DOI: 10.58254/viti.8.2025.13.153

УДК 621.396.662

д-р техн. наук Самохвалов Ю. Я. ORCID: 0000-0001-5123-1288 (ВІТІ ім. Героїв Крут)
канд. техн. наук Бовда Е. М. ORCID: 0000-0002-8267-2120 (ВІТІ ім. Героїв Крут)
Клименко В. М. ORCID: 0000-0001-7834-7213 (ВІТІ ім. Героїв Крут)
Устинов Д. А. ORCID: 0009-0004-3993-1096 (ВІТІ ім. Героїв Крут)

УПРАВЛІННЯ БАЛАНСУВАННЯМ НАВАНТАЖЕННЯ В ГІБРИДНИХ SDN-МЕРЕЖАХ НА ОСНОВІ АЛГОРИТМУ ІТЕРАЦІЙНОГО РОЗСЛАБЛЕННЯ З МАСШТАБУВАННЯМ І ОКРУГЛЕННЯМ

У статті розглянуто задачу оптимального перерозподілу трафіку в гібридних SDN-мережах, де поєднуються традиційні технології маршрутизації (OSPF) та програмно-керовані мережі (SDN). Показано, що класичні методи балансування навантаження (DNS-Round Robin, ECMP, ADC тощо) не забезпечують необхідної гнучкості у складних телекомунікаційних середовищах. Проаналізовано сучасні підходи, зокрема використання нейронних мереж, навчання з підкріпленням, багатошляхове та енергоефективне балансування. Встановлено, що для гібридних мереж актуальним є врахування як QoS, так і обмежень апаратних ресурсів комутаторів (TCAM).

Формалізовано задачу багатопродуктового обмеженого поділу потоку (MCFS) у вигляді задачі оптимізації з урахуванням пропускної здатності каналів, кількості шляхів та обмежень пам’яті комутаторів. Доведено, що точне розв’язання задачі є NP-складним, тому запропоновано наближений метаевристичний метод – ітераційне розслаблення з масштабуванням та округленням (Iterative Relaxation with Scaling and Rounding, IRSR). Алгоритм поєднує розв’язання лінійного розслаблення, масштабування пропускної здатності та поетапне округлення з повторною оптимізацією.

Отримані результати підтверджують ефективність IRSR у зменшенні максимального навантаження на лінії зв’язку та підвищенні ефективності використання мережевих ресурсів. Запропонований підхід дозволяє автоматизувати процеси управління трафіком, запобігати перевантаженням та формувати адаптивну політику маршрутизації. Подальші дослідження спрямовані на інтеграцію інтелектуальних методів прогнозування для підвищення рівня автоматизації та продуктивності гібридних SDN-мереж.

Ключові слова: *гібридна SDN-мережа, оптимальний перерозподіл трафіку, балансування навантаження, багатопродуктовий потік, метаевристичні методи, IRSR-алгоритм, QoS, обмеження TCAM.*

Y. Samokhvalov, E. Bovda, V. Klimenko, D. Ustinov. Load balancing management in hybrid SDN networks based on iterative relaxation algorithm with scalation and rounding

The paper addresses the problem of optimal traffic redistribution in hybrid SDN networks that combine traditional routing technologies (OSPF) with Software-Defined Networking (SDN). It is shown that classical load balancing methods (DNS-Round Robin, ECMP, ADC, etc.) lack the required flexibility in complex telecommunication environments. A survey of recent approaches highlights the use of neural networks, reinforcement learning, multipath and energy-efficient balancing. For hybrid networks, special attention is given to both Quality of Service (QoS) requirements and hardware constraints of switches (TCAM).

The problem is formalized as a Multicommodity Constrained Flow Splitting (MCFS) optimization task considering link capacities, path limitations, and memory constraints of network devices. Since the exact solution is NP-hard, an approximate metaheuristic method – Iterative Relaxation with Scaling and Rounding (IRSR) – is proposed. The algorithm combines solving a relaxed linear program, scaling of link capacities, and iterative rounding with resource reallocation.

The obtained results demonstrate that IRSR effectively reduces maximum link utilization and improves bandwidth resource efficiency. The proposed approach enables automated traffic management, prevents network overloads, and supports adaptive routing policies. Future research will focus on integrating intelligent forecasting and decision-making techniques to enhance automation and performance in hybrid SDN environments.

Keywords: *hybrid SDN network, optimal traffic redistribution, load balancing, multicommodity flow, metaheuristic methods, IRSR algorithm, QoS, TCAM constraints.*

Вступ. Мережева інфраструктура центрів обробки даних (ЦОД) працює в умовах підвищеного навантаження та неоднорідності трафіку. Існує кілька методів керування навантаженням [1–6]: *DNS*-Round Robin, Link Aggregation, *ECM*P, *ADC*, *SDN* мережі. Серед цих методів використання *SDN* мереж дозволяє більш ефективно управляти трафіком (*traffic engineering*) в розподілених телекомунікаційних середовищах. В *SDN*-мережі управління

трафіком здійснює контролер, який регулює пропускну здатність мережі та реалізує вибрану політику балансування навантаження на основі заданих правил, а також спеціалізованих додатків. Балансування навантаження в *SDN*-мережах застосовується в дата-центрах, мережах операторів, *IoT* (Інтернет речей), бездротових мережах.

