НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО”
Навчально-науковий фізико-технічний інститут
кафедра математичного моделювання та аналізу даних

«На правах рукопису»
УДК 519.854.3

«До захисту допущено»
Завідувач кафедри
_____ Н. М. Куссуль
(підпис) *(ініціали, прізвище)*
“____” _____ 2022 р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-науковою програмою “Математичні методи моделювання, розпізнавання образів та комп’ютерного зору”
зі спеціальності 113 «Прикладна математика»
на тему: «Алгоритм розв’язку супермодулярних $(max, +)$ задач розмітки з самоконтролем на базі субградієнтного спуску»

Виконав: студент 2 курсу групи ФІ-01мн
Хоменко Руслан Олександрович
_____
(підпис)
Науковий керівник: к.т.н., Водолазський Євгеній Валерійович
_____
(підпис)
Рецензент: к.т.н., зав. відділом, Мацелло Вячеслав Васильович
_____
(підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ — 2022 року

ABSTRACT

The thesis contains 62 pages, 7 figures and 22 references.

We present an algorithm that can give a correct answer to one of two questions for any $(\max, +)$ labeling problem: either "What is the best labeling?" or "Is the problem supermodular?" Moreover, we prove that for every supermodular problem the algorithm gives optimal labeling. The algorithm is called self-driven because a user cannot decide which of the two questions will be answered — this decision is up to the algorithm. Also, the algorithm does not need to know the order of labels if the problem is supermodular. In this work, we describe the algorithm for integer weights of vertices and edges and use subgradient descent to guarantee the finite time of execution.

$(\mathrm{MAX}, +)$ LABELING PROBLEMS, SUPERMODULAR LABELING PROBLEMS, SELF-DRIVEN PATTERN RECOGNITION, DISCRETE OPTIMIZATION, GRAPHICAL MODELS, STRUCTURAL PATTERN RECOGNITION.

РЕФЕРАТ

Дисертація містить 62 сторінки, 7 ілюстрацій і бібліографію з 22 найменувань.

Дану роботу присвячено алгоритму, який для будь-якої поданої на вхід $(\max, +)$ задачі розмітки з цілочисельними якостями надасть одну з двох відповідей: або оптимальну розмітку, або "задача не супермодулярна", і ця відповідь гарантовано буде коректною. Самоконтроль полягає у тому, що не користувач вирішує, на яке питання треба відповісти, а сам алгоритм вирішує, що потрапляє у зону його компетентності. Іншою особливістю алгоритму є те, що він не потребує відомої впорядкованості міток для супермодулярних задач. Гарантію скінченної кількості кроків надає використання субградієнтного спуску і цілочисельність ваг вершин та ребер.

$(\mathrm{MAX}, +)$ ЗАДАЧІ РОЗМІТКИ, СУПЕРМОДУЛЯРНІ ЗАДАЧІ РОЗМІТКИ, САМОКОНТРОЛЬ У РОЗПІЗНАВАННІ ОБРАЗІВ, ДИСКРЕТНА ОПТИМІЗАЦІЯ, ГРАФОВІ МОДЕЛІ, СТРУКТУРНЕ РОЗПІЗНАВАННЯ ОБРАЗІВ.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 9
Вступ 10
1 Аналіз попередніх алгоритмів розв'язку задач розмітки 12
1.1 Загальні відомості 12
1.2 Постановка $(\max, +)$ задачі розмітки 16
1.3 Еквівалентні перетворення 18
1.4 Представлення у вигляді задачі лінійного програмування 19
1.5 Властивості задачі 24
Висновки до розділу 1 27
2 Пошук оптимальної розмітки 28
2.1 Субградієнтний спуск 28
2.2 Викреслювання другого порядку 34
2.3 Відновлення розмітки після оптимізації 40
Висновки до розділу 2 43
3 Запропонований алгоритм точного розв'язку 44
3.1 Алгоритм розв'язку $(\max, +)$ задач розмітки 44
3.2 Властивості розв'язку 49
Висновки до розділу 3 51
4 Результати експериментів 53
4.1 Приклад застосування для не супермодулярної задачі 53
4.2 Результати експериментів 55
Висновки до розділу 4 57
Висновки 59
Перелік джерел посилань 60

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Стандартні позначення

$\mathbb{N}$ — множина натуральних чисел,

$\mathbb{Z}$ — множина цілих чисел,

$\mathbb{R}$ — множина дійсних чисел,

$\mathbb{N}_0$ — множина $\mathbb{N}$ з 0 ($\mathbb{N}_0 = \mathbb{N} \cup \{0\}$),

$\mathbb{R}_+$ — множина невід'ємних чисел з $\mathbb{R}$,

$\emptyset$ — порожня множина,

X^n — n-вимірний векторний простір над множиною X,

$\langle x, y \rangle$ — скалярний добуток векторів x,y,

$|X|$ — потужність множини X, або кардинальне число множини X,

$X \times Y$ — декартів добуток множин X та Y,

$X \cup Y$ — об'єднання множин X та Y.

Позначення, введені в дисертації

T — множина об'єктів,

K — множина міток,

Γ — структура сусідства,

N_t — множина усіх сусідів об'єкту t.

ВСТУП

Дана робота завдячує Кригіну Валерію Михайловичу — молодшому науковому співробітнику Відділу обробки та розпізнавання образів Міжнародного науково-навчального центру інформаційних технологій і систем НАН України та МОН України.

Роботу виконано у межах теми „Створення інтелектуальних інформаційних технологій на базі методів і засобів образного мислення”, державний реєстраційний номер 01114U002068.

Актуальність роботи. Задачі розмітки відіграють важливу роль у структурному розпізнаванні зображень. Одним з важливих класів задач розмітки, які мають ефективний розв’язок, є клас супермодулярних $(\max, +)$ задач з відомою та невідомою впорядкованістю міток. Ми наводимо алгоритм розв’язку задач, який гарантовано за скінченну кількість кроків видасть правильну відповідь, а також є швидшим за існуючі методи.

Мета і завдання дослідження.

Об’єкт дослідження — $(\max, +)$ задачі розмітки.

Предмет дослідження — точний розв’язок задач розмітки.

Метою роботи є створення алгоритму, який розв’язує задачі класу $(\max, +)$ та є швидшим за існуючі алгоритми.

Завдання наступні:

1) проаналізувати існуючі алгоритми розв’язку $(\max, +)$ задач розмітки,
2) розробити алгоритм розв’язку $(\max, +)$ задач розмітки, який буде швидшим за існуючі методи з гарантією збіжності за скінченну кількість кроків,
3) реалізувати алгоритм програмно та провести експерименти.

Наукова новизна одержаних результатів.

Створено алгоритм розв’язку $(\max, +)$ задач розмітки, який швидший за існуючі аналоги та має гарантію збіжності за скінченну кількість кроків.

Практичне значення одержаних результатів.

За допомогою алгоритму можна отримати розв'язок для будь-яких супермодулярних і деяких не супермодулярних $(\max, +)$ задач розмітки за скінченну кількість кроків, а для інших задач отримати відмову від розпізнавання.

Публікації.

Стаття "Алгоритм розв'язку супермодулярних $(\max, +)$ задач розмітки з самоконтролем на базі субградієнтного спуску" пройшла повний цикл рецензування, доопрацювання та схвалена для публікації у журналі „Кібернетика та системний аналіз" у № 4, 2022 р.

1 АНАЛІЗ ПОПЕРЕДНІХ АЛГОРИТМІВ РОЗВ'ЯЗКУ ЗАДАЧ РОЗМІТКИ

Перший розділ присвячено огляду існуючих підходів та методів розв'язування супермодулярних задач розмітки. Даний опис дає змогу краще зрозуміти проблематику та складнощі, які треба вирішити, а також розглядаються обмеження та недоліки існуючих методів.

1.1 Загальні відомості

Структурне розпізнавання зображень зводиться до розв'язання специфічних задач дискретної оптимізації [1–4]. Незважаючи на велике різноманіття даних задач, весь клас таких задач можна представити як $(\max, +)$ задачі розмітки. До відшукання оптимальних розміток $(\max, +)$ задачі зводиться багато прикладних задач структурного розпізнавання образів. Серед них сегментація, стереозір, знешумлення, бінаризація та багато інших [3, 5, 6].

Далеко не завжди є можливість знайти точний розв'язок до масштабних задач оптимізації. Всі прикладні задачі відносяться до цього класу. За певних умов точний розв'язок можливий. Наприклад, алгоритм динамічного програмування знаходить глобальний мінімум задачі за умови, якщо граф є ациклічною структурою [7]. Для ациклічних моделей складність алгоритму росте пропорційно розміру задачі. На жаль, не для всіх прикладних задач можливий такий розв'язок. Більш широким класом розв'язних задач є супермодулярні задачі. Супермодулярність для $(\max, +)$ задач розмітки — це свого роду „аналог" опуклості функції в теорії оптимізації. За таких умов існують алгоритми, які дозволяють знайти точний розв'язок задачі за скінченну кількість кроків. Відмітимо, що будь-яка задача на ациклічній структурі є супермодулярною.

