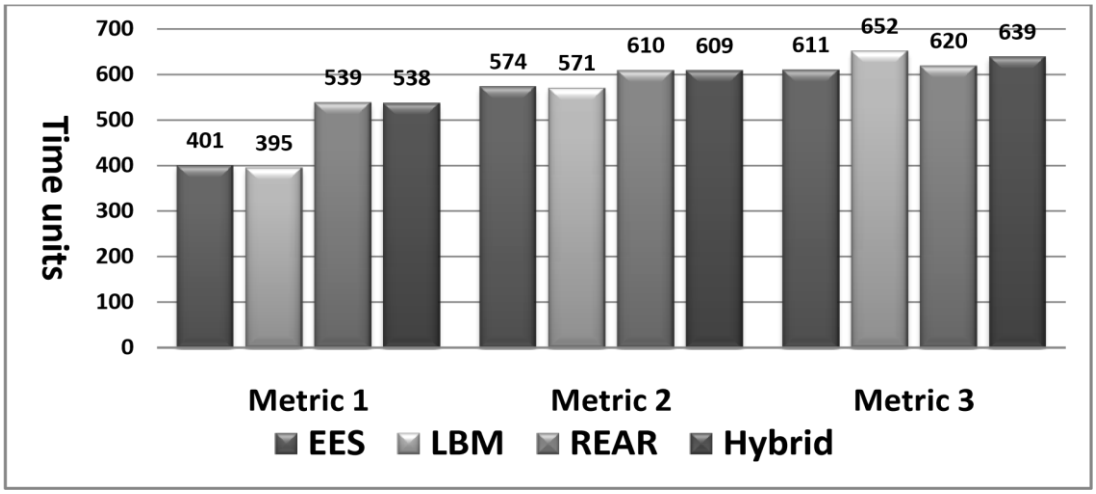


Рисунок 3.8 - Тривалість життя мережі в мережевій мережі з 400 датчиками

Вплив дальності передачі

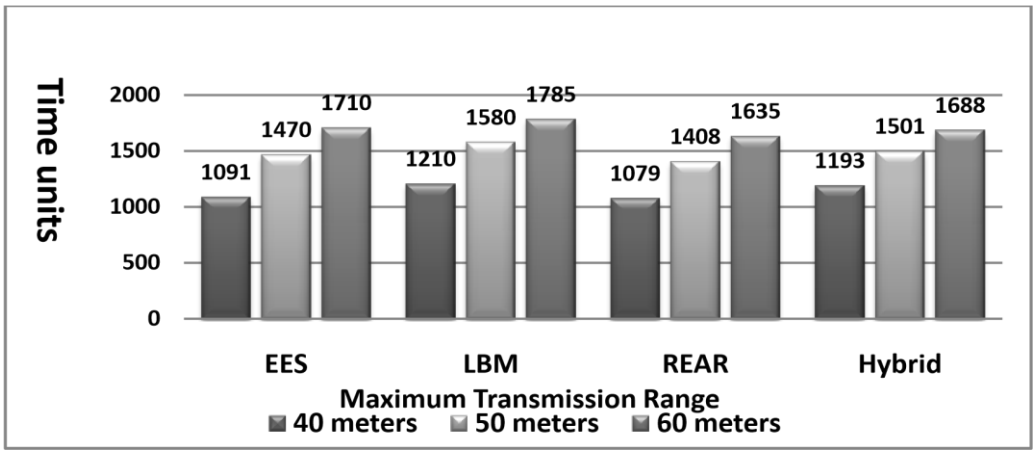


Рисунок 3.9 - Термін служби мережі при різному максимальному діапазоні передачі в мережі зі 100 датчиками