До основних політик балансування навантаження в *SDN* відносяться:

алгоритми балансування навантаження [7–13];

застосування нейронних мереж для прогнозування навантаження, виявлення аномалій і прийняття рішень щодо перебалансування трафіку в режимі реального часу [14–16];

архітектурні підходи (централізоване балансування, розподілене балансування, ієрархічне балансування) [17–20].

В межах політик балансування навантаження в *SDN* мережах на сьогодні виділяються декілька напрямів наукових досліджень по перерозподілу навантаження: використання штучних нейронних мереж (далі – ШНМ) або навчання з підкріпленням для динамічного балансування навантаження; балансування навантаження з урахуванням *QoS* (Quality of Service); балансування невеликої кількості дуже великих потоків («слонів»), які займають більшу частину пропускної здатності і величезної кількості дрібних потоків («ми шей»); балансування багатошляхового навантаження; енергоефективне балансування навантаження; балансування навантаження в мультиконтролерних середовищах *SDN*-мереж. Нижче проведено аналіз відомих робіт по балансуванню навантаження.

Аналіз останніх досліджень і публікацій. Робота [21] присвячена механізму роботи *OpenFlow*, який дозволяє реалізувати балансування навантаження. В роботі [22] дається інформація про різні методи детекції та управління слонівними потоками, які є основою для їх балансування. В роботі [23] розглянуто приклад розподілу навантаження на контролери *SDN*. В роботі [24] показано використання глибокого навчання з підкріпленням для завдань трафік-інженерінга, що включає балансування навантаження. В роботі [25] розглянуто приклад реалізації багатошляхового балансування. В роботі [26] розглянуто приклад, який поєднує енергоефективність з машинним навчанням. В роботі [27] пропонується використовувати поглиблене навчання для побудови адаптивної системи балансування навантаження в *SDN*. Розглядається, як глибокі нейронні мережі можуть аналізувати складні стани мережі та динамічно приймати рішення щодо маршрутизації трафіку для оптимізації продуктивності. Такі підходи дозволяють системі самостійно навчатися та адаптуватися до мінливих моделей трафіку без явного програмування правил. В роботі [28] пропонується механізм балансування навантаження для самих контролерів *SDN*, який враховує їх «довіру» та «репутацію», що є критично важливим у розподілених середовищах з декількома контролерами. Це дозволяє не тільки розподілити навантаження, але і забезпечити надійність і захист від потенційно скомпрометованих хостів. В роботі [29] розглянуто методи, які спрямовані на досягнення низької затримки та високої пропускної здатності, і що вони є основними цілями балансування навантаження. Обговорюються різні підходи та алгоритми, які *SDN* надає для оптимізації продуктивності мережі, включаючи конкретні методи для зменшення затримки, що особливо актуально для додатків, чутливих до часу. В роботі [30] досліджується гібридний підхід до балансування навантаження в мережах *SDN* з кількома контролерами. Де «гібридний» означає комбінацію традиційних технологій маршрутизації та технології *SDN*. У великих, складних мережах, де повна централізація може бути неефективною, гібридні моделі відіграють ключову роль у забезпеченні більшого балансування навантаження та підвищеної масштабованості.

На основі проведеного аналізу маємо зробити висновок, що переважна більшість вище наведених авторів проводили свої дослідження за різної конфігурації мереж, для окремих видів мереж. Разом з тим, в неповній мірі враховуються завдання перерозподілу навантаження в мережі у випадку одночасного використання традиційних технологій маршрутизації в поєднанні з технологіями *SDN* та оптимізації пропускної здатності ліній зв’язку.

В статті розглянуто підхід до управління навантаженням в гібридних *SDN*-мережах, який дозволяє знаходити можливі конфігурації поділу потоку трафіку. Це дає змогу формувати політики управління трафіком для попередження перевантажень та оптимізації використання ресурсів пропускної здатності мережі.

Метою статті є створення алгоритму управління в гібридних *SDN*-мережах для мінімізації максимального використання ліній зв’язку в мережі (балансування навантаження) та оптимізації ресурсів пропускної здатності мережі.

Постановка задачі оптимального перерозподілу трафіку в гібридних *SDN*-мережах.

Під гібридною *SDN*-мережею розуміють мережу, у якій поєднуються традиційні технології маршрутизації з технологіями *SDN* (*SDN*-комутатори, *OSPF*-маршрутизатори, комутатори) (рис. 1). Такі мережі є найбільш поширеними на практиці, оскільки, по-перше, базуються на перевірених технологіях передавання даних, а по-друге, застосування *SDN*-контролерів і комутаторів (до 30 % обладнання) суттєво підвищує гнучкість та ефективність мережі [21; 22; 30].