1.1.1 Частковий випадок на дві мітки

Одним із перших алгоритмів розв'язування є алгоритм [8], який дозволяє знаходити точний розв'язок $(\max, +)$ задачі розмітки при обмеженні, що задача має лише 2 мітки. Прикладом такої задачі може бути задача знешумлення бінарного зображення. Ідея згаданого алгоритму полягає в тому, щоб представити задачу як орієнтований граф спеціального виду таким чином, щоб мінімальний зріз цього графу відповідав розв'язку прямої задачі розмітки. В теорії графів існують алгоритми, які можуть знайти розв'язок за поліноміальний час [2, 3, 9]. Суттєвим обмеженням алгоритму є те, що задача повинна містити лише 2 мітки. На жаль, для більшості прикладних задач цього недостатньо.

1.1.2 Зведення задачі до пошуку максимального потоку

Підхід [10] є у певному сенсі узагальненням попереднього алгоритму на довільну скінченну кількість міток. Додатковою умовою застосування є відома впорядкованість множини міток. Як і у попередньому методі, основна ідея полягає в побудові спеціального орієнтованого графу, який відповідає $(\max, +)$ задачі розмітки. Розв'язком буде значення мінімального розрізу цього графу.

1.1.3 Перетворення k в 2

Перетворення k в 2 [11] дає можливість зводити супермодулярні задачі із довільною кількістю міток до задач із двома мітками. Суть підходу полягає в тому, що будь-яку $(\max, +)$ задачу розмітки можна представити як бінарну задачу, а далі використати найменший розріз для знаходження точного розв'язку. Отже, всі отримані результати для бінарних задач узагальнюються до загального випадку $(\max, +)$ задачі розмітки із довільною кількістю міток.

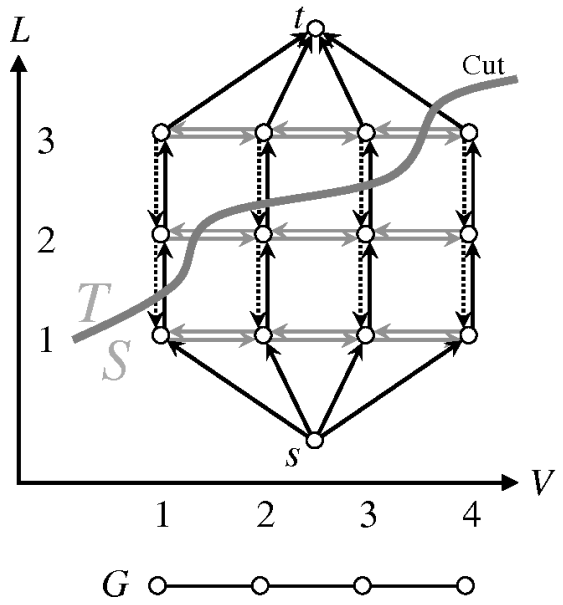


Рисунок 1.1 — Приклад графу [10]

1.1.4 Ітеративні методи

Інший спосіб розв'язання $(\max, +)$ задач розмітки полягає у застосуванні ітеративних алгоритмів. Часто ітеративні алгоритми застосовуються до цільової функції двоїстої задачі оптимізації [1, 12]. Перевагою даного підходу є те, що ітеративні алгоритми не накладають жорстких умов на вхідні дані задачі, тому множина прикладних задач, до яких вони застосовні, розширюється. Варто визнати, що не всі ітеративні методи гарантовано дають точний розв'язок. Деякі з них наближаються до точного розв'язку лише у ліміті, що є недоліком з практичної точки зору.

Мінімізація цільової функції двоїстої задачі є доволі універсальним способом відшукання найкращої нечіткої розмітки для довільної задачі [1, 4, 12]. Проте на даний момент існують лише окремі спроби вирішити цю задачу, а практично гарні алгоритми невідомі. Однією з таких спроб є алгоритм дифузії [4, 12]. Відомо, що за скінченну кількість кроків алгоритм дифузії для будь-якої супермодулярної задачі знайде рішення, яке відрізняється від найкращого не більше, ніж на ε, для будь-якого наперед заданого $\varepsilon > 0$.

Алгоритм дифузії дозволяє розв'язувати широкий клас задач, який містить всі

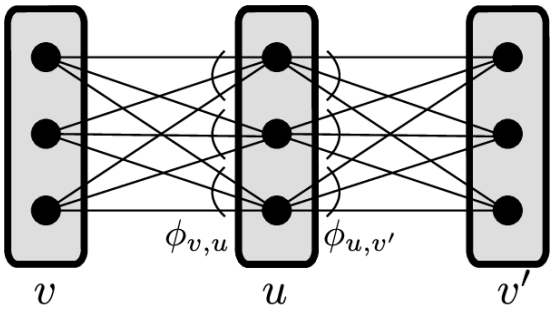


Рисунок 1.2 — 1 крок алгоритму дифузії — оптимізація за блоком змінних [10]

супермодулярні задачі. Проте немає гарантії знаходження мінімального значення цільової функції двоїстої задачі. Цим недоліком володіють також методи, що засновані на субградієнтній оптимізації [13].

Для субградієнтної оптимізації є додаткова складність у необхідності визначення умови зупинки алгоритму та способі відшукання оптимальної розмітки після досягнення оптимуму. Обидві ці проблеми частково вирішуються в [14].

1.1.5 Висновки

У даному розділі описано методи, які по-різному підходять до розв'язування $(\max, +)$ задач розмітки. Звичайно, найбільш бажаними є методи, які знаходять точний розв'язок задачі, але, як бачимо, вони всі мають недоліки: додаткові обмеження, які накладаються на вхідні умови задачі, або розв'язок лише обмеженого класу задач, що на практиці часто звужує коло застосовності цих методів.

На противагу прямим методам розв'язування ітеративні методи часто дозволяють знаходити розв'язок до більш широкого кола задач, не накладаючи додаткових обмежень. Недоліками цих методів часто є їх повільність та відсутність будь-яких гарантій на відшукання точного розв'язку, а також проблема відшукання оптимальної розмітки після досягнення прийнятного значення цільової функції. Для даних алгоритмів це є окремою задачею, яка часто є обчислювально більш складною, ніж сама оптимізація цільової функції.

Нові методи розв'язування такого класу задач часто є спробами або розши-

рити клас задач, які розв'язуються точно, або пришвидшити та вдосконалити ітеративні методи, надати їм теоретичні гарантії на відшукання прийнятного розв'язку.

1.2 Постановка $(\max ,+)$ задачі розмітки

Задано скінченну непорожню множину T об'єктів та скінченну непорожню множину K міток. Функцію $k : T \to K$ будемо називати розміткою, що для кожного об'єкту $t \in T$ визначає мітку $k(t) \in K$. На множині t об'єктів визначимо структуру сусідства $\Gamma \subset T^2$, яка є асиметричною

$$(t,t') \in \Gamma \implies (t',t) \notin \Gamma.$$

Надалі буде використовуватися запис tt' замість (t,t'). Множину всіх сусідів об'єкту t будемо позначати

$$N_t = \{t' : tt' \in \Gamma \cup \Gamma^{-1}\}.$$

Впорядковану пару (t, k), $t \in T$, $k \in K$ будемо називати вершиною. Для кожної пари сусідів $tt' \in \Gamma$ пару $((t,k),(t',k'))$, $k \in K$, $k' \in K$ будемо називати ребром. Будемо казати, що вершина (t^*, k^*) належить розмітці k, якщо $k(t^*) = k^*$. Ребро $((t^*,k^*),(t^{**},k^{**}))$ належить розмітці k, якщо обидві вершини (t^*,k^*) та (t^{**},k^{**}) належать k.

Введемо цілочисельну функцію $q : T \times K \to \mathbb{Z}$ якостей вершин, цілочисельну функцію $g : \Gamma \times K^2 \to \mathbb{Z}$ якостей ребер. Будемо позначати якості наступним чином: $q_t(k)$ — якість вершини (t,k), $g_{tt'}(k,k')$ — якість ребра $((t,k)),(t',k')$. Якістю розмітки будемо називати функцію $G : K^T \to \mathbb{Z}$

$$G(k) = \sum_{t \in T} q_t(k_t) + \sum_{tt' \in \Gamma} g_{tt'}(k_t,k_{t'}).$$

Якість розмітки — це сума якостей всіх вершин та ребер, які їй належать.