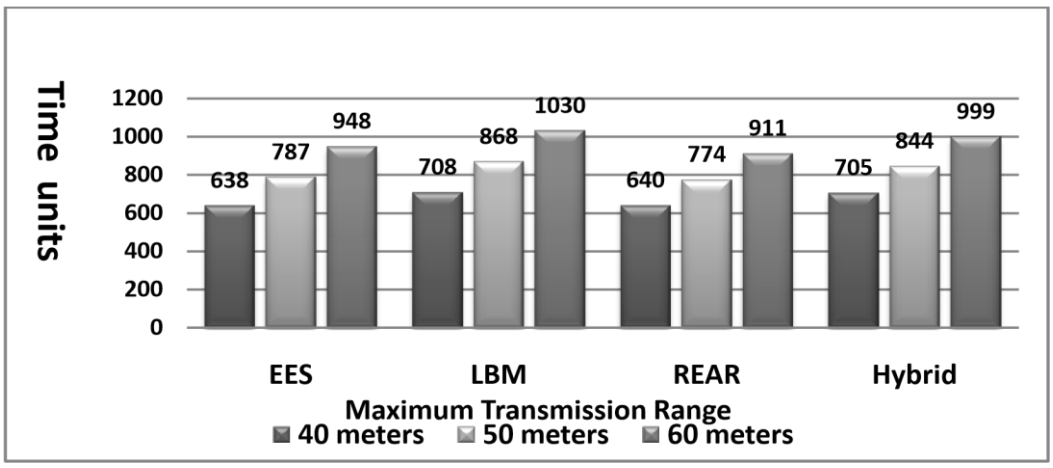


Рисунок 3.10 - Термін служби мережі при різному максимальному діапазоні передачі в мережі з 200 датчиками

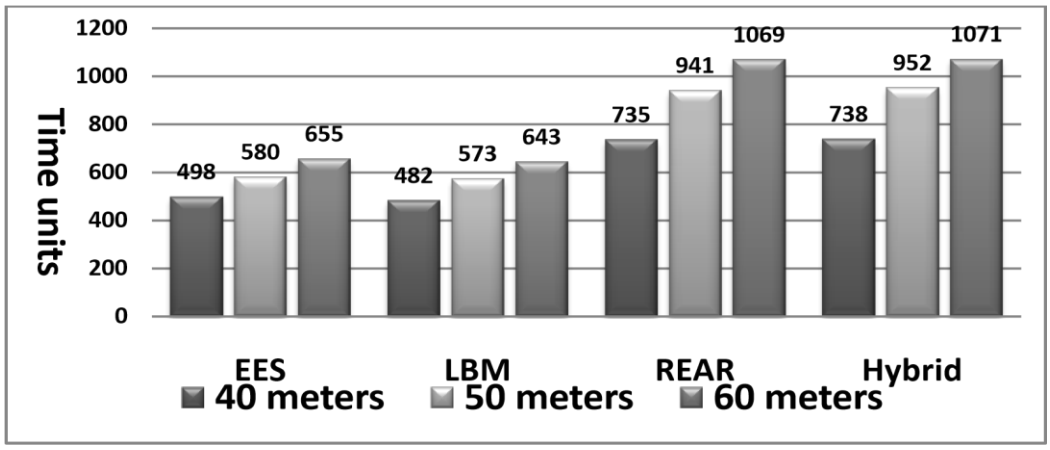


Рисунок 3.11 - Метрика 1 при різному максимальному діапазоні передачі в мережі з 100 датчиками

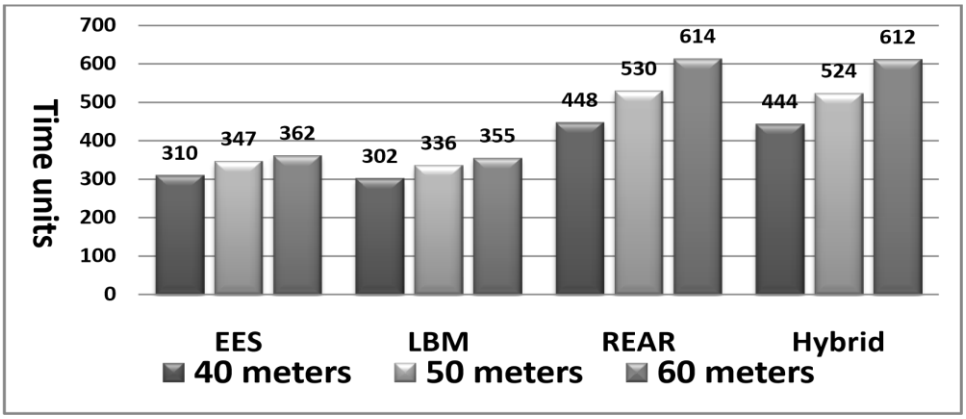


Рисунок 3.12: Метрика 1 при різному максимальному діапазоні передачі в мережі з 200 датчиками

