МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ І ПРИКЛАДНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

СУЛІМА ВІКТОР КОНСТЯНТИНОВИЧ

Допускається до захисту:
в.о. завідувача кафедри
інформаційних технологій,
д-р. техн. наук, професор
_____Наталія ВЕСЕЛОВСЬКА
«____»_____ 2026 р.

ПРОГРАМНА РЕАЛІЗАЦІЯ АЛГОРИТМІВ АНАЛІЗУ ДИНАМІКИ ІНФОРМАЦІЙНИХ КАСКАДІВ

Спеціальність 122 Комп’ютерні науки

Кваліфікаційна (бакалаврська) робота

Керівник:
Надія ПОТАПОВА, доцент кафедри
інформаційних технологій,
к. е. н., доцент

_____

Оцінка: _____ / _____ / _____
(бали/за шкалою ЄКТС/за національною шкалою)

Голова ЕК: _____

Вінниця - 2026

АНОТАЦІЯ

Суліма В. К. Програмна реалізація алгоритмів аналізу динаміки інформаційних каскадів. Спеціальність 122 «Комп'ютерні науки», освітня програма «Комп'ютерні науки». Донецький національний університет імені Василя Стуса, Вінниця 2026.

У кваліфікаційній (бакалаврській) роботі досліджено та проаналізовано теоретичні основи поширення інформації в соціальних мережах, виділено ключові топологічні та часові принципи формування інформаційних каскадів. За допомогою сучасних технологій було розроблено програмний комплекс аналізу динаміки інформаційних потоків, функціонування якого направлено на виявлення структурних аномалій, оцінювання швидкості дифузії контенту та прогнозування масштабів каскадних поширень у соціальних медіа.

Ключові слова: інформаційний каскад, спрямований граф, NetworkX, FastAPI, Independent Cascade Model, React.

76 ст. 15 рис., 17 табл., 2 дод., 50 джерел.

ABSTRACT

Sulima V. K. Software Implementation of Algorithms for Analyzing the Dynamics of Information Cascades. Specialty 122 «Computer Science», educational program «Computer Science». Vasyl Stus Donetsk National University, Vinnytsia 2026.

In the qualification (bachelor's) work, the theoretical foundations of information propagation within digital networks are investigated and analyzed, highlighting key topological and temporal principles of information cascade formation. Using modern technologies, a software suite has been developed to analyze the dynamics of information flows; its operation is designed to identify structural anomalies, assess the rate of content diffusion, and predict the scale of cascading propagation on social media.

Keywords: information cascade, information diffusion, directed graph, NetworkX, FastAPI, Independent Cascade Model, React.

ЗМІСТ

ВСТУП

Висока швидкість обміну повідомленнями, алгоритмічне ранжування контенту та активна участь користувачів сприяють виникненню явища інформаційних каскадів – процесів масового поширення інформації, які можуть набувати вірусного характеру. Такі каскади відіграють важливу роль як у формуванні громадської думки, так і в розповсюдженні новин, маркетингових кампаній або дезінформації.

Особливу актуальність дослідження набуває в умовах інформаційного перевантаження, коли користувачі щоденно стикаються з великою кількістю контенту різної якості та достовірності. Вірусне поширення інформації може мати як позитивні наслідки (швидке інформування, популяризація соціально важливих тем), так і негативні (поширення фейкових новин, маніпуляцій, інформаційних атак). У зв’язку з цим виникає потреба у створенні ефективних інструментів для вирішення таких проблем.

Складність полягає в їх нелінійному характері, залежності від поведінкових факторів користувачів та впливі алгоритмів соціальних платформ. Традиційні методи аналізу даних не завжди дозволяють повною мірою описати такі процеси, тому виникає необхідність у застосуванні сучасних підходів, зокрема методів теорії графів.

У цьому контексті особливого значення набуває програмна реалізація алгоритмів, здатних автоматизувати процес збору, обробки та аналізу інформації про поширення контенту. Вони дозволяють не лише фіксувати факт виникнення каскаду, але й оцінювати його параметри, прогнозувати подальший розвиток та виявляти аномальні або маніпулятивні патерни.

Метою бакалаврської роботи є комплексне дослідження та розробка алгоритмів аналізу динаміки інформаційних каскадів у соціальних мережах, а також їх програмна реалізація для оцінювання параметрів поширення інформації.

Об’єктом дослідження є інформаційні процеси у соціальних мережах,

зокрема механізми поширення інформації між користувачами.

Предметом дослідження є алгоритми аналізу динаміки інформаційних каскадів, їх математичні моделі та програмні засоби реалізації.

Для реалізації даного дослідження були поставлені наступні *завдання*:

1. Провести аналіз сучасного стану досліджень інформаційних процесів у соціальних мережах, акцентуючи увагу на сутності поняття інформаційного каскаду, визначення його основних характеристик та параметрів;
2. Дослідити існуючі алгоритми аналізу та моделювання поширення інформації;
3. Обґрунтувати вибір методів і підходів до аналізу інформаційних каскадів;
4. Розробити алгоритм аналізу динаміки інформаційного каскаду;
5. Реалізувати програмний модуль для обробки та аналізу даних соціальних мереж;
6. Провести експериментальну перевірку функціонування запропонованого алгоритму та проаналізувати отримані результати.

Основна частина роботи складається з трьох розділів. У першому розділі досліджено теоретичні основи інформаційних каскадів та особливості їх поширення в соціальних мережах. У другому розділі розроблено математичну модель інформаційного каскаду, алгоритми його побудови та аналізу структурної і часової динаміки. У третьому розділі описано програмну реалізацію системи аналізу інформаційних каскадів, включаючи архітектуру застосунку, модулі збору даних, побудови графів та обчислення метрик поширення інформації.

Практична цінність бакалаврської роботи визначається можливістю безпосереднього застосування розроблених алгоритмів та програмного комплексу у реальних сценаріях моніторингу, аналізу та управління інформаційними потоками у соціальних мережах. На відміну від суто теоретичних досліджень у галузі мережевого аналізу, дана робота орієнтована на отримання результатів, придатних до інтеграції у практичні системи.

Однією з найбільш актуальних практичних областей застосування є виявлення та нейтралізація дезінформаційних кампаній. Розроблені алгоритми дозволяють на ранніх стадіях поширення розрізняти органічні каскади від скоординованих.

Апробація результатів даного дослідження була здійснена на IV Всеукраїнської науково-практичної конференції студентів, аспірантів та молодих вчених «Прикладні інформаційні технології» (м. Вінниця, 18 липня 2023 року), V Всеукраїнської науково-практичної конференції здобувачів, аспірантів та молодих вчених «Прикладні інформаційні технології» (м. Вінниця, 20 лютого 2025 року), Міжнародна наукова інтернет-конференція на тему: "Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення", випуск 110 (м. Тернопіль, 7-8 травня 2026 року), VII Всеукраїнської науково-практичної конференції здобувачів, аспірантів та молодих вчених «Прикладні інформаційні технології» (м. Вінниця, травня 2026 року), журнал «Наука і техніка сьогодні» випуск №5 (м. Київ, 3 червня 2026 року).

РОЗДІЛ 1
ТЕОРЕТИЧНІ ОСНОВИ АНАЛІЗУ ІНФОРМАЦІЙНИХ КАСКАДІВ

1.1. Особливості функціонування інформаційних процесів у соціальних мережах

Соціальні мережі це одна з найбільш динамічних форм цифрової комунікації – за даними аналітичної платформи DataReportal, станом на 2024 рік кількість активних користувачів соціальних мереж у світі перевищує 5,1 мільярда осіб, що становить понад 62% населення планети [1]. Такий масштаб охоплення пояснює особливу роль цих платформ як каналів поширення інформації – від новин і наукових відкриттів до дезінформації та маніпулятивного контенту. Соціальну мережу доцільно моделювати у вигляді графа, де вершини – користувачі, а ребра – їхні взаємодії. Ключовими характеристиками є ступінь вузла, коефіцієнт кластеризації, середня довжина шляху та наявність центральних вузлів, що визначають швидкість переходу інформації між групами [2].

На рисунку 1.1 представлено простий орієнтований граф та його ребра з вершинами.

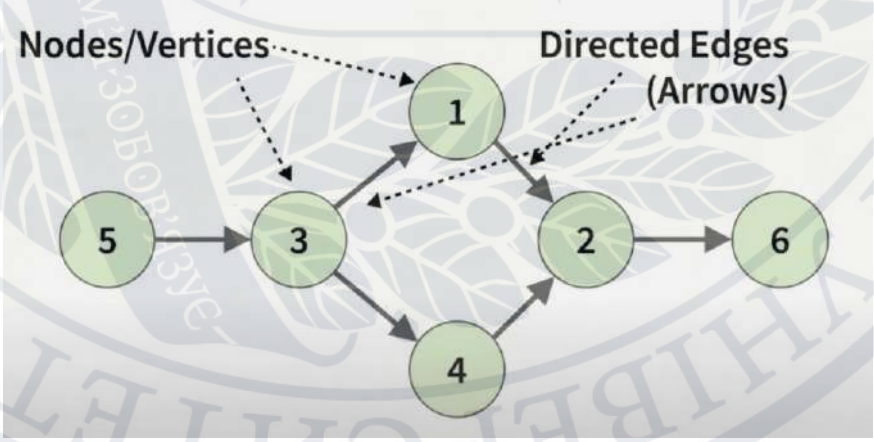

Рисунок 1.1 – Візуалізація орієнтованого графа

Кожен вузол характеризується набором атрибутів: активністю, впливовістю та поведінковими патернами – саме ця структура визначає

швидкість та маршрути руху інформаційних потоків.

Цифрові комунікації у соціальних мережах реалізуються через кілька ключових механізмів:

- асинхронна передача повідомлень (публікації, коментарі, репости);
- синхронні взаємодії (прямі ефіри, Stories, Spaces);
- алгоритмічна дистрибуція (рекомендаційні системи на основі машинного навчання).

Інформаційні потоки у соціальних мережах мають нелінійний характер і на відміну від традиційних ЗМІ з моделлю «один до багатьох», соціальні платформи реалізують модель «багато до багатьох», де кожен учасник одночасно є і споживачем, і автором контенту. Оскільки поширення повідомлень теж відбуватися нелінійно, формуючи розгалужені інформаційні каскади, що ускладнює їх аналіз і прогнозування, а ефективне виявлення маніпулятивного контенту потребує одночасного врахування як змісту повідомлень, так і структури їхнього поширення [3]. Це породжує складні мережеві ефекти, що описуються законом Меткалфа – цінність мережі пропорційна квадрату кількості її учасників n^2 [4]. Онлайн-взаємодія користувачів формує базові типи відносин у мережі, які представлено в таблиці 1.1.

Таблиця 1.1 Типи онлайн-взаємодії користувачів у соціальних мережах

Тип взаємодії	Характеристика	Приклад платформи	Вплив на поширення інформації
Симетрична (дружба)	Двосторонній зв’язок	Facebook	Помірний – замкнені групи
Асиметрична (підписка)	Односторонній зв’язок	Twitter/X, Instagram	Високий – широке охоплення
Тимчасова (взаємодія)	Ситуативний контакт	Reddit, Telegram	Вибірковий – тематичні хвилі
Групова (спільнота)	Колективна приналежність	Discord, Facebook Groups	Концентрований – в межах кластера

Важливою характеристикою цифрових комунікацій є їх часова динаміка, у дослідженні Vosoughi, Roy та Aral (2022) підтвердили, що неправдива інформація поширюється у Twitter у 6 разів швидше, ніж достовірна, оскільки вона несе елемент новизни і викликає сильніші емоційні реакції.

Вірусне поширення інформації є центральним об'єктом дослідження у галузі обчислювальних соціальних наук. Термін «вірусний контент» метафорично відображає подібність між поширенням інформації та епідемічними процесами: в обох випадках спостерігається експоненційне зростання охоплення за умови перевищення порогового значення репродуктивного числа $R_0 > 1$ [5].

При цьому інформаційна загроза не зводиться до факту неправдивості повідомлення – небезпечними можуть бути напівправдиві твердження, емоційно забарвлені інтерпретації та вирвані з контексту факти, що обумовлює необхідність багатокритеріального підходу до оцінювання ризику [6].

Механізми поширення інформації у соціальних мережах класифікуються за двома основними моделями:

1. Модель незалежних каскадів (Independent Cascade Model, IC) – кожен активований вузол має одну спробу активувати кожного зі своїх сусідів з деякою ймовірністю p(u,v). Математично процес описується рівнянням:

$$P(\text{акт.}\, v) = 1 - \prod_{u \in N_{act}(v)} \left(1 - p_{u,v}\right) \quad (1.1)$$

де $p_{u,v}$ – це ймовірність впливу вузла u на вузол v;

$\Pi_{u \in N_{act}(v)}$ – це добуток всіх сусідів вузла v.

2. Модель лінійного порогу (Linear Threshold Model, LT) – вузол активується, коли сумарна вага впливу активованих сусідів перевищує індивідуальний поріг θ_v [7].

Порівняльна характеристика моделей наведена у таблиці 1.2. Повторне розповсюдження (resharing / retweeting) є ключовим механізмом підтримання каскаду. Аналіз 4,5 мільйона каскадів у Twitter, проведений Zhang et al. (2023), показав, що розподіл глибини каскадів підпорядковується степеневому закону з

показником $\alpha \approx 2{,}3$, що характерно для систем без масштабу [6]. Практично це означає, що більшість каскадів залишаються малими (1–3 рівні), тоді як незначна частина досягає тисяч рівнів вкладеності.

Таблиця 1.2 Порівняльний аналіз моделей поширення інформації

Критерій	IC-модель	LT-модель	Модель SIR	SEIR-модель
Тип активації	Ймовірнісна	Порогова	Компартментна	Компартментна з латентністю
Пам’ять процесу	Відсутня	Кумулятивна	Відсутня	Наявна
Зворотна активація	Неможлива	Неможлива	Неможлива	Неможлива
Складність обчислення	O(n·m)	O(n·m)	O(n)	O(n)
Застосування	Маркетинг, новини	Норми, переконання	Епідемії, чутки	Дезінформація
Точність на реальних даних	Висока	Середня	Середня	Висока

На рисунку 1.2 схематично відображено типову структуру інформаційного каскаду з трьома рівнями поширення:

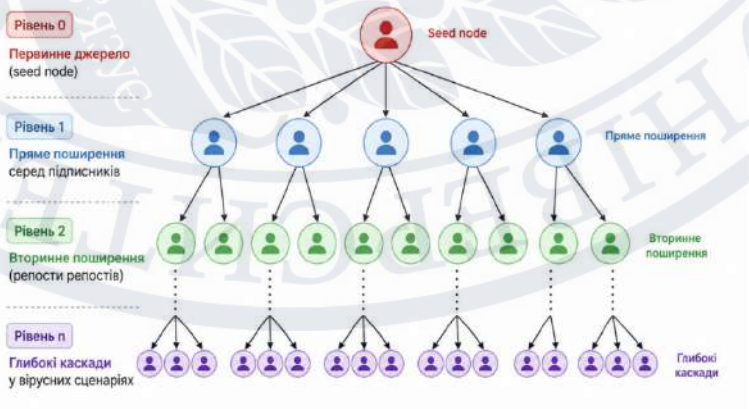

Рисунок 1.2 – Рівні поширення повідомлення в інформаційних каскадах

У таблиці 1.3 висвітлені фактори популярності контенту, які детальніше були досліджені у роботі Ferrara та Yang (2022) – де було виділено п’ять ключових груп чинників, які впливають на каскад [9].

Таблиця 1.3 Фактори вірусного поширення контенту

Група факторів	Конкретні показники	Вплив на каскад (%)	Метод вимірювання
Емоційний	Валентність, збудження, страх, гнів	34–41%	Sentiment analysis, LIWC
Структурний	Довжина тексту, наявність медіа, хештеги	18–25%	Feature extraction
Мережевий	Ступінь вузла-джерела, центральність	22–29%	Graph metrics
Тимчасовий	Час публікації, день тижня, сезонність	10–15%	Time-series analysis
Тематичний	Новизна, релевантність, трендовість	15–20%	Topic modeling (LDA)

Особливу роль відіграє емоційний фактор: контент, що викликає гнів або страх, поширюється на 70% швидше порівняно з нейтральним, тоді як контент, що викликає радість, демонструє вищі показники глибини каскаду. Також в цьому процесі задіяні і рекомендаційні системи платформ, які автоматично просувають контент із високим рівнем взаємодії. Це призводить до того, що популярні повідомлення отримують ще більшу видимість, формуючи ефект «інформаційного підсилення».