Задача відшукання найкращої строгої розмітки полягає в тому, щоб знайти розмітку k^*

$$k^* \in \arg\max_{k \in K^T} G(k). \tag{1}$$

Складність задачі полягає в тому, що вона є задачею дискретної оптимізації, в якій для кожного об'єкта необхідно вибрати лише одну мітку. Послабимо цю умову і для кожного об'єкта будемо обирати "суміш" міток.

Для кожних $t \in T$ і $k \in K$ введемо $\alpha_t(k) \in \mathbb{R}$, яке ми будемо називати вагою вершини (t,k), а для кожних $tt' \in \Gamma$, $k \in K$ і $k' \in K$ введемо $\beta_{tt'}(k,k') \in \mathbb{R}$, яке будемо називати вагою ребра $((t,k),(t',k'))$. Позначимо α — набір $(\alpha_t(k)|t \in T, k \in K)$ ваг вершин, β — набір $(\beta_{tt'}(k,k')|tt' \in \Gamma, k \in K, k' \in K)$ ваг ребер. Пару (α, β) будемо називати ваговою функцією. Вагова функція називається нечіткою розміткою, якщо вона задовольняє

$$\begin{cases} \alpha_t(k) = \sum\limits_{k' \in K} \beta_{tt'}(k,k'), & t \in T, k \in K, t' \in N_t, \\ \sum\limits_{k \in K} \alpha_t(k) = 1, & t \in T, \\ \beta_{tt'}(k,k') \geq 0, & tt' \in \Gamma, k \in K, k' \in K. \end{cases}$$

Якість нечіткої розмітки

$$G(\alpha, \beta) = \sum_{t \in T} \sum_{k \in K} \alpha_t(k) \cdot q_t(k) + \sum_{tt' \in \Gamma} \sum_{k \in K} \sum_{k' \in K} \beta_{tt'}(k,k') \cdot g_{tt'}(k,k').$$

Задача нечіткої розмітки полягає у тому, щоб знайти розмітку з найкращою якістю

$$(\alpha^*, \beta^*) = \arg\max_{(\alpha,\beta)} G(\alpha, \beta). \tag{2}$$

Неважко побачити, що, якщо обмежити значення $\alpha_t(k)$ та $\beta_{tt'}(k,k')$ лише цілими числами, то задача (2) стає еквівалентною задачі пошуку строгої розмітки (1). В

цьому випадку $\alpha_t(k') = 1$ в задачі нечіткої розмітки означає, що $k(t) = k'$ в задачі строгої розмітки.

1.3 Еквівалентні перетворення

Існує достатня умова оптимальності нечіткої розмітки. Якщо нечітка розмітка (α, β) задовольняє умовам

$$\begin{cases} q_t(k) < \max\limits_{\ell \in K} q_t(\ell), & \Longrightarrow \ \alpha_t(k) = 0, \\ g_{tt'}(k,k') < \max\limits_{\ell \in K, \ell' \in K} g_{tt'}(k,k'), & \Longrightarrow \ \beta_{tt'}(k,k') = 0, \end{cases}$$

то розмітка є оптимальною нечіткою розміткою. Дана умова є дуже строгою, тому лише невеликий клас задач підпадає під цю умову. Задачі, які задовольняють цю умову, називаються тривіальними. Можна послабити умову (1.3), використовуючи концепцію еквівалентних перетворень [1, 15]. Дві задачі (q^1,g^1) та (q^2,g^2) нечіткої розмітки, що визначені на однаковій множині об'єктів T та однаковій множині міток K, називаються еквівалентними (позначаємо $(q^1,g^1) \sim (q^2,g^2)$), якщо якості кожної нечіткої розмітки однієї задачі дорівнюють якості цієї ж нечіткої розмітки іншої задачі. Також відомо, що для кожної задачі нечіткої розмітки (q,g) існує тривіальний еквівалент. Задача (q^*,g^*), за якої значення

$$E(q',g') = \sum_{tt' \in \Gamma} \max_{k \in K, k' \in K} g'_{tt'}(k,k') + \sum_{t \in T} \max_{k \in K} q'_t(k),$$

є мінімальним з можливих, називається потужністю класу еквівалентності

$$(q^*,g^*) = \operatorname*{arg\,min}_{(q',g') \sim (q,g)} E(q',g').$$

Дві задачі розмітки із якостями q^1, g^1 і q^2, g^2 відповідно є еквівалентними тоді й тільки тоді, коли існує такий набір чисел $\varphi_{tt'}(k), t \in T, t' \in N(t), k \in K$, який задовольняє систему рівностей

$$\begin{cases} q_t^1(k) = q_t^2(k) - \sum\limits_{t' \in N(t)} \varphi_{tt'}(k), & t \in T, k \in K, \\ g_{tt'}^1(k,k') = g_{tt'}^2(k,k') + \varphi_{tt'}(k) + \varphi_{t't}(k), & tt' \in \Gamma, k \in K, k' \in K. \end{cases}$$

Таким чином, потужність еквівалентно трансформованої задачі може бути явно виражена як функція φ

$$E(\varphi) = \sum_{tt' \in \Gamma} \max_{k \in K, k' \in K} [g_{tt'}(k,k') + \varphi_{tt'}(k) + \varphi_{t't}(k)] + \sum_{t \in T} \max_{k \in K} [q_t(k) - \sum_{t' \in N(t)} \varphi_{tt'}(k)], \quad (3)$$

і зведення задачі до тривіальної може бути виконано шляхом мінімізації (3) без жодних обмежень на φ.

1.4 Представлення у вигляді задачі лінійного програмування

Представимо $(\max, +)$ задачу розмітки як задачу лінійного програмування [4, 16]. Для кожної вершини (t,k), $t \in T$, $k \in K$ введемо число $\alpha_t(k) \in [0,1]$. Для кожного ребра $(t,k),(t',k')$, $t \in T$, $k \in K$, $t' \in N_t$, $k' \in K$, яке поєднує мітки k і k' в сусідніх об'єктах t і t', введемо число $\beta_{tt'}(k,k')$.

Елементи вектору $\alpha_t(k)$, що відповідають вершинам, мають бути узгодженими з з розміткою k, тобто, якщо виконується $\alpha_t(k^*) = 1$ для якоїсь мітки $k^* \in K$, то має також виконуватися $k(t) = k^*$. За визначенням для задання розмітки $k \in K^T$ потрібно в кожному об'єкті $t \in T$ обрати єдину мітку $k \in K$. Тому на елементи ве-

ктора $\alpha_t(k)$ накладаються обмеження однозначності для кожної вершини в об'єкті

$$\sum_{k \in K} \alpha_t(k) = 1, t \in T.$$

Відмітимо, що для випадку $\alpha_t(k) \in \{0,1\}$ ця умова означає вибір єдиної мітки в об'єкті, бо для обраної мітки k^* буде виконуватися $\alpha_t(k^*) = 1$, а для всіх інших міток $k' \in K : k' \neq k^*$

$$\sum_{k \in K, k \neq k^*} \alpha_t(k) = 0.$$

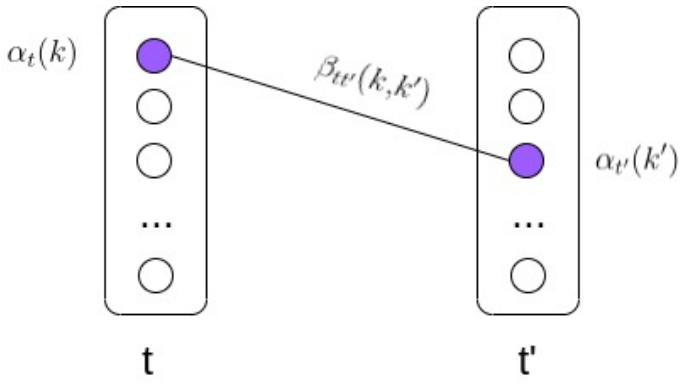


Рисунок 1.3 — Поєднуючі обмеження

Нехай об'єкт $t' \in T$ є сусідом об'єкту $t \in T$, тобто $tt' \in \Gamma$. Якщо в об'єкті t' була обрана мітка $k' \in K$, то має існувати таке ребро, яка поєднує якусь із вершин об'єкту $t' \in T$ і вершину (t',k'), тобто

$$\sum_{k \in K} \beta_{tt'}(k,k') = \alpha_{t'}(k'), \forall t \in T, t' \in N_t, k' \in K.$$

