ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ТЕЛЕКОМУНІКАЦІЙ
Кафедра Мобільних та відеоінформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка методів використання віртуалізації функцій мережі для покращення ефективності та гнучкості мобільних систем»

на здобуття освітнього ступеня магістра
зі спеціальності 172 Електронні комунікації та радіотехніка
освітньо-професійної програми Апаратура радіозв'язку, радіомовлення і телебачення

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Михайло КОЦЮБА
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Виконав: здобувач вищої освіти
групи АРДМ-62
Михайло КОЦЮБА

Керівник: к.т.н. доц. Лариса ДАКОВА
Рецензент: В. Скрипник

Київ – 2023

ДЕРЖАВНИЙ УНІВЕРСИТЕТ

ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут телекомунікацій

Кафедра Мобільних та відеоінформаційних технологій

Ступінь вищої освіти Магістр

Спеціальність 172 Телекомунікації та радіотехніка

Освітньо-професійна програма Апаратура радіозв'язку, радіомовлення і телебачення

ЗАТВЕРДЖУЮ

Завідувачка кафедрою МВТ

_____ Наталія РУДЕНКО

«___» _____ 2023 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Коцюбі Михайлу Валерійовичу

1. Тема кваліфікаційної роботи: «Розробка методів використання віртуалізації функцій мережі для покращення ефективності та гнучкості мобільних систем»,
керівник кваліфікаційної роботи Лариса ДАКОВА, к.т.н. доцент,
затверджені наказом Державного університету інформаційно-комунікаційних технологій від 19.10.2023 р. №145.

2. Строк подання кваліфікаційної роботи 25.12.2023 р.

3. Вихідні дані до кваліфікаційної роботи:
   1. Телекомунікаційна інфраструктура університету.
   2. Хмарні технології.
   3. Алгоритм глибокого навчання з підкріпленням.
   4. Науково-технічна література.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
   1. Теоретичні основи та концептуальні засади віртуалізації функцій мережі.
   2. Глибокий аналіз та оцінка віртуалізації функцій мережі у контексті мобільних технологій.
   3. Стратегії VNF: оптимізація, управління та оцінка для мобільних систем.

5. Перелік ілюстративного матеріалу: презентація

6. Дата видачі завдання

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1.	Підбір науково-технічної літератури	19.10.23	Викон.
2.	Теоретичні основи та концептуальні засади віртуалізації функцій мережі	31.10.23	Викон.
3.	Глибокий аналіз та оцінка віртуалізації функцій мережі у контексті мобільних технологій	15.11.23	Викон.
4.	Стратегії VNF: оптимізація, управління та оцінка для мобільних систем	29.11.23	Викон.
5.	Висновки, вступ, реферат	11.12.23	Викон.
6.	Розробка презентації	18.12.23	Викон.

Здобувач вищої освіти _____ Михайло КОЦЮБА
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник кваліфікаційної роботи _____ Лариса ДАКОВА
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 58 стор., 15 рис., 1 табл., 19 дж., 2 додатка.

Мета роботи – розробка алгоритму для ефективного та автоматичного розміщення VNF.

Об'єкт дослідження – віртуалізація функцій мережі.

Предмет дослідження – розподіл груп функцій віртуалізації мережі.

Короткий зміст роботи: В магістерський роботі було проведено дослідження теоретичних та емпіричних аспектів розробки методів використання віртуалізації функцій мережі. Для перевірки працездатності алгоритму було використано OMNeT++, для отримання даних про затримку і відсоток втрати пакетів, щоб оцінити якість розгорнутих віртуальних з'єднань. Агент DRL написаний на Python із використанням бібліотеки Keras.

КЛЮЧОВІ СЛОВА: ВІРТУАЛІЗАЦІЯ ФУНКЦІЙ МЕРЕЖІ, VNF, АГЕНТ DRL, ETSI, VNF-FGs, OMNET++, ГЛИБОКЕ НАВЧАННЯ З ПІДКРІПЛЕННЯМ, DRL, ВІРТУАЛІЗАЦІЯ, NFV, NFVI, NFVO, NFVC, NFV-MANO, ЯКІСТЬ ОБСЛУГОВУВАННЯ, QOS, NGMN, 5G, ГЛИБОКІ НЕЙРОННІ МЕРЕЖІ, DDPG

ABSTRACT

Text part of the master's qualification work: 58 pages, 15 pictures, 1 table, 19 sources, 2 applications.

The purpose of the work – development of an algorithm for efficient and automatic placement of VNFs.

Object of research – virtualization of network functions.

Subject of research – distribution of network virtualization function groups.

Summary of the work: In his master's thesis, he studied the theoretical and empirical aspects of developing methods for using virtualization of network functions. OMNeT++ was used to test the algorithm's performance, to obtain data on latency and packet loss percentage to assess the quality of deployed virtual connections. The DRL agent is written in Python using the Keras library.

KEYWORDS: NETWORK FUNCTION VIRTUALIZATION, VNF, DRL AGENT, ETSI, VNF-FGS, NFV, NFVI, OMNET++, DEEP REINFORCEMENT LEARNING, VIRTUALIZATION, NFVO, NFVC, NFV-MANO, QUALITY OF SERVICE, QOS, NGMN, 5G, DEEP NEURAL NETWORKS, DDP

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
Навчально-науковий інститут телекомунікацій

ПОДАННЯ
ГОЛОВІ ЕКЗАМЕНАЦІЙНОЇ КОМІСІЇ
ЩОДО ЗАХИСТУ КВАЛІФІКАЦІЙНОЇ РОБОТИ
на здобуття освітнього ступеня магістра

Направляється здобувач Коцюба М.В до захисту кваліфікаційної роботи
(прізвище та ініціали)

за спеціальністю 172 Телекомунікації та радіотехніка
освітньо-професійної програми Апаратура радіозв'язку, радіомовлення і телебачення

на тему: Розробка методів використання віртуалізації функцій мережі для покращення ефективності та гнучкості мобільних систем

Кваліфікаційна робота і рецензія додаються.

Директор ННІТ _____ Владислав КРАВЧЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Висновок керівника кваліфікаційної роботи

Здобувач Коцюба Михайло Валерійович показав гарну теоретичну та інженерну підготовку, уміння володіти технологіями нового покоління, вміло працювати з програмно-апаратними засобами мобільного зв'язку, довідковою і науково-технічною літературою. Працюючи над завданнями, які доручались керівником, проявив ініціативність, сумлінність та хист до інженерної роботи.
Все це дозволяє оцінити виконану кваліфікаційну роботу Коцюби Михайла Валерійовича на оцінку «добре» та присвоїти йому кваліфікацію «Магістр з телекомунікацій та радіотехніки».

Керівник кваліфікаційної роботи _____ Лариса ДАКОВА
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

" " _____ 2023 року

Висновок кафедри про кваліфікаційну роботу

Кваліфікаційна робота розглянута. Здобувач Коцюба М.В.
(прізвище та ініціали)

допускається до захисту даної роботи в Екзаменаційній комісії.

Завідувачка кафедри
Мобільних та відеоінформаційних технологій _____ Наталія РУДЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

" " _____ 2023 року

ВІДГУК РЕЦЕНЗЕНТА

на кваліфікаційну магістерську роботу

здобувача вищої освіти Коцюби Михайла Валерійовича

на тему: «Розробка методів використання віртуалізації функцій мережі для покращення ефективності та гнучкості мобільних систем»

Актуальність:

У нинішній час дослідження у сфері VNF має велике значення для розвитку мережевих технологій, особливо в умовах сучасних вимог до масштабованості, ефективності та гнучкості мережевих систем. Дослідження питання використання методів віртуалізації функцій мережі, дозволяє вирішити ці питання.

Тому, теоретичні та практичні дослідження направлені на вирішення даних завдань є актуальними і своєчасними.

Позитивні сторони:

1. Зміст магістерської роботи відповідає завданню. Її автор, показав високий рівень знань і ступінь підготовленості до майбутньої роботи за фахом.

2. Текст викладений грамотно, ясно, послідовно. Графічний матеріал оформлений якісно. Широко використовується сучасна науково-технічна література, включаючи англомовні видання.

Недоліки:

1. В роботі занадто багато уваги приділено теоретичним засадам віртуалізації функцій мережі.

2. В роботі чітко не відображено як здійснюється розгортання віртуалізацій функцій мережі.

Відзначені зауваження не впливають на загальну позитивну оцінку кваліфікаційної роботи.

Висновок: кваліфікаційна робота заслуговує оцінку «добре», а здобувач КОЦЮБА Михайло Валерійович заслуговує присвоєння кваліфікації: «Магістр з телекомунікацій та радіотехніки».

Рецензент: PhD _____ В. В. Сидоренко

(науковий ступінь, вчене звання) *(підпис)* *(Ім'я, ПРІЗВИЩЕ)*

ЗМІСТ

ВСТУП 8
1 ТЕОРЕТИЧНІ ОСНОВИ ТА КОНЦЕПТУАЛЬНІ ЗАСАДИ ВІРТУАЛІЗАЦІЇ ФУНКЦІЙ МЕРЕЖІ 10
1.1 Огляд віртуалізації функцій мережі (VNF) 10
1.2 Архітектура та компоненти VNF 13
1.3 Принципи роботи VNF 21
2 ГЛИБОКИЙ АНАЛІЗ ТА ОЦІНКА ВІРТУАЛІЗАЦІЇ ФУНКЦІЙ МЕРЕЖІ У КОНТЕКСТІ МОБІЛЬНИХ ТЕХНОЛОГІЙ 24
2.1 Аналіз сучасного стану використання VNF у мобільних системах 24
2.2 Вивчення ефективності та гнучкості VNF 31
2.3 Проблеми та виклики при інтеграції VNF 37
3 СТРАТЕГІЇ VNF: ОПТИМІЗАЦІЯ, УПРАВЛІННЯ ТА ОЦІНКА ДЛЯ МОБІЛЬНИХ СИСТЕМ 44
3.1 Оптимізація розміщення VNF 44
3.2 Алгоритми управління VNF 50
3.3 Тестування та оцінка алгоритмів VNF 58
ВИСНОВКИ 65
ПЕРЕЛІК ПОСИЛАНЬ 67
Додаток А 69
Додаток В 71
ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ 73

ВСТУП

У світі, що стрімко розвивається, важливість та складність телекомунікаційних систем постійно зростають, особливо у контексті мобільних мереж. Однією з ключових технологій, яка визначає майбутнє цих систем, є віртуалізація функцій мережі (VNF). Вона надає значні переваги у термінах збільшення ефективності, гнучкості та масштабованості мережевих ресурсів.

Напрямок віртуалізації функцій мережі є ключовим у розвитку мережевих технологій, відкриваючи шлях для переходу від традиційних мереж, що залежать від фізичного обладнання, до більш гнучких, програмно-конфігурованих систем. Завдяки VNF змінюються підходи до управління мережевими ресурсами, що пропонує нові можливості та внесення значного вкладу у економіку та оперативне управління, знижуючи капітальні та операційні витрати, а також забезпечуючи велику гнучкість у впровадженні нових послуг та швидкій реакції на зміни в мережевих потребах.

VNF включає віртуалізовані версії традиційних мережевих функцій, що можуть бути легко розміщені на віртуальних ресурсах, дозволяючи створювати гнучку та ефективну мережеву інфраструктуру, яка швидко адаптується до змінних потреб та технологічних вимог. Стандартизація та розробка відкритих інтерфейсів для VNF сприяють створенню сумісних рішень між різними виробниками та розробниками, вносячи значний вклад у інновації в галузі.

У сукупності, дослідження у сфері VNF має велике значення для розвитку мережевих технологій, особливо в умовах сучасних вимог до масштабованості, ефективності та гнучкості мережевих систем. Це особливо актуально для України, де модернізація інфраструктури та впровадження інноваційних технологій можуть значно покращити якість та доступність мережевих послуг, сприяючи економічному розвитку та технологічній інтеграції країни.

Ефективне та автоматичне розміщення мережевих послуг безумовно є одним із найважливіших технологічних елементів для переходу до повністю

автоматизованої мережі (тобто мережі з нульовим втручанням). Відповідно, дослідники з академічних та промислових сфер посилили свої зусилля для вирішення цього важливого питання. Було проведено кілька досліджень, деякі з них орієнтовані на архітектурні рішення, такі як ті, що розроблені Європейським інститутом телекомунікаційних стандартів (ETSI), інші пропонують більш алгоритмічні рішення, на які зосереджується ця робота.

У літературі вже було запропоновано кілька рішень [12]. Деякі техніки базуються на формулюванні та вирішенні задач оптимізації, які мають обмеження через неможливість застосування в реальному контексті, враховуючи затримку у вирішенні чи невідповідність у реальному контексті1. Інші пропонують швидкі евристичні методи, але не обов'язково досліджують величезний простір можливих дій. Хоча ці техніки ефективні, вони можуть мати проблеми зі збіжністю та тому повільністю. Техніки, засновані на глибокому навчанні, демонструють деяку ефективність, але є просто неефективними, коли простір дій дуже широкий.

У цій роботі буде розглядатися розподіл груп функцій віртуалізації мережі (VNF-FGs) для реалізації мережевих послуг. Також буде модельовано проблему розподілу VNF-FG як процес прийняття рішень Маркова (MDP) та вирішення її за допомогою підходу глибокого навчання з підкріпленням (DRL). Ця робота застосовує підхід DRL для вирішення вимог QoS. Метою даної роботи є:

1. Дослідження принципів роботи та архітектури VNF.
2. Аналіз сучасного стану реалізації та проблем VNF.
3. Дослідження оптимізації розміщення та управління VNF.
4. Дослідження глибокого навчання з підкріпленням, яка базується на Deep Deterministic Policy Gradient для покращення продуктивності агента DRL.

1 ТЕОРЕТИЧНІ ОСНОВИ ТА КОНЦЕПТУАЛЬНІ ЗАСАДИ ВІРТУАЛІЗАЦІЇ ФУНКЦІЙ МЕРЕЖІ

1.1 Огляд віртуалізації функцій мережі (VNF)

Віртуалізація функцій мережі (VNF) представляє собою один з найважливіших напрямків еволюції мережевих технологій, який корінним чином змінює підходи до управління мережевими ресурсами та послугами. Ця технологія відображає перехід від традиційних мереж, побудованих на фізичному апаратному забезпеченні, до гнучких, програмно-конфігурованих систем, що забезпечують нові рівні ефективності та адаптивності.

VNF дозволяє відокремити мережеві функції від фізичного обладнання, реалізуючи їх як програмне забезпечення, що може бути розміщене на загальних обчислювальних платформах. Це надає величезні переваги в термінах зниження капіталовкладень і операційних витрат, а також забезпечує більшу гнучкість у впровадженні нових послуг та швидкій реакції на зміни в мережевих потребах.

Основною ідеєю VNF є перехід від спеціалізованого мережевого обладнання до віртуалізації мережевих функцій, що забезпечує значні переваги в плані масштабованості, управління, впровадження та обслуговування мереж. Такий підхід дозволяє операторам мереж швидко розгортати та масштабувати мережеві сервіси відповідно до зростаючих або змінюваних потреб користувачів.

Віртуалізація – це процес запуску віртуального екземпляра комп'ютерної системи на рівні, абстрагованому від реального апаратного забезпечення. Найчастіше це стосується одночасного запуску декількох операційних систем на комп'ютерній системі. Для додатків, що працюють поверх віртуалізованої машини, може здаватися, що вони знаходяться на власній виділеній машині, де операційна система, бібліотеки та інші програми є унікальними для гостьової віртуалізованої системи і не пов'язані з хост-операційною системою, яка знаходиться під нею.

Існує багато причин, чому люди використовують віртуалізацію в комп'ютерах. Для користувачів настільних комп'ютерів найпоширенішою є можливість запускати програми, призначені для іншої операційної системи, без необхідності перемикання комп'ютерів або перезавантаження в іншу систему. Для адміністраторів серверів віртуалізація також пропонує можливість запускати різні операційні системи, але, можливо, важливіше те, що вона пропонує спосіб сегментувати велику систему на багато менших частин, що дозволяє більш ефективно використовувати сервер декількома різними користувачами або додатками з різними потребами. Вона також забезпечує ізоляцію, захищаючи програми, що працюють у віртуальній машині, від процесів, що відбуваються в іншій віртуальній машині на тому ж хості.

Гіпервізор – це програма для створення та запуску віртуальних машин. Вони традиційно поділяються на два класи: гіпервізори першого типу, або "голі", які запускають гостьові віртуальні машини безпосередньо на апаратному забезпеченні системи, по суті, працюючи як операційна система. Гіпервізори другого типу, або "розміщені", поводяться більше як традиційні програми, які можна запускати і зупиняти, як звичайні програми. У сучасних системах цей поділ менш поширений, особливо у системах типу KVM. KVM, скорочення від kernel-based virtual machine – це частина ядра Linux, яка може запускати віртуальні машини безпосередньо, хоча ви можете використовувати систему з віртуальними машинами KVM як звичайний комп'ютер.

Важливою характеристикою VNF є її взаємодія з технологією софтверно-визначених мереж (SDN), яка дозволяє централізовано управляти розподілом мережевих ресурсів. Ця синергія між VNF та SDN відкриває нові можливості для створення гнучких, адаптивних мережевих архітектур, здатних задовольнити різноманітні потреби сучасних мережевих середовищ.

Програмно-визначені мережі (SDN) – це архітектура, яка абстрагується від різних рівнів мережі, щоб зробити її гнучкою та маневреною. Метою SDN є покращення контролю над мережею, що дозволяє підприємствам та постачальникам послуг швидко реагувати на мінливі бізнес-вимоги.

У програмно-визначеній мережі, мережевий інженер або адміністратор може формувати трафік з централізованої консолі управління, не торкаючись окремих комутаторів у мережі. Централізований контролер SDN спрямовує комутатори на надання мережевих послуг там, де вони потрібні, незалежно від конкретних з'єднань між сервером і пристроями.

Цей процес є відходом від традиційної мережевої архітектури, в якій окремі мережеві пристрої приймають рішення щодо трафіку на основі своїх налаштованих таблиць маршрутизації. SDN відіграє важливу роль у мережевих технологіях вже протягом десятиліття і вплинула на багато інновацій в мережі.

Типове представлення архітектури SDN складається з трьох рівнів: прикладного рівня, рівня управління та інфраструктурного рівня. Ці рівні взаємодіють за допомогою північного та південного прикладних програмних інтерфейсів (API).

Розвиток VNF бере свій початок від появи концепту софтверно-визначених мереж (SDN) та загальної тенденції до віртуалізації в IT-індустрії. Перші ідеї та концепції VNF з'явилися як відповідь на потребу в більш гнучких та ефективних способах розгортання та управління мережевими ресурсами. Це було частково викликано стрімким розвитком технологій, зростаючим обсягом даних та потребою в підвищенні продуктивності мережі.

Важливим кроком у розвитку VNF стало стандартизування та розробка відкритих інтерфейсів, що дозволило різним виробникам та розробникам працювати над створенням сумісних рішень. Це призвело до появи багатьох інноваційних продуктів та рішень, які використовують принципи VNF для реалізації мережевих послуг.

У міру розвитку технології VNF, стало зрозуміло, що вона може відігравати ключову роль у підвищенні гнучкості, зниженні витрат та підвищенні продуктивності мереж. Це спонукало до подальших досліджень і розробок у цій області, зосереджуючись на покращенні та оптимізації VNF для використання в різноманітних мережевих середовищах.

З розвитком хмарних обчислень та Інтернету речей, VNF набуває ще більшого значення, відкриваючи нові можливості для інтеграції та управління мережевими послугами. Це включає розробку гнучких мережевих архітектур, здатних підтримувати величезну кількість підключених пристроїв та забезпечувати високий рівень продуктивності та безпеки.

1.2 Архітектура та компоненти VNF

Мережева функція (Network Function, NF) – це компонент мережі з чітко визначеними зовнішніми інтерфейсами та функціональною поведінкою. Прикладами мережевих функцій можуть бути DHCP-сервери, брандмауери, шлюзи SAE, MME, eNodeB. Віртуальна мережева функція (Virtual Network Function, VNF) – це програмна версія такої мережевої функції, яка може бути легко розміщена на віртуальних ресурсах, таких як віртуальні машини (VM).

Послуга, у свою чергу, є пропозицією, що надається постачальником послуг, і може складатися з однієї або декількох VNF. Одним з прикладів такої послуги є vEPC (Virtual Evolved Packet Core – віртуальне еволюційне пакетне ядро), яка включає в себе віртуалізовані версії мережевих функцій, таких як MME, обслуговуючий шлюз, пакетний шлюз, HSS та PCRF.

З розвитком програмного забезпечення, ці мережеві функції можуть бути адаптовані для роботи на віртуальних машинах. В такому випадку вони перетворюються на VNF. Коли всі ці VNF взаємодіють між собою через зовнішні інтерфейси, формується vEPC (Virtual EPC – віртуальний EPC). Це дозволяє створити гнучку та ефективну мережеву інфраструктуру, яка може швидко адаптуватися до змінних потреб та технологічних вимог.