У моделюванні вірусного поширення інформації активно використовуються епідеміологічні моделі, зокрема моделі типу SIR та SI. У таких моделях процес передачі інформації описується аналогічно поширенню інфекційних захворювань: користувачі можуть бути «неінформованими», «активними поширювачами» або «пасивними спостерігачами».

Топологія мережі є визначальним чинником динаміки інформаційних

каскадів, тому дослідження у галузі мережевої науки переконливо доводять, що швидкість, масштаб та форма каскаду залежать не лише від змісту повідомлення, а передусім від архітектурних характеристик графа.

Зв'язки між користувачами класифікуються за теорією Грановеттера на сильні та слабкі. Згідно з гіпотезою «сили слабких зв'язків» (strength of weak ties), саме містки між слабко пов'язаними кластерами забезпечують масштабне поширення інформації [10]. Кількісно зв'язок між вузлами і та j характеризується вагою:

$$w_{ij} = \frac{|N(i) \cap N(j)|}{|N(i) \cup N(j)|} \quad (1.3)$$

де N(i) та N(j) – множини сусідів відповідних вузлів (індекс Жаккара).

Ключові метрики, що характеризують зв'язність мережі та впливають на динаміку каскадів, наведені у таблиці 1.4.

Таблиця 1.4 Мережеві метрики та їх роль у динаміці каскадів

Метрика	Формула	Діапазон	Вплив на каскад
Ступінь вузла (degree)	$k(v) = \|N(v)\|$	[0, n-1]	Визначає початкове охоплення
Коефіцієнт кластеризації	$C(v) = 2e / k(v)(k(v)-1)$	[0, 1]	Визначає «замкненість» поширення
Центральність за посередництвом	$BC(v) = \Sigma \sigma(s,t\|v)/\sigma(s,t)$	[0, 1]	Визначає роль «мосту»
Центральність за власним вектором	$EC(v) = (1/\lambda)\Sigma a(v,u) \cdot EC(u)$	[0, 1]	Відображає вплив через впливових сусідів
Середня довжина шляху	$L = (1/n(n-1))\Sigma d(i,j)$	≥1	Визначає швидкість досягнення аудиторії

Кластери у соціальних мережах формуються внаслідок гомофілії – тенденції до утворення зв'язків між схожими користувачами. З точки зору алгоритмічної реалізації, виявлення кластерів (community detection) є ключовою задачею аналізу мережі. Серед найпоширеніших виділяють алгоритми наведені у таблиці 1.5 [11]:

Таблиця 1.5 Алгоритми виявлення спільнот у соціальних мережах

Алгоритм	Часова складність	Якість (modularity Q)	Масштабованість	Особливості
Louvain	O(n log n)	0.3–0.5	Висока (>10^6 вузлів)	Жадібна оптимізація модульності
Girvan-Newman	O(n·m^2)	0.4–0.6	Низька (<10^4)	Видалення ребер з максимальним BC
Leiden	O(n log n)	0.4–0.6	Висока	Покращений Louvain, гарантує зв’язність
Label Propagation	O(n+m)	0.2–0.4	Дуже висока	Нестабільність результатів
Infomap	O(n log n)	0.3–0.5	Висока	Мінімізація опису потоків

Хаби – вузли з аномально високим ступенем зв’язності і є рушієм масштабних каскадів. У безмасштабних мережах (scale-free networks) розподіл ступенів вузлів підпорядковується степеневому закону: P(k) ~ k^(-γ) де, γ ∈ [2, 3] для більшості соціальних платформ [5].

Це означає, що невелика кількість «суперхабів» (лідери думок, верифіковані акаунти, боти з мільйонною аудиторією) контролює переважну частину інформаційних потоків. У таблиці 1.6 наведено класифікацію хабів і їхню роль у каскаді.

Таблиця 1.6 Класифікація вузлів-хабів за роллю в інформаційних каскадах

Тип хабу	Ступінь (degree)	Роль у каскаді	Приклади	Метод ідентифікації
Суперхаб (Super-spreader)	>10^4	Ініціація масштабних каскадів	Верифіковані акаунти ЗМІ	PageRank > 0.01
Локальний хаб	10^2–10^4	Підтримання каскаду в кластері	Блогери, лідери спільнот	Degree centrality
Міст (Bridge hub)	10^2–10^3	Міжкластерний трансфер	Акаунти з різноманітною аудиторією	Betweenness centrality
Ехо-камера	Будь-який	Рециркуляція в замкненому кластері	Тематичні боти	Clustering coefficient > 0.7

1.2. Поняття та структура інформаційного каскаду

Поняття «інформаційний каскад» було введено у науковий обіг Бікхчандані, Хіршлейфером та Велчем у 1992 році і спочатку описувало ірраціональну поведінку агентів, які ігнорують власну інформацію на користь спостереження за діями інших. У сучасній інтерпретації, адаптованій до реалій цифрових платформ, інформаційний каскад визначається як послідовний процес поширення одиниці інформації через мережу взаємодіючих агентів, за якого рішення кожного наступного агента про передачу інформації залежить від поведінки попередників.

На рисунку 1.3 наочно зображена робота інформаційних каскадів.

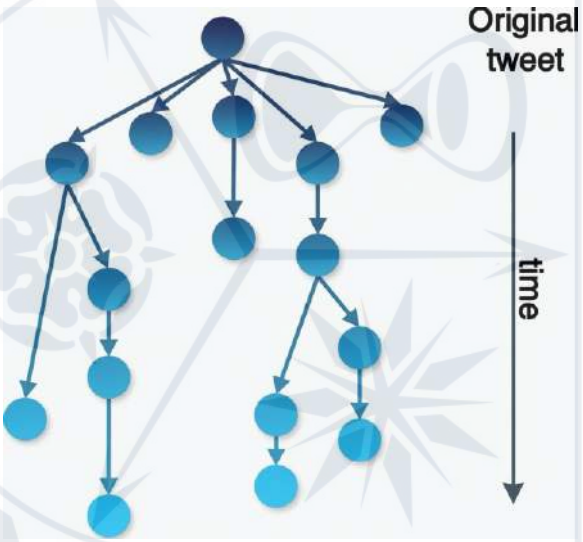

Рисунок 1.3 – Візуалізація роботи інформаційних каскадів

Документація бібліотеки NetworkX, яка є одним з найпоширеніших інструментів мережевого аналізу, формально описує каскад як орієнтований граф поширення, де кожне ребро відображає акт передачі інформації між двома вузлами у конкретний момент часу [12].

Принципово важливим є розмежування між інформаційним каскадом та суміжними поняттями. На відміну від «тренду», каскад має чітко простежувану деревоподібну структуру передачі з ідентифікованими ланцюжками ретрансляції, тоді як тренд відображає лише агреговану популярність без

урахування мережевих шляхів.

В онлайн-середовищі та реальному житті інформаційні каскади мають різну природу та наслідки:

- Вірусні меми та флешмоби: користувачі беруть участь у челенджах або поширюють певний медіафайл просто тому, що він уже став масовим у їхній стрічці новин.
- Каскади паніки та чуток: швидке поширення неперевіреної інформації про фінансову нестабільність банків, дефіцит товарів чи загрози безпеці, що змушує людей масово копіювати поведінку натовпу.
- Алгоритмічні каскади оцінювання: ситуації на платформах типу Reddit або Amazon, де перші 5–10 позитивних чи негативних оцінок товару/посту визначають його подальшу долю, запускаючи каскад однотипних оцінок від наступних користувачів, які підсвідомо підлаштовуються під створений рейтинг.

Класифікація інформаційних каскадів є багатовимірною задачею: залежно від обраного критерію один і той самий каскад може належати до різних типів одночасно. У сучасній літературі та документації інструментів мережевого аналізу виділяють кілька незалежних типів для класифікації – топологічна структура, механізм поширення, швидкість, достовірність контенту та джерело ініціації. У Додатку А коротко наведено інформацію по кожному з типів.

Класифікація за топологічною структурою – це найбільш фундаментальна класифікація, що описує геометрію дерева поширення. Документація NetworkX виділяє такі базові топологічні форми графів поширення як лінійний, деревоподібний, зіркоподібний, сітчастий і гібридний каскади [12].

Лінійний характеризується послідовною передачею інформації по ланцюжку, де кожен вузол має не більше одного попередника та одного наступника. Документація бібліотеки graph-tool описує такі структури як вироджені дерева (degenerate trees) з коефіцієнтом розгалуження $b = 1$:

$$\forall v \in V_{\text{cascade}}\colon \deg_{\text{in}}(v) \leq 1 \wedge \deg_{\text{out}}(v) \leq 1 \qquad (1.3)$$

де $\forall$ – квантор загальності;

$v \in V_{\text{cascade}}$ – вершину v, яка бере участь у каскаді;

$\deg_{\text{in}}(v)$ – вхідний ступінь вузла;

$\deg_{\text{out}}(v)$ – вихідний ступінь вузла.

Лінійні каскади типові для закритих месенджерів (WhatsApp, Telegram у приватних чатах) та пересилання електронної пошти. Ефективний коефіцієнт передачі для лінійного каскаду є постійним:

$$\beta_{\text{eff}} = \frac{\Delta N}{\Delta t} \approx \text{const} \tag{1.4}$$

де ΔN – приріст кількості активованих вузлів;

Δt – проміжок часу.

Деревоподібний каскад є найбільш поширеним типом вірусного поширення у відкритих соціальних мережах. Він характеризується рекурсивним розгалуженням, де кожен активований вузол може ініціювати власний підкаскад. Такі структури можна описати через процес Гальтона-Ватсона з середнім числом нащадків μ[13]. Ступінь розгалуження (branching factor) b визначає динаміку каскаду: при $b > 1$ каскад зростає експоненційно, при $b < 1$ – загасає [7].

Зіркоподібний каскад виникає, коли інформація поширюється переважно від одного або кількох суперхабів безпосередньо до великої кількості реципієнтів без подальшого перепоширення. Характеризується максимальним охопленням при мінімальній глибині (1–2 рівні). Типові платформи: Instagram, YouTube, офіційні акаунти медіа.

$$\exists s \in V\colon \deg_{\text{out}}(s) \gg \deg_{\text{out}}(v),\ \forall v \neq s \tag{1.5}$$

де $\exists$ – квантор існування;

s – спеціальний вузол;

V – множина всіх вузлів (вершин) графа;

$\deg_{\text{in}}(v)$ – вхідний ступінь вузла;

$\deg_{\text{out}}(v)$ – вихідний ступінь вузла.

Сітчастий каскад виникає, коли інформація поширюється через множину паралельних і взаємопов’язаних шляхів одночасно, утворюючи не дерево, а

повноцінний орієнтований граф із циклами. Graph-tool описує такі структури як strongly connected components у графі поширення [14]:

Сітчасті каскади типові для ситуацій, коли кілька незалежних джерел одночасно поширюють один і той самий контент – наприклад, під час великих новинних подій, коли десятки медіа-організацій публікують матеріал паралельно. Виявляються такі структури через алгоритм Косараджу для пошуку сильно зв'язних компонент.

Гібридний каскад поєднує ознаки двох або більше топологічних типів. Репозиторій CasCN на GitHub пропонує кількісний індекс гібридності $H \in [0,1]$[15]:

$$H = 1 - \frac{\deg_{\max}(v)}{\sum_v \deg_{\text{out}}(v)} \tag{1.6}$$

де $\deg_{\max}(v)$ – максимальний вихідний ступінь у всій мережі;

$\sum_v \deg_{\text{out}}(v)$ – сума всіх вихідних ступенів усіх вузлів мережі.

$H = 0$ відповідає чистому зіркоподібному типу, $H = 1$ – чистому деревоподібному. Більшість реальних масштабних каскадів є гібридними: розпочинаються як зіркоподібні (суперхаб), а потім набувають деревоподібної структури.

Класифікація за механізмом поширення описує не форму, а причину, з якої вузол вирішує передати інформацію. Розрізняють такі типи: каскад соціального впливу, каскад незалежних рішень, каскад наслідування і алгоритмічний каскад.

Каскад соціального впливу (Social Influence Cascade). Вузол активується під тиском соціального оточення – коли достатня кількість його сусідів вже передала інформацію. Описується моделлю лінійного порогу (Linear Threshold Model):

$$\sum_{u \in N(v),\, \Phi(u)=1} w_{uv} \geq \theta_v \tag{1.7}$$

де θ_v – індивідуальний поріг активації,

w_{uv} – вага зв'язку.

Типовий приклад – це поширення петицій або участь у флешмобах.

Каскад незалежних рішень – це коли кожен активований вузол має рівно

одну спробу активувати кожного сусіда з фіксованою ймовірністю p_{uv}, незалежно від стану інших вузлів. Цей механізм добре описує поширення новин, де рішення поділитися є переважно індивідуальним.

Каскад наслідування (Imitation Cascade) – вузол передає інформацію, бо спостерігає, що це роблять авторитетні для нього особи, незважаючи на власну оцінку контенту.

Алгоритмічний каскад (Algorithmic Cascade) – відносно новий тип, виділений після широкого впровадження рекомендаційних систем. Алгоритм ранжування сам ініціює та підтримує каскад, просуваючи контент у стрічках користувачів незалежно від їхніх прямих соціальних зв'язків [15]. У цьому разі «активація» вузла визначається не мережевою структурою, а рішенням алгоритму платформи.

Інформаційний каскад розвивається у часі та проходить кілька характерних етапів, кожен із яких має свої особливості.

Етапи розвитку каскадів:

Зародження (Emergence Phase) – фаза зародження охоплює часовий інтервал від публікації контенту до моменту, коли швидкість поширення стає стабільно позитивною. Описати цю фазу можна через умову:

$$\frac{d^2N(t)}{dt^2} > 0 \text{ та } \frac{dN(t)}{dt} > \varepsilon \tag{1.8}$$

де $N(t)$ – кількість активованих вузлів;

ε – мінімальний поріг швидкості.

Документація платформи Twitter Developer Portal зазначає, що алгоритм ранжування стрічки враховує швидкість отримання перших реакцій як основний сигнал для подальшого просування публікації – саме це робить фазу зародження критичною [16]. За даними датасетів SNAP, понад 90% потенційно вірусного контенту «гасне» у цій фазі.

Активне поширення (Active Diffusion Phase) – фаза активного поширення починається, коли $R > 1$, і описується логістичним рівнянням:

$$\frac{dN}{dt} = r \cdot N(t) \cdot \left(1 - \frac{N(t)}{K}\right) \tag{1.9}$$

де r– внутрішній коефіцієнт росту;

K– максимальна ємність мережі для даного контенту.

Ця фаза формує «хвилі» активності, інтервал між якими залежить від середнього часу відповіді на конкретній платформі [17]. Дані по платформах систематизовано у Таблиці 1.7.

Таблиця 1.7 Характеристики фази активного поширення на різних платформах

Платформа	Час між хвилями	Пікова швидкість	Тривалість фази	Коефіцієнт b
Twitter/X	12–25 хв	10^3–10^5 реп./хв	2–6 год	2,3–4,7
Facebook	45–90 хв	10^2–10^4 реп./хв	6–24 год	1,8–3,2
TikTok	5–15 хв	10^3–10^6 реп./хв	1–48 год	3,5–8,1
Telegram	3–10 хв	10^2–10^3 реп./хв	1–4 год	1,5–2,8
Reddit	20–60 хв	10^2–10^4 реп./хв	4–12 год	2,1–5,3

1.3. Графове представлення інформаційних каскадів

Теорія графів є математичним фундаментом сучасного аналізу соціальних мереж. Граф є абстракцією, що дозволяє формалізувати будь-яку систему попарних відношень між об’єктами. Можна визначити граф як структуру даних, що складається з вузлів (nodes) та ребер (edges), і надає єдиний інтерфейс для роботи з усіма різновидами графових об’єктів. Саме ця формалізація робить теорію графів універсальним інструментом для аналізу інформаційних каскадів. Розберемо кожен з термінів детальніше.

Вершини (nodes, vertices) є базовими елементами графової моделі соціальної мережі, і в контексті аналізу інформаційних каскадів кожна вершина $v \in V$ відповідає окремому агенту – користувачу платформи, акаунту, боту або організації. Вершині можна приписати довільний набір атрибутів.

У задачах аналізу каскадів вершини додатково мають стан активації $\Phi(v,t) \in \{0,1\}$, що змінюється в часі. У Таблиці 1.8 показано систематиматизацію типів вершин за їхньою роллю у каскаді і методом який був

використаний для індетифікації.