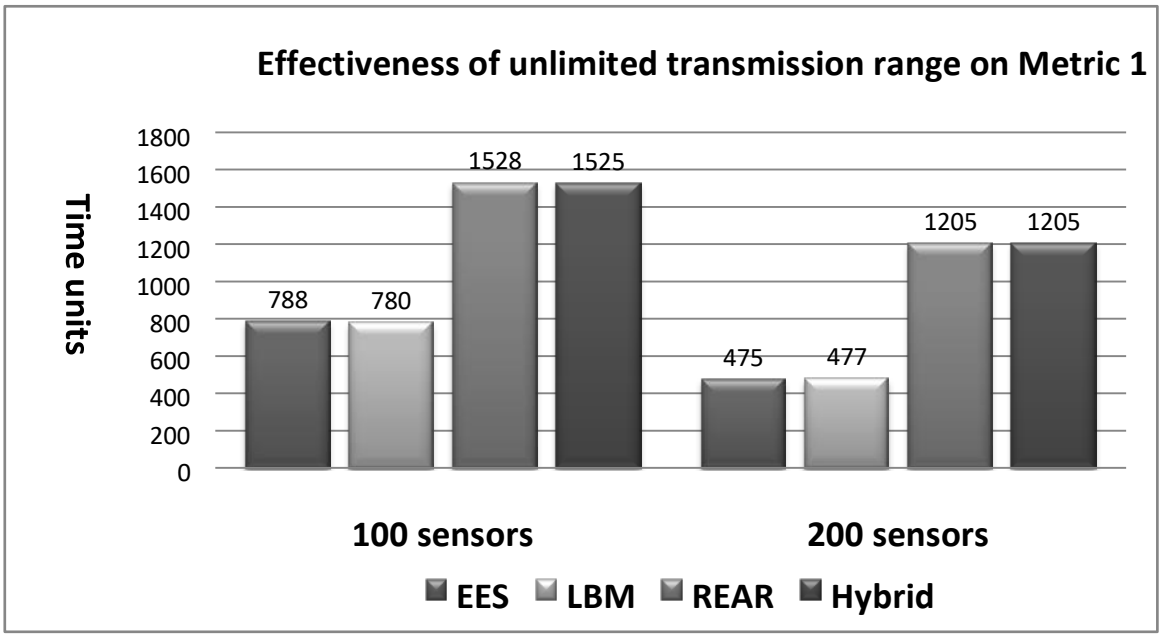


Рисунок 3.13 - Показник 1 без максимального діапазону передачі.

На рисунках 3.9 і 3.10 показано ефект застосування різних максимальних діапазонів передачі для сенсорних вузлів. Рисунки 3.11 і 3.12 показують ефективність для Метрики 1 з різним максимальним діапазоном передачі. Зі збільшенням максимального діапазону передачі збільшується термін служби мережі, оскільки датчики можуть досягати більш віддалених вузлів для передачі даних і можна знайти більше доступних шляхів. Тому можна встановити більше маршрутів, і мережа може працювати довше, ніж мережі з коротким діапазоном передачі. Крім того, чим більше діапазон передачі, тим менше датчиків потрібно для передачі даних. Це призводить до зменшення використання кожного датчика, тому один вузол датчика використовується рідше і може працювати довше. Якщо ми усунемо обмеження максимального діапазону передачі в нашому експерименті, результати на рисунку 3.13 показують, що LBM не покращує продуктивність, оскільки кожен датчик може надсилати дані безпосередньо на приймальний вузол, і проміжний вузол не потрібен. Тому кожен датчик рівний у «Навантаженні». Однак REAR значно покращує продуктивність у збереженні споживання енергії для кожного вузла, оскільки вони уникають використання сенсорних вузлів з низькою залишковою енергією

та використовують додатковий проміжний сенсорний вузол з більшою енергією для пересилання даних.

Зі збільшенням максимальної дальності передачі ефективність запропонованих схем залишається підтвердженою, але покращення зменшується зі збільшенням дальності. Це тому, що більший діапазон передачі призводить до меншої кількості стрибків, необхідних на оптимальному маршруті. Загалом LBM і REAR ефективні для управління розсіюванням енергії між сенсорними вузлами.

3.3 Реалізація API REST для пристрою IoT за використанням MEAN

Простий приклад серверної служби REST IoT було створено за допомогою MEAN, щоб додатково вивчити її корисність для такої програми.

A. Веб-сервіс Node & Express

Приклад веб-сервісу було створено за допомогою Node & Express, з відповідними даними, які зберігаються в базі даних MongoDb, що працює на тому ж сервері. Оскільки сервер був створений як експериментальна платформа, він забезпечував повну реалізацію REST API, дозволяючи чотири операції (POST, GET, PUT & DELETE) з усіма даними. У реальній реалізації (особливо в тій, де була вимога цілісності даних) було б хорошою практикою не підтримувати операції PUT, що робить дані незмінними. Аналогічно, надання операції DELETE для окремих або групових даних може бути небажаним.