Внутрішня архітектура VNF (Virtual Network Function) представлена на рис. 1.1, де детально зображено сутності та інтерфейси, що мають значення для обговорення у цьому розділі. При цьому свідомо відкидаються аспекти, які стосуються інфраструктури, управління та оркестрування NFV (Network Function Virtualization).

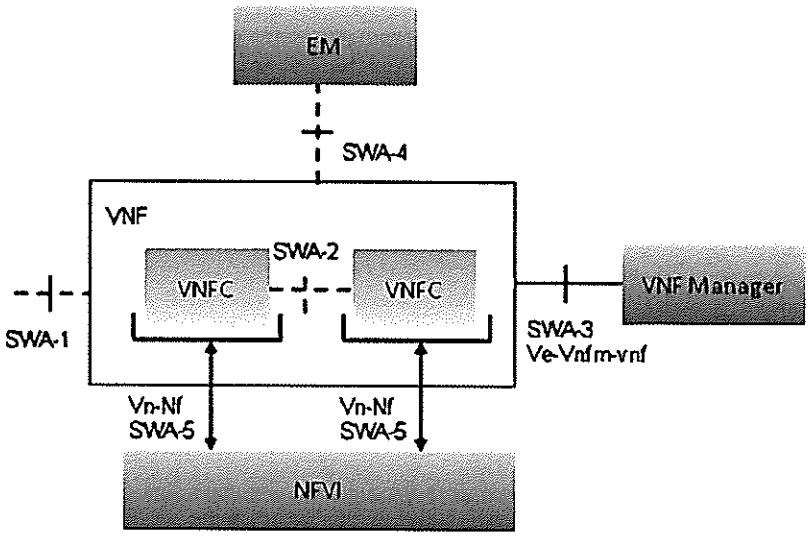


Рис. 1.1. Внутрішня архітектура VNF

VNF – це мережева функція, яка може функціонувати в рамках інфраструктури NFV (NFVI) і підлягає управлінню NFV Orchestrator (NFVO) та VNF Manager. Вона має чітко визначені інтерфейси для взаємодії з іншими VNF, VNF Manager, Element Management (EM) та NFVI, а також має чітко визначену функціональну поведінку.

VNF може реалізовувати один мережевий елемент з інтерфейсами та поведінкою, визначеними стандартизаційними організаціями, як-от 3GPP або IETF. Альтернативно, VNF може включати групу мережевих елементів. У випадку реалізації групи елементів, внутрішні інтерфейси між ними зазвичай не відкриті для зовнішнього доступу. Незалежно від структури, усі VNF надають чітко визначені інтерфейси та поведінку, забезпечуючи ясність та передбачуваність у мережевих операціях.

Наприклад, мережевий елемент P-GW (Packet Data Network Gateway) може бути інтегрований з іншими мережевими елементами, такими як MME (Mobility Management Entity) і S-GW (Serving Gateway), щоб створити уніфікований мережевий елемент, відомий як "EPC VNF" (Evolved Packet Core Virtual Network Function). "EPC VNF" не обмежується лише сумісністю зі стандартними 3GPP інтерфейсами, які регулюють взаємодію між внутрішніми компонентами системи, але також пропонує чітко визначені зовнішні інтерфейси для цілої групи.

Це означає, що "EPC VNF" забезпечує взаємодію не тільки в межах своєї внутрішньої структури, але й взаємодію з зовнішніми системами та мережами. Такий підхід дозволяє ефективно інтегрувати та управляти ключовими функціями мережі 4G LTE.

Аналогічно, VNF можуть бути використані для реалізації менших функціональних одиниць, як-от TDF (Traffic Detection Function) або PCEF (Policy and Charging Enforcement Function). Це демонструє гнучкість VNF у плані масштабування та адаптації до різних функціональних потреб мережі, дозволяючи створювати більш складні та інтегровані мережеві рішення.

Під час процесу проектування та розробки програмного забезпечення постачальники VNF можуть розділити це програмне забезпечення на окремі програмні компоненти. Ці компоненти є частинами архітектури програмного забезпечення та можуть бути згруповані та упаковані в один або декілька образів для розгортання. Отже, ці програмні компоненти, визначені постачальником VNF, відомі як компоненти VNF (VNFC).

VNF реалізуються за допомогою одного або декількох VNFC, і передбачається, що кожен екземпляр VNFC відповідає інтерфейсу віртуального контейнера NFVI (Network Function Virtualization Infrastructure) у співвідношенні один до одного. Це означає, що кожен VNFC працює в своєму віртуальному контейнері, і така структура дозволяє програмному забезпеченню VNF ефективно функціонувати в мережевому середовищі, використовуючи можливості NFVI.

Спосіб, яким постачальник VNF організує VNF із набору VNFC (Virtual Network Function Components), залежить від цілого ряду факторів. Ці фактори можуть включати пріоритетність задач, продуктивність, масштабованість, надійність, безпеку, а також інші нефункціональні аспекти. До таких аспектів належать інтеграція компонентів від інших постачальників VNF, операційні міркування, а також існуюча база коду.

Важливо зазначити, що кожен постачальник VNF має свій унікальний підхід до побудови VNF, і неможливо створити два однакових VNF, складені з ідентичних VNFC. Різноманітність у підходах та вимогах до реалізації VNF означає, що кожен

VNF буде унікальним у своєму складі та функціональності, відповідаючи специфічним потребам та цілям постачальника.

Ще однією типовою характеристикою VNFC є те, що їхня функціональність і інтерфейси між VNFC змінюються з часом як частина безперервного інноваційного процесу. Наприклад, постачальник VNF може вирішити збільшити продуктивність VNF шляхом покращення інтерфейсу між двома VNFC. Важливо зазначити, що такі зміни не впливають на зовнішні компоненти і обмежені в межах самої VNF. Це означає, що інтерфейси та функціональна поведінка VNFC не можуть вважатися постійними або важливими поза контекстом програмного забезпечення, розробленого постачальником VNF. Інтерфейси та поведінка VNFC не є чітко визначеними і можуть змінюватися від однієї версії програмного забезпечення до іншої, не залишаючись обов'язково стабільними. Ця гнучкість є ключовою для адаптації та інновацій у мережевих технологіях, але також вимагає від розробників уваги до управління змінами та сумісності між версіями.

VNF є абстрактним поняттям, що визначає програмний контракт. Конкретний екземпляр VNF, який функціонує в реальному часі, називається екземпляром VNF. VNFC є специфічним компонентом, що надається постачальниками VNF, тоді як VNFCI (Virtual Network Function Component Instance) є виконавчими компонентами, що формують екземпляр VNF.

Для реалізації віртуальної мережевої функції, визначеної VNF, VNF Manager створює один або декілька VNFCI, кожен з яких розміщується у своєму віртуалізованому контейнері. Ці VNFCI взаємодіють між собою, забезпечуючи функціональність VNF та реалізуючи необхідні інтерфейси.

Вимоги до стану розгортання VNF описані у дескрипторі VNF (VNFD), включаючи відносини між VNFCI, які є внутрішніми для VNF і не видимими ззовні. Операційні можливості після розгортання, такі як міграція віртуальних машин, що містять VNFCI, збільшення чи зменшення масштабу, масштабування та зміни мережевих з'єднань, також включені у VNFD.

VNFCI може здійснювати певні зміни у з'єднаннях між компонентами, але ті, що потребують змін у NFVI (окрім тих, що вносяться, наприклад, через апаратні

драйвери), реалізуються через NFV Management та Orchestration. Ці суб'єкти можуть виконувати зміни автоматично або за вказівками від VNF через VNFM для конкретної VNF.

Кожна віртуальна мережева функція має асоційований з нею єдиний VNFD. В VNFD, постачальник VNF надає гарантії як функціонального, так і нефункціонального характеру, такі як продуктивність, безпека та надійність. Ці гарантії можуть бути виконані лише до тих пір, поки зберігається цілісність VNF. Іншими словами, якщо один або декілька програмних компонентів VNF будуть замінені іншими компонентами, запевнення в VNFD втрачають свою силу.

Важливо зазначити, що в цілому VNF упаковується одним постачальником VNF. Якщо постачальник VNF інтегрує сторонні програмні компоненти у свою реалізацію VNF, то він несе відповідальність за надання комплекту VNF. Однак це не виключає можливості повторного використання програмних компонентів, які є спільними для декількох VNF. Для повторного використання такого спільного компонента, він повинен бути вилучений з оригінальних VNF та поміщений у новий тип VNF із чітко визначеними інтерфейсами.

Однією з основних цілей NFV є створення структури, в якій ключові мережеві можливості можуть бути відокремлені як VNF від інфраструктури, на якій VNF функціонують. Це дозволяє VNF масштабуватися при необхідності. Щоб правильно віртуалізувати ці можливості, важливо добре їх розуміти, визначати та каталогізувати для належного включення в оркестрування послуг.

Інтерфейс є точкою, де відбувається взаємодія між двома елементами, які можуть бути програмними або апаратними послугами, або ресурсами. Програмні інтерфейси забезпечують чітке розділення між специфікацією взаємодій програмних елементів та їх фактичною реалізацією у програмному забезпеченні. Вони створюють визначені точки для взаємодії між програмними компонентами та обмежують спілкування між ними цими точками. Цей документ не охоплює апаратні інтерфейси.

Зазвичай виокремлюють п'ять типів інтерфейсів, що стосуються програмної архітектури VNF. На рис. 1.2 представлено набір інтерфейсів, які були ідентифіковані в архітектурній структурі, визначеній в документі ETSI GS NFV 002.

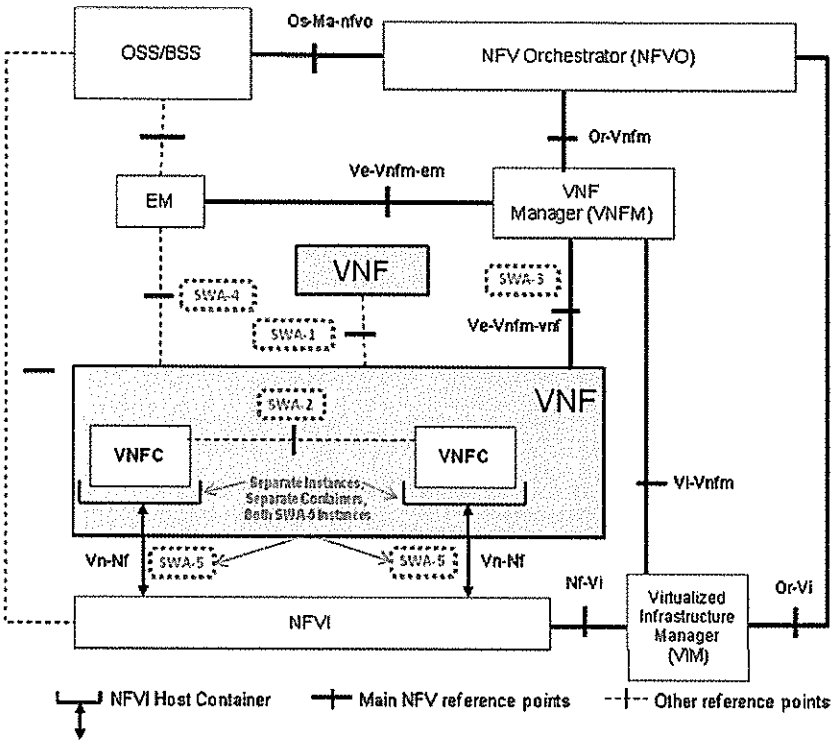


Рис. 1.2. Інтерфейси VNF та опорні точки архітектурної структури NFV

На рис. 1.2. представлено еволюційний огляд структури архітектури NFV, який надає детальну інформацію про аспекти, що стосуються програмної архітектури VNF. Ця інформація допомагає зрозуміти, як компоненти програмної архітектури VNF взаємодіють між собою.

При створенні двох або більше компонентів VNFC, кожен екземпляр VNFC розташовується в своєму унікальному контейнері віртуалізації. Інфраструктура NFV використовує декілька окремих екземплярів інтерфейсу SWA-5 (Vn-Nf), що дозволяє забезпечити доступ до середовища виконання кожного екземпляра VNFC. Це означає, що кожен VNFC може бути незалежно розміщений і управляється в рамках загальної інфраструктури NFV.

SWA-1 є чітко визначеним інтерфейсом, який забезпечує зв'язок між різними мережевими функціями всередині одного мережевого сервісу або перетинаючи

декілька сервісів. Цей інтерфейс може представляти з'єднання для передачі даних або для управління мережею, залучаючи як віртуальні мережеві функції, так і фізичні мережеві функції (PNF). SWA-1 розташований між парою VNF та PNF або між VNF та кінцевою точкою в мережі. Окрема VNF може підтримувати декілька інтерфейсів SWA-1.

Мережеві оператори або постачальники послуг використовують ці інтерфейси для створення мережевих послуг. Для того, щоб послуга була функціональною та ефективною, VNFD повинен надавати детальну інформацію про підтримувані SWA-1 інтерфейси. Це дозволяє оператору мережі або постачальнику послуг переконатися, що дана VNF сумісна з іншими VNF, які можуть бути отримані від інших постачальників VNF.

SWA-1 інтерфейси є логічними інтерфейсами, які використовують мережеві підключення доступні через інший інтерфейс, SWA-5. Це означає, що вони використовують існуючу інфраструктуру для забезпечення з'єднання між різними мережевими компонентами.

Інтерфейси SWA-2 відносяться до внутрішніх інтерфейсів усередині VNF, які використовуються для забезпечення комунікації між окремими VNFC. Визначення цих інтерфейсів лежить у відповідальності постачальників VNF, тому характеристики цих інтерфейсів можуть відрізнятися залежно від конкретного постачальника.

Зазвичай, інтерфейси SWA-2 встановлюють параметри продуктивності, такі як пропускна спроможність та затримка, щодо базової віртуалізованої інфраструктури. Однак, ці параметри зазвичай не є видимими для кінцевих користувачів VNF. Зазвичай вони висувають вимоги до функцій оркестрації та управління NFV, таких як VNF Manager, NFV Orchestrator (NFVO) та Virtual Infrastructure Manager (VIM), та загальної інфраструктури NFV.

Інтерфейси SWA-2 використовують основні механізми зв'язку, які реалізовані через інфраструктуру NFV. Це логічні інтерфейси, які переважно опираються на мережеві підключення, доступні через інтерфейс SWA-5.

SWA-3 є інтерфейсом, який забезпечує з'єднання між VNF та компонентами управління та оркестрації NFV, зокрема, із менеджером VNF (VNF Manager).

Основна роль SWA-3 полягає в тому, що через ці інтерфейси відбувається управління життєвим циклом VNF. Це включає процеси, як-от інстанціювання (створення екземплярів), масштабування (збільшення або зменшення ресурсів), та інші операції управління, які можуть бути застосовані до одного або групи VNF.

Щодо атрибутів з'єднання, інтерфейси SWA-3 можуть використовувати мережеві з'єднання, що базуються на IP (Інтернет-протокол) або на L2 (другий рівень OSI моделі, рівень каналу передачі даних), для забезпечення комунікації між VNF та системами управління.

Інтерфейс SWA-4 використовується Element Management (EM) для встановлення зв'язку з VNF. Цей інтерфейс управління дозволяє EM керувати VNF у процесі їх роботи, відповідно до принципів Fulfilment (Виконання замовлень), Assurance (Забезпечення якості), Billing (Білінг), відомих як FAB модель, та FCAPS (Fault, Configuration, Accounting, Performance, Security) - фреймворку управління мережею, який включає управління неполадками, конфігурацією, обліком, продуктивністю та безпекою.

SWA-5 визначає інтерфейси між кожною VNF та NFVI, яка є їх основою. Існує багато типів VNF, а також їхні колекції, такі як VNF Forwarding Graphs і VNF Sets. Кожен тип VNF використовує свій власний набір ресурсів NFVI, що може включати мережеві, обчислювальні, сховища та апаратні прискорювачі, доступ до яких забезпечується через інтерфейс SWA-5.

Інтерфейс SWA-5 відкриває доступ до віртуалізованих ресурсів NFVI, які були виділені для VNF, включаючи всі віртуальні обчислювальні, мережеві ресурси та засоби зберігання, що використовуються VNF. Таким чином, SWA-5 описує віртуалізоване середовище, в якому функціонує розгорнутий екземпляр VNF.

Інтерфейси SWA-5 служать як абстракція для всіх підінтерфейсів між NFVI та VNF. Кожен із цих підінтерфейсів має свої сценарії використання, ролі та атрибути для міжмережевого з'єднання, що визначають спосіб взаємодії VNF з інфраструктурою NFVI.

1.3 Принципи роботи VNF

ETSI (European Telecommunications Standards Institute) розробила ряд стандартів, серед яких одним з найбільш значущих є стандарт, що демонструє, як NFVI сприяє сегрегації апаратного та програмного забезпечення. Це відділення дає можливість більш гнучкої та ефективної роботи мережі, дозволяючи незалежно оновлювати та розвивати апаратне та програмне забезпечення.

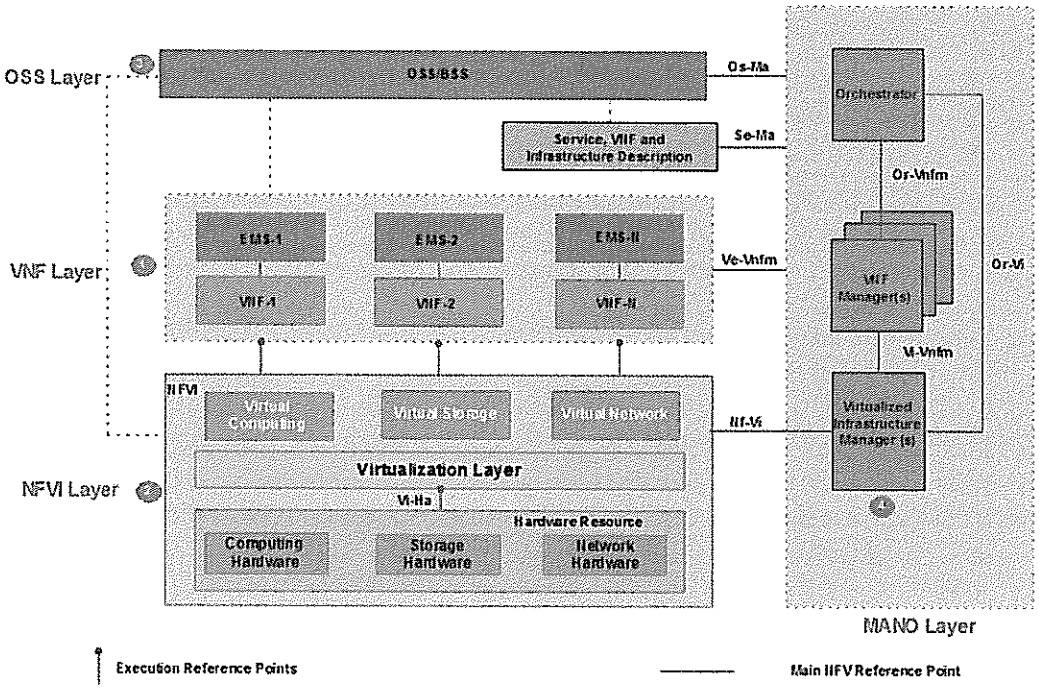


Рис. 1.3. Основні компоненти NFV

На рис. 1.3. представлені основні компоненти NFV. Ця структура поділяється на чотири ключові рівні:

- Рівень Мережевих Функцій Віртуалізації (VNF).
- Рівень Інфраструктури NFV (NFVI).
- Рівень Підсистеми Підтримки Експлуатації (OSS).
- Рівень Управління та Оркестрування (MANO).

Рівень Мережевих Функцій Віртуалізації (VNF) складається з двох основних компонентів: Віртуальної Мережевої Функції (Virtual Network Function, VNF) і Системи Управління Елементами (Element Management System, EMS).

Віртуальна Мережева Функція (VNF) є основою архітектури NFV. Вона представляє собою віртуалізовану версію традиційних мережевих функцій. Наприклад, віртуалізований маршрутизатор називається Router VNF, віртуалізована базова станція - базовою станцією VNF, і так далі. Це може бути також DHCP-сервер VNF або брандмауер VNF. Якщо віртуалізується специфічна підфункція мережевого елемента, вона також вважається VNF. Наприклад, у складі віртуалізованого Evolved Packet Core, компоненти, такі як MME, шлюзи та HSS, можуть бути окремими VNF.

VNF зазвичай розгортаються на віртуальних машинах (VM). Одна VNF може бути розгорнута на декількох VM, де кожна VM виконує одну функцію VNF, або весь VNF може бути розгорнутий на одній віртуальній машині.