Таблиця 1.8 Класифікація вершин графа соціальної мережі за роллю у каскаді

Тип вершини	Позначення	Ступінь k	Стан Ф	Роль у каскаді	Метод ідентифікації
Seed (джерело)	s_0	Будь-який	1 (першим)	Ініціація каскаду	Часова мітка t_0
Суперхаб	v_h	$k > 10^4$	1	Масштабування охоплення	Degree centrality
Міст	v_b	10^2–10^3	1	Міжкластерний трансфер	Betweennes s centrality
Рядовий ретранслятор	v_r	10^1–10^2	1	Підтримання глибини	Degree + активність
Кінцевий реципієнт	v_e	Будь-який	1(останній)	Завершення гілки	$\deg_{out} = 0$
Неактивований	v_0	Будь-який	0	Відсутня	Стан $\Phi = 0$

Ребра (edges) відображають відносини між вершинами. У соціальних мережах ребро $e = (u, v) \in E$ може означати підписку, дружбу, взаємодію або безпосередній акт передачі інформації. Документація graph-tool розрізняє прості ребра, мультиребра (кілька зв'язків між тими самими вузлами) та зважені ребра з атрибутом $w_{uv} \in \mathbb{R}^+$[18].

У контексті аналізу каскадів виділяють два принципово різних типи ребер:

1. Структурні ребра – відображають постійні соціальні зв'язки (підписки, дружба). Утворюють граф соціальної мережі $G_S = (V, E_S)$.

2. Каскадні ребра – відображають конкретні акти передачі інформації. Утворюють граф поширення $G_C = (V_C, E_C)$, де $V_C \subseteq V$, $E_C \subseteq E_S$.

У таблиці 1.9 представлені різні типи ребер у графах, та їх приклади у

соціальних мережах.

Таблиця 1.9 Типи ребер у графових моделях соціальних мереж

Тип ребра	Позначення	Атрибути	Семантика	Приклад
Соціальний зв'язок	$(u, v) \in E_S$	Вага, тип	Постійна підписка	Фоловінг у Twitter
Каскадне ребро	$(u, v, t) \in E_C$	Час, вага	Конкретний репост	Ретвіт з часовою міткою
Потенційне ребро	$(u, v) \in E_S \setminus E_C$	–	Зв'язок без передачі	Підписник, що не поширив
Мультиребро	$(u, v)^k$	k копій	Повторні взаємодії	Кілька репостів одного контенту

Для аналізу процесів поширення інформації у соціальних мережах використовуються різні типи графових структур. Вони дозволяють візуалізувати взаємозв'язки між користувачами, напрямки передачі інформації та особливості формування інформаційних каскадів. Залежно від способу представлення зв'язків і рівня деталізації, застосовуються орієнтовані графи, каскадні графи та дерева поширення.

Орієнтований граф (directed graph, digraph) є природною моделлю асиметричних відносин у соціальних мережах. Вікіпедія-сторінка «Directed graph» на математичному розділі описує digraph як пару $G = (V, A)$, де $A \subseteq V \times V$ – множина упорядкованих пар (дуг), що відрізняються від ребер наявністю напрямку [19]. Для аналізу каскадів орієнтованість є критичною: інформація передається від ретранслятора до його підписників, але не навпаки.

Для орієнтованого графа вводяться окремі поняття вхідного та вихідного ступеня:

$$\deg_{\text{in}}(v) = |\{u: (u, v) \in A\}| \tag{1.10}$$

$$\deg_{\text{out}}(v) = |\{u: (v, u) \in A\}| \tag{1.11}$$

де u – вершина, яку ми вивчаємо;

A – множина всіх дуг в графі;

(u, v)– дуга, яка починається в точці u і закінчується в точці v.

У каскадному графі $\deg_{in}(v) = 1$ для переважної більшості вузлів, тоді як $\deg_{out}(v)$ варіюється від 0 (кінцеві реципієнти) до $10^4 +$ (суперхаби).

Для аналізу окремих фрагментів соціальної мережі використовуються підграфи. Підграф каскаду (cascade subgraph) – це обмеженням повного графа соціальної мережі до множини вузлів та ребер, що беруть участь у конкретному каскаді. Вони дозволяють досліджувати локальні взаємодії користувачів, виділяти окремі спільноти та аналізувати особливості поширення інформації всередині них.

Принципово важливо розрізняти два різновиди підграфів каскаду:

1. Індукований підграф (induced subgraph) – містить усі ребра між активованими вузлами, що існують у вихідній мережі, навіть якщо вони не були використані для передачі інформації.

2. Каскадне дерево (cascade spanning tree) – містить лише ребра, що фактично використовувались для передачі інформації, утворюючи остове дерево індукованого підграфа.

На рисунку 1.4 візуалізовано різницю між графом, підграфом та його різновидами.

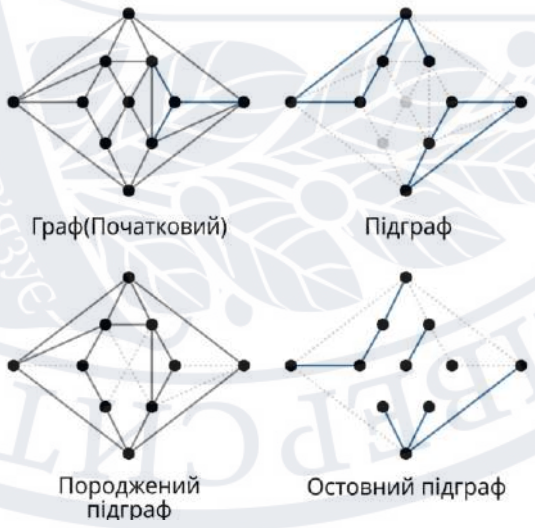

Рисунок 1.4 – Візуальне порівняння графу, підграфу і його різновидів

Ступінь вузла (node degree) є найбільш фундаментальною характеристикою, що визначає кількість ребер, інцидентних даному вузлу. Для орієнтованого каскадного графа розрізняють вхідний та вихідний ступінь, а також інші, що показано в таблиці 1.10.

Таблиця 1.10 Метрики на основі ступеня вузла та їх роль в аналізі каскадів

Метрика	Формула	Діапазон	Роль у каскаді
Вхідний ступінь	$\deg_{in}(v)$	$[0, n-1]$	Популярність, кількість джерел впливу
Вихідний ступінь	$\deg_{out}(v)$	$[0, n-1]$	Потенціал поширення
Нормований ступінь	$k_{norm} = k/(n-1)$	$[0, 1]$	Порівняння між мережами різного розміру
Середній ступінь	$\langle k \rangle = E/n$	$[0, n-1]$	Загальна щільність зв'язків
Максимальний ступінь	$k_{max} = \max_v k(v)$	$[0, n-1]$	Наявність суперхабів
Degree centrality	$DC(v) = k(v)/(n-1)$	$[0, 1]$	Локальна впливовість

Для аналізу каскадів кластеризація відіграє подвійну роль: висока кластеризація сприяє швидкому поширенню всередині спільноти (ефект «луна-камери»), але гальмує міжкластерний трансфер, оскільки вузли з високим $C(v)$ мають переважно «надлишкові» зв'язки в межах одної групи. У мережах з $\bar{C} > 0.5$ каскади мають тенденцію «застрявати» у початковому кластері та не досягають глобального масштабу без участі вузлів-мостів ($C(v) \approx 0$) [21]. У таблиці 1.11 представлена роль у каскаді, відповідно до інтерпритації.

Коефіцієнт кластеризації (clustering coefficient) вимірює схильність сусідів вузла бути також з'єднаними між собою, тобто ступінь «трикутності» локального оточення. Локальний коефіцієнт кластеризації вузла v [22]:

$$C(v) = \frac{|\{(u,w) \in E : u,w \in N(v)\}|}{k(v)(k(v)-1)/2} \quad (1.13)$$

де $N(v)$ – множина сусідів вузла v;

$k(v)(k(v)-1)/2$ – максимально можлива кількість ребер між сусідами.

Таблиця 1.11 Інтерпретація коефіцієнта кластеризації у контексті каскадів

Значення $C(v)$	Структура оточення	Роль у каскаді	Тип вузла
$C(v) \approx 0$	Зірка або міст	Міжкластерний трансфер	Bridge node
$0 < C(v) < 0.3$	Розріджений граф	Збалансоване поширення	Regular spreader
$0.3 \leq C(v) < 0.7$	Помірна кластеризація	Внутрішньокластерне поширення	Community member
$C(v) \approx 1$	Повний підграф (кліка)	Ехо-камера, рециркуляція	Echo chamber node

Середня довжина шляху (average path length, APL) характеризує «компактність» мережі – наскільки швидко інформація може теоретично досягти будь-якого вузла від будь-якого іншого.

Феномен «шести ступенів розділення» (six degrees of separation), підтверджений емпірично для Facebook ($L \approx 4.57$ при мільярдах вузлів), пояснює, чому інформаційні каскади здатні охопити глобальну аудиторію за лічені години [23].

Для каскадного дерева середня довжина шляху від кореня є важливішою характеристикою, ніж глобальна APL:

$$L_{s_0} = \frac{1}{|V_C|} \sum_{v \in V_C} d(s_0, v) \quad (1.14)$$

де s_0 – початковий вузол (або множина вузлів), з якого розпочався каскад;

V_C – всі вузли, які стали частиною каскаду;

$d(s_0, v)$ – це довжина найкоротшого шляху в графі каскаду від джерела s_0 до конкретного вузла v.

Ця метрика відображає «ефективність» каскаду: низьке L_{s_0} означає, що більшість активованих вузлів отримали інформацію безпосередньо від джерела (зіркоподібний каскад), а високе – що інформація пройшла через численні проміжні ретрансляції (деревоподібний каскад).

Окремим інструментом структурного аналізу каскадного графа є розфарбування графу (graph coloring). Основна ідея полягає в присвоєнні кожній вершині певного кольору таким чином, щоб суміжні вершини мали різні кольори [24]. У контексті аналізу каскадів хроматичне число $\chi(G)$ є непрямим індикатором щільності конфліктних зв'язків у мережі поширення: чим вище $\chi(G)$, тим більш «заплутаною» є структура каскаду та тим складніше інформації долати локальні кластери без повторного охоплення одних і тих самих вузлів

1.4. Аналіз сучасних підходів до дослідження каскадів

Сучасні підходи до вивчення каскадів спрямовані на аналіз механізмів поширення інформації, прогнозування вірусності контенту та виявлення маніпулятивних інформаційних кампаній. Через складність структури соціальних мереж і велику кількість взаємодій між користувачами для аналізу каскадів застосовуються методи теорії графів, математичного моделювання, машинного навчання та аналізу великих даних. Графова аналітика є основою обчислювального аналізу каскадів. Документація Apache Spark GraphX описує набір базових примітивів для роботи з великими графами: агрегацію по вершинах, поширення повідомлень (Pregel API) та ітеративні алгоритми [25]. На основі цих примітивів будуються алгоритми аналізу структури каскадів. Основні класи алгоритмів графової аналітики, систематизовані у таблиці 1.13.

Окремим важливим інструментом є GraphSAGE – алгоритм генерації ембедингів вузлів графа, що враховує агреговану інформацію про їхнє оточення. Документація бібліотеки PyTorch Geometric описує його як основу для більшості сучасних GNN-підходів до аналізу каскадів [26].

Прогнозування каскадів є одним з найактивніших напрямів прикладного машинного навчання у мережевому аналізі.

Таблиця 1.13 Алгоритми графової аналітики для аналізу каскадів

Клас алгоритмів	Алгоритм	Складність	Реалізація	Застосування
Центральність	PageRank	$O(k \cdot (n + m))$	NetworkX, Spark GraphX	Виявлення впливових вузлів
	Betweenness centrality	$O(nm)$	NetworkX, graph-tool	Виявлення вузлів-мостів
	HITS (Hubs & Authorities)	$O(k \cdot m)$	NetworkX	Авторитетність джерел
Спільноти	Louvain	$O(n\log\ n)$	python-louvain, iGraph	Виявлення кластерів
	Leiden	$O(n\log\ n)$	leidenalg	Поліпшений Louvain
Шляхи	BFS/DFS	$O(n + m)$	NetworkX	Реконструкція дерева каскаду
	Dijkstra	$O((n + m)\log\ n)$	NetworkX, SciPy	Найкоротші шляхи поширення
Розміщення	ForceAtlas2	$O(n^2)$	Gephi, fa2	Візуалізація структури каскаду
Декомпозиція	k-core decomposition	$O(n + m)$	NetworkX	Виявлення ядра мережі
Динаміка	Temporal BFS	$O(n + m)$	custom	Хронологічна реконструкція

Еволюцію підходів до прогнозування каскадів можна простежити через чотири покоління методів (таблиця 1.14).

Таблиця 1.14 Покоління методів прогнозування інформаційних каскадів

Покоління	Період	Підхід	Ключові моделі	Точність (Accuracy/MSLE)	Обмеження
I	до 2015	Статистичні	SIR, SEIR, Hawkes	MSLE ≈ 3.2	Не враховують структуру
II	2015–2018	Feature-based ML	Gradient Boosting, SVM на мережевих ознаках	MSLE ≈ 2.1	Ручний feature engineering
III	2018–2021	Deep Learning	DeepCas (GRU+attention), LSTM на послідовностях	MSLE ≈ 1.4	Не враховують граф повністю
IV	2021–2024	Graph Neural Networks	CasSeqGCN, CASC, Hi-CVAE	MSLE ≈ 0.8	Висока обчислювальна вартість

Моделі четвертого покоління на основі графових нейронних мереж (GNN) є сучасним стандартом. Документація PyTorch Geometric надає реалізації GCN, GAT та GraphSAGE, що використовуються як базові блоки для побудови каскадних предикторів [26]. Ключова перевага GNN – здатність одночасно враховувати структуру графа, атрибути вузлів та часову динаміку.

Під час аналізу інформаційних каскадів у соціальних мережах виникає низка технічних і методологічних проблем, пов’язаних зі складністю сучасного цифрового середовища, і основні з них це: маштабованість, складність обробки великих графів, проблеми пам’яті, динамічності і гетерогенності, неповнота

даних, приховані зв’язки та обмеження доступу до API.

Масштабованість є першочерговим викликом для всіх алгоритмів аналізу каскадів. Більшість класичних алгоритмів мережевого аналізу (betweenness centrality, all-pairs shortest path) мають поліноміальну складність, що робить їх непридатними для реальних соціальних мереж масштабу Twitter ($n \approx 5 \times 10^8$). Документація бібліотеки GraphSAINT пропонує підхід до вирішення проблеми масштабованості GNN через вибіркове навчання на підграфах – замість обробки всього графа одночасно модель навчається на випадково вибраних підграфах, що зменшує вимоги до пам’яті з $O(n)$до $O(b)$, де b– розмір батчу [27].

Складність обробки великих графів. Навіть при наявності достатніх обчислювальних ресурсів, обробка великих графів каскадів пов’язана з рядом специфічних труднощів, що виходять за межі простої масштабованості.

Проблема пам’яті. Повне зберігання матриці суміжності графа соціальної мережі масштабу Twitter вимагає $O(n^2)$ пам’яті, що при $n = 5 \times 10^8$ є фізично неможливим. Одне із запропонованих рішень – розріджені формати (CSR, COO) як обов’язкова умова роботи з реальними мережами.

Проблема динамічності. Реальні каскади розгортаються у часі, що перетворює статичний граф на темпоральний граф (temporal graph).

Проблема гетерогенності. Реальні соціальні мережі є гетерогенними: вузли та ребра мають різні типи (користувачі, медіа, боти; підписка, репост, лайк). Стандартні алгоритми, що передбачають гомогенний граф, втрачають частину інформації.

Неповнота даних є, мабуть, найбільш фундаментальним обмеженням сучасних досліджень каскадів, оскільки вона стосується не обчислювальних ресурсів, а самої природи доступних даних.

Проблема прихованих зв’язків. Значна частина поширення інформації відбувається через канали, що не реєструються платформами: особисті повідомлення, копіювання тексту без ретвіту, скріншоти, офлайн-спілкування. Спостережуване дерево каскаду може відображати лише 20–40% реального поширення.

Проблема обмеженого доступу до API. Після обмежень API Twitter/X у 2023 році, що збільшили вартість доступу до даних у 100 разів, більшість академічних досліджень вимушені працювати з неповними вибірками. Блог розробників Twitter Developer Platform підтверджує, що безкоштовний рівень API обмежений 500,000 твітів на місяць, тоді як реальний обсяг платформи становить понад 500 мільйонів на добу [28].

Проблема вибіркового спостереження. Навіть при повному доступі до API дослідник спостерігає лише вузли та ребра, що потрапили у вибірку. Математично це означає роботу з спостережуваним підграфом $\hat{G} \subseteq G$, де $\hat{G}$може суттєво відрізнятися від реального графа G за ключовими структурними характеристиками.