Обмеження такого виду називають поєднуючими (рис. 1.3). Якщо $\beta_{tt'}(k,k') \in \{0,1\}$, $\forall tt' \in \Gamma, k \in K, k' \in K$, то з цих обмежень також випливає, що між двома сусідніми об'єктами $tt' \in \Gamma$ може бути обране лише одне ребро, тобто додатково накладаються

обмеження однозначності для ребер між парами сусідніх об'єктів

$$\sum_{k\in K,k'\in K} \beta_{tt'}(k,k') = 1, \forall tt' \in \Gamma.$$

Отримали наступну множину обмежень

$$L \equiv \begin{cases} \sum\limits_{k'\in K} \beta_{tt'}(k,k') = \alpha_t(k), & t \in T, k \in K, t' \in N_t, \\ \sum\limits_{k\in K} \alpha_t(k) = 1, & t \in T, \\ \beta_{tt'}(k,k') \geq 0, & tt' \in \Gamma, k \in K, k' \in K, \\ \alpha_t(k) \geq 0, & t \in T, k \in K. \end{cases}$$

Позначимо множину всіх вершин і ребер графу як I

$$I = \{(t,k) : t \in T, k \in K\} \cup \{((t,k),(t',k')) : t \in T, t' \in \Gamma, k \in K, k' \in K\}.$$

Позначимо множину всіх якостей задачі як θ

$$\theta = \{q_t(k) : t \in T, k \in K\} \cup \{g_{tt'}(k,k') : t \in T, t' \in \Gamma, k \in K, k' \in K\}.$$

Множину всіх чисел α, β позначимо як

$$\mu = \{\alpha_t(k) : t \in T, k \in K\} \cup \{\beta_{tt'}(k,k') : t \in T, t' \in \Gamma, k \in K, k' \in K\}.$$

Тоді $(\max, +)$ задачу розмітки можна представити як задачу лінійного програмування

$$\max_{\mu\in L\cap\{0,1\}} \langle\theta,\mu\rangle.$$

Легко побачити, що виконується рівність

$$\max_{k\in K^T} G(k) = \max_{\mu\in L\cap\{0,1\}} \langle\theta,\mu\rangle.$$

Розпишемо скалярний добуток правої частини рівності (1.4) у явному вигляді

$$\begin{gathered}\max_{\mu\in L\cap\{0,1\}} \langle\theta,\mu\rangle = \\ = \max_{\mu\in L\cap\{0,1\}} \left[\sum_{t\in T}\sum_{k\in K} \alpha_t(k)\cdot q_t(k) + \sum_{tt'\in\Gamma}\sum_{k,k'\in K} \beta_{tt'}(k,k')\cdot g_{tt'}(k,k')\right].\end{gathered} \tag{4}$$

Дуалізуємо поєднуючі обмеження задачі (4). Новий доданок буде мати вигляд

$$\sum_{t\in T}\sum_{t'\in N_t}\sum_{k\in K} \varphi_{tt'}(k)\cdot\left[\sum_{t'\in T}\beta_{tt'}(k,k') - \alpha_t(k)\right], \tag{5}$$

де змінні $\varphi_{tt'}(k)\in\mathbb{R}$, $t\in T$, $t'\in N_t$, $k\in K$ є двоїстими. В подальшому будемо називати їх потенціалами. Перепишемо функцію (4) з урахуванням нового доданку (5), в якому розкриємо дужки

$$\begin{gathered}\langle\theta^{\varphi},\mu\rangle = \sum_{t\in T}\sum_{k\in K} \alpha_t(k)\cdot q_t(k) + \sum_{tt'\in\Gamma}\sum_{k,k'\in K} \beta_{tt'}(k,k')\cdot g_{tt'}(k,k') + \\ \sum_{t\in T}\sum_{t'\in N_t}\sum_{k\in K} \varphi_{tt'}(k)\cdot\sum_{t'\in T}\beta_{tt'}(k,k') - \sum_{t\in T}\sum_{t'\in N_t}\sum_{k\in K}\varphi_{tt'}(k)\cdot\alpha_t(k).\end{gathered}$$

Згрупуємо доданки в такому порядку: перший і останній, другий і третій

$$\begin{gathered}\langle\theta^{\varphi},\mu\rangle = \sum_{t\in T}\sum_{k\in K}\alpha_t(k)\cdot\left[q_t(k) - \sum_{t'\in N_t}\varphi_{tt'}(k)\right] + \\ \sum_{tt'\in\Gamma}\sum_{k,k'\in K}\beta_{tt'}(k,k')\cdot\left[g_{tt'}(k,k') + \varphi_{tt'}(k) + \varphi_{t't}(k')\right].\end{gathered} \tag{6}$$

Введемо позначення для репараметризованої якості мітки $k \in K$ в об'єкті $t \in T$

$$q_t^{\varphi}(k) = q_t(k) - \sum_{t' \in N_t} \varphi_{tt'}(k). \tag{7}$$

Репараметризована якість у вершині отримується шляхом віднімання потенціалів, що виходять з даної вершини (t,k), $t \in T$, $k \in K$ в усі сусідні об'єкти $t' \in N_t$, від вихідної якості в даній вершині. Введемо позначення для репараметризованої якості вибору пари міток $k \in K$, $k' \in K$ у двох сусідніх об'єктах $tt' \in \Gamma$

$$g_{tt'}^{\varphi}(k,k') = g_{tt'}(k,k') + \varphi_{tt'}(k) + \varphi_{t't}(k'), \tag{8}$$

тобто репараметризована якість за вибір ребра $((t,k),(t',k')$, $t \in T$, $t' \in N_t$, $k \in K, k' \in K)$ отримується шляхом додавання потенціалів, що виходять в об'єкти t і t', які дане ребро поєднує, до вихідної якості даного ребра. Використаємо позначення у виразі (6)

$$\langle \theta^{\varphi}, \mu \rangle = \sum_{t \in T} \sum_{k \in K} \alpha_t(k) \cdot q_t^{\varphi}(k) + \sum_{tt' \in \Gamma} \sum_{k,k' \in K} \beta_{tt'}(k,k') \cdot g_{tt'}^{\varphi}(k,k').$$

Твердження. Після перетворень (7) та (8) значення штрафної функції не зміниться для будь-якої розмітки $k \in K^T$.

Для доведення цього твердження запишемо штрафну функцію з репараметризованими якостями

$$G^{\varphi}(k) = \sum_{t \in T} q_t^{\varphi}(k) + \sum_{tt' \in \Gamma} g_{tt'}^{\varphi}(k,k') =$$

$$= \sum_{t \in T} \left[q_t(k) - \sum_{t' \in N_t} \varphi_{tt'}(k) \right] + \sum_{tt' \in \Gamma} \left[g_{tt'}(k,k') + \varphi_{tt'}(k) + \varphi_{t't}(k') \right].$$

Розкриємо дужки в останньому виразі

$$G^{\varphi}(k) = \sum_{t \in T} q_t(k) - \sum_{t \in T} \sum_{t' \in N_t} \varphi_{tt'}(k) +$$
$$+ \sum_{tt' \in \Gamma} g_{tt'}(k,k') + \sum_{tt' \in \Gamma} [\varphi_{tt'}(k) + \varphi_{t't}(k')] = G(k),$$

тому що перший і третій доданки в сумі дають $G(k)$, а другий і четвертий доданки в сумі дорівнюють нулю. Отримали

$$G^{\varphi}(k) = G(k), \forall k \in K^T.$$

Маємо двоїсту задачу, цільову функцію якої будемо мінімізувати

$$\min_{\Phi} \max_{\mu \in L \cap \{0,1\}} \langle \theta, \mu \rangle,$$

де

$$\Phi = \{\varphi_{tt'}(k) \in \mathbb{R} | t \in T, t' \in N_t, k \in K\}.$$

Отримали остаточний вигляд двоїстої цільової функції, яку будемо мінімізувати по набору двоїстих змінних φ.

1.5 Властивості задачі

Задача розмітки називається супермодулярною, якщо існує таке відношення $n : T \times K \to 1, \ldots, |K|$, що для будь-яких значень $k \in K$, $k' \in K$, $\ell \in K$, $\ell' \in K$ і $tt' \in \Gamma$ з умов $n_t(k) \geq n_t(\ell)$ та $n'_t(k') \geq n'_t(\ell')$ випливає

$$g_{tt'}(k,k') + g_{tt'}(\ell,\ell') \geq g_{tt'}(k,\ell') + g_{tt'}(\ell,k').$$

Ребра $((t,k),(t',k'))$ і $((t,\ell),(t',\ell'))$ називають паралельними, а ребра $((t,\ell),(t',k'))$ називають перехресними, тому наведену нерівність можна інтерпретувати наступним чином: сума якостей паралельних ребер є не гіршою за суму якостей перехресних ребер (рис. 1.4). При цьому на функцію q обмежень не накладається.