Система Управління Елементами (EMS) відповідає за управління функціональними аспектами VNF. Це включає управління несправностями, конфігурацією, обліком, продуктивністю та безпекою VNF. EMS може керувати кожною VNF через свої власні інтерфейси. Може існувати окрема EMS для кожної VNF, або одна EMS, яка управляє кількома VNF. Сама EMS також може бути реалізована як VNF.

Рівень Інфраструктури NFV (NFVI) складається з апаратних та програмних компонентів, які разом створюють середовище для розгортання, управління та виконання VNF. INFV може фізично охоплювати кілька локацій, з мережею, яка забезпечує зв'язок між цими місцями, щоб вони могли функціонувати як єдина інфраструктура NFV.

Компоненти Інфраструктури NFV:

- Апаратні ресурси.
- Рівень Віртуалізації.

З точки зору VNF, рівень віртуалізації та апаратні ресурси вважаються єдиним цілим, оскільки разом вони забезпечують необхідні ресурси для функціонування VNF.

Апаратні ресурси це ресурси які включають обчислювальні засоби, системи зберігання даних та мережеве обладнання. Вони відіграють ключову роль у

забезпеченні потужності для обробки даних, зберігання інформації та підключення VNF до мережі. Ресурси обчислення та зберігання зазвичай використовуються в пулі, що дозволяє ефективно управляти та розподіляти ці ресурси серед різних VNF.

Рівень віртуалізації, відомий також як гіпервізор, відіграє ключову роль у архітектурі NFV. Його основні функції включають:

- Абстрагування та Логічне Розділення Фізичних Ресурсів: Гіпервізор абстрагує фізичні ресурси (наприклад, обчислювальні потужності, пам'ять, мережеві інтерфейси) та представляє їх у вигляді віртуальних ресурсів, доступних для VNF. Це дозволяє створити рівень абстракції, який відокремлює програмне забезпечення VNF від фізичного обладнання.
- Надання Можливостей для Реалізації VNF: Гіпервізор дозволяє програмному забезпеченню VNF використовувати базову інфраструктуру віртуалізації. Це означає, що VNF можуть бути розгорнуті та виконуватися на віртуалізованій інфраструктурі без залежності від конкретного фізичного обладнання.
- Надання Віртуалізованих Ресурсів для VNF: Гіпервізор забезпечує VNF необхідними віртуалізованими ресурсами, такими як віртуальні машини, віртуальне сховище та мережеві засоби. Ці ресурси є необхідними для ефективного виконання VNF.

Гіпервізор також виконує роль проміжного рівня між віртуальними машинами та фізичними ресурсами. Це означає, що програмне забезпечення може бути розгорнуте на різних фізичних ресурсах, надаючи необхідну гнучкість та масштабованість для різноманітних мережевих сервісів і застосунків.

2 ГЛИБОКИЙ АНАЛІЗ ТА ОЦІНКА ВІРТУАЛІЗАЦІЇ ФУНКЦІЙ МЕРЕЖІ У КОНТЕКСТІ МОБІЛЬНИХ ТЕХНОЛОГІЙ

2.1 Аналіз сучасного стану використання VNF у мобільних системах

Наступне покоління мобільних мереж, 5G, відкриває шлях для нових типів додатків, послуг та сценаріїв використання, які базуються на трьох основних областях:

- Покращений Мобільний Широкосмуговий Зв'язок (eMBB): Цей аспект зосереджений на забезпеченні значно вищої швидкості передачі даних та покращеній пропускній спроможності порівняно з попередніми поколіннями мобільних мереж.
- Наднадійний Зв'язок з Низькою Затримкою (URLLC): Це включає технології, спрямовані на забезпечення високої надійності та низької затримки, що є критично важливим для додатків, таких як автономні транспортні засоби та промислові автоматизовані системи.
- Масовий Зв'язок Машинного Типу (mMTC): Сфокусований на підключенні великої кількості IoT (Internet of Things) пристроїв, що вимагають мінімального енергоспоживання та невеликої пропускної спроможності.

5G використовує широкий спектр передових технологій, включаючи хмарні обчислення, NFV, програмно-визначені мережі (SDN), нове радіо (NR), штучний інтелект (AI), і багато інших. Зокрема, NFV дозволяє відокремити програмні компоненти мережевих функцій від спеціалізованого апаратного забезпечення, перетворюючи мобільну мережу на гнучкий ланцюг модульних мережевих функцій, розгорнутих у хмарному середовищі, замість використання великих монолітних систем з застарілим обладнанням.

Ця нова парадигма NFV значно прискорює розгортання різних обчислювальних майданчиків, які налаштовуються відповідно до потреб

оркестрації та сервісів. Ми можемо виділити дві основні категорії обчислювальних майданчиків: периферійні хмари та центральні хмари.

- Центральні Хмари: Центральна хмара зазвичай включає кілька потужних обчислювальних ресурсів та систем зберігання, які можуть бути розміщені на значній відстані один від одного, до кількох сотень кілометрів. Ці хмари ідеально підходять для виконання інтенсивних віртуалізованих робочих навантажень, які толерантні до затримок у мережі.
- Периферійні Хмари: Вони розташовуються близько до RAN (Radio Access Network), що дозволяє зменшити затримки та джиттер. Периферійні хмари мають менші обчислювальні ресурси та можливості зберігання порівняно з центральними хмарами, але вони критично важливі для забезпечення послуг з високою пропускною здатністю та низькою затримкою.

ETSI API для мобільних периферійних обчислень (Mobile Edge Computing, MEC) підтримує нові сервіси, які вимагають високої пропускної здатності, низької затримки та/або розподілених обчислень. Ці сервіси будуть розміщуватися в нових точках присутності операторів (Points of Presence, PoP). Ця парадигма також дозволяє більш гнучко керувати пулом ресурсів, пропонуючи можливість розгортання VNF за вимогою.

Основними вимогами до прямого з'єднання між периферійним центром обробки даних та точкою радіодоступу є забезпечення низької затримки та високої пропускної здатності, що є критично важливим для ефективності 5G мереж.

Залежно від конкретних сценаріїв використання, VNF можуть бути налаштовані різними способами для досягнення ефективності 5G Slicing. У 5G, зв'язок надається через створення мережевих "зрізів", кожен з яких представляє собою комбінацію декількох доменів – від хмари до кінцевого пристрою – з унікальною конфігурацією для кожного окремого випадку використання.

Традиційні технології віртуалізації можуть зіткнутися з труднощами, пов'язаними з масштабованістю, доступністю та швидкістю роботи, що робить

створення зрізів на рівні VNF складним завданням. Однак, підхід Cloud Native (CN) пропонує вирішення цих проблем шляхом застосування нових принципів проектування мережевих функцій.

Перехід від застарілої інфраструктури до CN включає не тільки розвиток VNF за рахунок відмови від емуляції апаратного забезпечення та застосування модульного дизайну, але й впровадження абсолютно нової архітектури. Ця архітектура орієнтована на такі вимоги, як розподіленість, автоматизація, слабкий зв'язок між компонентами, бездержавність (statelessness) та самовідновлення. Це включає також інтеграцію спостережуваності та комунікації через API.

Оператори мобільного зв'язку (MNO) активно співпрацюють з організаціями з розробки стандартів, щоб визначити ключові компоненти 5G NFV, такі як оркестрація, управління життєвим циклом, вимоги до розгортання, продуктивності та безпеки. В цьому контексті пропонується короткий огляд їхньої діяльності, включаючи розуміння принципів проектування та застосування Cloud Native, а також приклади використання цих підходів. Виокремлено ключові виклики, пов'язані з переходом систем NFV від застарілого дизайну до новітнього підходу Cloud Native.

Також представлено пропозицію щодо фреймворку 5G VNF, який підтримує технології мультирадіозв'язку, забезпечуючи високошвидкісний зв'язок для гравців Over-The-Top (OTT) та операторів мобільних віртуальних мереж (MVNO). Ця пропозиція включає використання відкритого програмного забезпечення і Cloud Native дизайну для ефективного та багатокористувацького доступу до хмарних ресурсів мобільних операторів. Вона зосереджена на двох основних аспектах: визначенні викликів і можливостей, що виникають при переході від традиційного дизайну NFV до Cloud Native, а також на фреймворку з відкритим вихідним кодом, який використовується для Cloud Native VNF.

Кілька проектів з відкритим вихідним кодом пропонують власні версії фреймворку NFV-MANO (Network Functions Virtualization Management and Orchestration) з різними рівнями зрілості. Наприклад, існують такі проекти, як Open Source MANO (OSM), Open Networking Automation Platform (ONAP) з відкритим

вихідним кодом, і OpenBaton. Деякі з цих проектів є результатом промислової співпраці та надають гарантії щодо підтримки та стабільності проекту.

Більшість цих проектів традиційно використовують OpenStack як Virtual Infrastructure Manager (VIM) за замовчуванням. Однак, нещодавно деякі з них розпочали інтеграцію з Kubernetes, розширюючи свої можливості для підтримки контейнерізованих середовищ. В табл. 2.1. підсумовано основні проекти та їх відповідні сценарії використання, що дозволяє краще зрозуміти їх функціонал та призначення.

Таблиця 2.1. Основні проекти VNF та їх відповідні сценарії використання

Проект	ETSI NFV-MANO	Організація	Підтримуванні VIMs	Приклади застосування
OSM	NFVO, VNFM, VIM, NFVI, VNF Catalog, NS Catalog	ETSI	OpenStack, VMware, OpenVIM	service chaining 5G slicing MEC
ONAP	NFVO, VNFM, VIM, NFVI, VNF Catalog, NS Catalog	Linux Foundation	OpenStack	vCPE VoLTE
OPNFV	NVFI	Linux Foundation	OpenStack, Kubernetes	vEPC vCDN
Cloudify	NFVO, VNFM, VIM, NFVI	Cloudify Platform Ltd.	OpenStack, VMware, AWS, Docker	vCPE SD-WAN
Open Baton	NFVO, VNFM, VIM, NFVI	Fraunhofer FOKUS and TU Berlin	OpenStack, Docker	multi-site extensibility interoperability vIMS
SONATA	NFVO, VNFM, VIM, NFVI, VNF Catalog, NS Catalog	5G PPP 5GTANGO	OpenStack, OpenVIM	IoT vCDN vEPC

Open Source MANO (OSM), який був представлений у 2016 році, є проектом ETSI (European Telecommunications Standards Institute) з відкритим вихідним кодом, реалізуючи стандарт NFV-MANO. Проект OSM включає у себе широку екосистему учасників, що об'єднує понад 100 організацій, включаючи хмарних провайдерів, розробників програмного забезпечення, телекомунікаційні компанії, виробників обладнання, а також представників з академічного світу.

На даний момент OSM досяг четвертого релізу. Один з ключових елементів цього релізу це NorthBound Interface (NBI), який надає універсальний інтерфейс для майбутньої інтеграції з різними клієнтами. Компоненти OSM мають хмарний дизайн, що дозволяє оптимізувати розмір та прискорити процес розгортання.

OSM зосереджений на таких варіантах використання, як ланцюжок послуг (Service Chaining) та нарізка мережі (Network Slicing), оркестрація в архітектурі Mobile Edge Computing (MEC) та різних вертикальних сценаріях 5G. Ці функції роблять OSM важливим інструментом для розвитку і реалізації сучасних телекомунікаційних та хмарних технологій.

Проект Open Network Automation Platform (ONAP) Фонду Linux є відкритим проектом, ініційованим оператором мобільного зв'язку AT&T, який швидко набуває популярності в телекомунікаційній індустрії. ONAP зосереджений на спрощенні процесів проектування, створення, оркестрування, моніторингу та управління життєвим циклом VNF, SDN та послуг вищого рівня.

ONAP пропонує два основних фреймворки:

1. Час Проектування (Design-Time, DT): Цей фреймворк включає три ключові компоненти для проектування та створення сервісів, управління автоматизацією замкненого циклу, а також VNF SDK для створення ланцюжків послуг та інтеграції мережевих сервісів.
2. Час Виконання (Run-Time, RT): Цей фреймворк забезпечує глобальне уявлення топології та розподілених ресурсів, інтегруючись з VNFM, VIM та NFVI. Він відповідає за наскрізний контроль та управління сервісами, включаючи аналітику для моніторингу системи.

ONAP фокусується на таких сценаріях використання, як віртуальні клієнтські приміщення обладнання (vCPE) та передача голосу через LTE (VoLTE), пропонуючи новітні рішення для управління та автоматизації мережевих послуг.

Cloudify – це проект оркестрування NFV, який базується на специфікаціях TOSCA (Topology and Orchestration Specification for Cloud Applications). Ця платформа призначена для управління та контролю життєвого циклу VNF та базового NFVI.

Механізм оркестрування в Cloudify включає в себе механізми топології, робочих процесів та політики, що дозволяють ефективно управляти та автоматизувати різноманітні аспекти мережевих функцій та інфраструктури. Система Cloudify також пропонує набір інструментів проектування, включаючи Інтерфейс Управління Розгортанням та Blueprint Composer, які допомагають у проектуванні та реалізації комплексних схем для NFV.

Ці інструменти та можливості роблять Cloudify важливим рішенням для оркестрування у сфері NFV, дозволяючи операторам та розробникам ефективно управляти життєвим циклом VNF і NFVI.

Відкрита Платформа для Віртуалізації Мережевих Функцій (OPNFV) – це колаборативна платформа з відкритим вихідним кодом, призначена для оцінки інтеграції та розгортання VNF (Віртуальні Мережеві Функції) і базової інфраструктури, що є важливою для майбутнього 5G. Платформа була запущена Фондом Linux Foundation у 2014 році з метою об'єднати велику частину учасників телекомунікаційної індустрії, включаючи постачальників, операторів телекомунікацій, а також провідних IT-гравців.

Основний акцент OPNFV на інтеграції NFVI і розгортанні VNF. Проект активно використовує та інтегрує такі відомі технології, як OpenDaylight, OpenStack, KVM, Open vSwitch та Linux, проводячи їхнє всебічне тестування.

Наразі, проект OPNFV працює над своїм шостим релізом, під назвою Fraser, в якому продовжує розвивати концепцію Cloud Native VNFs, яка була вперше представлена у попередньому релізі. Ця робота є ключовою для підтримки розвитку і розгортання інноваційних телекомунікаційних рішень на базі 5G.

Open Baton – це проект з відкритим вихідним кодом, який був запущений Фраунгофером FOKUS і Берлінським Технічним Університетом. Цей проект створено з метою забезпечення середовища NFV, базуючись на специфікаціях NFV-MANO. Open Baton включає NFVO для управління та оркестрування VNF.

Проект використовує декілька VNFM (VNF Managers) у співпраці з NFVO для управління життєвим циклом VNF. Також Open Baton надає SDK для сприяння розробці нових VNFM. Нещодавно в рамках проекту було розроблено новий VIM (Virtual Infrastructure Manager) драйвер для інтеграції з Docker, що дозволяє використовувати Docker контейнери в середовищі Open Baton.

Open Baton зосереджений на розширюваності, щоб забезпечити інтеграцію з різними VIM та підтримку сценаріїв, які охоплюють багато сайтів. Це робить Open Baton гнучким інструментом для розвитку та впровадження NFV-рішень у різноманітних середовищах.

SONATA – це проект, що належить до ініціативи 5GPPP (5G Public-Private Partnership) в рамках програми H2020, який забезпечує реалізацію специфікацій архітектури NFV-MANO. Він використовує декілька програмних компонентів з проекту 5GPPP 5GTANGO.

Крім основних функціональних модулів NFV-MANO, таких як NFVO, VNFM та VIM, SONATA пропонує багату екосистему, яка включає сховища даних для мережевих функцій та послуг, інструменти моніторингу та засоби для розподілу інфраструктурних ресурсів. Також, проект надає SDK для пакування VNF, управління сервісами та моніторингу.

SONATA вважається однією з найбільш передових реалізацій специфікацій NFV-MANO серед проектів 5GPPP. Проект орієнтований на різноманітні сценарії використання, включаючи Інтернет речей (IoT) для моніторингу та оптимізації мережевого трафіку, віртуальні мережі доставки контенту (vCDN) для підвищення масштабованості та автоматичної конфігурації, а також віртуальне еволюційне пакетне ядро (vEPC) для покращення масштабованості та відмовостійкості, оптимізуючи розміщення мережевих ресурсів.

NGMN (Next Generation Mobile Networks) описує різноманітні нові бізнес-моделі, які очікуються з появою 5G. Серед них виділяються такі типи постачальників, як постачальники активів, постачальники послуг зв'язку та провайдери партнерських послуг, які працюють за моделлю XaaS (Everything as a Service). Одна з ключових вимог цих моделей полягає в тому, що постачальники послуг повинні мати можливість конфігурувати та управляти послугою, в той час як оператори відповідатимуть за керування та розвиток мережі.

5G розглядається як програмна платформа для надання послуг (5G as a Service, 5GaaS), яка підтримує технології мульти-радіодоступу через VNF та надає можливості підключення для OTT-плеєрів (Over-The-Top) та MVNO (Мобільних Віртуальних Мережевих Операторів). Ця платформа також може використовуватися у сценаріях спільного використання радіочастотного спектра та участі у глобальній еволюції RAN (Радіодоступної Мережі).

2.2 Вивчення ефективності та гнучкості VNF

Системи управління та оркестрування NFV (MANO) були розроблені для забезпечення гнучкого управління віртуалізованими мережевими послугами в контексті мереж 5G. В цьому контексті ETSI ISG NFV (European Telecommunications Standards Institute Industry Specification Group for Network Functions Virtualization) визначила стандартну систему NFV MANO, яка служить як еталон для постачальників систем MANO, а також для проектів MANO з відкритим вихідним кодом. Це включає розробку систем, які можуть гнучко управляти та оркеструвати мережеві послуги в швидко розвиваючихся мережах 5G, що є критично важливим для ефективності та адаптивності мережевих операцій.

Відсутність ключових показників ефективності (KPI) для систем управління та оркестрування NFV (MANO) створює труднощі для операторів мобільного зв'язку в процесі вибору найбільш підходящої системи MANO, яка б задовольняла їхні потреби. Без чітко визначених KPI, продуктивність системи MANO не може бути кількісно оцінена або порівняна, залишаючи операторів лише з можливістю

порівняти перелік функцій, заявлених різними постачальниками. Тому ці KPI стають важливими для аналізу та порівняння продуктивності різних систем MANO і виявлення можливих прогалин у продуктивності.

Рівні MANO представлено на рис. 2.1.

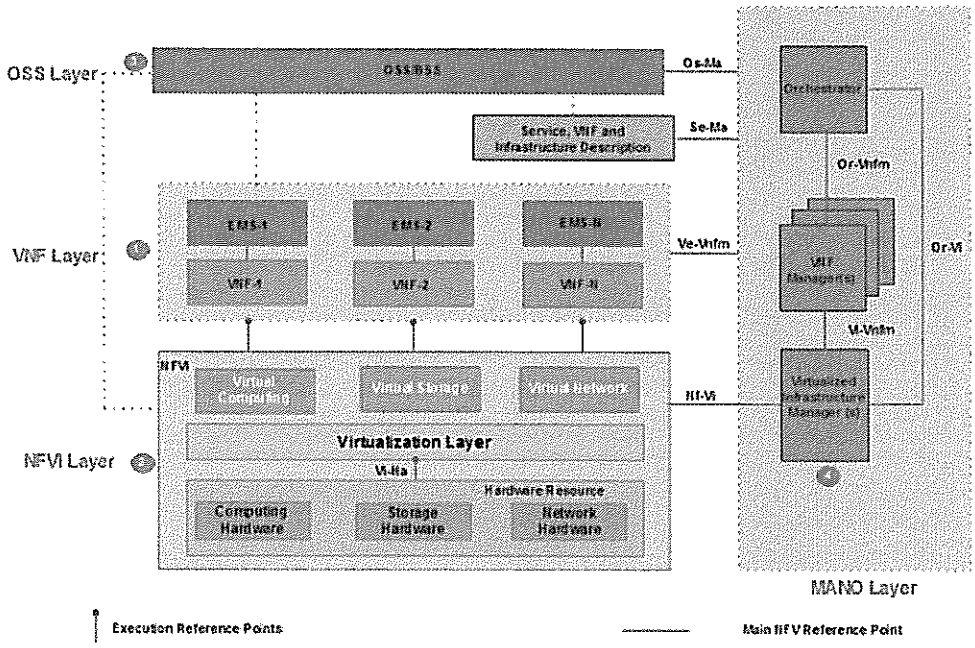


Рис. 2.1. Основні рівні MANO

У процесі бенчмаркінгу KPI для систем управління та оркестрування NFV (MANO) виникають певні виклики:

- Доступність: Важливо забезпечити доступність мережевих елементів та послуг. Це означає, що мережеві ресурси мають бути доступні та надійні для користувачів та послуг, що використовують їх.
- Використання: Необхідно гарантувати максимально ефективне використання мережевих ресурсів. Це означає оптимізацію розподілу ресурсів для запобігання їх недостатності або надлишку.
- Угоди про Рівень Обслуговування (SLA): Важливо забезпечувати, що користувачі та послуги не використовують ресурси понад передбачені обсяги, відповідно до SLA.