Підсумовуючи, сучасні підходи до аналізу інформаційних каскадів досягли значного прогресу у точності прогнозування та масштабованості обчислень, проте залишаються суттєві обмеження, що зумовлюють актуальність розробки нових програмних рішень.

Ручний аналіз інформаційних каскадів є практично неможливим через обсяги даних. За даними Twitter Developer Platform, лише одна велика новинна подія генерує від 100,000 до 10,000,000 ретвітів протягом перших 24 годин [28]. Ручне відстеження навіть тисячної частки цього потоку перевищує можливості будь-якої дослідницької групи. Необхідність автоматизації зумовлена кількома взаємопов’язаними причинами.

По-перше, обсяг даних. Обробка таких обсягів потребує не лише алгоритмічної автоматизації, а й відповідної програмної інфраструктури.

По-друге, відтворюваність. Ручний аналіз є суб’єктивним і важко відтворюваним. Підкреслюють, що автоматизовані пайплайни аналізу даних є необхідною умовою відтворюваності наукових результатів.

Після окреслення і аналізу проблематики було вирішено реалізувати програмний комплекс як багатошарову систему, кожен рівень якої використовує спеціалізований набір технологій, обраних відповідно до функціональних вимог конкретного шару. Таблиця 1.15 надає зведений огляд технологічного стеку.

Таблиця 1.15 Зведений стек технологій програмного комплексу

Шар	Технології	Призначення
Збір даних	Playwright, BeautifulSoup	Отримання публікацій та взаємодій із Twitter/X
Графовий аналіз	NetworkX, CascadeGraph	Побудова орієнтованого графа каскаду та розрахунок метрик
Моделювання поширення	Independent Cascade Model, SIR-подання	Моделювання розвитку інформаційного каскаду
Backend	FastAPI, Pydantic	Реалізація API та координація етапів аналізу
Зберігання даних	SQLite, L1 In-Memory Cache	Кешування та збереження результатів аналізу
Frontend	React, Vite, D3.js	Візуалізація каскадів та результатів аналізу

Playwright використовується для автоматизації браузера та отримання динамічного контенту Twitter/X. Це важливо, оскільки значна частина сторінок соціальних мереж формується після виконання JavaScript-коду. У системі Playwright відповідає за відкриття сторінки, роботу з авторизованою сесією, прокручування вмісту та отримання HTML-структури сторінки.

BeautifulSoup використовується на наступному етапі для розбору отриманого HTML та вилучення потрібних елементів: тексту публікації, ідентифікаторів користувачів, часових міток, показників взаємодії та службових метаданих. Такий розподіл відповідальності дозволяє відокремити рендеринг сторінки від безпосереднього парсингу даних.

Для зменшення ризику блокування під час збору даних у системі передбачено використання випадкових затримок між діями, збережених авторизаційних сесій, обмеження завантаження другорядних ресурсів та налаштування параметрів браузерного середовища.

Центральним обчислювальним компонентом є графовий модуль. Для роботи з графами використовується бібліотека NetworkX. У системі застосовується орієнтований граф, де вузли відповідають акаунтам користувачів, а ребра – взаємодіям між ними: відповідям, цитуванням або поширенням повідомлення. На основі такого графа обчислюються глибина каскаду, коефіцієнт розгалуження, центральність вузлів, часові затримки та інші характеристики поширення. Для моделювання розвитку інформаційного каскаду використовується модель незалежного каскаду, доповнена часовим згасанням і SIR-поданням станів вузлів. Це дозволяє не лише описати вже побудований граф, а й оцінити можливу подальшу динаміку поширення повідомлення. Backend-шар реалізовано за допомогою FastAPI. Він відповідає за прийом запитів, запуск pipeline аналізу, взаємодію між scraper-модулем, CascadeGraph та модулями моделювання. Pydantic використовується для опису структур даних і перевірки коректності вхідних та вихідних об’єктів.

Для локальної розробки, тестування та кешування результатів використовується SQLite. У межах прототипу SQLite зберігає історію аналізів, результати побудови каскадів і метрики. Окремо використовується in-memory cache для швидкого доступу до нещодавно отриманих результатів. У разі промислового розгортання систему можна адаптувати до використання PostgreSQL або MySQL як основної СУБД для production-середовища – які можуть працювати з паралельними запитами на відміну від SQLite.

Frontend-частина реалізована за допомогою React та Vite. Вона забезпечує відображення графа каскаду, структурних метрик, часових характеристик та результатів моделювання. Для графової візуалізації використовується D3.js, що дозволяє будувати інтерактивне представлення вузлів, ребер і ключових елементів поширення інформації.

Docker використовується для контейнеризації компонентів системи. Backend і frontend запускаються в окремих контейнерах із власними залежностями. Docker Compose координує їх спільний запуск, налаштовує порти, змінні середовища та взаємодію між сервісами.

РОЗДІЛ 2
РОЗРОБКА АЛГОРИТМІВ АНАЛІЗУ ДИНАМІКИ ІНФОРМАЦІЙНИХ КАСКАДІВ

2.1. Формалізація моделі інформаційного каскаду

Для аналізу динаміки поширення інформації у соціальних мережах доцільно використовувати графове подання структури взаємодій між користувачами. Такий підхід дозволяє формалізувати розповсюдження повідомлень та виконувати подальший аналіз інформаційних каскадів із використанням методів теорії графів [35]. У такому випадку соціальна мережа подається у вигляді орієнтованого графа, де вершини відповідають окремим акаунтам, а ребра описують інформаційні взаємодії між ними.

У загальному вигляді соціальна мережа визначається як граф:

$$G = (V, E) \tag{2.1}$$

де V – множина вершин графа, що є користувачам соціальної мережі;

E – множина ребер, які описують взаємодії між користувачами.

Оскільки поширення інформації має напрямок передачі повідомлення, від одного користувача до іншого, в межах моделі використовується саме орієнтований граф [36]. Ребро між вершинами формується у випадку інформаційної взаємодії: репосту, відповіді, цитування або іншого способу поширення контенту. Напрямок ребра визначає напрямок передачі інформації між акаунтами.

У системі аналізу вершина графа містить не лише ідентифікатор користувача, а й набір додаткових характеристик, які можуть використовуватись під час подальшого аналізу. До таких параметрів можуть належати:

- кількість підписників;
- активність акаунта;
- часові характеристики взаємодій;
- оцінка ботоподібної активності;
- службові параметри профілю.

Таким чином вершину графа можна подати як:

$$v_i = (id_i, a_i, t_i, b_i) \quad (2.2)$$

де id_i – ідентифікатор користувача;

a_i – параметри активності акаунта;

t_i – часові характеристики;

b_i – оцінка поведінкових або ботоподібних ознак.

Ребра графа також містять додаткові параметри, що характеризують особливості взаємодії між користувачами. Враховується тип взаємодії, часовий інтервал між подіями та інтенсивність поширення інформації.

Формально ребро можна подати так:

$$e_{ij} = (v_i, v_j, \tau_{ij}, r_{ij}) \quad (2.3)$$

де v_i, v_j – вершини графа;

τ_{ij} – часовий інтервал взаємодії;

r_{ij} – тип інформаційної взаємодії.

Така структура дозволяє аналізувати не лише те що користувачі взаємодіяли один між одним, а й часову динаміку поширення інформації [37]. Це важливо для дослідження інформаційних каскадів, оскільки швидкість поширення, структура взаємодій та активність окремих вузлів можуть суттєво впливати на загальний характер розвитку каскаду.

Граф взаємодій формується на основі даних, отриманих скрапер-модулем, після отримання інформації про повідомлення та взаємодії між користувачами виконується побудова орієнтованого графа взаємодій, при цьому окремо враховуються реальні взаємодії між акаунтами та додаткові синтетичні вузли, які можуть використовуватись для моделювання розширення каскаду за умов неповноти даних.

Графове подання соціальної мережі створює основу для подальшого формування інформаційного каскаду, аналізу структури поширення повідомлення та обчислення метрик динаміки інформаційних потоків. Використання орієнтованого графа також дозволяє застосовувати алгоритми

аналізу центральності вузлів, оцінювання глибини каскаду та дослідження структурних характеристик інформаційного поширення.

Після формалізації соціальної мережі у вигляді графа необхідно визначити структуру інформаційного каскаду та особливості його формування. У загальному випадку інформаційний каскад можна розглядати як процес послідовного поширення повідомлення між користувачами соціальної мережі через інформаційні взаємодії [38]. Формування каскаду починається з початкового повідомлення та надалі розширюється внаслідок репостів, цитувань, відповідей або інших типів взаємодій між акаунтами.

У межах моделі інформаційний каскад визначається як підграф соціальної мережі:

$$C = (V_c, E_c) \tag{2.4}$$

де $V_c \subseteq V$ – множина вузлів, що беруть участь у поширенні повідомлення;

$E_c \subseteq E$ – множина ребер взаємодії між цими вузлами. Таким чином каскад є окремою частиною загального графа соціальної мережі, що формується навколо конкретного інформаційного повідомлення детальніше на рисунку 2.1.

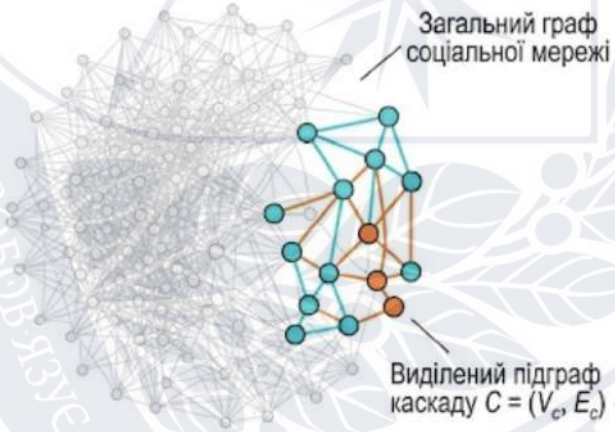

Рисунок 2.1 – Виділення підграфа інформаційного каскаду

Початковий вузол каскаду $v_0 \in V_c$ відповідає джерелу поширення інформації та визначає початок формування каскаду. Саме від цього вузла починається подальше розповсюдження повідомлення між користувачами.

Кожне нове ребро у структурі каскаду відповідає окремій інформаційній взаємодії між акаунтами.

Для опису часової динаміки кожному вузлу v_i ставиться у відповідність момент взаємодії t_i, що характеризує час появи повідомлення або реакції користувача. Використання часових параметрів дозволяє аналізувати швидкість розвитку каскаду, часові затримки між взаємодіями та зміну інтенсивності поширення інформації у часі [39].

У межах моделі ребра каскаду формуються лише між користувачами, між якими зафіксовано передачу або поширення інформації. Формально ребро каскаду можна подати так:

$$e_{ij} = (v_i, v_j) \tag{2.5}$$

де v_i – користувач-джерело інформації;

v_j – користувач, який взаємодіє з повідомленням.

Подібне подання дозволяє формувати орієнтовану структуру поширення інформації та виконувати подальший аналіз характеристик каскаду. У системі окремо враховуються часові залежності між взаємодіями, що дає змогу визначати інтенсивність розвитку каскаду та виявляти нетипові сценарії поширення інформації.

Особливістю реальних соціальних мереж є неповнота доступних даних, оскільки частина взаємодій може бути недоступною через обмеження платформи або налаштування приватності користувачів. Тому у межах системи каскад розглядається як наближене представлення реального процесу поширення інформації. Для зменшення впливу неповноти даних можуть використовуватись додаткові синтетичні вузли або моделі розширення структури каскаду.

Таким чином формалізація інформаційного каскаду створює основу для подальшого аналізу структури поширення інформації, обчислення метрик каскаду та дослідження часової динаміки розвитку інформаційних потоків.

Для аналізу поширення конкретного повідомлення із загального графа

соціальної мережі виділяється окремий підграф каскаду, який містить лише вузли та взаємодії, пов’язані з розповсюдженням відповідного інформаційного повідомлення [40].

Підграф каскаду подається як:

$$C_s = (V_s, E_s) \tag{2.6}$$

де $V\_s \subseteq V\_c$ – множина вузлів підграфа каскаду;

$E\_s \subseteq E\_c$ – множина ребер поширення інформації.

Формування підграфа виконується шляхом послідовного додавання вузлів та ребер у процесі виникнення взаємодій із повідомленням. Якщо користувач виконує репост, цитування або відповідь, відповідний вузол та інформаційний зв’язок додаються до структури каскаду.

Під час побудови підграфа враховується часовий порядок взаємодій:

$$t_i < t_j \tag{2.7}$$

де t_i – час появи повідомлення у вузлі v_i;

t_j – час наступної взаємодії у вузлі v_j.

Часова впорядкованість використовується для визначення послідовності поширення інформації та коректного формування структури каскаду [41]. Оскільки частина взаємодій у соціальних мережах може бути недоступною через обмеження платформи або налаштування приватності, сформований підграф розглядається як наближене представлення реального процесу поширення інформації.

2.2. Розробка алгоритму побудови інформаційного каскаду

Побудова інформаційного каскаду починається з отримання набору даних, що описує початкове повідомлення та зафіксовані взаємодії користувачів із ним. Для подальшого аналізу використовуються текст повідомлення, часові мітки взаємодій, типи реакцій користувачів та базові характеристики акаунтів [36].

На етапі підготовки даних виконується приведення отриманої інформації до єдиного формату – це необхідно для того, щоб різні типи взаємодій(відповіді, цитування, репости або інші реакції) могли бути використані під час побудови

єдиної структури каскаду. Основними операціями цього етапу є перевірка наявності обов'язкових полів, усунення дубльованих записів, нормалізація часових значень та відокремлення некоректних або неповних елементів.

Після підготовки кожна взаємодія зберігається як окремий елемент вхідного набору даних із зазначенням користувача, типу взаємодії та часу її появи. Такий формат дозволяє надалі послідовно додавати вузли й ребра до каскаду відповідно до порядку виникнення подій.

Окремо враховується якість отриманих даних. Якщо частина взаємодій або профільних характеристик недоступна, такі записи не вилучаються автоматично, а позначаються як неповні. Це дає змогу зберегти доступну інформацію та використати її під час подальшого формування каскаду без штучного розширення або спотворення структури даних. Після підготовки вхідних даних алгоритм переходить до формування структури інформаційного каскаду. На цьому етапі використовується впорядкований набір взаємодій, отриманий після очищення та нормалізації даних.

Першим кроком створюється початковий вузол, що відповідає автору або джерелу повідомлення. Далі алгоритм послідовно обробляє кожну зафіксовану взаємодію. Якщо користувач ще відсутній у структурі каскаду, для нього створюється новий вузол із доступними характеристиками. Якщо вузол уже існує, оновлюються лише параметри, пов'язані з новою взаємодією.

Для кожної взаємодії визначається її тип: відповідь, цитування, репост або інша форма поширення. На основі цього формується зв'язок між відповідними учасниками каскаду. Такий підхід дозволяє зберегти не лише факт участі користувача у поширенні повідомлення, а й характер його взаємодії з інформаційним потоком. Під час формування структури каскаду алгоритм перевіряє, чи належить взаємодія до досліджуваного повідомлення. До каскаду не включаються сторонні записи, які не мають зв'язку з початковим повідомленням або вже сформованими елементами каскаду. Це дозволяє уникнути змішування різних інформаційних потоків.

Якщо частина зв'язків між користувачами відсутня, алгоритм не припиняє

побудову каскаду, а використовує доступні взаємодії як основу для формування наближеної структури. У випадках недостатньої кількості реальних зв'язків можуть застосовуватись додаткові вузли, що імітують можливе розширення каскаду без заміни фактичних даних.

Результатом цього етапу є сформована структура каскаду, у якій кожен вузол відповідає учаснику поширення, а кожен зв'язок – зафіксованій або змодельованій інформаційній взаємодії. Отримана структура передається до наступних етапів аналізу часових залежностей та обчислення характеристик каскаду.

Після формування базової структури каскаду алгоритм уточнює порядок взаємодій за часовими мітками. Для цього події впорядковуються за моментом появи, після чого перевіряється узгодженість напрямку ребер із послідовністю поширення повідомлення. Такий підхід дає змогу уникнути ситуацій, коли зв'язок між вузлами формується всупереч реальному порядку появи взаємодій.

Часові параметри також використовуються для визначення затримок між окремими етапами поширення. Якщо між початковим повідомленням і наступними взаємодіями спостерігається короткий інтервал, каскад може характеризуватися різким початковим зростанням. Якщо взаємодії розподілені рівномірніше, розвиток каскаду розглядається як поступовий. Подібне врахування часової структури є важливим для аналізу інформаційних потоків у соціальних мережах [42].