Розглядаються два випадки. Якщо задача є супермодулярною і відображення n відоме, задача називається супермодулярною з відомою впорядкованістю. Коли відомо, що існує таке n, проте саме відображення невідоме, задача називається супермодулярною з невідомою впорядкованістю. Пряма перевірка супермодулярності

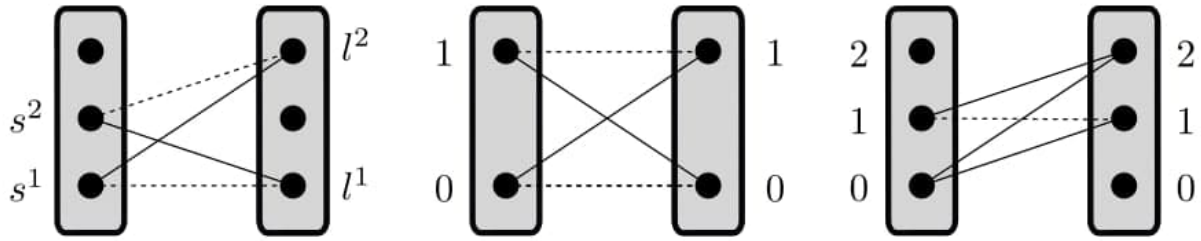


Рисунок 1.4 — Властивість супермодулярності для ребер [10]

є досить складною задачею. Для однієї пари об'єктів $tt' \in \Gamma$ складність перевірки $\mathcal{O}(|K^2|)$, а перевірка всієї задачі на супермодулярність — $\mathcal{O}(|K^2|\cdot|\Gamma|)$. Така перевірка стає проблемою у випадку, коли множини T, K досить великі, що не є рідкістю на практиці.

Досить багато штрафів можна представити у вигляді супермодулярних функцій. Наведемо деякі з них.

Модель Ізінга

$$g_{tt'}(k,k') = \lambda_{tt'} \cdot [\![t \neq t']\!]$$

для деяких констант $\lambda_{tt'}$, $tt' \in \Gamma$, $k \in K$, $k' \in K$, $|K| = 2$. Можна показати, що використовуючи репараметризацію, будь-яку бінарну функцію якостей можна перетворити в форму моделі Ізінга.

Модель Поттса є узагальненням моделі Ізінга для випадку, коли $|K| \geq 2$.

Бінарні якості мають аналогічний вигляд

$$g_{tt'}(k,k') = \lambda_{tt'} \cdot [\![t \neq t']\!].$$

Моделі Потса та Ізінга часто використовуються, коли задачі мають дискретний характер. Іноді ж потрібно на лише штрафувати за неправильну мітку, а штрафувати за за неправильну мітку пропорційно відстані до правильної. Для цього також необхідно, щоб на множині K існував порядок.

Пропорційні відстані

$$g_{tt'}(k,k') = |k - k'|^n, n \geq 2.$$

Прикладами задач, де можуть застосовуватися штрафи такого типу: сегментація, стереозір, постеризація, домальовування (inpainting), та багато інших.

Твердження. Сума 2 супермодулярних функцій $g^1_{tt'}(k,k')$ та $g^2_{tt'}(k,k')$ також є супермодулярною

$$g_{tt'}(k,k') = g^1_{tt'}(k,k') + g^2_{tt'}(k,k').$$

Твердження. Репараметризація не впливає на властивість супермодулярності, тобто, якщо функція була супермодулярною, після репараметризації вона також буде супермодулярною. Розглянемо довільну пару сусідів $tt' \in \Gamma$ і функцію якостей $g_{tt'}(k,k'), k \in K, k' \in K$. Якщо $g_{tt'}(k,k')$ є супермодулярною, то

$$g^{\varphi}_{tt'}(k,k') = g_{tt'}(k,k') + \varphi_{tt'}(k) + \varphi_{t't}(k')$$

також є супермодулярною. Треба перевірити чи виконується нерівність

$$g^{\varphi}_{tt'}(k,k') + g^{\varphi}_{tt'}(\ell,\ell') \geq g^{\varphi}_{tt'}(k,\ell') + g^{\varphi}_{tt'}(\ell,k').$$

Розпишемо репараметризовані якості

$$g_{tt'}(k,k') + \varphi_{tt'}(k) + \varphi_{t't}(k') + g_{tt'}(\ell,\ell') + \varphi_{tt'}(\ell) + \varphi_{t't}(\ell') \geq$$
$$\geq g_{tt'}(k,\ell') + \varphi_{tt'}(k) + \varphi_{t't}(\ell') + g_{tt'}(\ell,k') + \varphi_{tt'}(\ell) + \varphi_{t't}(k').$$

Перенесемо всі доданки з φ в одну частину

$$g_{tt'}(k,k') + g_{tt'}(\ell,\ell') \geq g_{tt'}(k,\ell') + g_{tt'}(\ell,k') +$$
$$+[\varphi_{tt'}(k) + \varphi_{t't}(\ell') + \varphi_{tt'}(\ell) + \varphi_{t't}(k') -$$
$$-\varphi_{tt'}(k) - \varphi_{t't}(k') - \varphi_{tt'}(\ell) - \varphi_{t't}(\ell')].$$

Видно, що всі доданки з φ скорочуються і потрібна рівність виконується.

Висновки до розділу 1

В розділі було розглянуто основні методи розв'язування $(\max, +)$ задач розмітки. Було описано переваги та недоліки кожного з них. Алгоритми, які розв'язують задачу точно, є більш бажаними, але часто вони вирішують задачі лише обмеженого класу (задачі з двома мітками, супермодулярні задачі із відомою впорядкованістю міток тощо). Ітеративні методи дозволяють розв'язувати значно ширший клас задач. Основними недоліками таких алгоритмів є їх повільність та відсутність будь-яких гарантій на відшукання точного розв'язку, а також проблема відшукання оптимальної розмітки після досягнення прийнятного значення цільової функції.

Наведено постановку $(\max, +)$ задачі розмітки та показано перехід до двоїстої задачі. Розглянуто властивість супермодулярності. Ці дані є необхідними для конструювання алгоритму у наступному розділі дисертації.

2 ПОШУК ОПТИМАЛЬНОЇ РОЗМІТКИ

Другий розділ присвячено задачам пошуку оптимальної розмітки. Загалом, дану процедуру можна представити двома етапами. Спочатку проводиться оптимізація цільової функції, а потім відновлюється розмітка, для якої був досягнутий оптимум.

2.1 Субградієнтний спуск

Цільовою функцією, яку ми будемо оптимізувати, є функція $E(\varphi)$

$$\min_{\varphi \in \Phi} E(\varphi).$$

$$E(\varphi) = \sum_{tt' \in \Gamma} \max_{k \in K, k' \in K} g^{\varphi}_{tt'}(k,k') + \sum_{t \in T} \max_{k \in K} q^{\varphi}_{t}(k) =$$

$$= \sum_{tt' \in \Gamma} \max_{k \in K, k' \in K} [g_{tt'}(k,k') + \varphi_{tt'}(k) + \varphi_{t't}(k)] + \sum_{t \in T} \max_{k \in K} \left[q_t(k) - \sum_{t' \in N(t)} \varphi_{tt'}(k) \right].$$

Функція E є випуклою по змінній φ, проте вона є кусково-лінійною, тому її розв'язок можна шукати за допомогою субградієнтного методу.

Субградієнтні методи [4, 13, 14] — це ітеративні методи вирішення задач опуклої оптимізації. Методи є збіжними навіть тоді, коли коли застосовуються навіть до недиференційованої цільової функції. Субградієнтний спуск — це узагальнення алгоритму градієнтного спуску [17] для задач з недиференційованою цільовою функцією.

Приведемо основні етапи оптимізації методом. Нехай $f : \mathbb{R}^n \to \mathbb{R}$ є опуклою (не обов'язково диференційованою) функцією з доменом $\mathbb{R}^n$. Субградієнтний метод

полягає у ітеративному повторенні кроків

$$x^{(k+1)} = x^{(k)} - \alpha_k \cdot g^{(k)},$$

де $g^{(k)}$ є одним із субградієнтів функції f в точці $x^{(k)}$, а k — номер ітерації($k \geq 0$). Відмітимо, що якщо функція f є диференційованою, то субградієнт буде співпадати із градієнтом. Також потрібно обрати крок субградієнта α.

Існують декілька найбільш розповсюджених способів вибору кроку.