- Затримка: Необхідно гарантувати, що послуги надаються в межах встановленого бюджету затримки, щоб підтримувати якість обслуговування (QoS) та якість досвіду (QoE).
- Джиттер: Важливо забезпечити, що час між прибуттям пакетів не відхиляється від середньої затримки більше, ніж на визначений рівень.
- Помилки/Впорядкування: Необхідно забезпечувати підтримку наскрізної цілісності сервісу, захищеного від помилок та збоїв.

Ці виклики важливі для забезпечення, що системи MANO можуть ефективно управляти мережевими ресурсами та послугами в мережах 5G, задовольняючи високі стандарти надійності, продуктивності та якості обслуговування.

Існує велика кількість вимірюваних параметрів, які вписуються у загальні категорії ключових показників ефективності, які моніторяться та підтримуються системами управління та оркестрування мережевих функцій (NMS). Ці KPI добре відомі, мають чітке визначення, і включають у себе вимірювання продуктивності обчислювальних та мережевих ресурсів NFVI, а також VNF.

Наприклад, продуктивність віртуалізованої функції маршрутизатора може бути оцінена за допомогою традиційних KPI, аналогічних тим, що використовуються для оцінки традиційних маршрутизаторів. Це означає, що хоча технології мережевої віртуалізації є відносно новими, методи оцінки та управління їх продуктивності можуть бути подібними до тих, що вже використовуються в традиційних мережевих середовищах.

Проблема визначення ключових показників ефективності для порівняльного аналізу продуктивності систем NFV MANO є значною. Особливо важливим це стає, враховуючи, що система NFV MANO, крім здійснення традиційного управління FCAPS (Fault, Configuration, Accounting, Performance, Security), також відповідає за управління життєвим циклом (LCM) VNF та мережевих послуг (NS).

Важливість наявності KPI для кількісної оцінки продуктивності систем MANO підкреслюється гнучким, динамічним характером віртуалізованих послуг. Швидкість реакції системи NFV MANO на моніторинг подій і виконання відповідних дій управління життєвим циклом є критично важливою, зважаючи на

суворі вимоги до продуктивності різних вертикалей, які використовують спільні ресурси NFVI.

Така порівняльна система MANO дозволить клієнтам вибирати відповідне рішення MANO, яке найкраще відповідає їхнім оперативним потребам, забезпечуючи таким чином оптимальне використання ресурсів і забезпечення високого рівня продуктивності та надійності мережевих послуг.

KPI системи управління та оркестрування NFV можуть бути розділені на дві основні категорії: функціональні KPI та операційні KPI.

1. Функціональні KPI описують характеристики системи MANO, які не залежать від часу виконання. До цієї категорії відносяться:
   - Ресурсний слід розгорнутої системи MANO.
   - Різноманітність платформ VIM, які може підтримувати система MANO.
   - Кількість віртуальних машин, якими може ефективно керувати одна платформа MANO.
   - Максимальна кількість VNF, яку система MANO може ефективно відстежувати та керувати в NFVI.
   - Палітра функцій, таких як підтримка DevOps, управління пакетами VNF, управління зображеннями VNF, інтегрована система моніторингу тощо.
2. Операційні KPI характеризують операції, які відбуваються під час виконання. Вони зазвичай кількісно вимірюються шляхом визначення часової затримки певної процедури або завдання управління життєвим циклом та її ефективності.

Ці KPI дають уявлення про швидкість та ефективність системи MANO у відповіді на різні управлінські завдання. На рис. 2.2., зображені різні операції, які можуть бути виконані на MANO.

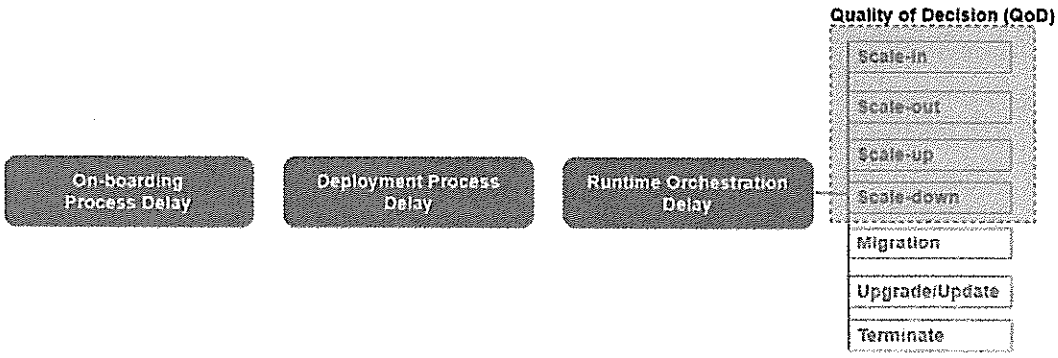


Рис. 2.2. Різні операції, які можуть бути виконані на MANO

Затримка процесу завантаження (OPD) описує час, який необхідний для завантаження образу віртуалізованої мережевої функції, тобто віртуальної машини з усіма її ресурсами. Після завантаження ВМ можна використовувати для розміщення та запуску служби VNF. Цей процес схожий на час розгортання сервісу, що передбачає створення образу програмного забезпечення VNF у форматі, який розпізнається системою MANO, і завантаження цього пакету до відповідного репозиторію.

Пакет включає не тільки програмне забезпечення VNF, але й файл дескриптора VNF (VNFD), який містить всю інформацію про конфігурацію, мережеві вимоги, вимоги до ресурсів, політику маршрутизації/безпеки, діапазони IP-адрес, вимоги до продуктивності, інтерфейси тощо. До пакету також входить дескриптор мережевих послуг (NSD).

NSD є шаблоном, який описує вимоги до мережевої послуги з погляду функцій, роботи, безпеки, зв'язку, якості обслуговування (QoS), якості досвіду (QoE), надійності та підключення. Він включає граф переадресації VNF, який визначає типи VNF та порядок їх підключення, а також характеристики віртуальних каналів, що з'єднують складові VNF для створення мережевої послуги. Затримка процесу завантаження (OPD) залежить від вимог до ресурсів сервісу, визначених в VNFD.

Затримка процесу розгортання (DPD) стосується часу, який необхідний для розгортання та створення екземпляра віртуалізованої мережевої функції у завантаженій віртуальній машині та налаштування функціонуючої мережевої

служби. У цьому процесі екземпляр служби створюється шляхом аналізу файлів дескриптора мережевих послуг та файлів VNFD.

Всі VNF, які входять до складу мережевої служби, створюються на основі відповідних вбудованих образів. Система MANO відповідає за надання необхідних ресурсів для створення інстанції VNF та їх з'єднання через відповідні віртуальні лінії у випадку складної NS. Потім система MANO налаштовує кожен VNF відповідно до інформації про конфігурацію, наданої у відповідних NSD і VNFD.

Швидкість розгортання VNF або NS є критично важливою, особливо коли VNF/NS потрібно масштабувати для задоволення раптового збільшення навантаження. Це важливо для забезпечення ефективного та гнучкого управління мережевими ресурсами у динамічному мережевому середовищі.

Затримка оркестрування під час виконання (ROD) стосується часу, який потрібен для виконання різних процедур управління в рамках системи управління та оркестрування NFV. Ця затримка визначається як різниця в часі між початком і завершенням кожної окремої дії управління.

Наприклад, система MANO, яка може швидко завершити операцію масштабування або міграції високонавантаженої VNF з мінімальним перериванням обслуговування, вважається високопродуктивною. ROD залежить від ефективності системи моніторингу, яка постійно відстежує активні екземпляри VNF/NS на протязі всього часу їх експлуатації для виявлення будь-яких відхилень у продуктивності або збоїв.

Таким чином, система MANO, яка виконує операції з низькою затримкою під час виконання і при цьому не створює значного навантаження на систему моніторингу, вважається ефективною з точки зору управління мережевими ресурсами та послугами.

Якість прийняття рішень (QoD) є важливою метрикою для кількісної оцінки продуктивності системи MANO, особливо з точки зору її ефективності при виконанні операцій управління життєвим циклом, таких як масштабування та міграція віртуалізованих мережевих функцій. QoD визначається наступними критеріями:

1. Ефективність рішення щодо управління ресурсами:
   - Чи будуть задоволені довгострокові та короткострокові потреби в ресурсах керованої VNF у вибраному обчислювальному вузлі.
   - Наскільки ненав'язливими були управлінські дії для інших VNF, які вже забезпечені ресурсами у вибраному обчислювальному вузлі. Це оцінюється за тим, наскільки керована віртуальна машина VNF вплине на продуктивність інших VNF у вибраному обчислювальному вузлі з точки зору доступності ресурсів.
2. Кількість спроб, необхідних для визначення найбільш підходящого обчислювального вузла для міграції/масштабування керованої VNF:
   - Цей критерій вимірює ефективність системи MANO у визначенні оптимального розташування для VNF.
3. Своєчасність обчислень та виконання дій MANO LCM (управління життєвим циклом):
   - Оцінка часу, необхідного для здійснення та виконання рішень, що стосуються управління життєвим циклом VNF.

QoD, таким чином, відображає здатність системи MANO швидко та ефективно адаптуватися до змін у потребах та навантаженнях мережі, гарантуючи при цьому оптимальне використання ресурсів та мінімальний вплив на інші функції та послуги мережі.

2.3 Проблеми та виклики при інтеграції VNF

У поспіху з пропозицією віртуалізованих мережевих функцій багато постачальників обрали шлях простого перенесення свого існуючого програмного забезпечення з традиційного апаратного середовища, такого як ASIC, FPGA або мережеві процесори, на комерційно доступні серверні платформи (COTS), зокрема на базі x86. Вони не змогли (або не бажали) здійснити повноцінну реархітектуру свого програмного забезпечення для ефективного використання можливостей

нового середовища NFV, що призвело до того, що їхні програмні рішення не використовують усіх переваг віртуалізації.

В результаті оператори стикаються з тим, що багато VNF потребують значних обхідних шляхів, щоб адекватно працювати на обраній універсальній платформі CPE. Для забезпечення сумісності між можливостями базового апаратного забезпечення та VNF, які працюють на ній, потрібно виконати велику роботу з інтеграції. Це може включати адаптацію апаратної платформи або налаштування самих VNF.

Поширена проблема, з якою стикаються віртуалізовані мережеві функції, полягає в відсутності повнофункціонального інтерфейсу програмування (API) для ефективної інтеграції з системами управління, розташованими на вищому рівні. Іноді для управління VNF необхідно використовувати спеціалізоване програмне забезпечення для Windows, замість того, щоб мати можливість користуватися простим веб-браузером.

Іншою проблемою є неадекватна документація для VNF, яка не надає достатньо інформації про процес розгортання та усунення несправностей. Часто бракує детальної та змістовної інформації в журналах подій та сигналах тривоги, що ускладнює діагностику та виправлення помилок під час встановлення та експлуатації VNF.

Крім того, оператори стикаються з практичними проблемами у роботі VNF. Наприклад, серійні номери VNF можуть встановлюватися за допомогою команд оболонки, які працюють коректно при введенні з консолі, але можуть викликати помилки при виконанні через захищену оболонку (SSH). Ці "диявольські деталі" часто ускладнюють процес інтеграції та використання VNF.

Однією з проблемних сфер для багатьох віртуалізованих мережевих функцій є використання методів для підвищення пропускної спроможності на пристроях, що базуються на x86 архітектурі. З переходом підприємств від традиційного підключення через TDM (Time Division Multiplexing) до підключень на основі Ethernet у глобальних мережах, пристрої CPE (Customer Premises Equipment),

розташовані на кордоні між WAN і LAN, все більше набувають ознак серверних платформ на базі x86, які використовуються в дата-центрах.

Серверна технологія на базі x86 давно здатна виконувати такі мережеві функції, як маршрутизація та захист брандмауером, проте до недавнього часу її продуктивність відставала від можливостей більш спеціалізованих пристроїв. Це означає, що хоча сервери на базі x86 здатні виконувати ці функції, вони не завжди можуть забезпечити ту ж рівень продуктивності та ефективності, що і пристрої, спеціально розроблені для цих цілей.

У проведених AT&T тестах було виявлено, що продуктивність обробки пакетів віртуальними маршрутизаторами на 20-35% нижча, ніж у фізичних маршрутизаторів із порівнянною обчислювальною потужністю. AT&T співпрацювала зі своїми постачальниками, щоб використовувати переваги новітніх технологій, таких як Data Plane Development Kit (DPDK) та Single Root I/O Virtualization (SR-IOV), на додаток до інших загальних налаштувань продуктивності програмного забезпечення. Це було зроблено з метою оптимізації вводу/виводу та обробки процесором для підвищення продуктивності обробки пакетів на пристроях COTS, які базуються на архітектурі x86.

Open vSwitch (OVS) був створений як альтернативний варіант комутатора для віртуалізації обладнання. Він пропонує додаткову функціональність порівняно зі стандартним мостом Linux, включаючи підтримку VXLAN, OpenFlow, GRE, QoS та детальне управління трафіком. Однак це призводить до зниження продуктивності через збільшення обсягу обробки даних для кожного перемикання кадрів.

Intel розробила Data Plane Development Kit для вирішення цієї проблеми. DPDK використовує спеціалізовані бібліотеки для прискорення обробки пакетів на обладнанні x86. Щоб використовувати ці переваги, модуль VNF повинен включити віртуальний драйвер, що підтримує віртуальне перемикання з DPDK. Таким чином, VNF здатен безпосередньо копіювати пакети з фізичної мережевої карти машини у свій простір пам'яті (і навпаки), уникаючи необхідності копіювання через простір ядра хост-ОС.

Хоча спочатку DPDK був розроблений Intel, наразі він є проектом з відкритим кодом, що підтримується на різних архітектурах процесорів, включаючи ARM та PowerPC.

Single Root I/O Virtualization (SR-IOV) значно покращує продуктивність віртуальної комутації, дозволяючи Ethernet-кадрам передаватися безпосередньо з фізичної мережевої карти на віртуальну мережеву карту VNF без обробки гіпервізором або проміжним віртуальним комутатором чи мостом, як це відбувається при використанні OVS або мосту Linux. З SR-IOV, кожна VNF має прямий доступ до фізичної мережевої карти, тобто коли кадр надходить на фізичну мережеву карту, він одразу поміщається у відповідну апаратну чергу, асоційовану з MAC-адресою або VLAN тегом кожної VNF.

Однак, SR-IOV має недолік: VNF не може бути легко переміщений на інший сервер, оскільки він прив'язаний до конкретного апаратного ресурсу. Такий підхід не підходить для динамічного середовища NFV, де VNF часто переміщаються між серверами, наприклад, у різних географічних локаціях. Однак, це не становить проблеми у випадку застосування на корпоративному CPE, де VNF постійно працюють на одному й тому ж обладнанні.

Враховуючи переваги SR-IOV порівняно з базовою OVS, було виявлено, що не всі VNF повністю підтримують SR-IOV. Це може призвести до того, що продуктивність VNF буде нижчою, ніж очікувалося.

Не всі мережеві функції ефективно працюють у рамках віртуалізованого середовища. Наприклад, був випадок, коли рішення для кібербезпеки, розроблене для роботи на сервері Windows, стало надзвичайно вимогливим до ресурсів, коли його запустили за допомогою гіпервізора KVM на Linux. Іншою проблемою є залежність від конкретного типу процесора. Деякі компоненти VNF можуть потребувати процесори з певними характеристиками, як-от розширені векторні розширення, доступні лише на певних сімействах процесорів Intel. В результаті, ці VNF можуть не функціонувати належним чином на універсальних CPE (uCPE), якщо вони використовують процесори, що не підтримують ці характеристики.

Оператори мереж зіткнулися з технічними викликами, пов'язаними з NFV, і мають створити репозиторій перевірених VNF для різних версій обладнання uCPE, яке планується розгорнути. Процес тестування має охоплювати керованість, масштабованість, функціональність, мережеву сумісність, можливість оновлення, продуктивність на рівні даних, а також продуктивність на різних апаратних платформах на базі x86. Тестування має йти далі, ніж просто ізольоване випробування VNF. Необхідно також перевіряти, як VNF функціонують у межах повного ланцюжка послуг.

Оператори, окрім створення репозиторію тестів VNF, повинні створити репозиторій протестованих ланцюжків послуг, які визначаються за допомогою дескриптора мережевих послуг. NSD описує ланцюжок VNF з інтерфейсами та вбудованими сервісами, які готові до розгортання на конкретному uCPE. Створення NSD є складним процесом, що вимагає глибоких знань та аналізу документації VNF.

Для з'єднання двох VNF, що працюють на різних віртуальних машинах, потрібна віртуальна мережа. Існує кілька способів створення такої віртуальної мережі, включаючи OVS, SR-IOV, DPDK, PCI passthrough, але не всі VNF підтримують ці технології. Це ставить перед проектувальниками NSD завдання враховувати особливості кожного VNF при розробці ланцюжка послуг.

Наприклад, віртуальна мережа може використовувати Open vSwitch (OVS) з кількома конфігураціями. Один з варіантів - OVS з інтерфейсами Virt.io, що забезпечує менший вплив на обчислювальні ресурси. Інший варіант включає використання OVS з інтерфейсами DPDK, які пропонують більшу продуктивність, але водночас споживають більше обчислювальної потужності. DPDK може бути налаштований з однією чергою на інтерфейс або з декількома чергами для більш швидкої обробки.

При виборі конфігурації для ланцюжка послуг важливо враховувати, що обрана конфігурація має бути підтримувана всіма VNF у ланцюжку. З огляду на постійні зміни у світі NFV, VNF і NSD вимагають регулярного оновлення, включаючи додавання нових функцій або розширення API. Це, в свою чергу,

потребує систематичного повторного тестування VNF та NSD після кожного оновлення.

NFV представляє не лише технічні виклики, але й організаційні проблеми. Операторам, які раніше зосереджувались виключно на мережевих технологіях, тепер потрібно оволодіти новими навичками у сфері віртуалізації, управління програмним забезпеченням та розробки YANG-моделей для опису своїх послуг.

Проблеми з VNF на uCPE можуть створити складнощі у визначенні відповідальності за неполадки. Винними можуть бути постачальники VNF, постачальники uCPE, інші VNF або PNF в ланцюжку послуг, або їхня певна комбінація. Коли декілька VNF працюють на одному обладнанні, ідентифікація первинної причини проблеми може бути складною.

Ці проблеми можуть ускладнити забезпечення відмовостійкості та підтримки SLA, що очікується від додатків або кінцевих користувачів. Все це вимагає від операторів глибшого розуміння та взаємодії між різними компонентами NFV.

Щоб уникнути технічних проблем, пов'язаних з VNF, існує спокуса надмірно оснастити uCPE, але це може призвести до занадто високих витрат. Надмірне забезпечення може зробити універсальну платформу для розгортання клієнтських пристроїв (uCPE) з віртуалізованими мережевими функціями навіть дорожчою, ніж традиційний набір фізичних пристроїв, які вона призначена замінити.

Іншою проблемою в управлінні VNF є незрозумілість меж управління між менеджером VNF та оркестратором ресурсів вищого рівня, особливо у контексті масштабування VNF. Більшість постачальників комунікаційних послуг (CSP) не мають досвіду в оркеструванні; вони часто продовжують використовувати статичні сценарії для налаштування та управління своїми пристроями та функціями.

Ще однією проблемою є рівень контролю, який оператори готові надати корпоративним клієнтам над універсальними платформами для розгортання клієнтських пристроїв та віртуалізованими мережевими функціями, які на них працюють. Наприклад, адміністратор корпоративної мережі може бажати внести зміни до налаштувань брандмауера, але може знадобитися звернутися до постачальника комунікаційних послуг (CSP), щоб здійснити ці зміни. Великі

підприємства бажають зберегти контроль над своїми системами безпеки та можуть не бачити значних переваг віртуалізації, якщо це призведе до втрати контролю.

Оператори зіштовхуються з великою кількістю варіантів у межах парадигми NFV, що ускладнює розуміння та вибір оптимальних рішень. Особливо це стає складним у середовищі, де використовуються різні версії VNF, кожна з яких має свої API для управління. Кожен новий випуск VNF вимагає від оператора перегляду обширної документації для з'ясування відомостей про управлінські API. Часті оновлення програмного забезпечення VNF створюють невизначеність щодо вибору найкращої версії. Постачальники послуг зв'язку (CSP) витрачають значні зусилля на дослідження налаштувань і конфігурацій, які пропонують різні VNF. Наприклад, SD-WAN VNF може пропонувати опції для оптимізації WAN і FEC, а також різні варіанти розгортання на одному чи декількох віртуальних процесорах (vCPU) із різною кількістю оперативної пам'яті.