У випадку неповних даних алгоритм не вилучає весь фрагмент каскаду, а обробляє доступну частину взаємодій. Якщо відсутні окремі профільні характеристики користувача, вузол усе одно може бути доданий до структури каскаду, але з позначенням неповноти атрибутів. Якщо відсутній точний зв'язок між окремими учасниками, такий елемент не використовується як підтверджене ребро, щоб не спотворювати структуру поширення.

Для збереження стабільності аналізу система розділяє фактичні та апроксимовані елементи каскаду. Реальні взаємодії використовуються як основа графа, а додаткові елементи можуть застосовуватися лише як допоміжний

механізм для моделювання можливого розширення каскаду. Це дозволяє не змішувати підтверджені дані з наближеними та коректніше інтерпретувати отримані метрики.

Отже, обробка часових залежностей і неповних даних забезпечує узгоджене формування каскаду навіть тоді, коли вхідна інформація є частковою. Після цього структура може використовуватися для обчислення метрик поширення, аналізу активності вузлів та виявлення нетипових сценаріїв розвитку каскаду. Загальну послідовність побудови інформаційного каскаду наведено на рисунку 2.2.

Рисунок 2.2 – Послідовність побудови інформаційного каскаду

2.3. Розробка алгоритмів аналізу каскаду

Після того як сформована структура інформаційного каскаду виконується обчислення його базових структурних характеристик. Основною метою цього є визначення параметрів які описують форму каскаду, масштаб поширення інформації та особливості зв’язків між учасниками інформаційного потоку [39].

Однією з основних характеристик є глибина каскаду, яка визначає максимальну відстань між початковим вузлом і найдальшим елементом поширення:

$$D_c = max\big(dist(v_0, v_i)\big) \tag{2.8}$$

де v_0 – початковий вузол каскаду;

v_i – вузол каскаду; $dist(v_0, v_i)$ – довжина шляху між вузлами.

Глибина каскаду дозволяє оцінити рівень послідовного поширення повідомлення між користувачами – невелика глибина, зазвичай, характерна для швидкого масового поширення, тоді як великі значення можуть свідчити про багаторівневу передачу інформації між окремими групами користувачів.

Окремо визначається кількість вузлів та ребер каскаду, які характеризують загальний масштаб інформаційного поширення. На основі цих параметрів обчислюється коефіцієнт розгалуження, що визначає середню кількість зв'язків між елементами каскаду:

$$B_c = \frac{|E_c|}{|V_c|} \tag{2.9}$$

де $|E_c|$ – кількість ребер каскаду;

$|V_c|$ – кількість вузлів каскаду.

Високі значення можуть свідчити про інтенсивне одночасне поширення повідомлення між великою кількістю користувачів – низькі значення, навпаки характерні для послідовного або локального розвитку каскаду.

Під час аналізу також враховується форма структури каскаду, і можуть виділятися каскади типу «broadcast», «chain» та «star», які відрізняються характером взаємодій між вузлами та способом поширення інформації, їх структура наведена на рисунку 2.3.

Аналіз структурної форми дозволяє оцінити особливості розвитку інформаційного потоку та виявити нетипові сценарії поширення повідомлень.

Результатом цього етапу є набір базових структурних характеристик каскаду, які використовуються під час подальшого аналізу центральності, часової динаміки та аномальної активності.

Після обчислення базових структурних характеристик виконується аналіз центральності вузлів каскаду. Основною метою цього етапу є визначення користувачів, які мають найбільший вплив на поширення інформації та відіграють ключову роль у розвитку інформаційного потоку [43].

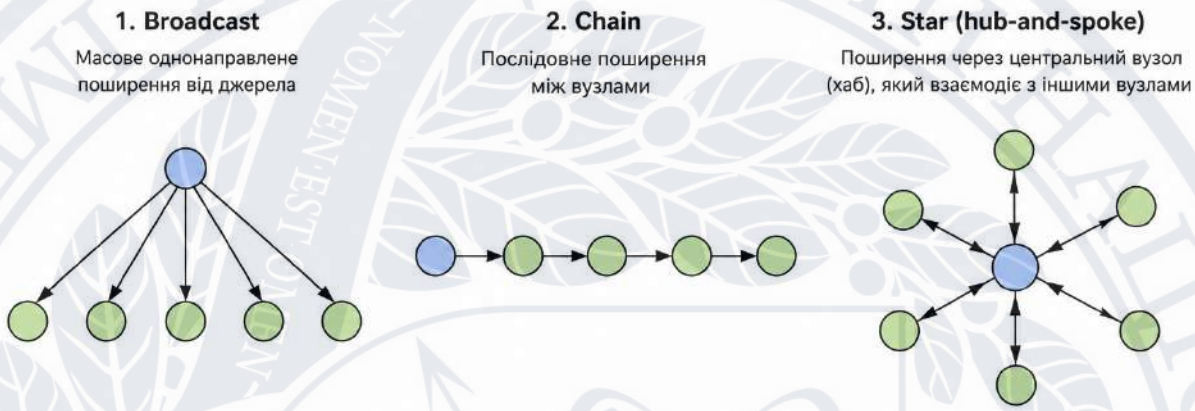

Рисунок 2.3 – Приклади структурних типів інформаційних каскадів

У межах моделі одним із основних показників є проміжність центральності, яка характеризує частоту проходження найкоротших шляхів через певний вузол графа:

$$BC(v_i) = \sum_{s \neq v_i \neq t} \frac{\sigma_{st}(v_i)}{\sigma_{st}} \tag{2.10}$$

де σ_{st} – кількість найкоротших шляхів між вузлами s та t;

$\sigma_{st}\ (v_i)$ – кількість шляхів, що проходять через вузол v_i.

Високі значення центральності можуть свідчити про те, що вузол бере участь у передачі інформації між різними частинами каскаду та впливає на загальну структуру поширення повідомлення. Подібні вузли можуть виступати як ключові посередники або центри координації інформаційного потоку.

Окремо враховується ступінь центральності, що визначає кількість безпосередніх зв’язків вузла з іншими учасниками каскаду:

$$DC(v_i) = \frac{\deg(v_i)}{|V_c| - 1} \tag{2.11}$$

де $\deg(v_i)$ – кількість зв'язків вузла v_i;

$|V_c|$ – кількість вузлів каскаду.

Ступінь центральності дозволяє оцінити локальну активність вузла та рівень його залучення до процесу поширення інформації. Високі значення цього показника можуть відповідати акаунтам, які активно взаємодіють із великою кількістю учасників каскаду.

Для оцінювання загальної форми поширення інформації використовується показник структури вірусності, який характеризує співвідношення між централізованим і багаторівневим поширенням повідомлення [44]:

$$SV = \frac{1}{N_c(N_c - 1)} \sum_{i \neq j} dist(v_i, v_j) \tag{2.12}$$

де SV – показник структури вірусності;

$dist(v_i, v_j)$ – відстань між вузлами каскаду;

N_c – кількість вузлів каскаду.

Невеликі значення структури вірусності характерні для каскадів, де основне поширення проходить безпосередньо від початкового джерела. Високі значення свідчать про багаторівневу передачу інформації між великою кількістю учасників.

Також враховується концентрація активності навколо окремих вузлів. Якщо значна частина взаємодій зосереджена навколо невеликої групи акаунтів, це може свідчити про централізований характер поширення або координовану інформаційну активність.

Результати центральності аналізу використовуються для подальшого дослідження часової динаміки каскаду, оцінювання ролі окремих вузлів у поширенні повідомлення та виявлення потенційно аномальної поведінки у структурі інформаційного потоку.

Після обчислення структурних характеристик та показників центральності проходить аналіз часової динаміки розвитку каскаду. Основною метою цього етапу є визначення швидкості поширення інформації, виявлення періодів підвищеної активності та пошук ознак нетипової або координованої поведінки

учасників [45].

Для оцінювання швидкості розвитку інформаційного потоку використовується показник інтенсивності поширення:

$$V_t = \frac{|E_c|}{t_{max} - t_{min}} \tag{2.13}$$

де $|E_c|$ – кількість взаємодій у каскаді;

t_{max} – час останньої взаємодії;

t_{min} – час початкового повідомлення.

Цей показник дозволяє оцінити середню швидкість формування нових взаємодій у межах каскаду. Високі значення можуть свідчити про різке поширення повідомлення або значну концентрацію активності протягом короткого проміжку часу.

Для аналізу часової структури каскаду також враховуються інтервали між послідовними взаємодіями:

$$\Delta t_i = t_i - t_{i-1} \tag{2.14}$$

де t_i – час поточної взаємодії;

t_{i-1} – час попередньої взаємодії.

Невеликі значення часових інтервалів між великою кількістю взаємодій можуть свідчити про хвилеподібне або координоване поширення інформації. Значні коливання часових параметрів, навпаки, характерні для більш природного розвитку інформаційного потоку.

Ще аналізується рання активність каскаду, для цього оцінюється кількість взаємодій, що виникають протягом початкового етапу поширення повідомлення:

$$E_{early} = \sum_{t_i < t_{lim}} 1 \tag{2.15}$$

де t_{lim} – граничний часовий інтервал початкового етапу розвитку каскаду.

Різке зростання кількості взаємодій на ранньому етапі може свідчити про штучне підсилення поширення або залучення групи акаунтів, які одночасно взаємодіють із повідомленням.

Для виявлення аномальної активності аналізуються:

- неприродно висока швидкість поширення;
- повторювані шаблони взаємодій;
- концентрація активності навколо окремих вузлів;
- синхронізовані часові інтервали між взаємодіями;
- різкі сплески активності у структурі каскаду.

Подібні ознаки можуть свідчити про використання ботоподібних акаунтів, координовану поведінку користувачів або штучне підсилення інформаційного потоку [46].

Окремо враховується розподіл активності у часових інтервалах. Якщо значна кількість взаємодій концентрується у невеликому часовому проміжку, це може вказувати на нетипову поведінку каскаду або організоване поширення повідомлення.

Результати аналізу часової динаміки використовуються для подальшого оцінювання характеру розвитку інформаційного каскаду, визначення рівня аномальної активності та виявлення потенційно маніпулятивних сценаріїв поширення інформації.

Під час аналізу інформаційних каскадів важливе значення має оцінка обчислювальної складності алгоритмів, оскільки зі збільшенням кількості вузлів і взаємодій суттєво зростає навантаження на систему – якщо в реальних соціальних мережах інформаційні каскади можуть містити велику кількість взаємодій, тому алгоритми повинні забезпечувати можливість стабільної обробки графових структур без критичного збільшення часу виконання [47].

Формування структури каскаду виконується шляхом послідовного додавання вузлів та ребер на основі отриманого набору взаємодій. За умови використання впорядкованого списку подій складність побудови графа можна оцінити як:

$$O(|V_c| + |E_c|) \tag{2.16}$$

де $|V_c|$ – кількість вузлів каскаду;

$|E_c|$ – кількість ребер каскаду.

Подібна складність є прийнятною для побудови каскадів навіть у випадку значного обсягу взаємодій, оскільки кожен вузол та ребро додаються до структури лише один раз.

Окреме навантаження виникає під час аналізу графових характеристик – найбільш ресурсомісткими є алгоритми, що використовують пошук шляхів між вершинами графа або аналіз центральності вузлів, рокрема, обчислення проміжної центральності потребує аналізу великої кількості маршрутів між вузлами каскаду, що призводить до збільшення кількості операцій зі зростанням розміру графа [48].

Складність аналізу також залежить від структури самого каскаду. Каскади з великою кількістю рівнів або високим коефіцієнтом розгалуження потребують більшого обсягу обчислень під час визначення шляхів поширення інформації, глибини структури та часових характеристик взаємодій. Окремо враховується обробка часових параметрів. Для коректного формування послідовності поширення інформації взаємодії впорядковуються за часовими мітками. Якщо дані вже отримані у впорядкованому вигляді, додаткове сортування не виконується, що дозволяє зменшити кількість допоміжних операцій під час побудови каскаду.

Таким чином основне обчислювальне навантаження системи пов’язане не з побудовою самого каскаду, а з подальшим аналізом його характеристик: центральності вузлів та часової динаміки поширення інформації, що створює необхідність подальшої оптимізації алгоритмів та використання локального аналізу підграфів замість повної структури соціальної мережі.

При аналізі інформаційних каскадів одним з завдань є забезпечення стабільної роботи алгоритмів у випадку збільшення кількості взаємодій або одночасного аналізу декількох інформаційних потоків. Для цього в межах системи використовується поетапна обробка даних, за якої кожен каскад формується та аналізується незалежно від інших структур поширення інформації.

На першому етапі виконується підготовка та очищення вхідних даних. Після цього формується локальний підграф каскаду, який містить лише вузли та взаємодії, пов’язані з конкретним повідомленням. Такий підхід дозволяє уникнути обробки повного графа соціальної мережі та суттєво зменшує кількість елементів, що беруть участь у подальших обчисленнях [47].

Окремо враховується можливість часткової обробки даних. Якщо частина взаємодій є недоступною або надходить із затримкою, система може виконувати аналіз доступної частини каскаду без необхідності повторного формування всієї структури після оновлення даних. Для зменшення навантаження часові параметри взаємодій впорядковуються ще на етапі підготовки даних, що дозволяє уникнути повторного сортування під час аналізу часової динаміки та формування структури поширення інформації. Оскільки окремі каскади не залежать один від одного, їх аналіз може виконуватись незалежно. Такий підхід створює можливість масштабування системи у випадку збільшення кількості повідомлень або одночасної обробки декількох інформаційних потоків. Використання локального аналізу підграфів та поетапної обробки даних забезпечується зменшення обчислювального навантаження й підвищення стабільності роботи алгоритмів під час аналізу великих інформаційних каскадів.

Під час аналізу інформаційних каскадів найбільше обчислювальне навантаження виникає під час розрахунку графових метрик, зокрема показників центральності та характеристик структури поширення інформації. Тому і використовуються підходи, спрямовані на зменшення кількості повторних обчислень і локалізацію аналізу лише до необхідної частини графа [48].

Одним із основних способів оптимізації є використання підграфа каскаду замість повної структури соціальної мережі. Обчислення центральності, глибини каскаду та часових характеристик виконуються лише для вузлів, які беруть участь у поширенні конкретного повідомлення. Це дозволяє суттєво скоротити кількість операцій під час аналізу великих інформаційних потоків.

Окремо оптимізується обчислення центральності вузлів. Оскільки алгоритми пошуку найкоротших шляхів мають високу складність для великих

графів, аналіз виконується лише після завершення формування структури каскаду. Це дозволяє уникнути повторного перерахунку показників після кожної нової взаємодії. Для зменшення навантаження частина структурних характеристик зберігається після первинного обчислення та повторно використовується на наступних етапах аналізу. Повторне обчислення виконується лише у випадку зміни структури каскаду або появи нових взаємодій, які можуть вплинути на значення метрик. Під час аналізу часової динаміки також використовується локальна обробка часових інтервалів без необхідності повторного перегляду всієї структури каскаду. Це дозволяє швидше визначати сплески активності, зміни швидкості поширення та аномальні часові залежності.

У результаті використання локального аналізу, повторного використання обчислених характеристик та обмеження області розрахунків забезпечується зменшення обчислювальних витрат під час аналізу інформаційних каскадів без втрати точності основних метрик.

2.4. Узагальнений алгоритм аналізу інформаційного каскаду

Після окремих етапів побудови та аналізу інформаційного каскаду формується узагальнений алгоритм обробки інформаційного потоку. Його основною метою є послідовне виконання етапів збору даних, побудови структури каскаду, аналізу графових характеристик та оцінювання динаміки поширення повідомлення [47].

Робота алгоритму починається з отримання вхідних даних про повідомлення та взаємодії користувачів. Після цього виконується попередня обробка інформації, яка включає очищення даних, перевірку коректності часових параметрів та формування впорядкованого набору взаємодій. На наступному етапі формується структура каскаду у вигляді орієнтованого графа. Початкове повідомлення використовується як кореневий вузол, після чого до структури послідовно додаються користувачі та зв’язки між ними відповідно до зафіксованих взаємодій. Після завершення побудови каскаду виконується перевірка часової узгодженості структури та коректності напрямків поширення

інформації. Далі алгоритм переходить до обчислення структурних характеристик каскаду. На цьому етапі визначаються:

- глибина каскаду;
- кількість вузлів та ребер;
- коефіцієнт розгалуження;
- показники центральності;
- структура вірусності;
- часові характеристики поширення.

Після обчислення базових характеристик виконується аналіз динаміки інформаційного потоку. Алгоритм оцінює швидкість поширення повідомлення, часові інтервали між взаємодіями, рівень концентрації активності та наявність ознак нетипової поведінки у структурі каскаду. Для виявлення аномальної активності окремо аналізуються:

- різкі сплески взаємодій;
- синхронізована активність користувачів;
- концентрація поширення навколо окремих вузлів;
- неприродно висока швидкість розвитку каскаду.