Початкова реалізація не використовувала жодного типу аутентифікації і дозволяла виконувати всі операції будь-якому підключеному користувачеві або пристрою. Хоча це дійсний підхід для експериментального сервера, що працює в приватній мережі, це явно не модель, яку слід використовувати для будь-якої загальнодоступної служби, підключеної до Інтернету. Розміщення служби на сервері за допомогою HTTPS і модифікація служби таким чином, щоб деякі або

всі виклики API вимагали аутентифікації за допомогою базової аутентифікації [19], дуже прості, і це було досліджено в подальшій роботі.

B. Апаратне забезпечення: простий пристрій IoT

Після того, як послуга була запроваджена, приклад сенсорного пристрою був створений із обладнання для розробки товарів. Пристрій складався з датчика температури Microchip MCP9808, який через інтерфейс I2C запитувався через заздалегідь визначений інтервал часу за допомогою коду, що працює на мікроконтролері Ateml ATMEGA328. Разом мікроконтролер і датчик температури створюють «блок датчиків»: який передає температуру у вигляді рядка ASCII по дротовому послідовному зв'язку на одноплатний комп'ютер Beagle Bone Black. Потім BeagleBone Black об'єднав вихідні дані з ідентифікатором пристрою та міткою часу, перш ніж передавати дані у веб-сервіс за допомогою скрипту Python, написаного за допомогою популярної бібліотеки запитів .

Використання спеціального мікроконтролера для роботи в якості інтерфейсу до апаратного забезпечення нижнього рівня означало, що апаратне забезпечення датчика стає узагальненим, дозволяючи мікроконтролеру надавати послідовний інтерфейс до частини шлюзу пристрою, дозволяючи замінити обладнання без істотних модифікацій пристрою.

Цей модульний підхід також надав максимальну гнучкість загальному дизайну пристрою. BeagleBone виступав як шлюз і може бути тривіально модифікований для роботи з кількома сенсорними блоками. Крім того, дротове послідовне з'єднання можна легко реалізувати для використання бездротового зв'язку — наприклад, модуля XBee ZigBee — якщо інсталяція вимагає, щоб розташування датчика було віддаленим від шлюзу.

Використання малопотужного модульного обладнання для зв'язку з потужнішими шлюзами базових станцій означає, що недорогі модулі можуть бути широко розгорнуті без обмежень зв'язком або потужністю.

Веб-сервіс MEAN записав дані, згенеровані цим обладнанням, у вигляді чотирьох пар ключ-значення JSON (ідентифікатор пристрою, позначка часу, записана температура та унікальний ідентифікатор MongoDb для елемента). Дані можна отримати зі служби або шляхом доступу до певних елементів даних (за ідентифікатором), або шляхом отримання всіх даних. У будь-якому випадку дані поверталися як JSON. Подальша обробка цих даних, наприклад, графік даних у часі, може бути проведена за допомогою будь-якого інструменту або програмного забезпечення, здатного отримати доступ до даних JSON через веб-API.

C. Веб-додаток Angular

Веб-сервіс також використовувався для забезпечення серверної частини односторінкового веб-додатку, реалізованого за допомогою комбінації Angular.js, Yeoman і Bootstrap, описаної в Розділі II-D.

Ця веб-додаток надає засіб для перегляду вихідних даних для даного ідентифікатора пристрою, а також надає засоби для видалення та редагування окремих точок даних. Також можна додати додаткові функції, такі як конкретна візуалізація даних, використання автентифікованого API та безпечних посилань даних, а також функціональні можливості для налаштування пристрою.

D. Використання Node.js на пристрої

Під час цієї роботи ми досліджували запуск JavaScript-коду через Node.js безпосередньо на пристрої. Існує ряд бібліотек з відкритим кодом, які дозволяють скриптовим мовам високого рівня, таким як Python і JavaScript, безпосередньо взаємодіяти з апаратним забезпеченням пристрою. Однією з таких бібліотек є бібліотека Node onoff [21], яка забезпечує прямий доступ до контактів GPIO з JavaScript на ряді підтримуваних апаратних платформ розробки: включаючи Raspberry Pi і BeagleBone. Інші бібліотеки підтримують доступ до апаратних шин даних: наприклад, I2C і SPI.