Для ефективного впровадження NFV, операторам необхідно співпрацювати з постачальниками, які здатні надати комплексні послуги системної інтеграції та організувати навчання для свого персоналу щодо ключових технологій, таких як Linux, віртуалізація, OpenStack та інструменти DevOps. Це допоможе забезпечити більш гладке розгортання NFV на підприємстві. Крім того, важливо вибирати постачальників uCPE, які підтримують широкий спектр сторонніх VNF, щоб забезпечити гнучкість та сумісність. Угоди з такими постачальниками повинні включати умови, що забезпечують підтримку нових VNF, а також оновлень існуючих VNF, включаючи тестування та оцінку їх впливу на існуючі ланцюги обслуговування.

3 СТРАТЕГІЇ VNF: ОПТИМІЗАЦІЯ, УПРАВЛІННЯ ТА ОЦІНКА ДЛЯ МОБІЛЬНИХ СИСТЕМ

3.1 Оптимізація розміщення VNF

Одним з найважливіших викликів, з якими стикаються постачальники інфраструктури (InPs), є проблема розгортання нових, обмежених послуг, які мають бути запроваджені майже миттєво. Щоб задовольнити ці нові вимоги, InPs переходять від рішень на апаратній основі, які відомі своєю негнучкістю, до набагато більш гнучких та масштабованих програмних рішень. Це бачення було реалізовано завдяки появі віртуалізації мережевих функцій, яка дозволила розгортати нові послуги на вимогу та майже в реальному часі. Проте розміщення цих послуг на інфраструктурі залишається дуже складним через обмеження цих послуг. Ця складність стає ще більш вираженою, коли послуги, які потрібно розгорнути, є композитними, часто у формі графа послуг (тобто VNF-FGs) [11].

Ефективне та автоматичне розміщення мережевих послуг, безумовно, є одним з найважливіших технологічних елементів на шляху до повністю автоматизованої мережі (тобто мережі з нульовим торканням). У зв'язку з цим дослідники з академічних та промислових сфер наростили свої зусилля, щоб вирішити це важливе питання. Було проведено декілька досліджень, деякі з них стосуються архітектурних рішень, таких як ті, що розроблені Європейським інститутом телекомунікаційних стандартів (ETSI), інші пропонують більш алгоритмічні рішення, на яких ми зосередимося у цьому розділі.

У літературі вже були запропоновані декілька рішень [12]. Деякі методи базуються на формулюванні та розв'язанні оптимізаційних задач, що мають недолік непридатності у реальному контексті через затримки у розв'язанні чи інші обмеження. Інші методи використовують швидкі евристики, але вони не завжди досліджують великий простір можливих дій. Метаевристики, зокрема еволюційні алгоритми, також були використані для вирішення цих задач, але вони можуть

стикатися з проблемами зі збіжністю та повільністю. Нещодавно були запропоновані методи, засновані на глибокому навчанні, які показують певну ефективність, але є неефективними у ситуаціях з широким простором дій [13]. Як показано в цьому розділі, навіть найкращі методи навчання вимагають адаптації для задовільного вирішення конкретної проблеми.

У цій роботі ми розглядаємо задачу розподілу віртуалізованих мережевих функцій у формі графа переадресації (VNF-FG) для реалізації мережевих послуг. Моделюючи цю задачу як Марківський процес прийняття рішень (MDP), ми вирішуємо її за допомогою методу глибокого навчання з підкріпленням. Припускаємо, що оператори задають свої запити, використовуючи VNF-FG, які включають типові елементи мережі, такі як балансувальники навантаження, брандмауери, та пристрої глибокої перевірки пакетів. Ця робота є розширенням дослідження [14], яке фокусується на використанні підходу глибокого навчання з підкріпленням (DRL) для вирішення задач, пов'язаних з вимогами до якості обслуговування (QoS). Головна мета полягає в тому, щоб ефективно розподілити максимальну кількість VNF-FG з відповідними вимогами до QoS на фізичні вузли в базових мережах.

Внесок цієї роботи полягає у трьох ключових аспектах:

1. Задачу розподілу віртуалізованих мережевих функцій у формі графа переадресації ми формалізували як Марківський процес прийняття рішень з визначенням відповідних станів та дій.
2. Запропоновано евристичний алгоритм, що перетворює нездійсненні рішення на здійсненні, тим самим покращуючи процес прийняття рішень.
3. Розроблено структуру глибокого навчання з підкріпленням, під назвою "Градієнт глибокої детермінованої політики розширеної розвідки", базовану на глибокому детермінованому градієнті політики (DDPG). Ця структура має на меті підвищити продуктивність агента DRL.

Результати моделювання демонструють, що запропонована система DRL забезпечує значно кращу продуктивність порівняно з алгоритмом розміщення на

основі цілочисельного лінійного програмування (ILP). Традиційне розміщення послуг, яке насправді є аналогом проблем bin-packing або knap-sack, є однією з найбільш вивчених проблем в літературі [15]. Цей випадок, хоча і дуже спрощений порівняно з прикладами, розглянутими в цій роботі, відомий як NP-складна проблема. Більш реалістичне розміщення послуг, таке як розміщення VNF-FG, яке ми розглядаємо, є значно складнішим. Послуги складаються з декількох підпослуг і мають численні обмеження, такі як процесор, оперативна пам'ять, дисковий простір тощо. Крім того, існують численні обмеження на лінії зв'язку, які з'єднують підсервіси або VNF, наприклад, пропускна здатність, затримка, втрати.

Хоча було проведено багато досліджень щодо розміщення простих послуг (тобто послуг, які складаються з однієї NFV), набагато менша увага була приділена розміщенню VNF-FG, яке фактично є досить новим питанням. Їх можна класифікувати за наступними чотирма основними категоріями.

Перший клас методів базується на математичній оптимізації. Багато робіт присвячено математичному моделюванню проблеми в контексті однієї чи декількох цілей. Залежно від характеристик проблеми, використовуються різні методи. Наприклад, деякі формулюють завдання оптимізації як цілочисельну лінійну програму (ЦЛП), інші як змішану цілочисельну лінійну програму (ЗЦЛП), а ще інші як цілочисельну нелінійну програму (INLP). Проблема, яку ми розглядаємо, є NP-складною і розв'язання ефективне лише для дуже малих екземплярів мережі, тому більшість підходів фокусуються на моделюванні і пропонують більш ефективні евристики для вирішення проблеми, і тому вони належать до іншої категорії. Крім того, проблема часто моделюється з сильними припущеннями. Наприклад, автори роботи [16] виходили з припущення, що кожен фізичний ланцюг має певну затримку, що не завжди відповідає дійсності. Подібне припущення про втрату швидкості можна знайти в [17]. Насправді затримка і трафік, що проходить через ланцюг, мають нелінійну залежність.

Щоб зменшити складність задачі, вживаються різні стратегії. Задачу розміщення ланцюжка функцій обслуговування (SFC) формулюють як багатоцільову змішану цілочисельну лінійну програму (ЗЦЛП), і деякі пропонують

звести цілі до однієї за допомогою методу обмежень. Щоб розв'язати залишену NP-складну задачу за поліноміальний час, автори пропонують лінійну релаксацію та округлення для наближення до оптимального розв'язку МІЛП. У інших дослідженнях, сформульовано задачу як ЗЛП, для якої запропонували рішення на основі рандомізованої лінійної програми з округленням, яке дає логарифмічний коефіцієнт апроксимації оптимуму. Та сформулювано задачу віртуалізації мережевих елементів (VNE) як задачу ILP з метою мінімізації витрат на пропускну здатність та кількості вузьких місць.

Ці підходи, хоча й ефективні, не гарантують знаходження найкращих рішень. Вони також мають певні обмеження. Оскільки деякі параметри доступні лише під час виконання, ці методи не можуть враховувати, наприклад, затримку, спричинену розміщенням, або втрати. Це робить ці підходи іноді неефективними (тобто якість отриманих рішень може бути не задовільною) в реальних умовах.

Більшість досліджень, знайдених у літературі, відносяться до категорії рішень на основі евристичних підходів. Методи, що входять до цього класу, зазвичай ефективні і можуть бути застосовані в реалістичних умовах. В роботі [18] автори пропонують евристичний підхід, який використовує багатоетапну конструкцію графа для представлення серверів, доступних для розміщення VNF. Після цього SFC розміщуються за допомогою максимального потоку між вершинами на різних етапах багатоетапного графа.

Для уникнення перебору всіх можливих рішень, включаючи деякі надмірні (наприклад, у випадку симетрії), автори в [16] запропонували ітеративний процес за допомогою бінарного пошуку для керування процедурою пошуку, яку традиційно виконують рішення-розв'язувачі. Аналогічно, в інших дослідженнях, запропоновано просту, але ефективну евристику, яка полягає в ітеративному розміщенні VNF поспіль, використовуючи процедуру пошуку найближчого рішення.

Замість того, щоб одночасно розглядати розміщення VNF та їх зв'язки, деякі підходи застосовують двоетапний метод. На першому етапі розміщують VNF, а потім на другому, з'єднують їх, вирішуючи проблему маршрутизації. Було

запропоновано алгоритм SFC-MAP, який розміщує VNF та з'єднує їх, використовуючи алгоритм найкоротшого шляху у багатошаровому графі, який створюється шляхом перетворення топології мережі з урахуванням обмежень SFC. Цей підхід також було розширено алгоритмом власної декомпозиції для розміщення VNF-FG. Головна перевага цього алгоритму полягає в тому, що його складність залежить в основному від розміру фізичної інфраструктури, що робить його дуже ефективним. Евристики підходять для стабільних систем, але можуть бути неефективними в умовах, де обмеження та цілі часто змінюються, оскільки вони можуть вимагати повного перепроектування. Також, використання евристик може призвести до швидкої збіжності, але з ризиком застрягти у локальному мінімумі, який може бути не найкращим рішенням.

У науковій літературі існують підходи, що базуються на використанні метаевристик. Більшість з цих робіт фокусується на застосуванні еволюційних алгоритмів для розміщення VNF-FG. Однією з пропозицій є використання генетичного алгоритму сортування без домінування (NSGA-II), який відомий своєю ефективністю у вирішенні багатоцільових оптимізаційних задач.

Щоб уникнути дослідження всіх можливих комбінацій, що значно сповільнило б збіжність, автори вирішили використовувати модифіковані версії алгоритму, щоб зосередитися лише на реалістичних рішеннях. Це підвищило ефективність за рахунок втрати деякої універсальності алгоритму NSGA-II. Деякі дослідження розглядають модифікації, в яких вибір розміщення VNF залишається випадковим, що сприяє глибокому аналізу простору можливих рішень. В інших підходах, щоб прискорити збіжність, було вирішено керувати розміщенням. Таким чином, була запропонована жадібна версія генетичного алгоритму сортування без домінування (NSGA-II), в якій вузли з вищими залишковими ресурсами отримують вищий пріоритет для розміщення, тоді як для розміщення зв'язків використовуються найкоротші шляхи.

Інші метаевристики також використовувалися в літературі. У роботі [19] автори запропонували новий підхід, який поєднує математичне програмування з метаевристикою пошуку в околі змінних (Variable Neighborhood Search, VNS) для

покращення аналізу простору рішень. Крім цієї останньої метаевристики, наскільки нам відомо, не було інших досліджень, які використовували б інші типи метаевристик для розміщення VNF-FG. Метаевристики дозволяють ефективно реагувати на проблему розміщення VNF-FG. Вони можуть легко інтегрувати нові цілі чи обмеження без необхідності перегляду всього рішення, на відміну від евристик. Також слід зазначити, що існують дослідження, що демонструють збіжність цих алгоритмів до оптимальних рішень за певних умов.

Навчання з підкріпленням (RL) – це галузь машинного навчання, де агент навчається приймати рішення на основі отриманої винагороди або покарання в результаті виконання однієї чи декількох дій. Здатність навчатися на основі минулого досвіду є цінною, оскільки це дозволяє адаптуватися до змін у навколишньому середовищі, таких як зміна типу трафіку або конфігурації мережі. Останні досягнення в цій сфері демонструють потенціал цих підходів для вирішення проблем комбінаторної оптимізації. Емпіричні дані підтверджують, що цей тип підходу може бути навіть більш ефективним, ніж контрольоване навчання, навіть у випадках, коли доступні оптимальні мітки.

В літературі було запропоновано кілька технічних підходів, пов'язаних з розміщенням віртуалізованих мережевих функцій. Більшість з них зосереджені на аспекті маршрутизації, який є частиною загальної проблеми розміщення VNF-FG. Проте, лише невелика кількість досліджень приділяє увагу цьому питанню в цілому. У роботі [13] автори використовують Марківський процес прийняття рішень (MDP) для моделювання стратегії розміщення VNF та їх з'єднання за допомогою алгоритму найкоротшого шляху. Оскільки знаходження оптимальної політики за такою стратегією є складною задачею в умовах великого простору станів, автори пропонують використання методу пошуку дерева Монте-Карло (MCTS) для розв'язання цієї проблеми. Незважаючи на ефективність цього підходу, він має високу складність через послідовне розміщення для кожного запиту. Тому автори також розглядають розподілений алгоритм на основі Q-Learning для розподілу VNF-FG. Для уникнення зростання кількості станів, що може ускладнити

застосування методу, автори пропонують дискретизувати простір станів. Однак це робить підхід непрактичним у реальних сценаріях використання.

Рішення, яке ми пропонуємо у цьому розділі для розміщення віртуалізованих мережевих функцій (VNF-FG), долає обмеження в масштабованості існуючих підходів за рахунок ефективнішого застосування апроксимаційних методів, зокрема через використання нейронних мереж. На відміну від підходу, описаного у [13], наше рішення дозволяє розміщувати один або декілька VNF-FG одночасно, і після фази навчання реакція на запит відбувається майже миттєво.

Наш підхід ґрунтується на останніх досягненнях у цій області та забезпечує високу ефективність, зокрема завдяки використанню концепції "відтворення пам'яті", що дозволяє інтегрувати переваги навчання під наглядом. Крім того, ми пропонуємо кілька вдосконалень існуючих методів, які забезпечують краще вивчення простору можливих рішень.

3.2 Алгоритми управління VNF

У розглядаються алгоритми, де рішення приймаються поетапно. Є особа або програма, яка відповідає за налаштування віртуальних мережевих функцій. Ці функції можуть працювати разом протягом різних етапів і потребують іноді змінювати своє місцезнаходження у мережі, але без перерв у роботі. Кожна така функція потребує певної кількості ресурсів, наприклад, обчислювальної потужності, пам'яті та місця для зберігання даних.

Для кожної функції є список, який показує, скільки кожного ресурсу вона потребує. В основному, функції потребують ресурсів для обчислень, тобто потужності процесора, оперативної пам'яті та місця на диску. Ідея полягає в тому, що ці вимоги можна складніше визначити, але якщо додати більше інформації в програму, вона зможе впоратися і з цим завданням. Тобто механізм може бути адаптований для роботи з більш складними ресурсами VNF, розширивши введення агента DRL.

Кожен зв'язок у віртуальній мережі має свої характеристики, як-от швидкість передачі даних, затримки у передачі і можливість втрати даних. Групи функцій у віртуальних мережах можна уявити як схему, де кожен елемент та зв'язок має своє визначене місце. Ця схема показує, які функції пов'язані між собою, і скільки є таких зв'язків. Уявляється, що ця схема як план, де зазначено місце кожної функції в мережі. Для того, щоб функція в мережі працювала правильно, вона має бути розміщена на вузлі, де є достатньо ресурсів. Це означає, що якщо функція потребує певну кількість ресурсів, наприклад, певну кількість пам'яті чи обчислювальної потужності, то вузол, на якому вона розміщена, має мати відповідні ресурси в достатній кількості. Також, кожна функція в мережі може бути розміщена лише на одному вузлі мережі, тому загальна кількість функцій на вузлі не може перевищувати одиницю. Зазвичай, коли планується, як дані будуть передаватися через мережу, основну увагу приділяють на те, наскільки швидко мережа може це зробити. Але є й інші важливі речі, які впливають на якість зв'язку, такі як затримки у передачі даних і можливість втрати інформації.

Щоб мережевий зв'язок працював добре, необхідно, щоб по мережі було достатньо "простору" для даних (пропускної здатності) та щоб дані доходили вчасно і без втрат. Це як вирішити, чи є достатньо місця на дорозі для всіх машин і чи не буде заторів або аварій. У роботі створено правила, які забезпечують, що зв'язки в мережі мають достатньо пропускної здатності і що затримки та втрати даних не перевищують певний рівень. Якщо мережевий зв'язок відповідає цим правилам, то вважається, що він працює добре.

Правила гарантують, що дані не застрягнуть у мережі, формуючи петлі, які можуть завадити передачі даних. Коли в мережі є два елементи, які повинні бути з'єднані, правила допомагають переконатися, що між ними є чіткий і прямий зв'язок. Якщо з'являється зайвий зв'язок, який створює петлю, правила запобігають його виникненню. Є певний показник, який використовується, щоб виміряти, чи всі необхідні частини мережі працюють. Якщо вони працюють, цей показник дорівнює одиниці. Це допомагає зрозуміти, чи в мережі все налаштовано правильно.

Уявіть програму, яка вчиться взаємодіяти з віртуальним світом крок за кроком. На кожному кроці вона спостерігає, що відбувається навколо, робить вибір і бачить, чи був цей вибір хорошим. Якщо вибір був вдалим, вона отримує "бонус" (нагороду). Припускається, що програма може бачити все в світі, в якому вона знаходиться, тому її рішення базуються на повній інформації про поточну ситуацію.

Програма намагається передбачити не тільки найкращу дію зараз, але й наскільки добре це рішення позначиться на майбутньому. Вона має певну "політику", за якою вирішує, як діяти у кожному можливому стані. Ціль програми зробити так, щоб загальна сума "бонусів", яку вона отримає з часом, була якомога більшою. Описи того, що програма спостерігає і які дії вона може виконати, залежать від конкретної задачі, яку вона має вирішити. У випадку розміщення функцій мережі, програма вивчає, як розташувати віртуальні компоненти і зв'язки між ними, щоб все працювало найкращим чином.

В основі цього розрахунку лежить те, що кожен вибір має свою цінність. Програма намагається вибрати дію з найбільшою цінністю, думаючи наперед і враховуючи можливі наслідки. За допомогою процесу, що повторюється багато разів, програма вчиться визначати ці цінності для кожного можливого вибору, щоб зрештою вибрати найкращий шлях дій.

Алгоритм Q-навчання – це спосіб, яким програма може навчитися вибирати найкращі дії в різних ситуаціях, але він добре працює лише коли варіантів дій не дуже багато. Коли можливостей для дій стає забагато, цей метод починає збоїти, тому використовують складніші комп'ютерні моделі, глибокі нейронні мережі (DNN), щоб впоратися з цією проблемою. Ці моделі зберігають інформацію про попередні дії та їх результати, щоб з часом все краще розуміти, що робити. Вони також використовують спеціальні техніки, щоб зробити навчання стабільнішим і не допустити, що програма "застрягне" на якихось неправильних висновках.

У випадку, коли програма вже має чіткий план дій, вона може використовувати цей план, щоб оцінити свої наступні кроки. Рішення, яке приймає програма, засноване лише на тому, що відбувається навколо неї, і на тому, які наслідки можуть бути від цього. Програма може самостійно виправляти свої

помилки, аналізуючи, наскільки вдалою була кожна дія, і вибираючи кращі варіанти на майбутнє. Це ніби вона сама себе перевіряє і вчиться на власних помилках.

В області розвитку програм, які самонавчаються (DRL), одним з популярних методів є так звані Deep Q Networks (DQN). Але вони не завжди ефективні, коли мова йде про велику кількість можливих дій. Тому був розроблений інший метод під назвою Deep Deterministic Policy Gradient (DDPG), який краще справляється. У цьому методі використовуються дві окремі нейронні мережі: одна допомагає визначити стратегію дій, а інша оцінює потенційну вигоду від цих дій.

Процес навчання проходить у декілька кроків (рис. 3.1):

1. Мережа, яка визначає стратегію, спостерігає поточну ситуацію і вирішує, яку дію розглянути.
2. Вона вираховує базову дію і додає трохи випадковості для розмаїтості вибору, після чого реалізує цю дію.
3. Як результат дії, мережа отримує винагороду (наприклад, позитивний результат) і нову інформацію про стан.
4. Інформація про те, що сталося, зберігається для подальшого аналізу.

Далі, в процесі навчання мережа проаналізує зібрану інформацію, щоб зрозуміти, як краще діяти в подібних ситуаціях у майбутньому. Мережа, яка оцінює вигоду, використовує цю інформацію для оцінки, чи була дія успішною і як її можна покращити. В кінці, мережі оновлюються з новою інформацією, щоб кожного разу приймати більш обґрунтовані рішення.