У мережі, що складається з *SDN*-елементів (контролерів і комутаторів *SDN*) та *OSPF*-маршрутизаторів, здійснюється керування трафіком із метою мінімізації максимального навантаження на канали мережі та оптимізації пропускної здатності мережі. *OSPF*-маршрутизатори періодично оновлюють свої таблиці, обчислюючи оптимальні маршрути, тоді як *SDN*-комутатори переналаштовуються за вказівками *SDN*-контролера. Для моніторингу мережі *SDN*-контролер періодично опитує *OSPF*-маршрутизатори та *SDN*-комутатори, отримуючи дані щодо навантаження потоків.

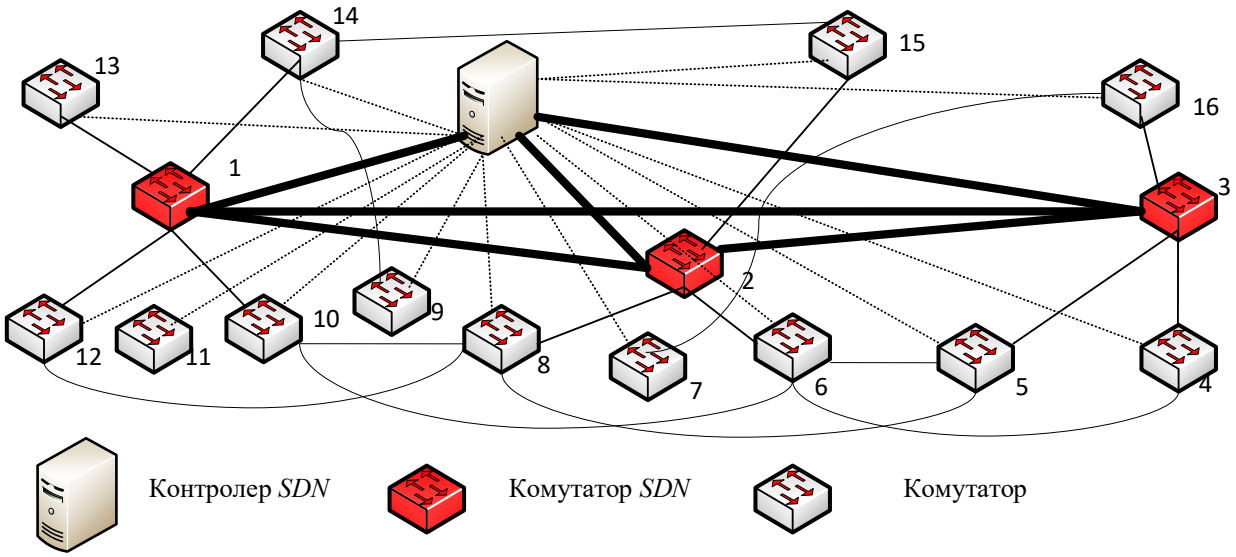

Рис. 1. Варіант використання традиційної технології маршрутизації в поєднанні з технологіями *SDN*

SDN-контролер містить самонавчальну базу знань, у якій зберігаються дані та можливі варіанти розподілу трафіку. На основі аналізу цієї інформації контролер, керований оператором, за допомогою системи підтримки прийняття рішень, формує впливи на *SDN*-комутатори, змінюючи їхню логіку роботи. Це дозволяє автоматизувати процеси маршрутизації, комутації, поділу трафіку, а також формувати політику управління потоками для запобігання перевантаженням та ефективної оптимізації пропускної здатності [22; 28; 29].

Формалізація задачі. Нехай гібридна мережа описується орієнтованим графом $G=(V, E)$, де V – множина вузлів мережі $V=\{\upsilon_1,\dots \upsilon_n\}$, E – множина спрямованих ліній зв’язків

$E=\{\boldsymbol{e_1, \dots, e_m}\} \in \boldsymbol{V \times V}$. Кожна лінія $\boldsymbol{e}$ характеризується b_e – пропускною здатністю, c_e – вартістю передачі одиниці трафіку та ρ_e – коефіцієнтом завантаження. C – множина *SDN*-комутаторів; D – множина комутаторів (маршрутизаторів); $w(e)$ – вага лінії зв'язку *OSPF*; набір $\underline{D} = \{\underline{D}_1, \underline{D}_2, \dots, \underline{D}_K\}$ потоків $к$ повинен бути маршрутизований через мережу. Кожен потік $к$ представляється як кортеж $\underline{D}_k = (h_k, t_k, T_k, P^k_p)$, де h_k і t_k є вузол-джерело та вузол-призначення; T_k – обсяг трафіку, що передається між вузлами; P^k_p – множина допустимих шляхів. $f_{k,p} \geqslant 0$ – реальна кількість трафіку (пропускної здатності) потоку k, яка призначена на шлях $p \in P_k$. T_{ht} – швидкість трафіку від вузла $\boldsymbol{s \in N}$ до деякого іншого вузла $d \in N$; W_{ud} – загальний обсяг трафіку для вузла призначення $d \in N$, що або бере свій початок або проходить через *SDN*-комутатор $e \in C$; P_{ht} – множина допустимих шляхів між h і t; I_{ut} – трафік, який вводиться за допомогою *SDN*-комутатора.