Результатом роботи алгоритму є сформований набір структурних та часових характеристик каскаду, які можуть використовуватись для подальшого аналізу інформаційних потоків, оцінювання особливостей поширення повідомлення та виявлення аномальних або координованих сценаріїв поширення інформації. Під час реалізації узагальненого алгоритму важливе значення має узгодженість окремих етапів аналізу. Побудова каскаду, обчислення графових характеристик та аналіз часової динаміки виконуються послідовно, оскільки результати попередніх етапів використовуються під час подальших розрахунків.

Для забезпечення стабільності роботи системи алгоритм допускає аналіз неповних наборів даних. Якщо окремі взаємодії або характеристики вузлів є недоступними, система виконує аналіз доступної частини каскаду без порушення цілісності структури інформаційного потоку.

Подібна структура дозволяє забезпечити послідовний аналіз

інформаційного каскаду та узгоджену обробку графових і часових характеристик поширення інформації [39]. Застосування такого алгоритму забезпечує узгоджене виконання всіх етапів аналізу інформаційного каскаду: від підготовки даних і побудови графа до обчислення структурних та часових характеристик.

Особливістю запропонованого підходу є поєднання структурного та часового аналізу. Структурні метрики дозволяють оцінити форму каскаду, глибину поширення, розгалуженість та роль окремих вузлів, тоді як часові характеристики дають змогу визначати швидкість розвитку інформаційного потоку, ранні сплески активності та можливі ознаки координованого поширення. У загальному вигляді послідовність роботи алгоритму зображена на рисунку 2.4.

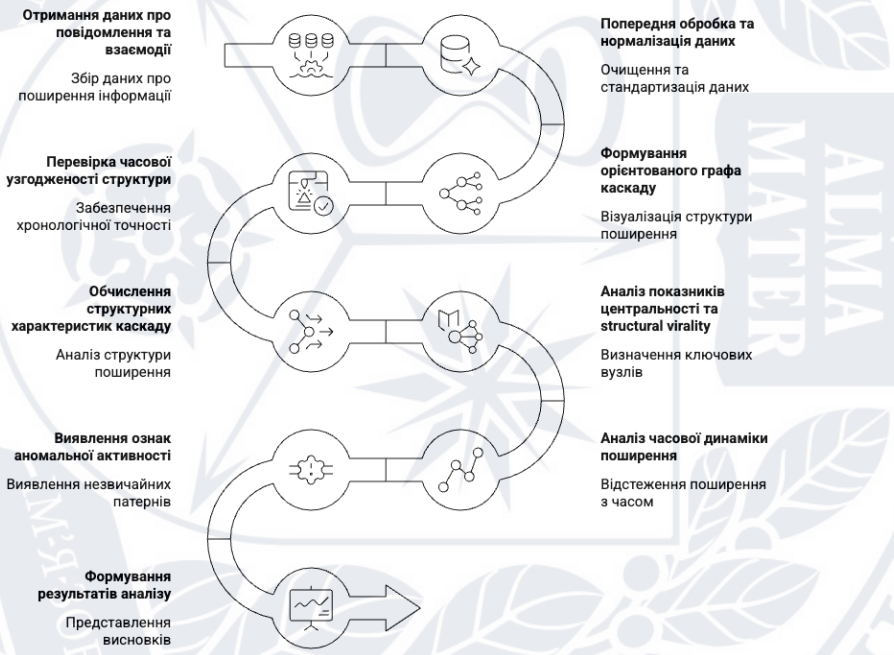

Рисунок 2.4 – Узагальнений алгоритм аналізу інформаційного каскаду

Таким чином, у межах розділу було формалізовано модель інформаційного каскаду, описано алгоритм його побудови, визначено основні структурні та часові метрики, а також розглянуто підходи до оптимізації обчислень. Запропонований алгоритм створює основу для подальшої практичної реалізації системи аналізу динаміки інформаційних каскадів та перевірки її роботи на даних соціальних мереж.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ АНАЛІЗУ ДИНАМІКИ ІНФОРМАЦІЙНИХ КАСКАДІВ

3.1. Архітектура системи аналізу інформаційних каскадів

У реалізованому додатку побудова інформаційних каскадів зроблена у вигляді послідовного pipeline, який виконує збір даних, побудову графової структури поширення, обчислення метрик каскаду та моделювання подальшої динаміки розповсюдження повідомлення.

Backend системи реалізований на основі FastAPI та відповідає за запуск повного циклу аналізу. Після отримання посилання на повідомлення система ініціює збір даних, передає отримані взаємодії до модуля побудови графа, після чого виконується розрахунок структурних і часових характеристик каскаду. Загальну архітектуру системи аналізу каскадів наведено на рисунку 3.1.

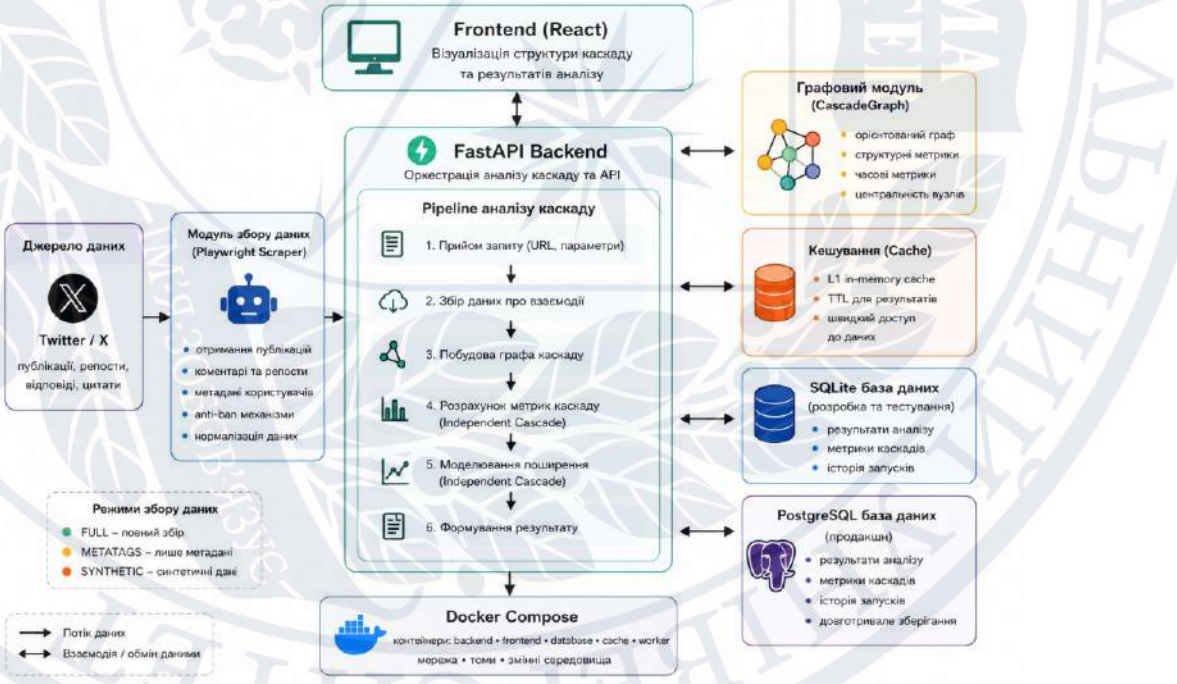

Рисунок 3.1 – Архітектура системи аналізу інформаційних каскадів

Логіка обробки реалізована, як послідовність етапів – отримання даних про публікацію, формування структури взаємодій, побудова орієнтованого графа,

розрахунок метрик каскаду та запуск моделі поширення. Такий підхід дозволяє розділити збір даних, графову обробку та моделювання динаміки на окремі функціональні блоки. Фрагмент pipeline аналізу наведено у лістингу послідовності обробки інформаційного каскаду:

```
post = await scraper.scrape(url)
manipulation = manipulation_detector.analyze(post)
bots = bot_detector.analyze_users(post.users)
graph = CascadeGraph()
graph.build_from_post(post)
graph.set_bot_scores(bots)
metrics = graph.compute_cascade_metrics()
simulation = graph.run_ic_model_with_temporal_decay()
```

У лістингу показано, що після збору даних система створює об’єкт CascadeGraph, будує граф каскаду на основі отриманих взаємодій, обчислює метрики та запускає модель подальшого поширення інформації. Саме цей pipeline є основою програмної реалізації алгоритмів аналізу динаміки каскадів.

Для зменшення повторних обчислень у системі використовується дворівневе кешування результатів – на першому рівні застосовується L1 in-memory cache, який забезпечує швидке повернення результатів нещодавно виконаних запитів. Якщо результат відсутній у пам’яті, система перевіряє L2-рівень – SQLite або PostgreSQL, у якому зберігаються результати попередніх аналізів та історія запусків. SQLite використовується тільки при розробці та первинному тестуванні, а PostgreSQL вже у продакшені, через те що SQLite це вбудована БД, яка має ряд обмежень, які не дозволять їй правильно функціонувати при умові одночасних запитів. Якщо дані відсутні на обох рівнях, запускається повний pipeline аналізу каскаду. Фрагмент логіки перевірки кешу наведено у лістингу використання кешу під час аналізу:

```
cached = l1_cache.get(url)
if cached:
    return cached

stored = sqlite_cache.get(url)
if stored:
```

```
    l1_cache.set(url, stored)
    return stored

result = await run_pipeline(url)
l1_cache.set(url, result)
sqlite_cache.save(url, result)
return result
```

Система спочатку перевіряє швидкий кеш у пам’яті, після цього звертається до SQL-сховища, а повний аналіз каскаду виконується лише за відсутності збережених результатів. Такий підхід зменшує кількість повторних звернень до scraper-модуля та скорочує час повторного аналізу повідомлення.

Використання кешування є важливим для аналізу каскадів, оскільки побудова графа, обробка взаємодій та моделювання поширення можуть потребувати додаткового часу при повторному аналізі одного й того самого повідомлення. Для зберігання результатів аналізу використовується SQL-сховище. У базі фіксуються URL повідомлення, дата аналізу, основні метрики каскаду та сформований результат. Це дозволяє зберігати історію запусків і використовувати її під час подальшого порівняння інформаційних каскадів. Окремо система підтримує контейнеризований запуск за допомогою Docker Compose. Контейнеризація спрощує розгортання backend, бази даних і допоміжних сервісів, а також забезпечує однакове середовище виконання на різних пристроях.

Таким чином, архітектура системи аналізу інформаційних каскадів побудована навколо pipeline обробки даних: збір інформації, побудова графа, розрахунок метрик, моделювання поширення та збереження результатів. Така структура відповідає задачам роботи, оскільки дозволяє досліджувати не лише окреме повідомлення, а повну динаміку його поширення у вигляді інформаційного каскаду.

3.2. Реалізація модуля збору даних та побудови каскадів

Модуль збору даних це початковий етап формування інформаційного каскаду. Його завдання полягає в отриманні даних про початкову публікацію,

доступні взаємодії користувачів, часові характеристики та структуру поширення повідомлення. Модуль не обмежується зчитуванням тексту поста, а формує структуру PostData, яка надалі використовується для побудови орієнтованого графа каскаду.

Збір даних виконується за допомогою scraper-модуля з використанням Playwright та BeautifulSoup. Playwright застосовується для відкриття динамічних сторінок соціальної мережі X, оскільки значна частина вмісту завантажується після виконання JavaScript-коду. BeautifulSoup використовується для подальшого розбору отриманої HTML-структури та виділення потрібних елементів сторінки. Такий підхід дозволяє працювати не лише з початковим HTML-кодом, а й з даними, які з’являються після завантаження сторінки у браузерному середовищі.

У scraper передбачено авторизований режим роботи. Для цього використовується збережений стан браузерної сесії, наприклад файл x_auth.json, який містить cookie та службові параметри входу. Це дозволяє відкривати сторінки у режимі звичайного користувацького перегляду та отримувати більше доступних публічних взаємодій, ніж у неавторизованому режимі. Використання збереженого стану сесії також зменшує потребу в повторному ручному вході під час кожного запуску scraper-модуля. Загальна логіка роботи scraper-модуля наведена у лістингу узагальненої логіки збору даних:

```
post = await scraper.scrape(url)
post_data = PostData(
    url=url,
    text=post.text,
    author=post.author,
    metrics=post.metrics,
    users=post.users,
    propagation_tree=post.propagation_tree,
    scraping_mode=post.scraping_mode,
    real_fields=post.real_fields
)
```

У результаті scraper формує об’єкт PostData, що містить текст публікації, автора, базові метрики активності, список користувачів, дерево поширення та режим збору. Така структура дозволяє передавати до наступних модулів не розрізнені дані, а повний опис каскаду.

Для підвищення стабільності у модулі застосовуються випадкові паузи між діями, зміна параметрів вікна браузера, очікування завантаження потрібних елементів сторінки та блокування ресурсів, які не впливають на аналіз. Ці механізми зменшують навантаження на сторінку та підвищують стабільність отримання даних.

Після отримання вмісту сторінки система виконує нормалізацію даних. На цьому етапі очищується текст, усуваються дублікати взаємодій, уніфікуються ідентифікатори користувачів, а часові значення приводяться до єдиного формату. Важливим параметром є delay_sec, який визначає затримку між початковою публікацією та появою конкретної взаємодії. Надалі цей параметр використовується під час аналізу часової динаміки каскаду.

Основною структурою для побудови графа є propagation_tree. У ній початкова публікація виступає кореневим вузлом, а відповіді, цитування та репости подаються як дочірні вузли. Кожен елемент дерева містить ідентифікатор вузла, посилання на батьківський вузол, тип взаємодії та часову затримку. Логічне представлення зв’язку в дереві поширення представлено у лістингу:

```
post_data.propagation_tree.append({
    "parent_id": parent_id,
    "target_id": node_id,
    "edge_type": edge_type,
    "delay_sec": delay_sec
})
```

У лістингу наведено узагальнений запис, що входить до складу структури PostData. Значення parent_id і target_id визначають напрям поширення інформації між вузлами, edge_type описує тип взаємодії, а delay_sec використовується для подальшого аналізу часової динаміки каскаду. Якщо певна

взаємодія пов’язана безпосередньо з початковою публікацією, її батьківським вузлом є кореневий вузол каскаду. Якщо ж взаємодія виникає як відповідь або реакція на інший елемент обговорення, вона розміщується на наступному рівні дерева.

Вебскрапінг, є популярною практикою побудови аналітичних систем збору даних із соціальних мереж [49]. Оскільки соціальна мережа X не завжди надає повний обсяг даних про взаємодії з публікацією, у системі передбачено врахування повноти отриманої інформації. Для цього у структурі PostData використовуються параметри scraping_mode та real_fields, які дозволяють визначити, які поля були отримані безпосередньо зі сторінки, а які залишилися недоступними або були отримані в обмеженому режимі. Це дає змогу коректніше інтерпретувати результати аналізу каскаду та не змішувати повні й часткові набори даних.

Після формування PostData дані передаються до модуля CascadeGraph, де дерево поширення перетворюється на орієнтований граф. Кожен елемент propagation_tree використовується для створення зв’язку між вузлами, а пара parent_id - target_id перетворюється на напрямлене ребро графа. Таким чином структура поширення переходить від деревоподібного представлення до графової моделі, придатної для обчислення метрик каскаду. У лістингу передачі структури каскаду до графового модуля показано перехід від підготовленої структури даних до графового аналізу:

```
graph = CascadeGraph()
graph.build_from_post(post_data)
metrics = graph.compute_cascade_metrics()
simulation = graph.run_ic_model_with_temporal_decay()
```

На цьому етапі scraper-модуль завершує свою роботу, а подальша обробка виконується модулем CascadeGraph. Такий розподіл відповідальності спрощує підтримку системи: один компонент відповідає за отримання й нормалізацію даних, інший – за побудову графа, обчислення метрик і моделювання поширення.

Оскільки збір даних зі сторінки є одним із найбільш тривалих етапів, перед запуском scraper-модуля система використовує механізм кешування, описаний у підрозділі 3.1. Якщо для відповідного URL уже існує збережений результат, повторне відкриття сторінки не виконується, а до подальшого аналізу передається раніше сформована структура PostData. Це зменшує кількість звернень до соціальної мережі та забезпечує відтворюваність аналізу одного й того самого каскаду.

Узагальнену структуру даних, що зберігаються у SQL-сховищі, наведено в таблиці 3.1.