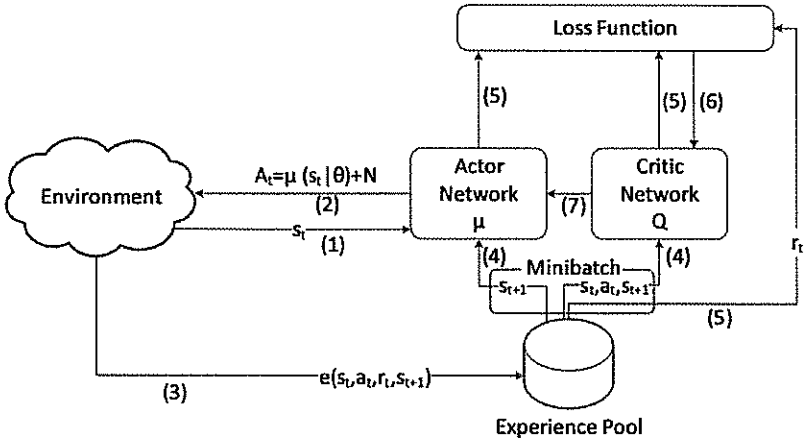


Рис. 3.1. Алгоритм DDPG

У сфері управління мережами, ми використовуємо список (матрицю) того, що потрібно нашим клієнтам, як карту для розміщення функцій віртуальної мережі. Це схоже на те, як ви б розпланували, які комп'ютерні програми (функції) запустити на яких серверах (фізичних мережах). Кожна програма має свої вимоги, наприклад, скільки пам'яті чи процесорного часу вона потребує, і ми маємо забезпечити ці ресурси. Також маємо врахувати, наскільки швидко дані можуть переміщатися між програмами, що важливо для підтримки якості послуг.

Тож створюємо список, який включає всі ці вимоги, і використовуємо його для навчання нашої програми для прийняття рішень (агента DRL). Цей агент вирішує, як найкраще "розмістити" кожну віртуальну функцію у мережі. Це допомагає нам забезпечити, щоб усі програми працювали ефективно і без збоїв у різних мережах.

Ми використовуємо додаткові правила, щоб визначити, яким чином краще розташувати різні частини нашої віртуальної мережі. Ці правила допомагають вирішити, як важливо розмістити певну частину мережі на конкретному сервері або через які з'єднання вона має передавати дані.

Коли вирішили, яку дію виконати, дивимося на всі ці правила разом, щоб вибрати найкращий варіант. Тому що ми повинні бути точними, тож використовуємо тільки визначені "так" або "ні" варіанти для кожного рішення.

Для того, щоб упорядкувати всі ці дії, розроблена система оцінювання, яка допомагає нам знайти найкращий спосіб розташувати частини мережі.

Ми використовуємо систему оцінювання для того, щоб визначити, наскільки добре працює наша програма для управління мережевими функціями. Якщо програма може встановити всі необхідні частини мережі і вони працюють правильно, вважаємо це успішним. Кожна частина має свої вимоги, наприклад, достатню швидкість передачі даних і низьку затримку. Якщо всі ці вимоги виконані, програма отримує позитивну оцінку.

Для оцінки того, наскільки добре програма справляється з кожним завданням, ми дивимося на те, наскільки вона виконує запити на обслуговування та як вона розміщує мережеві функції в мережі. Через складність мережевого середовища, нам

не завжди легко визначити, яка повинна бути нагорода за кожну дію. Однак, оскільки ми можемо вимірювати якість обслуговування, ми можемо визначити нагороду на кожному кроці, покращуючи таким чином нашу програму.

В нашому дослідженні ми розглядаємо спосіб розподілу віртуальних мережевих функцій по всій мережі. Вирішувати, де і як розмістити ці функції, – це складне завдання. Ми використовуємо метод глибокого навчання, званий DDPG, для розробки плану розподілу цих функцій. Цей метод допомагає нам розуміти, де краще розмістити кожну функцію, щоб вони працювали ефективно і задовольняли всі технічні вимоги, такі як пропускна здатність і затримка.

Ключовим елементом нашої системи є HFA, алгоритм, який допомагає перетворити запропоновані рішення від DDPG в практичні плани. Він визначає, які ресурси доступні і як найкраще розмістити VNF, зважаючи на обмеження мережі та вимоги до якості обслуговування. Уявіть, що це схоже на планування розташування меблів у кімнаті, де кожен предмет повинен ідеально вписуватися, максимально ефективно використовуючи доступний простір. Загалом, підхід дозволяє автоматизувати процес розподілу мережевих функцій, ефективно використовуючи ресурси мережі і забезпечуючи якісне обслуговування.

Алгоритм HFA (рис. 3.2.) вирішує, як найкраще розподілити віртуальні мережеві функції використовуючи пропозицію з програми, що допомагає приймати рішення. Він використовує особливий тип дій, званий "прото-акції", для призначення кожної VNF до певного місця. Ці прото-акції мають спеціальні ваги, що допомагають визначити, де краще розмістити VNF. Спочатку, алгоритм відокремлює ці ваги та використовує їх, щоб вирішити, до якого "вузла" (комп'ютера чи сервера) відправити кожну VNF. Вузли з більшою вагою розглядаються як пріоритетні. Якщо вузол має достатньо ресурсів для VNF, то VNF розміщується саме там. Після цього алгоритм перевіряє, скільки ресурсів залишилося, і оновлює цю інформацію. Коли всі VNFs розподілені, алгоритм шукає найкращий шлях для даних, які мають проходити між VNFs, щоби забезпечити, що всі дані можуть пройти без затримок і втрат. В останньому кроці, алгоритм

переконується, що всі з'єднання мають достатньо пропускної спроможності, щоб дані могли пройти без проблем.

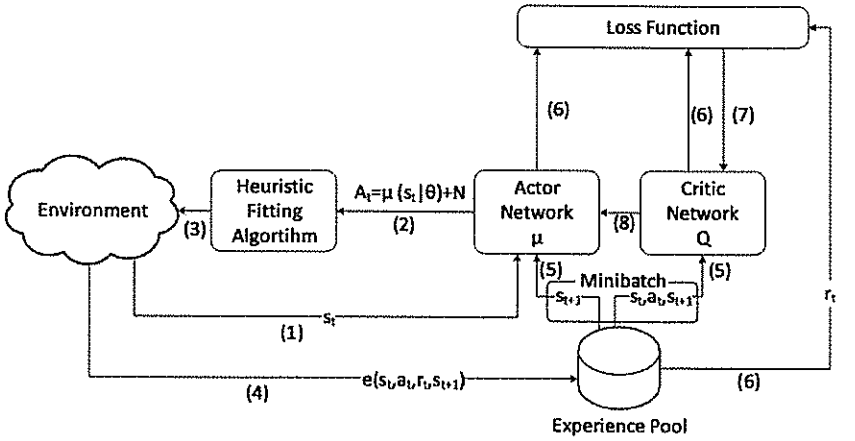


Рис. 3.2. Алгоритм HFA

Алгоритм E²D²PG, що показаний на рис. 3.3, використовує кілька кроків для того, щоб вирішити, які дії варто зробити. Спочатку він бере основну дію та додає до неї трохи випадковості для того, щоб дослідити нові можливості. Потім ці дії перевіряються спеціальним модулем, HFA, щоб визначити, чи можна їх використати для розміщення віртуальних мережевих функцій. Після цього експерт оцінює, яка з дій є найкращою, використовуючи різні методи оцінки. Обрана дія реалізується, і на основі цього визначається наступний крок і винагорода, які допомагають удосконалити процес прийняття рішень в майбутньому.

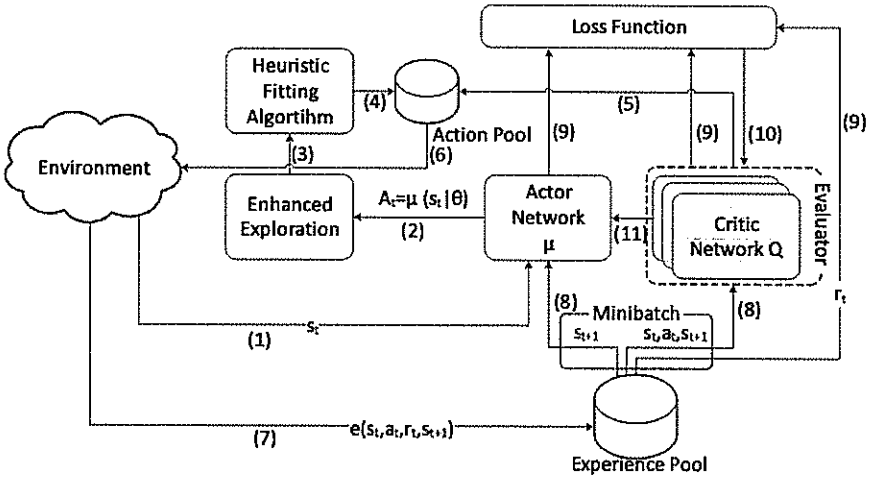


Рис. 3.3. Алгоритм E²D²PG

У звичайному методі глибокого навчання з підкріпленням, програма вибирає тільки одну дію кожного разу, що робить процес пошуку рішень досить повільним, особливо коли є багато можливих дій. Щоб вирішити цю проблему, ми вдосконалили цей процес, додаючи трохи випадковості до кожної дії. Це робиться за допомогою математичної формули, яка додає "шум" до запланованої дії. Таким чином, програма може розглянути більше варіантів і краще вивчити всі можливі дії, що допоможе знайти краще рішення.

У другому кроці нашого процесу, ми використовуємо HFA, щоб обрати дії з урахуванням раніше доданого шуму. Цей метод допомагає нам вибрати ті дії, які підходять для роботи віртуальних мережевих функцій та забезпечують достатньо широку пропускну спроможність для мережевих з'єднань. Після вибору, ці дії проходять оцінку експертом, про яку буде далі.

Експерт – це той, хто перевіряє, наскільки добре працює запропоноване рішення. У деяких дослідженнях важливу роль у виборі кращих рішень відіграє так званий "критик". Якість роботи критика безпосередньо впливає на те, наскільки добре ми можемо вибрати рішення. Щоб зробити роботу критика кращою, можна використати одночасне навчання декількох моделей. Проте, це вимагає багато окремих навчальних середовищ, які не завжди є у нашому розпорядженні. У випадку з певними мережевими функціями, нам доводиться працювати з реальними мережами, і тому ми не можемо просто створити багато однакових умов для навчання моделей.

У нашій роботі ми вдосконалюємо спосіб навчання комп'ютерних програм, використовуючи не одну, а кілька "критичних мереж", які допомагають програмі визначити, які дії краще робити в різних ситуаціях. Кожна з цих мереж пропонує своє рішення, а програма дивиться на середній результат від усіх, щоб вибрати найкращу дію. Мережа, яка пропонує дію з найкращим середнім результатом, використовується для вибору, що робити далі. Також ще перевіряємо, наскільки добре кожна критична мережа "вгадує" правильні дії, порівнюючи їх пропозиції з тим, що ми вважаємо правильним. Це допомагає нам вибрати найточнішу мережу,

щоб вона могла навчати інші частини програми. Таким чином, програма постійно вчиться і стає кращою в прийнятті рішень.

Алгоритм E^2D^2PG працює наступним чином: на початку усім "студентам" (мережам, які приймають рішення) дається однаковий старт. Вони вчяться на досвіді, який зберігається у "зошиті" (буфері повторень), та на прикладах, що мають випадкові елементи, щоб вони могли краще адаптуватися до різних ситуацій. Кожен "студент" пропонує свої ідеї (дії), і тоді ми вибираємо найкращу з них, щоб виконати. Ми також перевіряємо, як добре кожен "студент" впорався з завданням, порівнюючи його ідеї з правильними відповідями. Той, хто найточніше вгадав, стає "вчителем" (акторською мережею) для наступного раунду навчання. На кінці "зошит" (цільові мережі) оновлюється з новими знаннями, так що наступного разу всі "студенти" починають з трохи кращої позиції.

3.3 Тестування та оцінка алгоритмів VNF

Щоб оцінити ефективність нашого підходу, ми для перших експериментів використали топологію мережі BtEurope, яка має 24 вузли та 37 двосторонніх з'єднань. Пропускна здатність кожного зв'язку була визначена випадково і складала одне з наступних значень: 100 Мбіт/с, 150 Мбіт/с, 600 Мбіт/с або 1 Гбіт/с. Для отримання даних про затримку і відсоток втрати пакетів, щоб оцінити якість розгорнутих віртуальних з'єднань, використали симулятор дискретних подій OMNeT++ (версія 5.4.1).

Щоб випробувати різні способи налаштування мережі, виконується 10 тестів, кожен з яких використовує різні випадкові налаштування. Для створення схем мереж використовується спеціальна математична модель, яка допомагає визначити, як мережеві функції пов'язані між собою. Ця модель дозволяє мережевим з'єднанням формуватися так, щоб усі частини мережі були з'єднані. Кожен віртуальний зв'язок довільно запитує пропускну здатність в діапазоні від 1 Мбіт/с до 40 Мбіт/с, затримку від 1 мс до 100 мс та втрату пакетів від 0,1% до 10%..

У кожному з тестів використовуються різні кількості мережевих функцій та їхніх з'єднань, і кожен тест триває 200 "епізодів". В кожному епізоді відбувається 100 "кроків", на яких перевіряється, наскільки добре мережа працює. Завдання полягає в тому, щоб знайти спосіб налаштування мережі, який не лише виконає всі вимоги, але й буде найефективнішим у використанні ресурсів мережі.

Агент DRL написаний на Python із використанням бібліотеки Keras. Для запуску агента DRL та симуляції OMNeT++ було використано ноутбук з процесором Intel i5-1240P та 16 ГБ оперативної пам'яті. Агент DRL налаштований з наступними параметрами. Для навчання параметрів нейронної мережі використовується оптимізатор Adam. Швидкість навчання актора та критика становить 10^{-4} та 10^{-3} відповідно. Коефіцієнт дисконтування γ дорівнює 1.0. Параметри цільової мережі оновлюються з коефіцієнтом τ=0.001. Розмір пакету даних для навчання становить 32.

В алгоритмі E²D²PG ми використовуємо 10 критичних мереж для оцінювача, оскільки вони мають кращу оцінку потенційних дій шляхом усереднення обчислених Q-значень критичними мережами. Також на рис. 3.4 представлено порівняння роботи агента DDPG з HFA та без нього. Коефіцієнт прийнятності DDPG-HFA набагато вищий, ніж у DDPG, тому що DDPG не працює добре, коли йдеться про великомасштабний дискретний простір. Отже, за допомогою DDPG-HFA можна досягти вищого коефіцієнта прийняття.

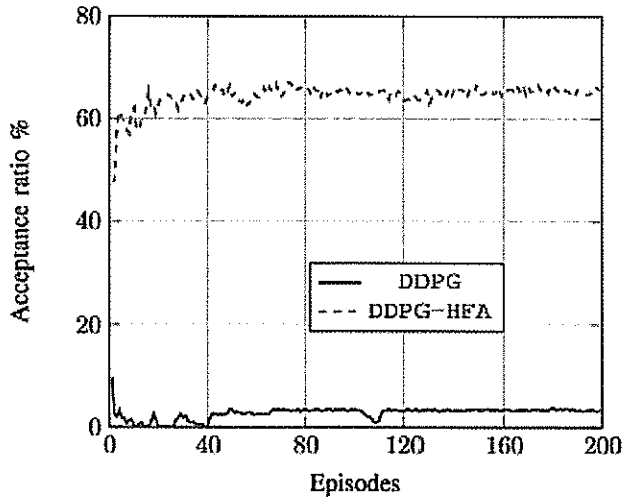


Рис. 3.4. Коефіцієнт сприйняття DDPG та DDPG-HFA

У цьому розділі ми досліджуємо продуктивність підходу E²D²PG. Для порівняння, ми вибрали евристичний підхід, First-Fit-Dijkstra (FFD), і DDPG-HFA. FFD використовує алгоритм First-Fit для розподілу VNF в базовій мережі та алгоритм Дейкстри (який визначає найкоротший шлях за кількістю проміжних вузлів) для створення шляхів VL. При заданому розподілі VNF, алгоритм Дейкстри може бути використаний для визначення шляху з'єднання VNF. У DDPG-HFA та E²D²PG місцеположення VNF та шляхи VL у базовій мережі визначаються нейронними мережами та HFA. Отже, якість дій, згенерованих агентом DRL, має значний вплив на продуктивність у DDPG-HFA та E²D²PG.

Ми розглядаємо три сценарії з 1, 3 та 5 VNF-FG. Чим більше кількість VNF-FG, тим більше вимог до ресурсів і складніше здійснити розгортання. У всіх сценаріях показники прийняття DDPG-HFA та E²D²PG вищі, ніж у FFD. Рис. 3.5. показує коефіцієнт сприйняття і відсоток розгорнутих VL, коли є один VNF-FG. Коефіцієнт сприйняття DDPG-HFA значно вищий, ніж у FFD (більше ніж на 20%). Крім того, DDPG-HFA вимагає менше часу для досягнення стабільного стану завдяки простоті запитів ресурсів. E²D²PG краще за DDPG-HFA як за коефіцієнтом сприйняття, так і за відсотком розгорнутих VL.

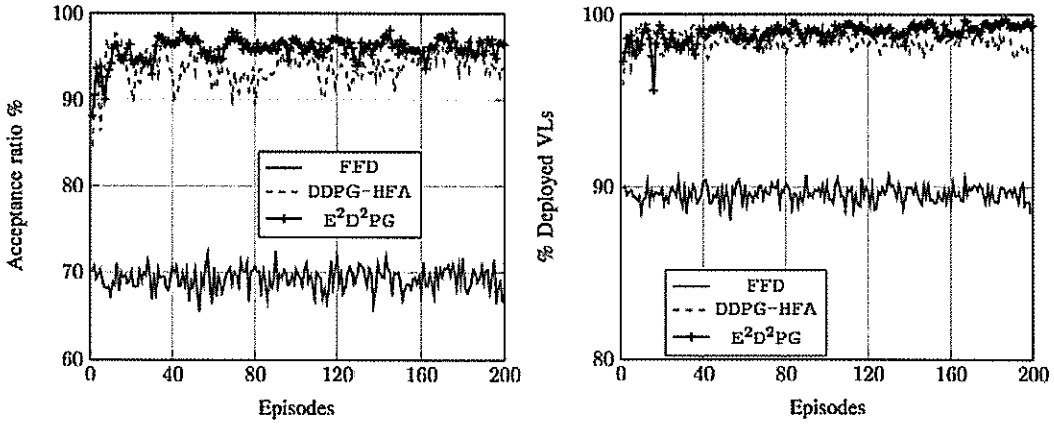


Рис. 3.5. Коефіцієнт сприйняття та відсоток розгорнутих VL при 1 VNF-FG

Рис. 3.6. показує коефіцієнт сприйняття та відсоток розгорнутих VL, коли є три VNF-FG. DDPG-HFA та E²D²PG мають кращий коефіцієнт сприйняття, ніж FFD. Хоча відсотки успішно розгорнутих VL у DDPG-HFA та FFD схожі, коефіцієнт

сприйняття у DDPG-HFA вищий. Завдяки своїй інтелектуальності, DDPG-HFA та E²D²PG здатні розумно розподіляти ресурси для VL, щоб оптимізувати коефіцієнт сприйняття VNF-FG.

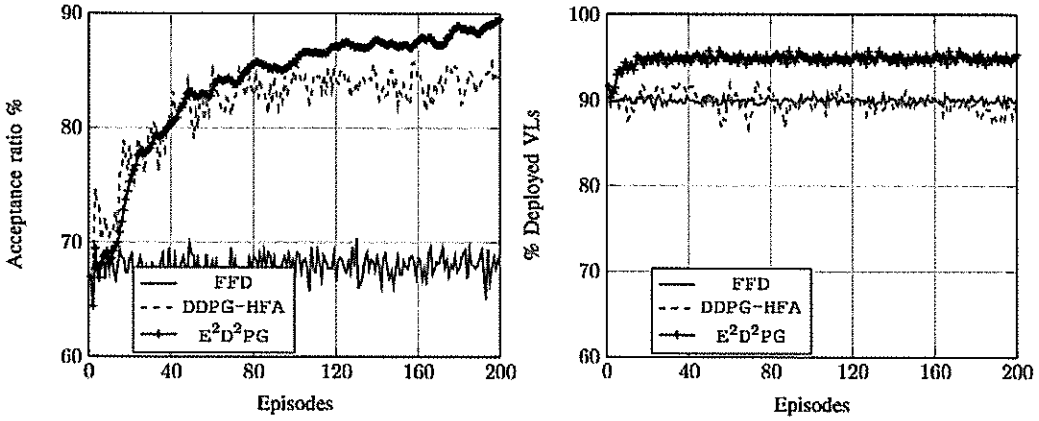


Рис. 3.6. Коефіцієнт сприйняття та відсоток розгорнутих VL при 3 VNF-FG

Рис. 3.7. показує коефіцієнт сприйняття E²D²PG, DDPG-HFA та FFD. У FFD найнижчий коефіцієнт сприйняття, коли кількість VNF-FG становить 5. Більше того, коефіцієнт сприйняття FFD не покращується з часом. Коефіцієнт сприйняття DDPG-HFA спочатку нижчий, ніж у FFD, оскільки DDPG-HFA на початку досліджує простір дій. Однак коефіцієнт сприйняття DDPG-HFA поступово зростає і він перевершує FFD після декількох епізодів. E²D²PG перевершує як DDPG-HFA, так і FFD навіть з самого початку. Коефіцієнт сприйняття E²D²PG на 10% кращий, ніж у DDPG-HFA.