Введемо наступні обмеження. *SDN*-комутатор u може вимірювати W_{ut} для всіх адресатів t, використовуючи розширену таблицю маршрутизації; значення T_{ht} для всіх пар вузлів (h, t) *SDN* контролера не відомо. Для більш ефективного використання мережевих ресурсів, кожен потік може бути передана через кілька шляхів. Однак, для того, щоб обмежити джиттер в загальному потоці, поділ трафіку по декількох шляхах (так званий поділ потоку) може бути виконано тільки на вході пристрою, використовуючи не більше P^k_p шляхів. Для кожного потоку в мережі повинні бути виконані умови (це збереження потоків):

Обмеження. Прийнятий обсяг для потоку k не перевищує обсягу трафіку, що передається між вузлами T_k:

$$\sum_{p \in P_k} f_{k,p} \leqslant T_k, \forall k \in K. \quad (1)$$

Обмеження ємності ребер

$$\sum_{k \in K} \sum_{p \in P_k} f_{k,p} \cdot \delta_{uv}^{k,p} \leqslant b_{uv}, \ \forall (u,V) \in E. \quad (2)$$

Обмеження на виділену смугу для кожного шляху потоку:

$$f_{k,p} \geq 0, \ f_{k,p} \leqslant T_k \cdot z_{k,p}, \ \forall k, \forall p \in P_k, \quad (3)$$

де $z_{k,p} \in \{0,1\}$ – бінарна змінна, що вказує, чи використовується шлях p для потоку k. Зв'язка $f_{k,p} \leqslant T_k \cdot z_{k,p}$ дозволяє лінійно пов'язати використання шляху з присутністю потоку на ньому.

Потрібно, щоб в гібридній мережі G перерозподіл трафіку здійснювався таким чином, щоб враховувалося балансування навантаження і при цьому оптимально використовувалися ресурси пропускної здатності мережі. Це можливо здійснити шляхом перерозподілу трафіку по кільком шляхах.

Механізм поділу потоків. Поділ трафіку на кілька шляхів у гібридній *SDN*-мережі здійснюється на рівні *SDN*-комутатора з використанням хеш-функції. Вона формує числове значення на основі заданих полів пакета (*IP*-адреса джерела та призначення, номери портів, протокол тощо). Комутатор отримує пакет, обчислює хеш і за його результатом вибирає один із кількох можливих шляхів.

Набір умов пересилання, визначений формулами (1)–(3), задає конфігурацію поділу потоку. Пересилання пакетів реалізується через дві таблиці: таблицю пересилання та групову таблицю. Перша відображає пакет у групу розщеплення, друга – розподіляє потоки між наступними переходами за допомогою хеш-функції. Нерівний поділ реалізується повторенням записів у груповій таблиці. Ці записи називають відрами (*buckets*). Вони задають частки

трафіку для кожного маршруту (рис. 2). Завдання *SDN*-контролера полягає в обчисленні конфігурацій відер для комутаторів та передачі відповідних управляючих команд.

Рис. 2. Процес поділу потоку D_1 на три субшляхи з використанням хеш-функції

На рисунку 2 наведено приклад, де потік D_1 розділено з використанням хеш-функції на 3 субшляхи з коефіцієнтами (1/6, 1/3, 1/2). Поділ виконується лише на вході. На проміжних вузлах кожного суб-шляху (наприклад, вузол 4) ніякого розщеплення потоку не допускається, і тільки одне відро потрібно для зберігання умов пересилання. Таблиця пересилання і групова таблиця комутатора u містять відповідно τ_u^F та τ_u^G відра, за умови, що їх сума не перевищує розмір тернарної асоціативної пам'яті *TCAM* комутатора *SDN* мережі. Ємність тернарної асоціативної пам'яті комутатора *SDN* мережі (*TCAM*), де зберігаються правила пересилання, обмежена (до кількох тисяч записів [30]). Це накладає обмеження на поділ і може спричиняти відхилення від оптимального розподілу

Хеш-поділ потоку може відрізнятися від оптимального фракційного поділу вирішенням завдання багатопродуктового потоку (кілька типів трафіку або трафік від конкретного вузла-джерела до конкретного вузла-призначення). Таким чином, постає задача багатопродуктового обмеженого поділу потоку (*Multicommodity Constrained Flow Splitting, MCFS*), яка полягає в максимізації прийнятого трафіку та мінімізації вартості маршрутизації за умови обмежень пропускної здатності, кількості шляхів і обсягу пам'яті *TCAM*.

Нижче наведені всі параметри завдання багатопродуктового обмеженого поділу потоку (*MCFS*) [31]:

h_k – вузол-джерело для потоку k;

t_k – вузол призначення для потоку k;

T_k – трафік, що генерується потоком k;

P^k_p – максимальне число шляхів для потоку k;

c_{uv} – вартість лінії зв'язку (u,v);

b_{uv} – пропускна здатність лінії зв'язку (u,v);

τ_u – TCAM розмір пам'яті вузла u.