Зі збільшенням доступності одноплатних комп’ютерів (і значним зниженням їх вартості), що спостерігається в останні роки, і значною обчислювальною потужністю, яку вони тепер можуть забезпечити: зменшується вимога до жорсткої оптимізації вбудованого коду, що працює на них. пристроїв у порівнянні з ситуацією навіть кілька років тому. Ця додаткова обчислювальна потужність (доступна за низькими витратами завдяки економії від масштабу в результаті революції смартфонів) означала, що вбудовані системи тепер можуть отримати вигоду від більшої абстракції між апаратним і програмним забезпеченням.

Хоча розробка програмного забезпечення з використанням мов сценаріїв дає певні переваги продуктивності, поки що не зрозуміло, наскільки добре вони підходять для використання для програмного забезпечення або мікропрограм, які запускаються на пристроях. Хоча ці типи мов можуть добре підходити для деяких видів діяльності (наприклад, раннє створення прототипів і використання любите- лями), використання цих мов має певні накладні витрати на продуктивність — і (критично для реальних додатків) значні накладні витрати на керування конфігурацією в порівнянні з статично зв'язаний код. Код, написаний з використанням однієї версії бібліотеки або модуля, може не працювати з іншою версією, і версії бібліотек часто прив’язані до певних версій основного інтерпретатора мови.

Таким чином, хоча використання мов сценаріїв для впровадження мікропрограми на пристрої може мати певні переваги, потрібна подальша робота, щоб більш формально оцінити переваги та витрати; і провести більш ретельну перевірку будь-яких проблем з продуктивністю та безпекою, які можуть виникнути.

3.4 Розширена реалізація сервісу IoT

У той час як варіант використання, представлений і досліджений , ідеально підходить для простих випадків; такий тип підключення лише для запису від

пристрою до Інтернету підходить лише для відносно невеликої кількості програм. Хоча така модель дозволяє підключеному до Інтернету датчику надсилати дані до центрального сховища, вона не забезпечує засобів для приводу, підключеного до Інтернету, для отримання даних команди та керування (наприклад, для ввімкнення світла або зміни параметра). температура, при якій працює система). Навіть пристрій, що працює лише з датчиком, може мати вимогу щодо зворотного шляху — наприклад, щоб змінити параметри датчика (наприклад, частоту сприйняття) або забезпечити засіб для оновлення програмного або мікропрограмного забезпечення пристрою.

Одним з підходів до реалізації цього вхідного зв'язку було б відкриття прямого з'єднання від кінцевого користувача (або його клієнтської програми) до пристрою; однак це створює ризики для безпеки пристрою (і, отже, системи), потенційно піддаючи пристрій шкідливим або помилково неформованим комунікаційним повідомленням від користувача. Він також встановлює вимогу до пристрою для аутентифікації вхідного зв'язку; і залишатися в стані, коли він готовий отримувати повідомлення.A. Server-based pull-communications

Альтернативою є надання зворотного шляху для зв’язку назад до пристрою, за допомогою якого пристрій періодично (з частотою, що відповідає додатку) опитує веб-сервіс, шукаючи оновлені командно-керівні повідомлення. Цей тип зв’язку з пристроєм за своєю суттю є більш безпечним, оскільки усуває необхідність прямого підключення до пристрою від віддаленого користувача. Керуючи всіма вхідними з’єднаннями на сервері, це також полегшує безпечну та остаточну аутентифікацію джерела отриманих командно-контрольних повідомлень. Забезпечення пристрою може безпечно встановити ідентичність веб-сервісу: за допомогою методів безпеки, поширених у веб-додатках, таких як захист транспортного рівня (TLS), і за умови, що пристрій (або користувач або клієнт користувача) безпечна автентифікація на сервері: самому пристрою ніколи не доводиться розглядати проблеми безпеки, пов’язані з безпосередньою автентифікацією користувача чи іншого командного агента. Ця аутентифікація

може використовувати або базову автентифікацію HTTP, або більш розширений протокол.

Це також потенційно дозволяє знизити енергоспоживання пристрою, оскільки він може переходити в стан сну протягом періодів між заздалегідь визначеними інтервалами зв'язку, а не постійно залишатися в стані більшої потужності, щоб отримувати повідомлення в невідомий час.

Використовуючи цю парадигму для зв'язку з витягуванням, пристрій IoT може збирати, а потім діяти на повідомлення: залежно від їх вмісту. Повідомлення можуть бути простою командою (наприклад, «увімкнути світло #0A49») або більш складним керуючим повідомленням (наприклад, «налаштуйте потужність нагрівача для підтримки температури в приміщенні 20,5°C»).