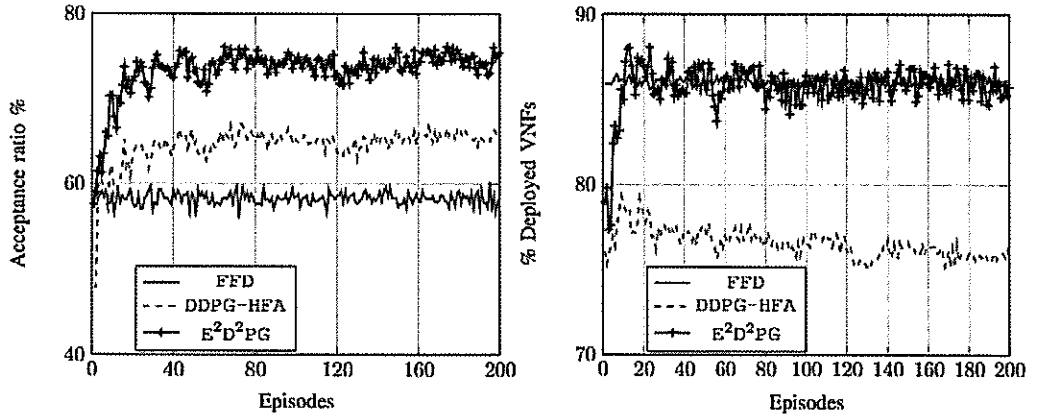


Рис. 3.7. Коефіцієнт сприйняття та відсоток розгорнутих VL при 5 VNF-FG

Окрім коефіцієнт сприйняття, ми також досліджуємо відсотки розгорнутих VNF та VL. Результати показані на рис. 3.8. Відсотки розгорнутих VNF та VL вищі, ніж коефіцієнт сприйняття, оскільки досягнення хорошого коефіцієнта сприйняття є складнішим. Наприклад, VNF-FG не приймається, навіть якщо тільки один VNF не розгорнуто успішно. Хоча відсотки розгорнутих VNF від FFD подібні до цих у E²D²PG, відсотки розгорнутих VL в E²D²PG значно вищі, ніж у FFD (близько 10%). Отже, показники у E²D²PG набагато вищі, ніж у FFD. Відсотки розгорнутих VNF та VL в DDPG-HFA нижчі, ніж у FFD, хоча його коефіцієнт сприйняття вищий, ніж у FFD. Незважаючи на те, що FFD може розгорнути більше VNF та VL, кількість VNF-FG, які відповідають вимогам QoS, нижча, ніж у DDPG-HFA.

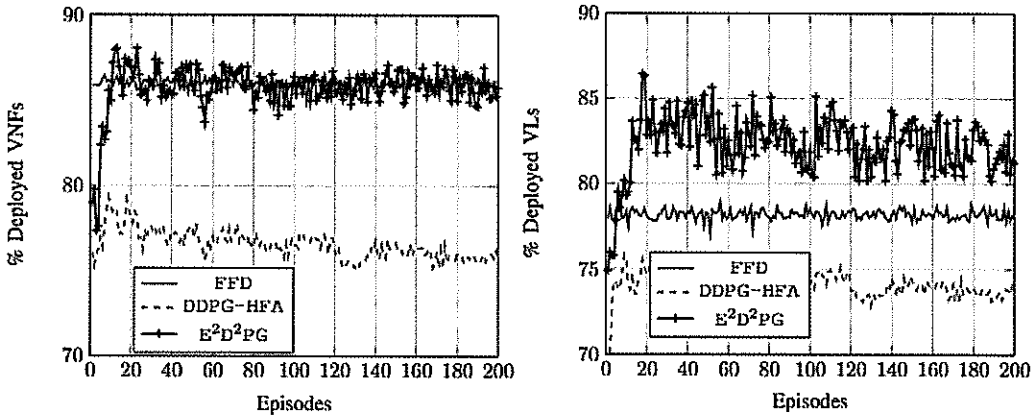


Рис. 3.8. Відсоток розгорнутих VNF та VL при 5 VNF-FG

Далі порівняємо коефіцієнт сприйняття E²D²PG з рішенням, заснованим на ILP. Зауважте, що рішення на основі ILP розв'язує модель у попередньому розділі без урахування обмежень QoS, оскільки відносини практичних метрик QoS та трафіку не є лінійними і складно мати точну модель для цих метрик QoS у багатоетапних мережах. Існуючі дослідження сформулювали проблему вбудовування VNF-FG як ILP-задачу (без урахування метрик QoS) розв'язали її, запропонувавши евристичні алгоритми. Рішення на основі ILP є оптимальним і вимагає вкрай високих обчислювальних ресурсів. У відповідь, ILP є верхньою межею евристичних алгоритмів. Однак, воно не є верхньою межею у розглянутих сценаріях через відсутність обмежень QoS. Рис. 3.9. представляє коефіцієнт

сприйняття рішення на основі ILP та E²D²PG. На початку коефіцієнт сприйняття E²D²PG нижчий, ніж у ILP. Однак він поступово зростає та перевищує ILP після кількох епізодів. Це тому, що E²D²PG здатний вивчити приховані візерунки середовища, взаємодіючі з підкладковою мережею.

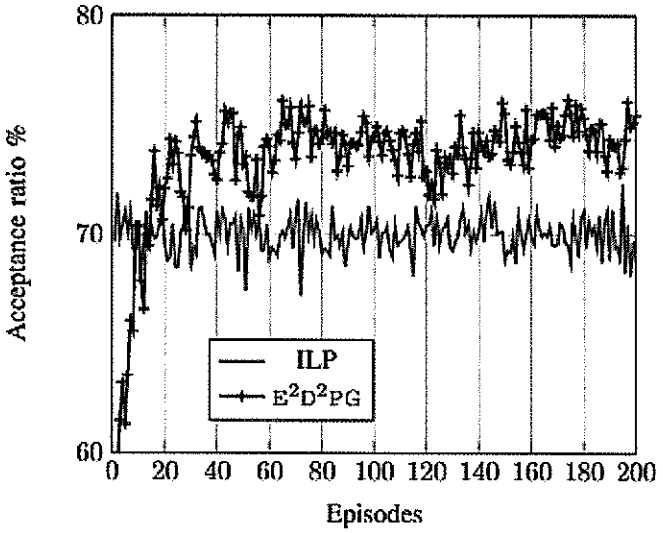


Рис. 3.9. Коефіцієнт сприйняття ILP та E²D²PG

Розглянемо більш складна підкладкову мережа для оцінки продуктивності запропонованих алгоритмів. Топологія підкладкової мережі буде Uninett2, яка має 74 вузли та 101 зв'язок. Ця підкладкова мережа має приблизно в 3 рази більше вузлів та зв'язків, ніж BtEurope. Ті ж нейронні мережі та конфігурації були використані у цих симуляціях. Коефіцієнт сприйняття представлений на рис. 3.10. Загалом коефіцієнт сприйняття, ніж у сценарії BtEurope. Це тому, що більша підкладкова мережа вимагає глибших нейронних мереж та більше зразків для витягування прихованих візерунків з середовища. Однак глибша нейронна мережа призводить до більш тривалого часу навчання та вищої обчислювальної складності. Без додавання більше шарів, DDPG-HFA та E²D²PG все ще значно перевищують FFD. Коефіцієнт сприйняття E²D²PG може бути вдвічі вищим за FFD. Коефіцієнт сприйняття E²D²PG також на 10% вищий, ніж у DDPG-HFA.

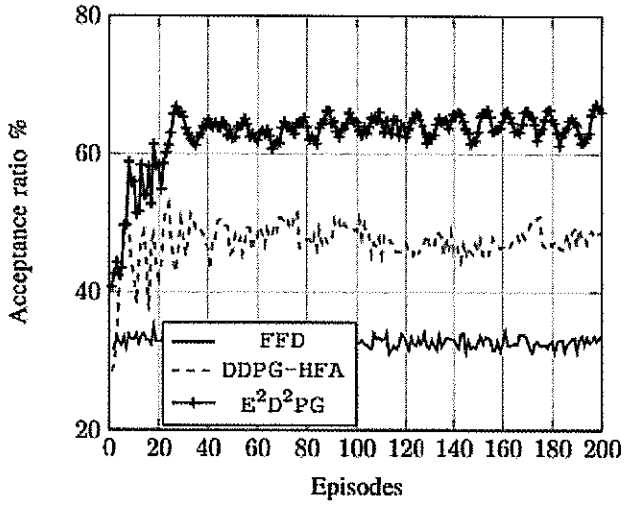


Рис. 3.10. Коефіцієнт сприйняття Uninett

NFV та оркестрація послуг полегшують управління та розгортання послуг в телекомунікаційних мережах. Ці послуги можуть бути описані у форматі VNF-FG; таким чином, розгортання послуги, насправді, є вбудовуванням відповідного VNF-FG в підкладкову мережу. Складність задачі вбудовування VNF-FG та невизначеність метрик мережі є основними викликами. На щастя, новітні підходи глибокого навчання відкривають перспективні рішення для вирішення цих викликів, особливо підходи глибокого навчання з підкріпленням (DRL). Підходи DRL можуть ефективно визначати майже оптимальні рішення. Більше того, вони можуть вивчити характеристики мережі, щоб покращити якість рішень. Однак, сучасні алгоритми DRL не можуть впоратися з проблемою з великим дискретним простором, як проблема вбудовування VNF-FG. У цій роботі адресується потреба в легковажному, але ефективному DRL для вирішення викликів у задачі вбудовування VNF-FG. Ми модифікували алгоритм DDPG з евристичним алгоритмом, який називається HFA, щоб покращити якість, яку надає мережа актора. Крім того, нові модулі, такі як оцінювач та поліпшене дослідження, були додані для формування нового алгоритму DRL, який називається Enhanced Exploration DDPG (E^2D^2PG). Запропонований алгоритм покращує дослідження агента DRL. Результати підтверджують, що запропонований алгоритм перевершує алгоритм DDPG, а також рішення на основі ILP.

ВИСНОВКИ

Дослідження принципів роботи та архітектури віртуалізації функцій мережі в першій частині дипломної роботи виявило її значний вплив на розвиток мережевих технологій. VNF, як ключовий напрямок еволюції мереж, корінним чином змінює підходи до управління мережевими ресурсами та послугами. Ця технологія дозволяє реалізувати мережеві функції як програмне забезпечення на загальних обчислювальних платформах, що надає переваги у зниженні капіталовкладень, операційних витрат і забезпечує гнучкість у впровадженні нових послуг та швидку реакцію на зміни в мережевих потребах. Постачальники VNF можуть розділяти програмне забезпечення на окремі компоненти (VNFC), які взаємодіють між собою у віртуальному середовищі, дозволяючи створювати більш складні та інтегровані мережеві рішення. Однак, кожен постачальник VNF має свій унікальний підхід до побудови VNF, що сприяє різноманітності в вимогах до реалізації VNF, забезпечуючи специфічні потреби та цілі кожного постачальника.

Аналіз сучасного стану реалізації та проблем VNF виявляє, що стандартизація та розробка відкритих інтерфейсів значно сприяли появі інноваційних продуктів та рішень. VNF забезпечує інтеграцію різних мережевих елементів і послуг, наприклад, в ролі "EPC VNF" для мереж 4G LTE, показуючи свою гнучкість і адаптивність до різних функціональних потреб мережі. Таким чином, підвищенні продуктивності мереж, спонукає до подальших досліджень і розробок у цій області, зосереджуючись на покращенні та оптимізації VNF для використання в різноманітних мережевих середовищах.

Дослідження оптимізації розміщення та управління VNF в даній дипломній роботі підкреслює важливість для постачальників інфраструктури, які стикаються з необхідністю швидкого впровадження нових послуг. Для вирішення цієї проблеми були запропоновані різні підходи, включаючи формулювання та розв'язання оптимізаційних задач, швидкі евристики, метаевристики, а також методи, засновані на глибокому навчанні. Але особливу увагу в дослідженні

приділено глибокому навчанню з підкріпленням (DRL) для оптимізації розміщення VNF-FG та задоволення вимог до якості обслуговування (QoS).

Результати моделювання демонструють, що система DRL забезпечує кращу продуктивність порівняно з традиційними алгоритмами, особливо в складних сценаріях, де послуги складаються з декількох та мають численні обмеження.

Що стосується алгоритмів управління VNF, рішення приймаються поетапно, і кожна функція VNF потребує певної кількості ресурсів, таких як обчислювальна потужність, пам'ять та місце для зберігання даних. Важливою частиною процесу є забезпечення, щоб кожна функція була розміщена на вузлі з достатніми ресурсами, а також врахування характеристик мережевого зв'язку, таких як швидкість передачі даних, затримки та можливість втрати даних.

Загалом, дослідження показує, що використання інноваційних підходів, особливо тих, що засновані на глибокому навчанні з підкріпленням, може значно покращити процес розміщення та управління VNF, забезпечуючи ефективне та гнучке впровадження мережевих послуг.

Дослідження глибокого навчання з підкріпленням, зокрема методом DDPG, для покращення продуктивності агента DRL, виявило значний потенціал у вирішенні складних завдань оптимізації в мережевих технологіях.

Використання DDPG для оптимізації розміщення віртуальних мережевих функцій (VNF-FG) демонструє високу ефективність. Особливо це стає важливим у випадках, коли простір дій є дуже великим та складним, як це часто буває в реальних мережевих середовищах. За допомогою DDPG агент DRL може ефективніше навчатися та адаптуватися до змін у мережевому середовищі, що підвищує якість та продуктивність розміщення та управління VNF. Таким чином, використання DDPG та інших методів DRL відкриває нові можливості для розв'язання складних оптимізаційних проблем у мережевих технологіях, пропонуючи інноваційні підходи до управління та розміщення VNF.

ПЕРЕЛІК ПОСИЛАНЬ

1. What is virtualization. URL: https://opensource.com/resources/virtualization
2. Software-defined networking (SDN). URL: https://www.techtarget.com/searchnetworking/definition/software-defined-networking-SDN
3. NFV Management and Orchestration [MANO] Key Performance Indicators [KPIs]. URL: https://www.techplayon.com/nfv-management-and-orchestration-mano-key-performance-indicators-kpis/
4. Network Function Virtualization (NFV) Architecture. URL: https://www.techplayon.com/network-function-virtualization-nfv-architecture/
5. Survey: The Top 5 Challenges Preventing SDN/NFV Deployment. URL: https://www.netcracker.com/blog/view-all/survey-top-5-challenges-preventing-sdn-nfv-deployment.html
6. Network Functions Virtualisation (NFV); Virtual Network Functions Architecture // ETSI GS NFV-SWA 001 – 2014.
7. Network Functions Virtualisation (NFV); Infrastructure Overview // ETSI GS NFV-INF 001 – 2015.
8. Sofiane I., Ayoub B., "Cloud Native 5G Virtual Network Functions: Design Principles and Use Cases," IEEE 8th International Symposium on Cloud and Service Computing (SC2), 2018.
9. Crawshaw J. "The Integration Challenges of Software-Defined & Virtualized Enterprise Networking," Heavy Reading white paper produced for Ekinops.
10. Quang T., Yassine H., Abdelkader O., "A Deep Reinforcement Learning Approach for VNF Forwarding Graph Embedding," IEEE Transactions on Network and Service Management, 16 (4), pp.1318-1331, 2019.
11. Gupta, B. Jaumard, M. Tornatore, and B. Mukherjee, "A Scalable Approach for Service Chain Mapping with Multiple SC Instances in a Wide-Area Network,"

IEEE Journal on Selected Areas in Communications, vol. 36, no. 3, pp. 529–541, March 2018.

12. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network Slicing and Softwarization: A Survey on Principles, Enabling Technologies, and Solutions," IEEE Communications Surveys Tutorials, vol. 20, no. 3, pp. 2429–2453, 2018.
13. S. Haeri and L. Trajkovi´c, "Virtual network embedding via Monte Carlo tree search," IEEE Transactions on Cybernetics, vol. 48, no. 2, pp. 510–521, 2018.
14. R. Riggio, A. Bradai, D. Harutyunyan, T. Rasheed, and T. Ahmed, "Scheduling wireless virtual networks functions," IEEE Transactions on Network and Service Management, vol. 13, no. 2, pp. 240–252, June 2016.
15. E. G. Coffman, Jr., M. R. Garey, and D. S. Johnson, "Approximation Algorithms for NP-hard Problems," D. S. Hochbaum, Ed. Boston, MA, USA: PWS Publishing Co., 1997, ch. Approximation Algorithms for Bin Packing: A Survey, pp. 46–93.
16. M. C. Luizelli, L. R. Bays, L. S. Buriol, M. P. Barcellos, and L. P. Gaspary, "Piecing together the NFV provisioning puzzle: Efficient placement and chaining of virtual network functions," in Proc. IFIP/IEEE IM, May 2015, pp. 98–106.
17. Q. Zhang, Y. Xiao, F. Liu, J. C. Lui, J. Guo, and T. Wang, "Joint optimization of chain placement and request scheduling for network function virtualization," in 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS). IEEE, 2017, pp. 731–741.
18. N. Tastevin, M. Obadia, and M. Bouet, "A graph approach to placement of service functions chains," in Proc. IFIP/IEEE IM, May 2017, pp. 134–141.
19. M. C. Luizelli, L. S. Buriol, and L. P. Gaspary, "A fix-and-optimize approach for efficient and large scale virtual network function placement and chaining," Computer Communications, vol. 102, pp. 67 – 77, 2017.