- Константний крок $\alpha_k = \alpha$, найменш розповсюджений, так як при наближенні до оптимуму, при невдало обраному α подальший спуск буде нестабільним.
- Кроки, які задовольняють умовам $\alpha_k \geq 0$, а також щоб ряд із квадратів кроків сходився, а ряд кроків розходився, тобто

$$\sum_{k=1}^{\infty} \alpha_k^2 < \infty, \sum_{k=1}^{\infty} \alpha_k = \infty.$$

- Послідовність спадаючих кроків, які збіжні в границі, але їх сума розбіжна, тобто

$$\lim_{k\to\infty} \alpha_k = 0, \sum_{k=1}^{\infty} \alpha_k = \infty.$$

Найбільш розповсюджений — останній варіант вибору кроку субградієнта, тому в подальшому будемо використовувати саме його. Розглянемо найбільш загальний метод субградієнтного спуску у контексті використання оптимізації цільової функції $(\max, +)$ задачі розмітки.

Ініціалізація

1) Визначимо послідовність кроків субградієнтного спуску як послідовність чисел γ_i, такі, що

$$\lim_{k\to\infty} \gamma_k = 0, \sum_{k=1}^{\infty} \gamma_k = \infty.$$

2) Для кожного ребра $((t,k),(t,k'))$, $t \in T$, $t' \in N_t$, $k \in K$, $k \in K$ введемо поча-

ткові значення якостей ребер $g^0_{tt'}(k,k') = g_{tt'}(k,k')$.

3) Для кожної вершини (t,k), $t \in T$, $k \in K$ введемо початкові значення якостей вершин $q^0_t(k) = q_t(k)$.
4) Присвоїти $i = 0$.

Субградієнтний спуск полягає в ітеративному обчисленні градієнтів цільової функції та відповідному оновленні репараметризації φ.

Ітерації

1) Починаємо із репараметризації, яка дорівнює нулю, $\varphi_{tt'}(k) = 0$, $\forall t \in T$, $t' \in N_t$, $k \in K$.
2) Для кожної пари сусідніх об'єктів $tt' \in \Gamma$ оберемо пару міток $k \in K, k' \in K$, які утворюють ребро із максимальною якістю, тобто виконується

$$g^i_{tt'}(k,k') = \max_{l \in K, l' \in K} g^i_{tt'}(l,l').$$

Оновлюємо градієнт за формулами

$$\varphi_{tt'}(k) := \varphi_{tt'}(k) - 1,$$

$$\varphi_{t't}(k') := \varphi_{t't}(k') - 1.$$

3) Для кожного об'єкту $t \in T$ оберемо мітку $k \in K$ таким чином, щоб

$$q^i_t(k) = \max_{l \in K} q^i_t(l).$$

Для всіх сусідніх об'єктів $t' \in N_t$ оновлюємо градієнт

$$\varphi_{tt'}(k) := \varphi_{tt'}(k) + 1.$$

4) Обчислюємо нові якості ребер $((t,k),(t',k'))$, $tt' \in \Gamma$, $k \in K$, $k' \in K$

$$g_{tt'}^{i+1}(k,k') := g_{tt'}^{i}(k,k') + \varphi_{tt'}(k) + \varphi_{t't}(k').$$

5) Обчислюємо нові значення якостей вершин (t,k), $t \in T$, $k \in K$

$$q_t^{i+1}(k) := q_t^i(k) - \sum_{t' N_t} \varphi_{tt'}(k).$$

6) Оновлюємо номер ітерації $i := i + 1$ і переходимо до пункту №1.

Нехай $E(\varphi^i)$ — якість розмітки на ітерації i. Тоді, згідно з теорією методу субградієнтного спуску, справедлива рівність [13]

$$\lim_{i \to \infty} E(\varphi^i) = \min_{\varphi} E(\varphi).$$

Тому алгоритму субградієнтного спуску вирішує задачу мінімізації, і таким чином знаходить якість оптимальної нечіткої розмітки.

Варто зауважити, що алгоритм вимагає більш точного формулювання. Перш за все, умова зупинки не була вказана. Також, хоча ми й знаємо якість оптимальної розмітки, ми нічого не можемо сказати про саму оптимальну розмітку. Субградієнтний метод не оперує поняттями, які можна якимось чином інтерпретувати як якість нечіткої розмітки.

Пошук строгого мінімуму — достатньо жорстка умова, часто замість цього намагаються розв'язати простішу задачу — задачу пошуку ε-оптимальної розмітки, тобто

$$\begin{cases} q_t(k) < \max\limits_{l \in K} q_t(l) - \varepsilon, & \Longrightarrow \ \alpha_t(k) = 0, \\ g_{tt'}(k,k') < \max\limits_{l \in K, l' \in K} g_{tt'}(k,k') - \varepsilon, & \Longrightarrow \ \beta_{tt'}(k,k') = 0. \end{cases} \tag{9}$$

Існування ε-оптимальної розмітки не гарантує глобального оптимуму, але показує,

що поточна якість є близькою до мінімальної якості, а також, що не менш важливо, що поточна нечітка розмітка є близькою до шуканої оптимальної розмітки.

Теорема.[14] Якщо якості (q,g) нечіткої розмітки із якістю $G(\varphi)$, для яких існує нечітка розмітка (α, β), яка задовольняє умовам (9), то це означає, що справедлива наступна нерівність

$$E(\varphi) \leq E(\varphi^*) + \varepsilon \cdot (|T| + |\Gamma|).$$

Для визначення умови зупинки алгоритму, послабимо задачу відшукання нечіткої розмітки

$$\begin{cases} \sum\limits_{t\in T} \sum\limits_{t'\in N_t} \sum\limits_{k\in K} \left[\alpha_t(k) - \sum\limits_{k'\in K} \beta_{tt'}(k,k')\right]^2 \leq \varepsilon, & \\ \sum\limits_{k\in K} \alpha_t(k) = 1, & t \in T, \\ \alpha_t(k) \geq 0, & t \in T, k \in K, \\ \beta_{tt'}(k,k') \geq 0, & tt' \in \Gamma, k \in K, k' \in K. \end{cases}$$

Відмітимо, що при $\varepsilon = 0$, початкова задача, і задача з послабленою умовою є еквівалентними. Для кожної ітерації i і отриманих значень якостей (q^i,g^i) визначимо

число S^i, яке назвемо нев'язкою.

$$\begin{cases} S^i = \min \sum\limits_{t\in T} \sum\limits_{t'\in N_t} \sum\limits_{k\in K} \left[\alpha_t(k) - \sum\limits_{k'\in K} \beta_{tt'}(k,k')\right]^2, & \\ \sum\limits_{k\in K} \alpha_t(k) = 1, & t \in T, \\ \alpha_t(k) \geq 0, & t \in T, k \in K, \\ \beta_{tt'}(k,k') \geq 0, & tt' \in \Gamma, k \in K, k' \in K, \\ q_t^i(k) < \max\limits_{l\in K} q_t^i(l) - \varepsilon, & \Longrightarrow\ \alpha_t(k) = 0, \\ g_{tt'}^i(k,k') < \max\limits_{l\in K, l'\in K} g_{tt'}^i(k,k') - \varepsilon, & \Longrightarrow\ \beta_{tt'}(k,k') = 0. \end{cases} \tag{10}$$

Таким чином, умова зупинки алгоритму субградієнтного спуску може бути визначена у термінах оцінки нев'язки на кожній ітерації, так, щоб нев'язка була меншою за наперед задане значення відхилення $\varepsilon \geq 0$.

Якщо для поточних значень якості вершин та ребер (q^i, g^i) значення нев'язки S^i є більшим за ε, то потрібно продовжувати ітерації субградієнтного спуску, якщо значення S^i менше за ε, то це значить, що задача є достатньо близькою до тривіальної, і будь-яка нечітка розмітка, яка задовольняє (9) є близькою до оптимальної шуканої розмітки.

Знаходження точного розв'язку до задачі (10) обчислювально є порівнюваною із початковою задачею. Одначе, не обов'язково знаходити точне значення S^i, щоб порівняти його із ε, буде достатньо знайти верхню і нижню межу S^i. Якщо верхня межа нев'язки S^i є меншою за ε, тоді зупиняємо субградієнтний спуск. Схожим чином, якщо нижня межа є більшою за ε, тоді продовжуємо ітерувати алгоритм субградієнтного спуску. Точний вираз для верхньої та нижньої межі можна знайти в [14].

2.2 Викреслювання другого порядку

Після того, як одним з методів було здійснено оптимізацію штрафної функції, виникає задача відновлення оптимальної розмітки — треба знайти таку розмітку k (або одну з таких розміток), якість якої дорівнює знайденій максимальній якості. Для цього часто використовується алгоритм викреслювання другого порядку, також відомий як алгоритм релаксації розмітки (relaxation labeling algorithm) [4, 18–20].