MCFS може бути сформульовані наступним чином:

$$max\zeta \sum_{k=1}^{K} \sum_{p=1}^{P_p^k} \sum_{(h_k;v)\in E} y_{h_k v}^{kp} - \sum_{k=1}^{K} \sum_{p=1}^{P_p^k} \sum_{(u,v)\in E} c_{uv} y_{uv}^{kp}, \quad (4)$$

$$\sum_{(h_k;v)\in E} x_{h_k v}^{kp} = 1, \ \sum_{(u;h_k)\in E} x_{u h_k}^{kp} = 0, \quad (5)$$

$$\sum_{(t_k;v)\in E} x_{t_k v}^{kp} = 0, \ \sum_{(u;t_k)\in E} x_{u t_k}^{kp} = 1, \quad (6)$$

$$\sum_{(u;v)\in E} x_{uv}^{kp} - \sum_{(v;u)\in E} x_{vu}^{kp} = 0, \ \forall u \in V\{h_k, t_k\}, \quad (7)$$

$$\sum_{k=1}^{K}\sum_{p=1}^{P_p^k} y_{uv}^{kp} \le b_{uv}, \tag{8}$$

$$y_{uv}^{kp} = T_k x_{uv}^{kp} \sum_{Q=1}^{\theta} \frac{q_Q^{kp}}{Q}, \tag{9}$$

$$\sum_{p=1}^{P_p^k} q_Q^{kp} = Q z_Q^k, \quad \sum_{Q=1}^{\theta} z_Q^k = 1, \tag{10}$$

$$\sum_{\substack{k\in K:\\ h_k=u}} \sum_{Q=1}^{\theta} Q z_Q^k + \sum_{\substack{k\in K:\\ h_k\neq u}} \sum_{p=1}^{P_p^k} \sum_{(u,v)\in E} h_{uv}^{kp} \le \tau_u, \tag{11}$$

$$h_{uv}^{kp} \le x_{uv}^{kp}, \quad h_{uv}^{kp} \le q_Q^{kp}, h_{uv}^{kp} \ge \frac{y_{uv}^{kp}}{T_k}, \tag{12}$$

$$x_{uv}^{kp}, h_{uv}^{kp} \in \{0,1\}, y_{uv}^{kp} \in R \tag{13}$$

$$q_Q^{kp} \in \{0, \dots, Q\}, z_Q^k \in \{0,1\}, \tag{14}$$

де ζ – досить велика постійна і $\theta = max\{\tau_u\}$ – це максимальне число правил, які можуть бути збережені в *TCAM*. Якщо цей параметр не заданий, діапазон змінних виглядає наступним чином:

$$k \in K,\ u \in V,\ \ p \in \left\{1, \dots, P_p^k\right\}, (u,v) \in E.$$

Цільова функція (4) в вищенаведеному формулюванні максимізує пропускну здатність, прийняту в мережі, в той час як другий додаток мінімізує витрати маршрутизації серед усіх допустимих рішень з максимально допустимою пропускною спроможністю. Для цього досить встановити, $set\ \zeta = |K||V| \max\limits_{k\in K} P_p^k \cdot \max\limits_{(u,v)\epsilon E} c_{uv}$ так, що збільшення кількості прийнятої смуги пропускання завжди буде пріоритетним в порівнянні зі зменшенням вартості маршрутизації.

Обмеження (5)–(7) по збереженню потоку, дозволяють спрямовувати трафік кожного потоку від свого вузла-джерела h_k до його вузла призначення t_k за P_p^k шляхом. Вони також запобігають циклам, що утворюються у джерела або цілі призначення.

Обмеження (8) гарантують, що обсяг трафіку, що передається по кожному спрямованому зв'язку в (u, v) не перевищує смуги пропускання b_{uv}. Обмеження (9) обчислюють частковий потік y_{uv}^{kp} (тобто пропускну здатність, що призначена шляху *p*) як добуток змінного шляху x_{uv}^{kp}, який вказує шлях *p* потоку *k* та використовує ребро (u, v), та частину потоку *k*, що виділений шляху *p*, а саме $T_k \sum_{Q=1}^{\theta} \frac{q_Q^{kp}}{Q}$. Ці нерівності є нелінійними, так як вони пов'язані з продуктом змінними x_{uv}^{kp} и q_Q^{kp}.

Набір обмежень (10) обмежує конфігурацію відра, визначаючи, скільки відер використовується для кожного шляху p, заданого для потоку *k*. Зокрема, друге обмеження в (10) обирає лише одну загальну кількість відер *Q*, щоб бути використаними для потоку *к* серед θ можливостей в більшості, і перше обмеження гарантує, що це число *Q* розподіляється і користується потоком *к* тому, що тільки одна величина *Q* є активною, змінні q_Q^{kp} можуть бути ненульовим тільки для одного значення *Q* в (10). Таким чином, вираз $\sum_{Q=1}^{\theta} \frac{q_Q^{kp}}{Q}$ в (9) дає частку потоку *k*, яка виділяється в шлях *p*.