B. Віддалені оновлення програмного забезпечення

Використовуючи цей підхід до сповіщень, також можна вказати повідомлення керування системою: наприклад, щоб інструктувати пристрій отримати оновлення програмного забезпечення з сервера. Ця парадигма аналогічна тому, як часто розповсюджуються виправлення операційної системи: віддалені системи не надсилають оновлення на комп'ютер, а навпаки, комп'ютер відстежує відоме сховище, чекаючи вказівки на доступність оновлень. Використання цього методу також означає, що комп'ютер не повинен мати пряму адресу з Інтернету: як це може бути у випадку, якщо він підключений до мережі, яка захищена брандмауером або знаходиться за маршрутизатором за допомогою трансляції мережевих адрес (NAT). . Коли оновлення доступні, вони завантажуються комп'ютером. Походження цих оновлень можна перевірити за допомогою TLS, а їх вміст перевіряти на цілісність за допомогою функції хешування, наприклад SHA-1.

Оновлення програмного забезпечення для пристрою IoT можуть варіюватися від виправлень безпеки та простих оновлень мікропрограмного забезпечення — аж до значної переконфігурації роботи пристрою. Така реконфігурація може модифікувати пристрій, щоб забезпечити нову

функціональність — і може використовувати раніше неактивне обладнання всередині пристрою. Наприклад, віддалений датчик можна переналаштувати, щоб він працював більш ефективно, на основі аналізу даних, які він раніше зібрав — інформації, яка не була відома під час початкового розгортання пристрою. Можливість віддалено переналаштувати пристрій, який був розгорнутий, дуже привабливий: хоча для цього існує ряд етичних міркувань.

Щоб забезпечити додаткову безпеку процесу, можна використовувати криптографічні алгоритми для підписання програмного забезпечення та повідомлень як автентичного походження із затвердженого джерела, а також за допомогою приватного ключа в пристрої, щоб гарантувати, що лише повідомлення, дані або оновлення програмного забезпечення, призначені для певний пристрій може використовуватися цим пристроєм.

С. Альтернативи базовій аутентифікації — OAuth2

Протокол OAuth2— це відкритий стандартний протокол, призначений для узгодження доступу до ресурсу та полегшення аутентифікації за допомогою маркерів доступу. Хоча найчастіше використовується для забезпечення механізму автентифікації користувача на веб-сайті, використовуючи служби ідентифікації від стороннього постачальника (наприклад, Google), не відкриваючи облікові дані користувача від цього постачальника сайту, який вони хочуть використовувати; Протокол також підтримує автентифікацію користувача на ресурсі, де сайт, на якому розміщено ресурс, є його власним постачальником ідентифікації.

Перевага цього (у порівнянні з більш спрощеним підходом, що використовує безпосередньо облікові дані користувача) полягає в тому, що клієнт ніколи не повинен зберігати пароль. Більш точна детальність, яку пропонує цей метод, означає, що користувач може скасувати доступ до певного маркера (наприклад, конкретної клієнтської програми), не впливаючи на інші маркери.

Протокол OAuth2 визначає чотири типи надання авторизації, які можна використовувати для узгодження доступу та видачі токена. Ці різні типи грантів найкраще підходять для різних заявок. Хоча чотири типи надання описують різний процес узгодження, після успішного завершення всі вони призводять до видачі маркера доступу, який використовується для аутентифікації з'єднання шляхом надсилання як частини заголовка HTTP (тому все одно вимагає використання TLS). Служба OAuth2 може, залежно від реалізації, також видавати маркер оновлення, щоб полегшити повторну видачу маркера доступу після того, як початковий маркер закінчився.

Надання коду авторизації та неявне надання в першу чергу призначені для того, щоб дозволити веб-сервісам отримати доступ до даних, що зберігаються в іншій веб-сервісі. Їх можна використовувати для автентифікації користувача веб-додатка, пов'язаного з пристроєм IoT, у порівнянні з API іншого веб-ресурсу, наприклад Twitter або Facebook. Наприклад, користувач може вибрати використання веб-програми пристрою IoT для обміну даними з цього пристрою — у вигляді твіту або повідомлення у Facebook.

Надання облікових даних власника ресурсу призначено для надання доступу до (напівнадійної) клієнтської програми. Користувачу необхідно вказати свій пароль під час першої автентифікації (або будь-якої наступної повторної автентифікації в разі закінчення терміну дії маркера або відкликання) — але він ніколи не зберігається клієнтською програмою. У цьому випадку використання користувач може мати клієнтську програму (чи то на мобільному пристрої, чи на більш традиційному комп'ютері) для доступу до веб-сервісу: за допомогою надання доступу до облікових даних власника ресурсу для отримання доступу до веб-служби.