Таблиця 3.1 – Узагальнена структура збереження результатів аналізу каскаду

Поле	Призначення
url	посилання на початкову публікацію
analyzed_at	дата й час виконання аналізу
post_data	структуровані дані публікації та взаємодій
cascade_metrics	основні метрики побудованого каскаду
ic_simulation	результат моделювання поширення
data_quality	характеристика повноти отриманих даних

Таблиця 3.1 відображає логічний склад інформації, яка зберігається для повторного використання результатів. У SQLite не обов’язково зберігається кожен елемент дерева поширення як окремий запис; структура каскаду може входити до серіалізованого результату аналізу або до структурованих даних публікації. Такий спосіб збереження дозволяє не запускати scraper повторно для вже оброблених посилань і водночас зберігати достатньо даних для подальшої перевірки графових алгоритмів.

Отже, даний модуль збору даних забезпечує перехід від сторінки соціальної мережі до формалізованої структури інформаційного каскаду. Він

отримує дані за допомогою Playwright і BeautifulSoup, підтримує авторизований режим через збережений стан сесії, виконує нормалізацію взаємодій, формує propagation_tree, враховує неповноту даних і передає підготовлену структуру до CascadeGraph для подальшого аналізу динаміки поширення.

3.3. Реалізація алгоритмів аналізу каскаду

Після завершення збору та нормалізації даних система переходить до алгоритмічного аналізу інформаційного каскаду. На цьому етапі вхідними даними є не сторінка соціальної мережі, а підготовлена структура PostData, яка містить дерево поширення propagation_tree, список користувачів, типи взаємодій та часові характеристики. Основним завданням цього етапу є перетворення підготовленої структури на орієнтований граф та обчислення показників, що описують форму, швидкість і характер поширення повідомлення.

Клас CascadeGraph відповідає за побудову графа, розрахунок метрик каскаду та запуск моделі подальшого поширення. Для подання структури каскаду використовується networkx.DiGraph, оскільки зв’язки між вузлами мають напрям: від початкової публікації або проміжної взаємодії до наступного елемента поширення. Загальну послідовність роботи модуля наведено у лістингу запуску графового аналізу каскаду:

```
graph = CascadeGraph()
graph.build_from_post(post_data)
metrics = graph.compute_cascade_metrics()
simulation = graph.run_ic_model_with_temporal_decay()
```

Метод build_from_post() формує граф на основі поля propagation_tree. Кожен елемент дерева поширення використовується для створення вузла або напрямленого ребра. Вузол зберігає доступні атрибути: ідентифікатор користувача, рівень у каскаді, часову затримку даних та інші службові характеристики, а ребро зберігає тип взаємодії: відповідь, цитування або репост. Узагальнений фрагмент додавання елементів до графа наведено у лістингу:

```
self.graph.add_node(
    node_id,
    user_id=user_id,
    depth=depth,
    delay_sec=delay_sec,
    is_real=is_real
)
self.graph.add_edge(
    parent_id,
    node_id,
    edge_type=edge_type)
```

Атрибут depth використовується для визначення рівня вузла відносно початкової публікації. Поле delay_sec застосовується у часовому аналізі, а edge_type дозволяє відрізняти типи взаємодій між вузлами. Завдяки цьому граф містить не лише зв'язки між елементами каскаду, а й інформацію, необхідну для наступних обчислень.

Після побудови графа виконується метод compute_cascade_metrics() – він формує набір структурних і часових характеристик каскаду. На рівні структури графа система визначає кількість вузлів і ребер, максимальну глибину, коефіцієнт розгалуження та тип каскаду. Ці значення обчислюються без повторного звернення до scraper-модуля, оскільки всі необхідні дані вже містяться у графі.

Окремо в межах compute_cascade_metrics() виконується аналіз центральності вузлів. Для цього використовується функціональність бібліотеки NetworkX, зокрема розрахунок betweenness_centrality. У результаті система визначає вузли, які мають найбільше значення для зв'язності каскаду. У практичній реалізації зберігається не повний набір значень для всіх вузлів, а список найбільш значущих вузлів, що зменшує обсяг результуючих даних і спрощує їх подальше використання.

Часова частина аналізу базується на атрибуті delay_sec. Система збирає

часові затримки вузлів, групує їх за інтервалами та обчислює показники ранньої активності. У результаті визначаються early_burst_5m, early_burst_10m, bucket_distribution, spike_ratio та timing_variance.

Узагальнений фрагмент розрахунку часових показників наведено у лістингу:

```
delays = [
    node["delay_sec"]
    for _, node in self.graph.nodes(data=True)
    if node.get("delay_sec") is not None
]
early_burst_5m = count_less_than(delays, 300) / len(delays)
early_burst_10m = count_less_than(delays, 600) / len(delays)
bucket_distribution = build_time_buckets(delays)
spike_ratio = max(bucket_distribution.values()) / len(delays)
timing_variance = variance(delays)
```

У лістингу показано, що часові характеристики розраховуються на основі атрибутів вузлів графа. Значення early_burst_5m та early_burst_10m показують частку взаємодій, що виникли протягом перших 5 і 10 хвилин. bucket_distribution описує розподіл активності за часовими інтервалами. spike_ratio визначає частку взаємодій у найбільш активному інтервалі, а timing_variance використовується для оцінювання нерівномірності появи реакцій.

Додатково розраховується coordination_index – він використовується як допоміжна характеристика часової синхронності дій у каскаді, цей показник не розглядається окремо від інших метрик, а застосовується разом зі структурними та часовими характеристиками для опису динаміки поширення.

Окремим етапом є моделювання подальшого поширення інформації. Для цього використовується метод run_ic_model_with_temporal_decay(). Він реалізує модель незалежного каскаду з урахуванням часового згасання активності. Під час симуляції активні вузли можуть передавати інформацію сусіднім вузлам, а

ймовірність такого переходу зменшується залежно від часової затримки. У моделі також враховується характеристика авторитетності вузла, пов’язана з розміром його аудиторії.

Результат моделювання формується як структура ICSimulation. Вона містить послідовність кроків моделі, кількість вузлів у станах S, I та R, крок пікової активності, загальну кількість охоплених вузлів і крок завершення поширення. Фрагмент запуску моделі наведено у лістингу:

```
simulation = graph.run_ic_model_with_temporal_decay()
peak_step = simulation.peak_step
total_reached = simulation.total_reached
convergence_step = simulation.convergence_step
```

У результаті роботи модуля CascadeGraph система отримує два основні результати: структуру метрик CascadeMetrics та результат моделювання ICSimulation. Перша структура описує фактичний стан побудованого каскаду, друга – можливу подальшу динаміку поширення. Обидва результати передаються до загальної відповіді API та використовуються для подальшої візуалізації й аналізу.

Алгоритми аналізу каскаду зосереджені в модулі CascadeGraph. Він перетворює дерево поширення на орієнтований граф, обчислює структурні та часові метрики, визначає тип каскаду і запускає IC/SIR-моделювання.

3.4. Візуалізація структури каскаду та динаміки поширення

Після алгоритмічного аналізу результати передаються до клієнтської частини системи. Frontend реалізовано як вебінтерфейс, який отримує від backend готові дані про вузли, ребра, метрики каскаду та результати моделювання. Під час проєктування цього компонента враховувалися ключові вимоги до сучасного розроблення frontend-частини додатків, а саме: забезпечення високої швидкодії, зручності користування та привабливості

інтерфейсу для взаємодії [50]. Основні обчислення виконуються на серверній стороні, а інтерфейс використовується для наочного подання структури поширення та перевірки результатів аналізу. Дані для побудови графа передаються у вигляді структурованого набору вузлів і ребер, тому frontend не змінює логіку аналізу, а лише відображає результат роботи. Такий розподіл дозволяє зберегти узгодженість між розрахованими даними та їх графічним представленням.

Основним елементом візуалізації є орієнтований граф – він показує початкову публікацію, наступні реакції та напрямлені зв’язки між ними. Вузли позначають окремі елементи каскаду, а стрілки показують напрям поширення інформації від початкового поста до відповідей, цитувань і подальших реакцій. Колір використовується для розрізнення типів взаємодій, що спрощує перегляд структури каскаду. Таке подання дозволяє швидко оцінити загальну форму поширення, кількість гілок і наявність проміжних вузлів у структурі каскаду. Загальний вигляд орієнтованого графа інформаційного каскаду наведено на рисунку 3.2.

Рисунок 3.2 – Загальний вигляд орієнтованого графа поширення каскаду

На рисунку 3.2 видно, що інтерфейс відображає не лише сам граф, а й короткі числові характеристики: кількість вузлів, кількість зв’язків, кількість взаємодій, глибину каскаду та рівень раннього сплеску. Ці значення дають змогу швидко оцінити масштаб і форму поширення без переходу до детального звіту.

Для зручності аналізу граф підтримує різні режими перегляду. У режимі часової динаміки вузли розташовуються відповідно до моменту появи взаємодії після початкової публікації, як показано на рисунку 3.3. Ранні взаємодії відображаються ближче до початкового вузла, а більш пізні – поступово зміщуються далі. З рисунку видно, що часове розташування вузлів допомагає інтерпретувати розвиток каскаду. Якщо більшість вузлів зосереджена біля початкової публікації, це свідчить про швидку появу реакцій після публікації. Якщо вузли розподілені більш рівномірно, каскад має поступовий характер розвитку.

Рисунок 3.3 – Відображення графа каскаду в режимі часової динаміки

Окремо в інтерфейсі передбачено перегляд характеристик вибраного вузла. Після вибору елемента графа система показує його тип, час появи після публікації, кількість подальших поширень і кількість підписників. Це дозволяє перейти від загальної структури каскаду до аналізу конкретного вузла та оцінити його роль у подальшому поширенні.

Рисунок 3.4 – Перегляд характеристик вибраного вузла каскаду

Для аналізу ролі окремих вузлів у структурі поширення система відображає індекс впливу та вузли-посередники. Індекс впливу показує, які акаунти найбільше вплинули на подальше розгалуження каскаду. Блок вузлів-посередників відображає елементи, через які проходять основні гілки поширення. Таке подання допомагає визначити не лише кількість реакцій, а й структурну роль окремих учасників.

Наступний блок інтерфейсу призначений для перегляду динаміки поширення та часових сплесків. Він показує, як змінюється кількість вузлів у різних станах моделі: ще не охоплені, активні та ті, що вже завершили участь у поширенні.

Рисунок 3.5 – Відображення індексу впливу та вузлів-посередників

Такий графік дозволяє оцінити момент пікової активності та швидкість згасання каскаду.

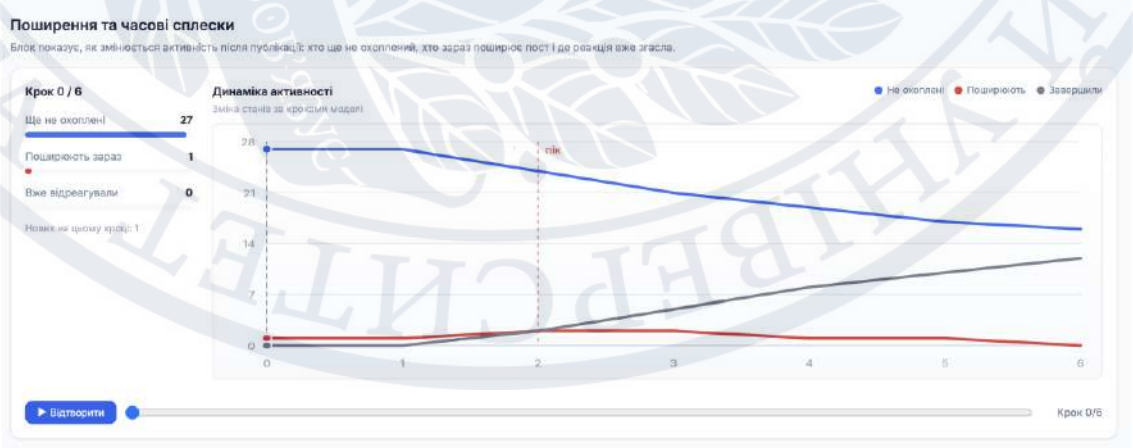

Рисунок 3.6 – Візуалізація динаміки поширення за кроками моделі

У нижній частині блоку моделювання відображаються параметри поширення та показник часової координації. Параметр згасання активності показує, як швидко зменшується інтенсивність поширення після перших хвилин. Параметр впливу великих акаунтів дозволяє оцінити, як на динаміку каскаду впливають вузли з більшою аудиторією. Показник часової координації використовується для оцінювання того, чи були взаємодії зосереджені в короткому часовому проміжку.

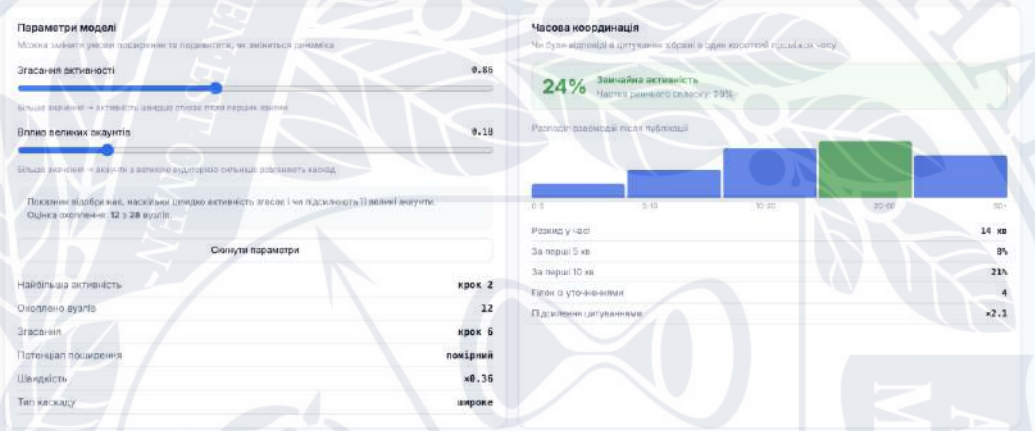

Рисунок 3.7 – Параметри моделі поширення та оцінка часової координації

Таким чином, frontend-частина системи забезпечує наочне подання результатів, сформованих backend-модулями. Орієнтований граф дозволяє перевірити структуру каскаду, блок вибраного вузла – оцінити окремі елементи поширення, індекс впливу та вузли-посередники – визначити ключові точки каскаду, а графік моделювання – простежити часову динаміку активності.

3.5. Перевірка роботи системи аналізу інформаційних каскадів

Перевірка роботи системи виконувалась для підтвердження коректності повного циклу аналізу публікації: формування PostData, побудови орієнтованого графа, розрахунку метрик, моделювання поширення та оцінювання ознак маніпулятивності контенту. Для перевірки використовувались підготовлені приклади публікацій і взаємодій, на основі яких система формувала propagation_tree, будувала граф у модулі CascadeGraph, обчислювала структурні

й часові показники та запускала IC/SIR-моделювання.

У межах перевірки оцінювалась робота трьох основних груп модулів. Перша група стосувалась графового аналізу: система мала побудувати дерево поширення, створити вузли й напрямлені ребра, визначити кількість вузлів і зв’язків, глибину, розгалуження та тип каскаду. Окремо перевірялось, що часові показники узгоджуються з розподілом реакцій у часі, а значення глибини відповідає найдальшому рівню поширення від початкової публікації.

Друга група перевірок стосувалась моделі поширення. Для IC/SIR-моделювання контролювалось формування послідовності станів S, I та R, визначення пікового кроку активності, загального охоплення та моменту завершення поширення.

Третя група перевірок стосувалась допоміжних модулів аналізу контенту та поведінкових ознак. Система повинна була формувати оцінку маніпулятивності публікації, повертати текстові маркери, категорію ризику та додаткові поведінкові характеристики акаунтів. Ці результати не замінювали графовий аналіз, але використовувались як додатковий контекст під час інтерпретації інформаційного каскаду.

Також було оцінено загальний час виконання аналізу. Вимірювався проміжок від надсилання посилання до отримання повного результату. Результати наведено в таблиці 3.2.

Таблиця 3.2 – Загальний час виконання аналізу публікації

Сценарій	Загальний час виконання	Характеристика
Перший повний аналіз публікації	6–9 хв	збір даних, побудова каскаду, метрики, моделювання, оцінка контенту

Продовження табл. 3.2

Сценарій	Загальний час виконання	Характеристика
Повторний аналіз тієї самої публікації	до 1 хв	використання вже підготовлених або збережених результатів

У результаті перевірки підтверджено, що система коректно формує дерево поширення, будує орієнтований граф, розраховує структурні й часові метрики, визначає тип каскаду, виконує IC/SIR-моделювання та формує оцінку маніпулятивності контенту.