Обмеження (11) гарантують, що правила пересилання визначені як для розщеплення потоку, що виникають на вузлі *u*, так шляху, які проходять через вузол *u*, та не перевищують ємність *TCAM u*. Для підрахунку, скільки відер повинні бути зарезервовано для пересилання правил, ми не можемо підсумувати всі вимоги змінних x_{uv}^{kp}, які вказують на активацію витікаючих зв'язків для шляху p. Дійсно, оптимальним рішенням можна вибрати шлях з нульовим потоком на ньому ($x_{uv}^{kp}=1$ та $y_{uv}^{kp}=0$). З цієї причини, ми використовуємо двійкові змінні h_{uv}^{kp} в другій сумі (11), і ми також визначили обмеження (12), (13) щоб встановити змінну $h_{uv}^{kp}=1$, тільки якщо деякий трафік потоку *к* направляється до місця призначення по шляху *p* по лінії зв'язку (u, v). Нарешті, обмеження (14) визначають області рішення наших змінних.

Ми бачимо, що вищевказані формулювання дійсні для мережі зі спрямованими зв'язками. Однак, ми можемо легко відновити формулювання з неорієнтованим зв'язком шляхом інтерпретації (u, v) як в протилежному напрямку (v, u), так і замінити y_{uv}^{kp} на $y_{uv}^{kp}+y_{vu}^{kp}$ в обмеженнях (8).

Щоб вирішити проблему *MCFS,* запропоновано метаеврістіческій підхід, який називається ітераційне розслаблення з масштабуванням і округленням Iterative Relaxation with Scaling and Rounding (*IRSR*). *IRSR* – відноситься до наближеного (евристичного) методу, який:

спочатку розслабляє початкову (часто цілочислову) задачу до задачі лінійного програмування (*LP*);

потім масштабує значення потоків (зменшує масштаб, щоб простіше округлювати);

далі округлює знайдене дробове рішення до цілочислового (або до значень, прийнятних для практичного використання в *SDN*);

і, на решті, ітеративно повторює процес, покращуючи рішення та виконуючи повторне виділення ресурсів, поки не буде досягнуто прийнятного рішення або поки не вичерпано обчислювальний ресурс;

дозволяє розв'язувати великі задачі швидше, ніж точні методи (такі як *ILP*);

отримувати наближене рішення з гарантіями щодо відхилення;

підтримувати адаптивність у реальному часі, що особливо важливо в *SDN* мережах, де маршрути можуть динамічно змінюватись [32].

Алгоритм ітераційного розслаблення з масштабуванням і округленням (*IRSR*). Оскільки вирішення цілочисельного лінійного програмування *MILP* нездійсненно з обчислювальної точки зору, розглянемо лінійне розслаблення, видаляючи обмеження шляху і відра. Вирішуючи отриманий *LP*, який ми називаємо розслабленим *LP* (*RLP*), дамо набір засобів для підмножини вимог [32]. Недоліком цього підходу є те, що вимога $\underline{D}_K$ в кінцевому підсумку, працює більш ніж по P_p^k шляхів, і оскільки обсяг потоку на кожному шляху є дробовим, то воно також не узгоджується з обмеженнями відра, яке розташоване в оригінальній *MILP*. Таким чином, ці порушення повинні бути виправлені для отримання відповідного рішення.

Припустимо, що вимога $\underline{D}_K$ була допущена в рішенні *RLP* і припустимо, що $\hat{P}_p^k > P_p^k$ шляхів були виділені на це. Розглянемо $\hat{P}_p^k > P_p^k$ шляхів з найменшою кількістю смуги пропускання $\underline{D}_K$, що виділяються на них. Ми видаляємо їх і рівномірно перерозподіляємо їх пропускні спроможності на інші P_p^k шляхи. Після цього перерозподілу шляху, ми виходимо з округленого відра в спробі знайти відповідне рішення в рамках розмірності відра, що дається на кожну вимогу відра обмежень.

При виконанні вищевикладеного може виявитися, що краї ємностей порушуються. Ми маємо справу з цим двома способами: по-перше, перш ніж вирішувати *RLP*, ми множимо краї ємності мережі на (1 – α), де $\alpha \in [0,1)$ – параметр. Можливості, що обмежуються таким чином, можуть змусити *RLP* вибрати більш дорожчі шляхи, але це дає нам можливість виконувати наступні фази перерозподілу і округлення (які, звичайно ж, виконуються з

фактичними ємностями країв). Насправді, оскільки ми не знаємо α апріорно, яке гарне значення α, ми пробуємо їх набір і вибираємо той, який дає кращий результат з точки зору виділеної смуги пропускання після фази округлення відра. Це можна зробити паралельно, оскільки платформи *SDN* мають велику обчислювальну потужність.