Структура графу аналогічна $(\max,+)$ задачі розмітки. Нехай T — скінченна множина об'єктів, K — скінченна множина міток, $\Gamma \subset T^2$ — множина сусідів, $N_t \subset T$ — множина сусідів об'єкту $t \in T$. Якості вершин та ребер — бінарні значення, $q : T \times K \to \{0,1\}$ якості вершин, $g : \Gamma \times K^2 \to \{0,1\}$ якості ребер. Вершина $(t,k), t \in T, k \in K$ вважається допустимою, якщо відповідна їй якість $q_t(k) = 1$. Аналогічним чином, ребро $((t,k),(t',k'))$, $t \in T$, $t' \in N_t$, $k \in K$, $k' \in K$ називається допустимим, якщо відповідна йому якість $g_{tt'}(k,k') = 1$.

Допустимістю розмітки будемо називати число $G \in \{0,1\}$

$$G = \bigvee_{k:T\to K} \bigwedge_{t\in T} q_t(k_t) \wedge \bigwedge_{tt'\in\Gamma} g_{tt'}(k,k').$$

$(\bigvee, \bigwedge)$-задача полягає пошуку відповіді на питання „Чи існує така розмітка, для якої всі вершини та ребра є допустимими?", тобто чи можна знайти таке значення G, що $G = 1$.

Правила викреслювання

Вершина вважається допустимою, якщо відповідна їй якість $q_t(k) = 1$, а також в неї входить принаймні одне допустиме ребро із кожного сусіднього об'єкта $t' \in N_t$, тобто

$$q_t(k) = q_t(k) \wedge \bigwedge_{t'\in N_t} \bigvee_{k\in K} g_{tt'}(k,k'),$$

де $t \in T$, $k \in K$, $k' \in K$, $tt' \in \Gamma$.

Операція називається викреслюванням вершини.

Ребро вважається допустимим, якщо відповідна їй якість $g_{tt'}(k,k') = 1$, а також якщо вершини, які утворюють ребро (t,k), (t',k') є допустимими, тобто

$$g_{tt'}(k,k') = g_{tt'}(k,k') \wedge q_t(k) \wedge q_{t'}(k'),$$

де $t \in T, k \in K, k' \in K, tt' \in \Gamma$.

Операція називається викреслюванням ребра.

З наведеною постановкою задачі та описом основних операцій, наведемо повний алгоритм викреслювання другого порядку.

Algorithm 1 Алгоритм викреслювання другого порядку

Вхід: $(\bigvee, \bigwedge)$-задача розмітки

Вихід: $G \in \{0,1\}$;

Ініціалізація:

$$q^0 = q, g^0 = g, i = 0.$$

Крок 1:

$$q_t^{i+1}(k) = q_t^i(k) \wedge \bigwedge_{t' \in N_t} \bigvee_{k \in K} g_{tt'}^i(k,k'),$$

$$\forall t \in T, \forall k \in K.$$

Крок 2:

$$g_{tt'}^{i+1}(k,k') = g_{tt'}^i(k,k') \wedge q_t^i(k) \wedge q_{t'}^i(k'),$$

$$\forall t \in T, \forall t' \in N_t, \forall k \in K, \forall k' \in K.$$

Крок 3:

$$i = i + 1.$$

Якщо за ітерацію жодне значення з якостей q, g не змінилося, повертаємося до кроку 1.

Умова зупинки:

Якщо на якомусь кроці $j > 0$, виконуються умови

$$q_t^j(k) = 0,$$

$$\forall t \in T, \forall k \in K,$$

$$g_{tt'}^j(k,k') = 0, \forall t \in T, \forall t' \in N_t, \forall k \in K, \forall k' \in K,$$

то це означає, що для задачі не існує такої розмітки, яка складається з допустимих вершин та ребер, повертаємо $G = 0$.

Якщо ж ні — то це означає, що існує така розмітка, яка складається з допустимих вершин та ребер, тоді повертаємо значення $G = 1$.

Алгоритм викреслювання другого порядку полягає в багаторазовому застосу-

ванні операцій „викреслювання вершини” та „викреслювання ребра”.

Алгоритм завершує роботу за скінченну кількість ітерацій, адже ніяка викреслена вершина або дужка не може знову стати допустимою. Якщо після завершення роботи алгоритму деякі вершини залишилися допустимими, то за певних умов [4] з множини допустимих вершин можна побудувати розмітку.

Загальна складність алгоритму викреслювання другого порядку — $\mathcal{O}(|T|^2 \cdot |K|^2)$

2.2.1 Застосування до $(\max, +)$ задачі розмітки

Після оптимізації цільової функції двоїстої задачі шукають відповідь на запитання чи існує допустима розмітка, і якщо існує, то як її знайти. Відповідь на перше запитання дає алгоритм викреслювання другого порядку. Для цього потрібно розглянути наступне представлення задачі.

Маємо двоїсту $(\min, +)$ задачу розмітки на множині об’єктів T, множині міток K, та з репараметризованими якостями вершин $q_t^{\varphi}(k), t \in T, k \in K$ та репараметризованими якостями ребер $g_{tt'}^{\varphi}(k,k')$, $tt' \in \Gamma$, $k \in K$, $k' \in K$. За оригінальною постановкою задачі, ми хочемо мінімізувати значення функції

$$E(\varphi) = \sum_{tt' \in \Gamma} \max_{k \in K, k' \in K} [g_{tt'}(k,k') + \varphi_{tt'}(k) + \varphi_{t't}(k)] +$$

$$\sum_{t \in T} \max_{k \in K} [q_t(k) - \sum_{t' \in N(t)} \varphi_{tt'}(k)] = \sum_{tt' \in \Gamma} \max_{k \in K, k' \in K} g_{tt'}^{\varphi}(k,k') + \sum_{t \in T} \max_{k \in K} q_t^{\varphi}(k).$$

$$\varphi^* = \min_{\varphi} E(\varphi).$$

Нехай алгоритмом оптимізації ми знайшли підходяще або достатньо близьке значе-

ння φ^*. Підставимо його у цільову функцію, і отримаємо

$$E(\varphi^*) = \sum_{tt' \in \Gamma} \max_{k \in K, k' \in K} g_{tt'}^{\varphi^*}(k,k') + \sum_{t \in T} \max_{k \in K} q_t^{\varphi^*}(k).$$

З цієї рівності чітко видно, що шукана розмітка — це та, для якої ми обираємо мітку з максимальною якістю для кожного об'єкту, і одночасно з цим, вибираємо такі ребра, які мають максимальну вагу. Якщо алгоритм оптимізації не досяг глобального оптимуму, то можуть виникнути проблеми. Розмітка — це не просто вибір мітки в кожному об'єкті, крім того, потрібно також щоб обрані мітки були з'єднані ребром.

Припустимо, що для якогось об'єкту $t \in T$ ми обрали мітку з максимальною якістю $k_t^* \in K$. Для сусіднього об'єкту $t' \in N_t$ обрали мітку $k_{t'}^* \in K$. Тепер потрібно обрати найкраще ребро для пари $tt' \in \Gamma$. Нехай ребро з найкращою якістю буде $((t,l),(t',l'))$, $t \in T$, $t' \in T$, $l \in K$, $l' \in K$. Проблемою в даному випадку буде те, що з обраного набору міток не можливо обрати розмітку (для пари об'єктів). Якщо така ситуація сталася при хоча б для однієї пари об'єктів $tt' \in \Gamma$, то отримати розмітку вже неможливо. Можливі також схожі випадки, наприклад $k_{t'}^* = l, k_{t'}^* \neq l'$, тобто ребро входить не в мітку з максимальною вагою, і т.д.

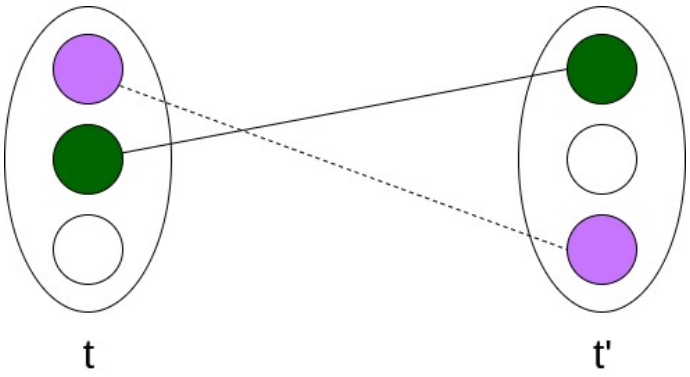


Рисунок 2.1 — Приклад випадку, коли неможливо обрати розмітку. Фіолетовим кольором позначені вершини з максимальною якістю q, зеленим кольором позначені вершини, які утворюють ребро з максимальною якістю g. Лініями позначені можливі ребра, які поєднують „кращі" вершини.