ВИСНОВКИ

У бакалаврській роботі було досліджено проблематику аналізу динаміки інформаційних каскадів у соціальних мережах та реалізовано програмно побудову, аналіз та візуалізацію структури поширення постів. За результатами дослідження отримано наступні висновки:

1. В процесі роботи було проаналізовано сучасні підходи до дослідження інформаційних процесів у соціальних мережах. Встановлено, що поширення повідомлень має мережевий і часовий характер, тому для його аналізу доцільно використовувати графове подання. У межах роботи інформаційний каскад розглянуто як орієнтований підграф, у якому вузли відповідають публікації та взаємодіям користувачів, а ребра відображають напрям поширення інформації.
2. Було визначено основні характеристики інформаційного каскаду: кількість вузлів і зв’язків, глибину поширення, коефіцієнт розгалуження, часові затримки, ранню активність, часові сплески, центральність вузлів та тип структури каскаду. Такі показники дозволяють оцінювати не лише масштаб поширення повідомлення, а й його форму, швидкість розвитку та роль окремих учасників у загальній структурі інформаційного потоку.
3. На основі аналізу існуючих підходів було обґрунтовано використання методів графового аналізу, часових метрик та IC/SIR-моделювання.
4. Розроблений алгоритм передбачає формування структури PostData, побудову дерева поширення propagation_tree, перетворення його на орієнтований граф, розрахунок структурних і часових метрик та моделювання подальшого розвитку каскаду.
5. Практичним результатом роботи стала програмна реалізація системи аналізу інформаційних каскадів. Backend-частина на FastAPI координує виконання pipeline аналізу, модуль збору даних формує структурований опис публікації та взаємодій, а модуль CascadeGraph будує граф, розраховує метрики, визначає тип каскаду та запускає моделювання поширення. Для повторного використання результатів застосовано кешування та збереження даних у SQLite.

Також реалізовано frontend-візуалізацію результатів аналізу. Інтерфейс відображає орієнтований граф поширення, основні метрики, характеристики вибраного вузла, індекс впливу, вузли-посередники та динаміку поширення. Це дозволяє поєднати числові результати з наочним поданням структури каскаду.

6. Проведена перевірка підтвердила працездатність реалізованої системи. Система коректно формує дерево поширення, будує орієнтований граф, розраховує структурні й часові метрики, визначає тип каскаду та виконує IC/SIR-моделювання. Додатково перевірялась робота допоміжних модулів оцінювання контенту й поведінкових ознак. Перший повний аналіз публікації займає орієнтовно 6–9 хвилин, тоді як повторний аналіз виконується швидше завдяки використанню збережених результатів.

Отримані результати підтверджують, що розроблений підхід може використовуватись для автоматизованого аналізу динаміки інформаційних каскадів у соціальних мережах. Подальший розвиток роботи може бути пов’язаний із додаванням нових метрик, удосконаленням моделі прогнозування поширення та розширенням механізмів виявлення аномальної активності в каскадах.

СПИСОК ВИКОРИСТАНИХ ПОСИЛАНЬ

1. Digital 2024: Global Overview Report. URL: https://datareportal.com/reports/digital-2024-global-overview-report (дата звернення: 09.05.2026)

2. Суліма В. К., Баценко Т. В. Вплив топології соціальних мереж на швидкість поширення інформаційних загроз. Міжнародна наукова інтернет-конференція на тему: "Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення", випуск 110 (м. Тернопіль, 7-8 травня 2026 року)

3. Потапова Н. А., Веселовська Н. Р., Лаптєва М. А., Суліма В. К. Проєкт та архітектура системи аналізу інформаційних потоків у соціальних мережах на основі інтеграції NLP та графових методів. Наука і техніка сьогодні. 2026. № 5. С. 5628-5641.

4. Hendler, J., & Golbeck, J. (2022). Metcalfe's law, web 2.0, and the semantic web. Web Semantics: Science, Services and Agents, 6(1), 14–20. DOI: 10.1016/j.websem.2007.11.008

5. Goel, S., Anderson, A., Hofman, J., & Watts, D. J. (2022). The structural virality of online diffusion. Management Science, 68(1), 180–196. DOI: 10.1287/mnsc.2021.4111

6. Потапова Н. А., Суліма В. К., Лаптєва М. А., Павлюк О.А. Концептуальні засади комплексної оцінки інформаційних загроз у соціальних мережах: теоретико-математична модель. Наука і техніка сьогодні. 2026. №5. С. 5642-5653.

7. Kempe, D., Kleinberg, J., & Tardos, É. (2021). Influential nodes in a diffusion model for social networks. Journal of the ACM, 68(5), 1–31. DOI: 10.1145/3461476

8. Zhang, Q., Li, J., & Wang, Y. (2023). Power-law distributions in information cascade depth: Empirical evidence from Twitter. Social Networks, 72, 88–101. DOI: 10.1016/j.socnet.2022.10.004

9. Ferrara, E., & Yang, Z. (2022). Measuring emotional contagion in social

media. PLOS ONE, 17(4), e0267311. DOI: 10.1371/journal.pone.0267311

10. Granovetter, M. (2022). The strength of weak ties revisited: Network theory in the digital age. Annual Review of Sociology, 48, 21–45. DOI: 10.1146/annurev-soc-030921-121010

11. Fortunato, S., & Hric, D. (2022). Community detection in networks: A user guide updated. Physics Reports, 965, 1–86. DOI: 10.1016/j.physrep.2022.04.004

12. NetworkX Documentation. (2024). Algorithms: Diffusion and Cascades. URL: https://networkx.org/documentation/stable/reference/algorithms (дата звернення: 09.05.2026)

13. Santa Fe Institute. (2023). Complexity Explorer: Spreading Processes on Networks. URL: https://www.complexityexplorer.org/courses/97-introduction-to-complexity (дата звернення: 09.05.2026)

14. graph-tool Documentation. (2024). Efficient Network Analysis in Python/C++. URL: https://graph-tool.skewed.de/static/doc/index.html (дата звернення: 12.05.2026)

15. CasCN Project Repository. (2023). Cascade Classification and Hybrid Index. GitHub. URL: https://github.com/CasCN/cascade-analysis (дата звернення: 12.05.2026)

16. Twitter Developer Platform. (2024). Home Timeline Ranking Signals. URL: https://developer.twitter.com/en/docs/twitter-api (дата звернення: 12.05.2026)

17. Computational Social Science Repository. (2023). Information-Diffusion-Dynamics: Jupyter Notebooks. GitHub. URL: https://github.com/computationalsocialscience/diffusion-dynamics (дата звернення: 12.05.2026)

18. graph-tool Documentation. (2024). Graph views, filtering and sparse representations. URL: https://graph-tool.skewed.de/static/doc/graph_tool.html (дата звернення: 12.05.2026)

19. Wikipedia. (2024). Directed graph – Mathematics. URL: https://en.wikipedia.org/wiki/Directed_graph (дата звернення: 14.05.2026)

20. Goel, S., Anderson, A., Hofman, J., & Watts, D. J. (2022). The structural

virality of online diffusion. Management Science, 68(1), 180–196. DOI: 10.1287/mnsc.2021.4111

21. NDlib Documentation. (2023). Epidemic and Information Spreading Models. URL: https://ndlib.readthedocs.io/en/latest (дата звернення: 14.05.2026)

22. Wolfram MathWorld. (2024). Clustering Coefficient. URL: https://mathworld.wolfram.com/ClusteringCoefficient.html (дата звернення: 14.05.2026)

23. Barabási, A.-L. (2024). Network Science: Scale-Free Networks and Hubs. Barabási Lab. URL: https://barabasilab.com/network-science-book

24. Суліма В. К., Гончар В. М. Алгоритми визначення мінімальної кількості кольорів, для розфарбування графу. Збірник тез IV Всеукраїнської науково-практичної конференції студентів, аспірантів та молодих вчених «Прикладні інформаційні технології» (м. Вінниця, 18 липня 2023 року). Вінниця: ДонНУ імені Василя Стуса, 2023. С. 186-189.

25. Apache Spark. (2024). GraphX Programming Guide: Pregel API and Graph Algorithms. URL: https://spark.apache.org/docs/latest/graphx-programming-guide.html (дата звернення: 14.05.2026)

26. PyTorch Geometric Documentation. (2024). GNN Models: GCN, GAT, GraphSAGE. URL: https://pytorch-geometric.readthedocs.io/en/latest (дата звернення: 14.05.2026)

27. GraphSAINT. (2023). Graph Sampling Based Inductive Learning Method. GitHub. URL: https://github.com/GraphSAINT/GraphSAINT (дата звернення: 14.05.2026)

28. Twitter Developer Platform. (2024). API Access Levels and Rate Limits. URL: https://developer.twitter.com/en/docs/twitter-api/getting-started/about-twitter-api (дата звернення: 15.05.2026)

29. Playwright Documentation. (2024). Browser Automation and Anti-Bot Evasion. URL: https://playwright.dev/python/docs/intro (дата звернення: 15.05.2026)

30. BeautifulSoup Documentation. (2024). HTML and XML Parsing. URL:

https://beautiful-soup-4.readthedocs.io/en/lates (дата звернення: 15.05.2026)

31. NetworkX Documentation. (2024). DiGraph — Directed Graph Class. URL: https://networkx.org/documentation/stable/reference/classes/digraph.html (дата звернення: 15.05.2026)

32. FastAPI Documentation. (2024). Async and Await, Background Tasks. URL: https://fastapi.tiangolo.com/async (дата звернення: 15.05.2026)

33. React Documentation. (2024). Component Architecture and State Management. URL: https://react.dev/learn (дата звернення: 15.05.2026)

34. D3.js Documentation. (2024). Force-Directed Graphs and SVG Rendering. URL: https://d3js.org/d3-force (дата звернення: 15.05.2026)

35. NetworkX Developers. NetworkX Documentation. URL: https://networkx.org/documentation/stable/ (дата звернення: 15.05.2026)

36. Vosoughi S., Roy D., Aral S. The spread of true and false news online. Science. 2018. DOI: 10.1126/science.aap9559

37. Yang K.-C., Varol O., Davis C. A., Ferrara E., Flammini A., Menczer F. Arming the public with artificial intelligence to counter social bots. Human Behavior and Emerging Technologies. 2019. DOI: 10.1002/hbe2.115

38. Zhou X., Zafarani R. A Survey of Fake News: Fundamental Theories, Detection Methods, and Opportunities. ACM Computing Surveys. 2020. Vol. 53, No. 5. Article 109. DOI: https://doi.org/10.48550/arXiv.1812.00315

39. Liu Y., Li Y., Wang J., Liu T. Modeling Information Diffusion in Online Social Networks: A Survey. IEEE Access. 2021. Vol. 9. P. 64682–64702. DOI: https://doi.org/10.48550/arXiv.1704.03261

40. Pierri F., Luceri L., Jindal N., Ferrara E. Propaganda and Misinformation on Facebook and Twitter during the COVID-19 Pandemic. Social Network Analysis and Mining. 2022. Vol. 12, No. 1.

41. Nizzoli L., Avvenuti M., Cresci S., Tesconi M. Coordinated Behavior on Social Media in 2019 UK General Election. Applied Network Science. 2021. Vol. 6, No. 1. DOI:10.1609/icwsm.v15i1.18074

42. Ferrara E. The History of Digital Spam and Disinformation.

Communications of the ACM. 2023. Vol. 66, No. 1. P. 44–51.

43. Stella M., Ferrara E., De Domenico M. Bots increase exposure to negative and inflammatory content in online social systems. Proceedings of the National Academy of Sciences. 2018. Vol. 115, No. 49. P. 12435–12440. DOI: 10.1073/pnas.1803470115

44. Pierri F., Luceri L., Jindal N., Ferrara E. Propaganda and Misinformation on Facebook and Twitter during the Russian Invasion of Ukraine. Proceedings of the 15th ACM Web Science Conference 2023. DOI: 10.1145/3578503.3583597

45. Wang J., Li J., Liu C., Peng Y., Li X. Fusing temporal and structural information via subgraph sampling and multi-head attention for information cascade prediction. Scientific Reports. 2025. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC11861288/

46. Pacheco D., Flammini A., Menczer F. Uncovering Coordinated Networks on Social Media: Methods and Case Studies. Proceedings of the International AAAI Conference on Web and Social Media. 2021. Vol. 15. P. 455–466. DOI: https://doi.org/10.1609/icwsm.v15i1.18075

47. Newman M. Networks. Oxford : Oxford University Press, 2018. 840 p.

48. Abu-El-Haija S., Perozzi B., Kapoor A., Al-Rfou R., Lee J. MixHop: Higher-Order Graph Convolutional Architectures via Sparsified Neighborhood Mixing. Proceedings of the 36th International Conference on Machine Learning. 2019. P. 21–29. DOI: https://doi.org/10.48550/arXiv.1905.00067

49. Суліма В. К., Потапова Н. А. Технології збору даних із соціальних мережах для аналітичних систем. Збірник тез VII Всеукраїнської науково-практичної конференції здобувачів, аспірантів та молодих вчених «Прикладні інформаційні технології» (м. Вінниця, 15 травня 2026 року). Вінниця: ДонНУ імені Василя Стуса, 2026. С. 198-199.

50. Суліма В. К., Зелінська О. В. Сучасні тенденції Frontend мобільних додатків. Збірник тез V Всеукраїнської науково-практичної конференції здобувачів, аспірантів та молодих вчених «Прикладні інформаційні технології» (м. Вінниця, 20 лютого 2025 року). Вінниця: ДонНУ імені Василя Стуса, 2025 С. 87-88.

ДОДАТКИ

ДОДАТОК А

Зведена класифікація типів інформаційних каскадів

Критерій	Тип	Ключова ознака	Типова платформа	Метод виявлення
Топологія	Лінійний	$b = 1$, ланцюг	Месенджери	Path analysis
	Деревоподібний	$b > 1$, рекурсія	Twitter, Reddit	Tree decomposition
	Зіркоподібний	1 суперхаб	Instagram, YouTube	Hub detection
	Сітчастий	Цикли у графі	Новинні події	SCC-алгоритми
	Гібридний	$H \in (0,1)$	Будь-яка	Індекс H
Механізм	Соціального впливу	Порогова активація	Facebook	LT-модель
	Незалежних рішень	Ймовірнісна активація	Twitter	IC-модель
	Наслідування	Байєсівське оновлення	Фінансові платформи	Behavioral analysis
	Алгоритмічний	Рекомендаційна система	TikTok, YouTube	Feed analysis
Швидкість	Миттєвий	Пік < 1 год	Twitter	Peak detection
	Поступовий	Пік > 48 год	Reddit, блоги	Trend analysis
	Резонансний	Кілька піків	Будь-яка	Multi-peak fitting
Достовірність	Верифікований	Підтверджені факти	Новинні сайти	Fact-check API
	Дезінформація	Хибний контент	WhatsApp	Hoaxy
	Маніпулятивний	Скоординований	Twitter	BotSlayer
Джерело	Органічний	Звичайний користувач	Будь-яка	Origin tracking
	Скоординований	Група акаунтів	Twitter	DARPA SocialSim
	Медійний	Верифікований акаунт	Всі платформи	Verified badge filter

ДОДАТОК Б

Порівняльна характеристика органічного та скоординованого інформаційного каскаду

Ознака	Органічний інформаційний каскад	Скоординований інформаційний каскад
Характер поширення	Формується природно через інтерес користувачів до повідомлення	Може бути штучно ініційований або підсилений групою акаунтів
Початкова активність	Зростає поступово або залежить від актуальності теми	Часто має різкий старт одразу після публікації
Часові інтервали між взаємодіями	Нерівномірні, оскільки користувачі реагують у різний час	Можуть бути надто короткими або схожими між собою
Розподіл активності у часі	Активність розподіляється хвилями або поступово згасає	Значна частина реакцій зосереджується у короткому часовому проміжку
Глибина каскаду	Може поступово збільшуватися через природне перепоширення	Часто має невелику глибину при високій кількості одночасних взаємодій
Структура графа	Має різноманітну форму з природними гілками поширення	Часто централізується навколо одного або кількох ключових вузлів
Роль ключових вузлів	Впливові вузли з’являються внаслідок природної реакції аудиторії	Окремі вузли можуть виконувати роль координаторів або підсилювачів
Поведінка учасників	Користувачі мають різні часові та поведінкові шаблони	Групи акаунтів можуть демонструвати синхронність дій
Тип взаємодій	Взаємодії різноманітні: відповіді, цитування, обговорення, поширення	Можуть переважати однотипні реакції або шаблонне поширення
Ознаки раннього сплеску	Можливі у випадку актуальної або резонансної теми	Можуть бути непропорційно високими відносно природної активності
Ймовірність ботоподібної активності	Зазвичай нижча, хоча окремі підозрілі акаунти можуть бути присутні	Часто вища через залучення автоматизованих або напівавтоматизованих акаунтів