Другий спосіб, з яким ми маємо справу з порушеннями пропускної здатності – це ітерація: вимоги, які відкидаються для порушення обмежень пропускної здатності, стають внеском в інше округлення алгоритму. Ми продовжуємо ітерацію до тих пір, поки не буде досягнуто якогось поліпшення в кількості виділень або розподілі всіх вимог.

Слід зазначити, що ми вирішуємо *RLP* з використанням генерації стовбцю (Column Generation – *CG*), що є підходом, який часто ефективний при роботі з *LP* з великим числом змінних. Однак *CG* можна не використовувати, і замість цього може бути використаний будь-який відомий алгоритм для вирішення *LP*, наприклад:

1. Масштабування. Завдання масштабів країв пропускної здатності (1 - α).

2. Розслаблений *LP* (*RLP*). Видалення номера шляху і обмеження відра, ослаблення цілих змінних і рішення.

3. Проміжні відра. Для кожного (повністю або частково) виділеного запиту $\underline{D}_K$ видаляється один запис у вигляді відра з кожного вузла по кожній шляху в розподілі (маршрутизація шляху через *TCAM* важить одному запису).

4. Відро округлення. Використати округлення в результуючий мережі і вимагати установки для отримання розміру *B* відра для кожного запиту.

5. Перерозподіл шляхів. Для кожного (повністю або частково) виділеного $\underline{D}_K$, якщо $\hat{P}_p^k > P_p^k$ йому були виділені шляхи, то рівномірно перерозподілити $\hat{P}_p^k > P_p^k$ надлишкові шляхи на найбільш сильно розподіленому P_p^k.

6. Округлення відер. Для кожного виділеного $\underline{D}_K$ для b= 1,..., B_k перерозподіляється потік між виділеними (не більше P_p^k) підшляхами $\underline{D}_K$ в одиницях d_k/b, щоб максимально точно відтворити попередній розподіл.

7. Фільтрація. Проведення перевірки, які вимоги тепер порушують обмеження пропускної здатності. Виконати це послідовно, завжди перевіряючи залишки мережі на вже виділені вимоги.

8. Оновлення залишкових відер. Оновлювати використання відер як на вихідних вузлах, так і на проміжних вузлах для вимог, які були виділені під час цієї ітерації.

9. Ітерація. Всі вимоги, відхилені на попередній фазі, стають внеском в повторення цього виділеного алгоритму.

Слід зазначити, що округлення відра є нетривіальним завданням. Підхід полягає в наступному: для кожного виділеного $\underline{D}_K$ виділяється відра по одному, і при цьому створюється профіль потоку для кожного шляху для $\underline{D}_K$ на основі цих відер. Якщо ми збираємося додати рівно *b* відер в цілому, то кожне відро додає d_k/b до мінімуму шляху, якому він призначений. Відро поміщається випадковим чином на розподіл ймовірності, що отримане з профілю потоку, яке викликане розподілом поточного відра і яке задається рішенням *RLP*. Більша відмінність цих профілів по конкретному шляху робить його більш імовірним, що відро буде прив'язане до цього шляху.

Переваги алгоритму *IRSR*. Обчислювальна ефективність – значно менші витрати часу порівняно з *ILP*-методами. Адаптивність – можливість повторного запуску при зміні стану мережі. Якість рішень – гарантує допустимий розподіл, близький до оптимального.

Висновки

Сформульовано та досліджено задачу оптимального розподілу потоків у гібридних *SDN*-мережах з урахуванням обмежень пропускної здатності та ресурсів *TCAM*. Доведено, що точне розв'язання є *NP*-складним, тому запропоновано використання наближеного алгоритму *IRSR*.

Отримані результати підтверджують ефективність підходу для балансування трафіку та запобігання перевантаженням. Подальші дослідження будуть спрямовані на інтеграцію методів прогнозування та інтелектуального керування для підвищення рівня автоматизації та ефективності трафік-інженерії.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Konrad Połys, Krzysztof Grochla, Michał Gorawski. LTE Load Balancing with Tx Power Adjustment and the Real Life Mobility Model // Computer Networks (CN 2018). Pp. 183–192.

2. Ahmed Hazim Alhilali, Ahmadreza Montazerolghaem. Artificial intelligence based load balancing in SDN: A comprehensive survey Networking and Internet Architecture (cs.NI). URL: https://doi.org/ 10.48550/arXiv.2308.02149.

3. Alejandro Aguilar-Garcia, Raquel Barco, Sergio Fortes, Pablo Muñoz. Load balancing mechanisms for indoor temporarily overloaded heterogeneous femtocell networks. *EURASIP Journal on Wireless Communications and Networking.* 2015. Article number: 29.

4. Електронне посилання від 06.10.2025. URL: https://avinetworks.com/what-is-load-balancing/.

5. Електронне посилання від 10.10.2025. URL: https://www.resonatenetworks.com/2020/05/25/what-are-the-different-types-of-load-balancers/.

6. Електронне посилання від 16.10.2025. URL: https://www.dnsstuff.com/what-is-server-load-balancing.