Нарешті, тип надання облікових даних клієнта використовується виключно для міжмашинного зв'язку, де не задіяний конкретний користувач: пристрій або клієнт аутентифікуються (як відомий і надійний клієнт) за допомогою служби. Для цього використовується дуже схожий процес із наданням власнику ресурсу,

але з використанням облікових даних, що стосуються самого клієнта, а не користувача-людини.

Гранти як власнику ресурсу, так і облікові дані клієнта добре підходять для використання в контексті Інтернету речей; і серверний елемент обох можна легко реалізувати за допомогою Node, Express та інших бібліотек .

Реалізація веб-сервісу з використанням гранту облікових даних власника ресурсу OAuth2 була створена під час подальшої роботи до початкової діяльності.

D. Контейнеризація

Однією з головних тенденцій останнього року стало зростання інтересу до контейнеризації та використання її для розгортання додатків — за допомогою таких інструментів, як Docker [6]. Docker дозволяє розробнику упакувати програму разом з усіма її залежностями в стандартизований і легкий контейнер, який потім можна розгорнути у виробничому середовищі. Ця техніка дає можливість розробнику гарантувати, що всі необхідні залежності будуть виконані, незалежно від стану конфігурації базової операційної системи на машині, на якій програма буде розгорнута.

Хоча, безсумнівно, існує потенціал для використання служб контейнеризації для полегшення веб-сервісу або програми MEAN, існують також інші (потенційно менші) підходи, які можна прийняти. Node надає менеджер пакетів вузлів (npm) [27]

— який використовує файл JSON для визначення конфігурації та залежностей програми Node; і може автоматично завантажувати весь необхідний код бібліотеки. Через рівень абстракції від базової ОС, який надає Node: цю техніку можна використовувати, щоб дозволити npm автоматично налаштовувати середовище для запуску будь-якої програми Node. Подібні інструменти керування пакетами існують і в інших мовах сценаріїв. Ця техніка до певної міри усуває необхідність контейнерного підходу під час використання програм Node. Однак у виробничому середовищі все ще необхідно забезпечити

наявність залежностей нижчого рівня (наприклад, правильні версії Node і MongoDb). Контейнеризація, така як Docker, забезпечує один із способів швидкого розгортання: і якщо такий метод використовується, його можна легко розширити, щоб включити програму Node та її залежності.

Хоча за межами цієї роботи, існує також потенціал використання цієї парадигми контейнеризації як частини процесу, який може забезпечувати атомарні транзакційні оновлення програмного або мікропрограмного забезпечення пристрою IoT. Планується подальша робота щодо дослідження доцільності цього підходу.

ВИСНОВКИ

Дана дипломна робота показала, що стек MEAN добре підходить для створення серверних служб для класу пристроїв Інтернету речей, які потребують API веб-сервісів. MEAN пропонує переваги в продуктивності та продуктивності в порівнянні з більш традиційними наборами інструментів, такими як LAMP.

Комбінація Node & Express.js з базою даних MongoDb забезпечує чудову реалізацію веб-сервісу на основі JSON; не в останню чергу тому, що всі інструменти в стеку будуть використовувати базові дані JSON. Такий веб-сервіс MEAN стека також дуже масштабований — завдяки перевагам як Node, так і MongoDb.

Хоча формальна оцінка переваг запуску Node та інших мов сценаріїв і інтерпретаторів безпосередньо на апаратному забезпеченні виходила за рамки цієї роботи; ця робота продемонструвала, що це можливо — і що вона заслуговує подальшого вивчення та розгляду варіантів використання.

Використання веб-сервісу для безпечного ввімкнення віддаленого оновлення програмного забезпечення є можливим.

Використання програмних контейнерів для полегшення масштабованого розгортання веб-сервісу стека MEAN також заслуговує подальшого вивчення в міру розвитку технології.