Додаток А

АЛГОРИТМ HFA (PYTHON)

```
def heuristic_fitting(vnfs, resources, proto_actions):
    """
    vnfs: список VNFs для розподілу.
    resources: словник з доступними ресурсами для кожного вузла.
    proto_actions: словник з прото-акціями та їх вагами для кожної VNF.
    """
    allocated_vnfs = {}  # Словник для зберігання інформації про розподіл VNF
    paths = {}  # Словник для зберігання шляхів між VNF

    # Перебір кожного VNF
    for vnf in vnfs:
        # Сортування вузлів за вагами в прото-акціях
        nodes_by_weight = sorted(proto_actions[vnf], key=proto_actions[vnf].get,
        reverse=True)

        for node in nodes_by_weight:
            # Перевірка чи вузол має достатньо ресурсів
            if all(resources[node][res] >= proto_actions[vnf][node][res] for res in
            resources[node]):
            allocated_vnfs[vnf] = node
            # Оновлення ресурсів вузла
            for res in resources[node]:
                resources[node][res] -= proto_actions[vnf][node][res]
            break

    # Знаходження шляхів між VNFs
    for vnf_source in vnfs:
        for vnf_destination in vnfs:
            if vnf_source != vnf_destination:
            # Знаходження найкращого шляху (припустимо, що в нас є функція
            find_path)
            path = find_path(allocated_vnfs[vnf_source], allocated_vnfs[vnf_destination],
            resources)
            # Перевірка на достатність пропускної спроможності
            if is_sufficient_bandwidth(path, resources):
                paths[(vnf_source, vnf_destination)] = path

    return allocated_vnfs, paths

# Приклад виклику функції:
```

```
# vnfs = ['VNF1', 'VNF2', ...]
# resources = {'Node1': {'CPU': 100, 'RAM': 200, ...}, 'Node2': {...}, ...}
# proto_actions = {'VNF1': {'Node1': {'CPU': 10, 'RAM': 20}, 'Node2': {...}, ...}, 'VNF2':
{...}, ...}
# allocated_vnfs, paths = heuristic_fitting(vnfs, resources, proto_actions)
```

Додаток В

АЛГОРИТМ E²D²PG (PYTHON)

```
import numpy as np
import copy

# Ініціалізація "студентів" (мереж)
students = [CriticNetwork(state_dim, action_dim) for _ in range(K)]
teacher = ActorNetwork(state_dim, action_dim)  # Акторська мережа
notebook = ReplayBuffer()  # "Зошит" - буфер повторень

# Навчальний процес
for episode in range(total_episodes):
    # Отримання початкового досвіду
    state = environment.reset()
    for t in range(max_steps):
        # "Студенти" пропонують ідеї на основі випадкових елементів
        actions = [student.propose_action(state) for student in students]
        # Вибір найкращої ідеї
        best_action = select_best_action(actions)
        # "Студент" виконує ідею і отримує новий досвід
        new_state, reward, done = environment.step(best_action)
        # Запис нового досвіду у "зошит"
        notebook.store(state, best_action, reward, new_state, done)
        state = new_state
        if done:
            break

    # Перевірка роботи "студентів" і вибір "вчителя"
    for student in students:
        sample = notebook.sample()
        student.learn(sample)
    best_student = select_best_student(students)
    teacher = copy.deepcopy(best_student)

    # Оновлення "зошита" з новими знаннями
    notebook.update(teacher)

# Функції, які використовуються в коді
def select_best_action(actions):
    # Вибирає найкращу дію з представлених
    # ...
```

```
def select_best_student(students):
    # Визначає найкращого "студента" (мережу з найкращою оцінкою)
    # ...
```

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

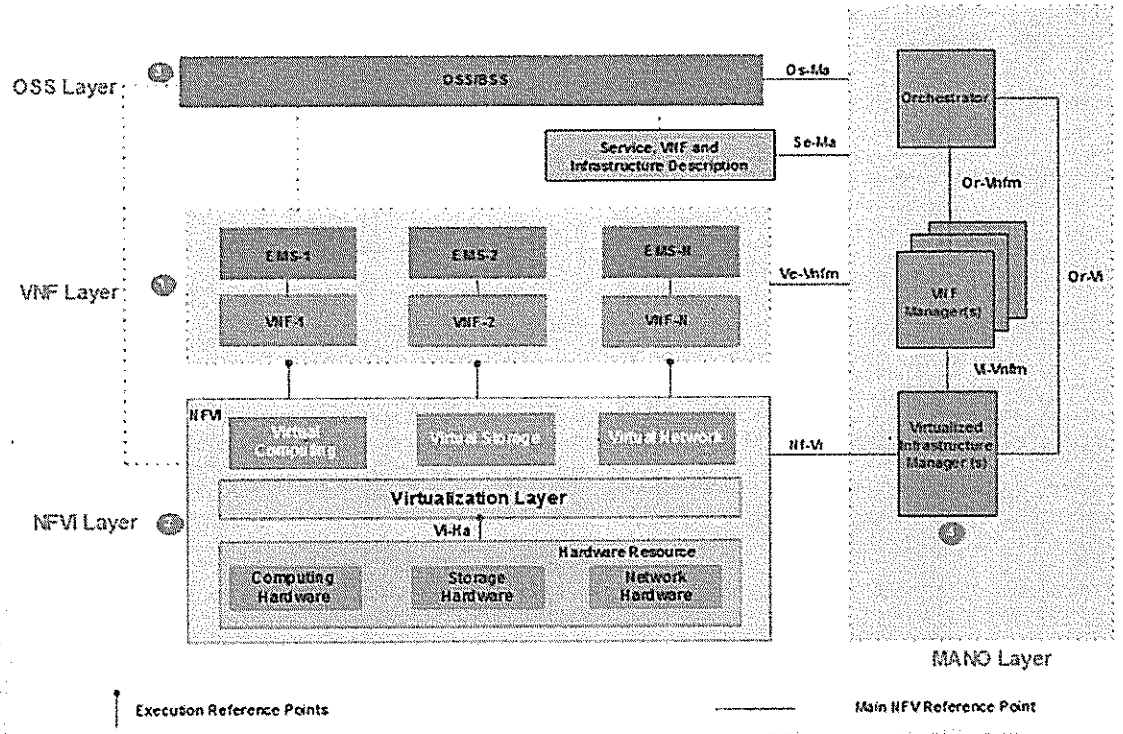
Розробка методів використання віртуалізації функцій мережі для покращення ефективності та гнучкості мобільних систем

Виконав: здобувач вищої освіти групи АРДМ-62

Михайло КОЦЮБА

Керівник: к.т.н. доц. Лариса ДАКОВА

Типова архітектура NFV



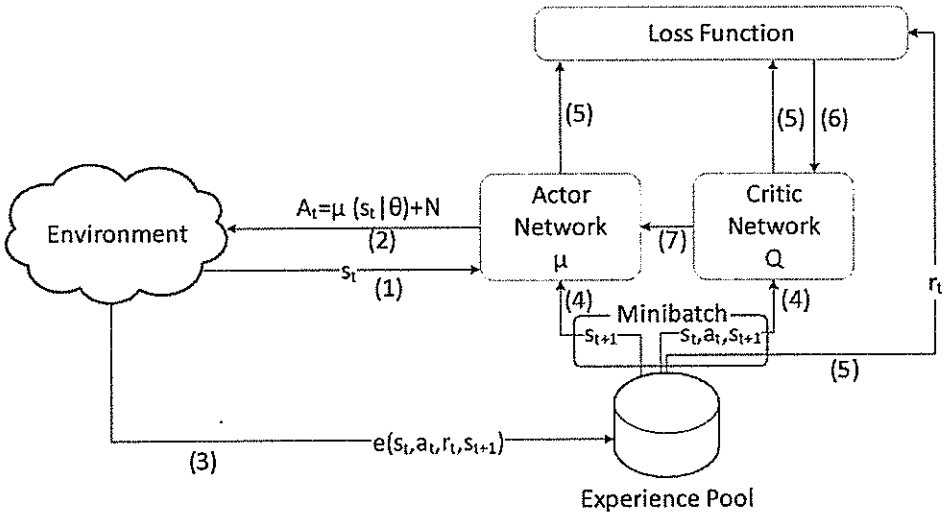


Fig. 1: DDPG

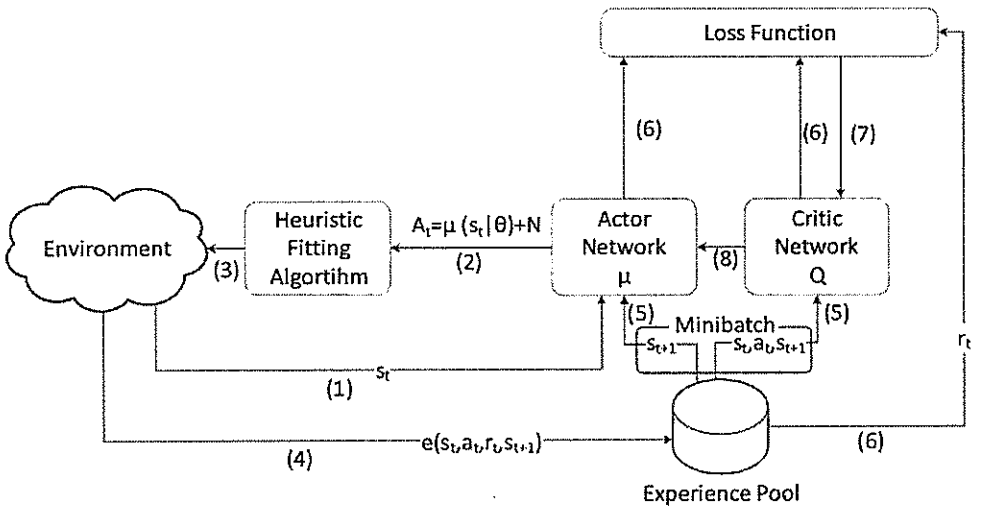


Fig. 2: DDPG-HFA

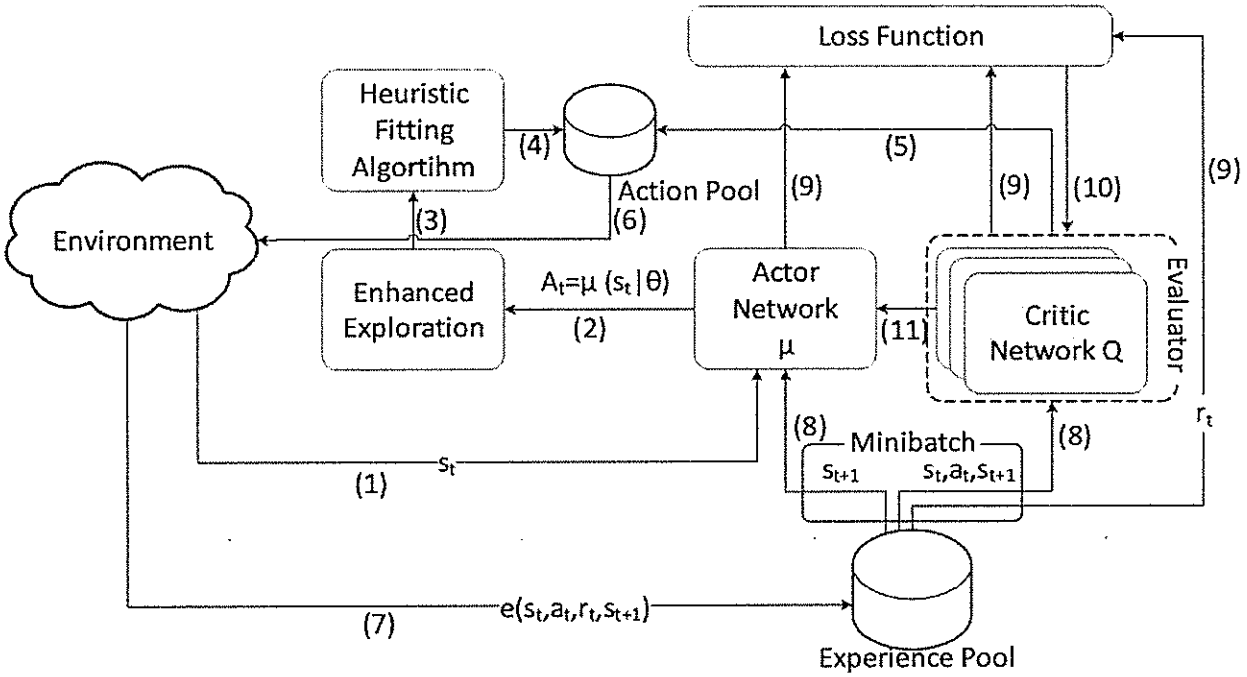
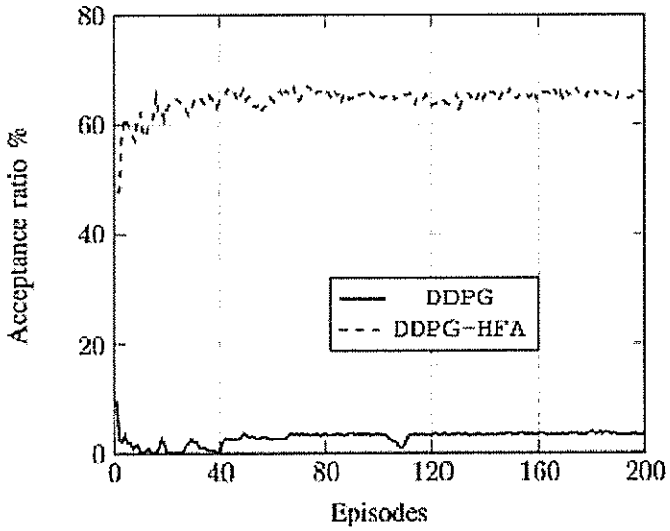
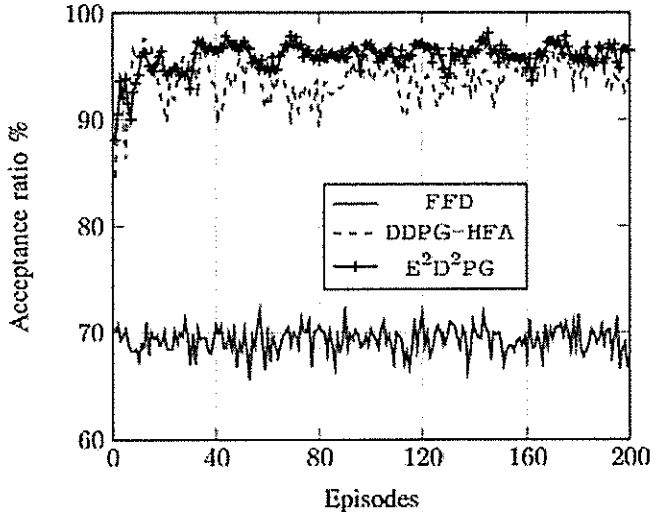
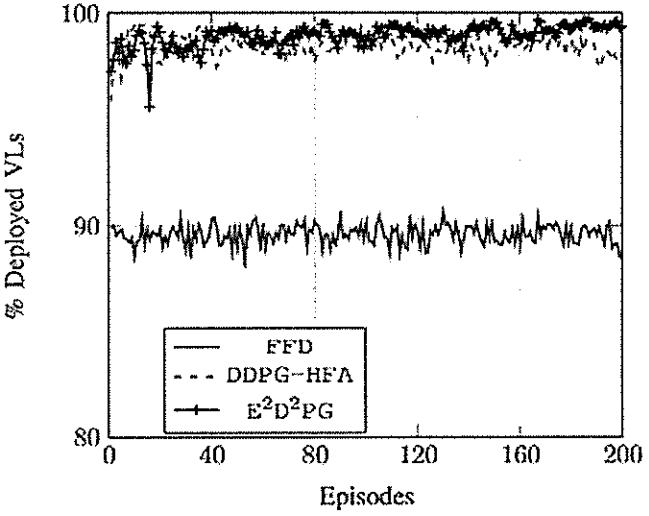

Loss Function
Heuristic Fitting Algortihm
(4)
Action Pool
(5)
(9)
(9)
(10)
(9)
Environment
(3)
(6)
Enhanced Exploration
$A_t=\mu(s_t|\theta)$
(2)
Actor Network μ
(11)
Critic Network Q
Evaluator
r_t
(8)
Minibatch
(8)
s_{t+1}
s_t,a_t,s_{t+1}
s_t
(1)
$e(s_t,a_t,r_t,s_{t+1})$
(7)
Experience Pool

Fig. 3: E^2D^2PG

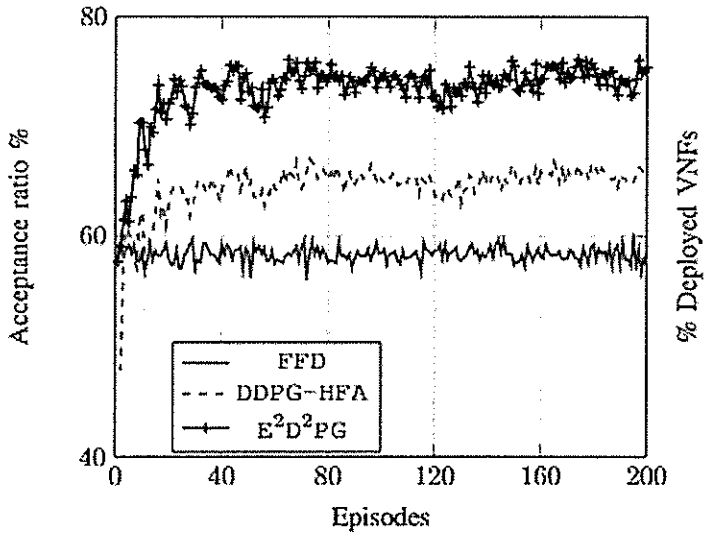

Acceptance ratio %
Episodes
DDPG
DDPG-HFA
0
20
40
60
80
0
40
80
120
160
200

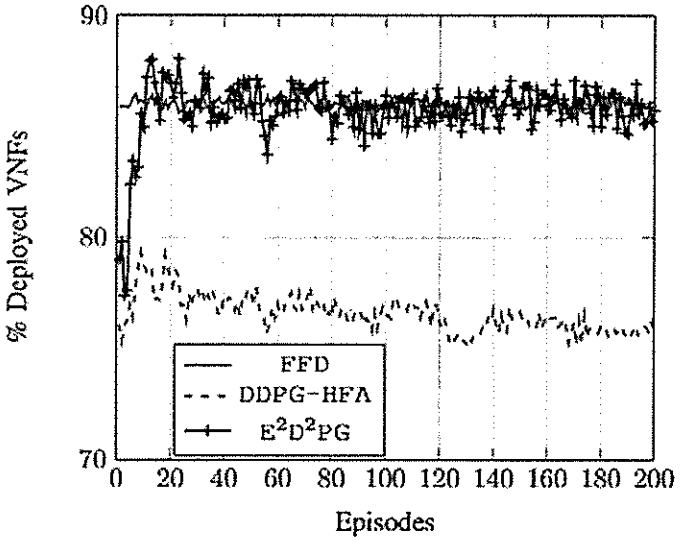

Acceptance ratio %
Episodes
FFD
DDPG-HFA
E2D2PG
60
70
80
90
100
0
40
80
120
160
200

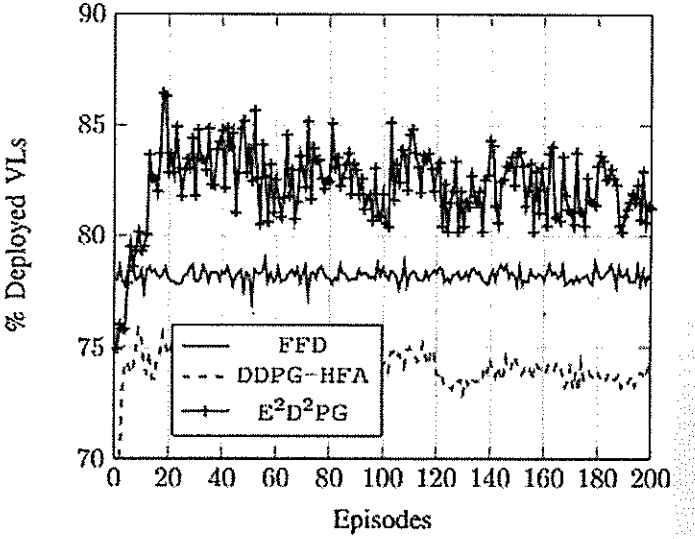

% Deployed VLs
Episodes
FFD
DDPG-HFA
E2D2PG
80
90
100
0
40
80
120
160
200

При 1 VNF-FG

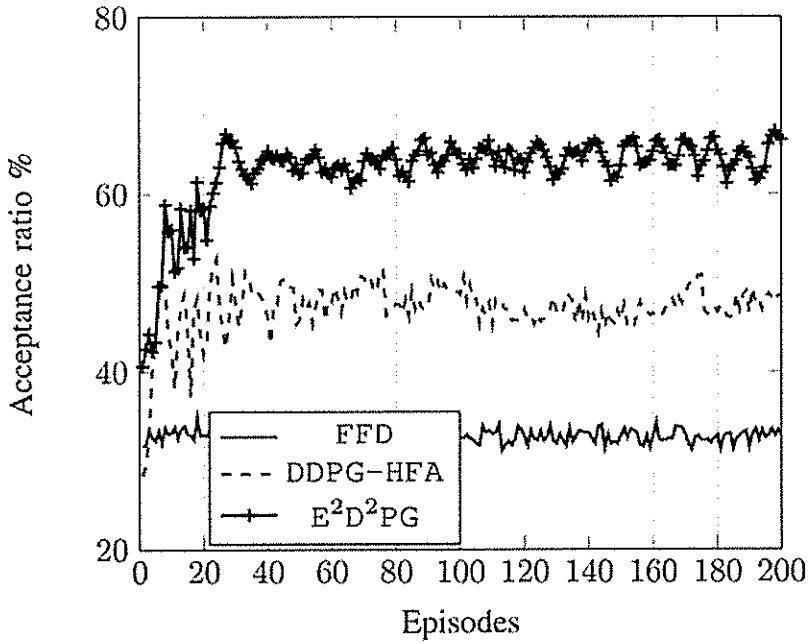
При 5 VNF-FGs


Acceptance ratio %
80
60
40
0
40
80
120
160
200
Episodes
FFD
DDPG-HFA
E²D²PG


% Deployed VNFs
90
80
70
0
20
40
60
80
100
120
140
160
180
200
Episodes
FFD
DDPG-HFA
E²D²PG


% Deployed VLs
90
85
80
75
70
0
20
40
60
80
100
120
140
160
180
200
Episodes
FFD
DDPG-HFA
E²D²PG

Інфраструктура Uninett



Протокол аналізу звіту подібності науковим керівником

Заявляю, що я ознайомився (-лась) з Повним звітом подібності, який був згенерований Системою виявлення і запобігання плагіату щодо роботи:

Автор: Михайло Коцюба

Назва: Розробка методів використання віртуалізації функцій мережі для покращення ефективност та гнучкості мобільних систем

Координатор: Лариса Дакова

Підрозділ: ННІТ

Коефіцієнт подібності 1:1.3

Коефіцієнт подібності 2:0.6

Тривога: 0

Після аналізу Звіту подібності констатую наступне:

☑ виявлені в роботі запозичення є сумлінними і не мають ознак плагіату. Тому робота визнається самостійною і допускається до захисту;

☐ виявлені в роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і самостійності її автора. Роботу направити на доопрацювання;

☐ виявлені в роботі запозичення є недобросовісними і мають ознаки плагіату або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень. У зв'язку з чим, робота не допускається до захисту.

…… ……

Дата *Підпис Наукового керівника*