7. A Survey on Static and Dynamic Load Balancing Algorithms in Cloud Computing (Mukati & Upadhyay, 2019).

8. A Comprehensive Study of Static, Dynamic and Hybrid Load Balancing (Hamadah, ~2014).

9. Load balancing in cloud computing: Methodological survey on different types of algorithm (~2016).

10. Dynamic Load Balancing in Cloud Computing: A Review and a Novel Approach (Laha et al., 2024).

11. Load balancing techniques in cloud computing environment (Shafiq et al., 2022).

12. Comparative analysis of metaheuristic load balancing algorithms for efficient load balancing in cloud computing (Zhou et al., 2023).

13. Classification of Load Balancing Optimization Algorithms (Moharamkhani, 2024).

14. Elastic neural network method for load prediction in cloud computing grid (Qaddoum et al., IJECE).

15. Host Load Prediction with Bi-directional Long Short-Term Memory in Cloud Computing (Shen & Hong, 2020).

16. A hybrid CNN-LSTM model for predicting server load in cloud computing (2024).

17. Hierarchical Load Balancing Strategy in Cloud Environment (RTIS 2017 / Springer, 2018).

18. Multi-Level Load Balancing Methods for Hierarchical Web Server Clusters (Yeon & Jeong, 2015).

19. Centralized and Decentralized Non-Cooperative Load-Balancing Games among Federated Cloudlets (Mondal et al., 2020).

20. A Load-Balance System Design of Microgrid Cluster Based on Hierarchical Petri Nets (Sicchar et al., 2018).

21. Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, Jonathan Turner. OpenFlow: Enabling Innovation in Campus Networks // ACM SIGCOMM Computer Communication Review (CCR). 2008. DOI: 10.1144/ccr.38.2.69.

22. S. A. Al-rubaye, Z. H. Zaidan, R. H. Zaidan, M. S. Mahmood, O. M. J. Al-Musa, T. M. Al-Jobouri. Elephant Flow Detection and Migration for Software-Defined Networks: A Comprehensive Review // Wireless Personal Communications. 2022. DOI: 10.1007/s11277-022-10023-x.

23. S. Al-rubaye, J. L. De La Fuente-Moya, J. A. Hernandez, P. S. S. De Silva SDN Controller Load Balancing Based on Reinforcement Learning. 2021 IEEE International Conference on Communications Workshops (ICC Workshops). DOI: 10.1109/ICCWorkshops50700.2021.9473859.

24. Y. Mao, Y. Ji, Y. Li, S. Mao, T. Zhang, B. Ning, N. Zhang. Deep Reinforcement Learning for Network Traffic Engineering: A Survey // IEEE Communications Surveys & Tutorials. 2021. DOI: 10.1109/COMST.2021.3060505.

25. K. A. H. Hassan, M. S. Rahman, M. A. A. Al-Haj, A. F. M. S. Islam. Multi-path Load Balancing in SDN Using Link Status and Traffic Type. 2020. 2nd International Conference on Advanced Information and Communication Technology (ICAICT). DOI: 10.1109/ICAICT50694.2020.9273574.

26. A. E. M. E. Elghoniemy, E. I. Hassan, M. M. Fouad Energy-Efficient Load Balancing in Software-Defined Data Centers: A Deep Reinforcement Learning Approach // IEEE Access. 2023. DOI: 10.1109/ ACCESS.2023.3243161.

27. J. R. Aguilar-Velazco, F. H. Rivera-Reyes, F. J. Ramos-Velasco, M. A. Portilla-Flores. Deep Learning-Based Load Balancing for Software-Defined Networking // Sensors. 2024. DOI: 10.3390/s24082531.

28. H. A. K. Al-Hasib, M. N. A. Khan, K. M. A. Salam, M. Z. Abedin. A Load Balancing and Fault Tolerance Mechanism for SDN Controllers Using Trust and Reputation System // IEEE Access. 2022. DOI: 10.1109/ACCESS.2022.3142079.

29. F. M. H. Aldughaim, M. K. Abdullah, S. D. Al-Sharabi. Traffic Engineering for Low Latency and High Throughput in Software-Defined Networks: A Survey // Journal of Network and Computer Applications. 2021. DOI: 10.1016/j.jnca.2021.103009.

30. H. N. Ahmed, M. I. Sarim, S. N. Che-Pazi, N. A. A. Nordin. Hybrid Load Balancing for Software Defined Networks with Multiple Controllers // 2020 IEEE International Conference on Internet of Things and Intelligence Systems (IoTaIS). DOI: 10.1109/IoTaIS50847.2020.9329402.

31. Бовда Е. М. Метод управління перерозподілом навантаження в SDN мережах // Зб. наук. праць / Військовий інститут телекомунікації та інформатизації. Київ, 2017. С. 6–16.

32.François Lamothe, Emmanuel Rachelson, Alain Haït, Cedric Baudoin, Jean-Baptiste Dupe. Randomized rounding algorithms for large scale unsplittable flow problems. URL: https://doi.org/ 10.1007/s10732-021-09478-w.