СПИСОК ПОСИЛАНЬ НА ДЖЕРЕЛА

1. G. Lawton, “LAMP lights enterprise development efforts,” Computer, vol. 38, no. 9, pp. 18–20, Sep. 2005. [Online]. Available: http://dx.doi.org/10.1109/MC.2005.304

2. MEAN.io. (2015) MEAN — Full-Stack JavaScript Using MongoDB, Express, AngularJS, and Node.js. [Online]. Available: http://mean.io/

3. S. Tilkov and S. Vinoski, “Node.js: Using JavaScript to Build High-Performance Network Programs,” IEEE Internet Computing, vol. 14, no. 6, pp. 80–83, 2010. [Online]. Available:http://doi.ieeecomputersociety.org/10.1109/MIC.2010.145

4. I. K. Chaniotis, K.-I. D. Kyriakou, and N. D. Tselikas, “Is Node.js a viable option for building modern web applications? A performance evaluation study,” Computing, pp. 1–22, 2014. [Online]. Available: http://dx.doi.org/10.1007/s00607- 014-0394-9

5. J. O’Dell, “Why Walmart is using Node.js,” 2012. [Online]. Available: http://venturebeat.com/2012/01/24/whywalmart-is-using-node-js/

6. Github. (2015) Projects, Applications, and Companies Using Node. [Online]. Available: https://github.com/joyent/node/wiki/Projects,-Applications,- and-Companies-Using-Node

7. T. Bray, “The JavaScript Object Notation (JSON) Data Interchange Format Interchange Format,” 2014. [Online]. Available: https://tools.ietf.org/html/rfc7159

8. Knowledge Base of Relational and NoSQL Database Management Systems. [Online]. Available: http://dbengines.com/en/ranking

9. A. Leff and J. T. Rayfield, “Web-application development using the Model/View/Controller design pattern,” Proceedings Fifth IEEE International Enterprise Distributed Object Computing Conference, pp. 118–127, 2001.

10. Yeoman: The web’s scaffolding tools for modern webapps. [Online]. Available: http://yeoman.io

11. Get Bootstrap. [Online]. Available: http://getbootstrap.com

12. R. T. Fielding and R. N. Taylor, “Principled design of the modern web architecture,” in Proceedings of the 22Nd International Conference on Software

Engineering, ser. ICSE ’00. New York, NY, USA: ACM, 2000, pp. 407–416. [Online]. Available: http://doi.acm.org/10.1145/337180.337228

13. R. Fielding and J. Reschke. (2014) Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. [Online]. Available: https://tools.ietf.org/html/rfc7231

14. J. Martin. (2015) Microsoft Forks Node.js to Support ARM. [Online]. Available: http://www.infoq.com/news/2015/05/nodejs-arm

15. L. Upton. (2015, 30/4/2015) Windows 10 for iot. [Online]. Available: https://www.raspberrypi.org/blog/windows-10-foriot/

16. Welcome to NGINX Wiki’s documentation. [Online]. Available: https://www.nginx.com/resources/wiki/

17. Internet of Things: MongoDB. [Online]. Available: https://www.mongodb.com/use-cases/internet-of-things

18. Z. Parker, S. Poe, and S. V. Vrbsky, “Comparing NoSQL MongoDB to an SQL DB,” Proceedings of the 51st ACM Southeast Conference on - ACMSE ’13, 2013. [Online]. Available: http://dl.acm.org/citation.cfm?id=2498328.2500047

19. R. Fielding and J. Reschke, “Hypertext Transfer Protocol (HTTP/1.1): Authentication,” 2014. [Online]. Available: http://tools.ietf.org/html/rfc7235

20. K. Reitz. Requests: HTTP for Humans. [Online]. Available: http://docs.python-requests.org/en/latest/

21. OnOff – GPIO access and interrupt detection with JavaScript. [Online]. Available: https://github.com/fivdi/onoff

22. T. Dierks and E. Rescorla, “The Transport Layer Security (TLS) Protocol Version 1.2,” 2008. [Online]. Available: https://tools.ietf.org/html/rfc5246

23. D. Hardt, “The OAuth 2.0 Authorization Framework,” 2012. [Online]. Available: https://tools.ietf.org/html/rfc6749

24. J. Hanson. OAuth2orize library. [Online]. Available: https://github.com/jaredhanson/oauth2orize

25. Passport : Simple, unobtrusive authentication for Node.js. [Online]. Available: http://passportjs.org

26. What is Docker? [Online]. Available: https://www.docker.com/what-docker

27. npm: the Node package manager. [Online]. Available: https://www.npmjs.co

БІБЛІОГРАФІЧНА ДОВІДКА

Тема дипломної роботи бакалавра: "Створення вебзастосунків стеком MERN (Mongo, Express, React, Node)"

Обсяг пояснювальної записки: 71 аркуш

Дата закінчення дипломної роботи 14 червня 2022р.

Підпис студента _____