Для приблизного розв'язку іноді опускають ці неточності, та обирають мі-

тку в об'єкті шляхом вибору ребра із максимальною якістю. Часто такий підхід дає прийнятні результати на практиці, хоча йому й бракує теоретичного підґрунтя.

Для уникнення таких випадків, ми маємо задачу, після того, як здійснили оптимізацію. Сформулюємо $(\bigvee, \bigwedge)$-задачу розмітки.

Множина об'єктів T та множина міток K, структура сусідства Γ задачі будуть такими ж, як і для $(\max, +)$ задачі. Нові якості задачі будуть задаватися наступними функціями

$$\begin{cases} q_t(k) = [\![q_t^{\varphi}(k) = \max\limits_{l \in K} q_t^{\varphi}(l)]\!], & t \in T, k \in K, \\ g_{tt'}(k,k') = [\![g_{tt'}^{\varphi}(k,k') = \max\limits_{l \in K, l' \in K} g_{tt'}^{\varphi}(l,l')]\!], & tt' \in \Gamma, k \in K, k' \in K. \end{cases}$$

При такій постановці ми якраз і будемо перевіряти, чи можливо обрати найкращі мітки у вершинах та ребрах таким чином, щоб з них можна було скласти розмітку. Варто відмітити, що виконання умов (2.2.1) можливе лише за умови, що функція $P(\varphi)$ — досягає глобального оптимуму в точці φ^*.

Іноді алгоритми оптимізації гарантовано сходяться до глобального оптимуму тільки при нескінченній кількості кроків, що неможливо для практичного застосування. Через це часто вводяться послаблені умови допустимості вершин та ребер.

$$\begin{cases} q_t(k) = [\![|q_t^{\varphi}(k) - \max\limits_{l \in K} q_t^{\varphi}(l)| \le \varepsilon]\!], & t \in T, k \in K, \\ g_{tt'}(k,k') = [\![|g_{tt'}^{\varphi}(k,k') - \max\limits_{l \in K, l' \in K} g_{tt'}^{\varphi}(l,l')| \le \varepsilon]\!], & tt' \in \Gamma, k \in K, k' \in K, \\ \varepsilon \in \mathbb{R}_+. \end{cases}$$

Фактично вирішується оптимізаційна задача виду

$$k^*(\varepsilon) \in \arg \bigvee_{k: T \to K} \bigwedge_{tt' \in \Gamma} [\![|g_{tt'}^{\varphi}(k,k') - \max_{l \in K, l' \in K} g_{tt'}^{\varphi}(l,l')| \le \varepsilon]\!] \wedge \bigwedge_{t \in T} [\![|q_t^{\varphi}(k) - \max_{l \in K} q_t^{\varphi}(l)| \le \varepsilon]\!].$$

Отримана розмітка називається ε-допустимою. Варто відмітити, що при такому послаблені, ми можемо й не досягти оптимального значення після оптимізації, (хоча й будемо ε-близькі до нього). Знайдена розмітка при умові $\varepsilon \to 0$ буде наближатися до оптимальної. Такі умови дозволяють точно вирішувати ширший клас задач за скінченний час, а також контролювати відхилення знайденого оптимуму від глобального, більше того, відхилення від ε слугує метрикою, з допомогою якої ми можемо балансувати швидкість розв'язку та точність вирішення.

Для нових якостей вершин $q_t(k)$, $t \in T$, $k \in K$ та якостей ребер $g_{tt'}(k,k')$, $tt' \in \Gamma$, $k \in K$, $k' \in K$ розв'яжемо $(\bigvee, \bigwedge)$-задачу розмітки. Якщо отримали $G = 0$, то це означає, що для даної знайденої репараметризації φ не існує допустимої розмітки, тобто принаймні для однієї пари об'єктів обрані кращі мітки для вершин не стикуються з обраними кращими мітками для ребер. Якщо ж $G = 1$, то це означає, що існує розмітка, для яка складається із допустимих вершин та ребер, які стикуються між собою.

Невикреслена множина вершин та ребер називається ε-узгодженим набором вершин та ребер. Якщо алгоритм завершив свою роботу із відповіддю $G = 0$, то ε-узгоджений набір вершин та ребер буде порожньою множиною.

Хоча алгоритм викреслювання другого порядку і дозволяє перевіряти наявність потрібної розмітки, алгоритм не дає змоги відшукати її.

2.3 Відновлення розмітки після оптимізації

Алгоритми з самоконтролем відомі для розв'язку задач лінійної класифікації, для інваріантних відносно оператора напівгратки задач розмітки, для інваріантних відносно мажоритарного оператора $(\min, \max)$ задач розмітки та більш широкого класу задач розмітки, інваріантних відносно мажоритарного оператора а також для супермодулярних $(\max, +)$ задач розмітки з цілочисельними якостями вершин та ребер.

Надалі ми будемо розглядати самоконтроль у контексті його застосування до $(\max,+)$ задач розмітки, а саме — пошуку оптимальної розмітки після оптимізації цільової функції.

2.3.1 Самоконтроль

Для $(\max,+)$ задач розмітки самоконтроль використовується для відшукання узгодженої розмітки. Часто такої розмітки може не існувати, такі випадки алгоритм також відпрацьовує. Не вдаючись в деталі, хочемо, щоб на вхід алгоритму подавали $(\bigvee,\bigwedge)$ задачу розмітки, після оптимізації цільової функції, а на вихід одне з двох — найкращу розмітку (якщо вона існує), або відповідь — для даної репараметризації не існує узгодженої розмітки. Самоконтроль полягає у тому, що не користувач вирішує, на яке питання треба відповісти, а сам алгоритм вирішує, що потрапляє у зону його компетентності.

У контексті $(\max,+)$ задачі пошуку ε-допустимої розмітки, задача виглядає наступним чином:

$$k^*(\varepsilon) \in \arg \bigvee_{k:T\to K^T} \bigwedge_{tt'\in\Gamma} [\![|g^{\varphi}_{tt'}(k,k') - \max_{l\in K, l'\in K} g^{\varphi}_{tt'}(l,l')| \le \varepsilon]\!] \wedge \bigwedge_{t\in T} [\![|q^{\varphi}_t(k) - \max_{l\in K} q^{\varphi}_t(l)| \le \varepsilon]\!],$$

де $q^{\varphi}_t(k)$, $t \in T$, $k \in K$ — репараметризовані значення якостей вершин, після оптимізації цільової функції, $g^{\varphi}_{tt'}(k,k')$, $tt' \in \Gamma$, $k \in K$, $k' \in K$ — репараметризовані значення якостей ребер після оптимізації. Нехай $f : K^T \to \{0,1\}$ — функція допустимості розмітки, тобто якщо $f(k) = 1, k \in K^T$ — то розмітка k є допустимою, інакше — ні. Тоді можна представити задачу як

$$k^*(\varepsilon) \in \arg \bigvee_{k:T\to K^T} f(k_\varepsilon).$$

Маємо алгоритм викреслювання другого порядку, який дає відповідь, чи існує

допустима розмітка для даного набору якостей вершин та ребер, тобто перевіряти рівність $f(k) = 1$, $k \in K^T$. Самоконтроль займається тим, що знаходить таке значення $k \in K^T$ для якого $f(k) = 1$. Алгоритм викреслювання другого порядку є ядром самоконтролю.

Вхід

Задача пошуку оптимальної розмітки

Вихід

Найкраща допустима розмітка $k \in K^T$ або „відмова від розпізнавання" або „немає рішення"

Ітерації

1) На першому кроці перевіримо, чи взагалі існує допустима розмітка, тобто

$$\begin{cases} \bigvee_{k:T\to K^T} f(k) = 0, & \text{відповідь — „немає рішення"}, \\ \bigvee_{k:T\to K^T} f(k) = 1, & \text{продовжуємо роботу, переходимо до кроку №2.} \end{cases}$$

2) Проходимося по всіх об'єктах і в кожному з них шукаємо мітку, при якій існує допустима розмітка

$$\forall t \in T,$$

проходимося по всіх мітках в об'єкті $t \in T$ і по черзі перевіряємо допустимість

$$\forall i \in \{1, \ldots, |K|\},$$

$$F_t(k_i) = \bigvee_{k_1\in K} \bigvee_{k_2\in K} \cdots \bigvee_{k_{i-1}\in K} \bigvee_{k_{i+1}\in K} \cdots \bigvee_{k_{|K|}\in K} f(k_1, k_2, \ldots, k_{i-1}, k_i, k_{i+1}, \ldots, k_{|K|}).$$

Якщо існує така мітка, при якій існує допустима розмітка, то запам'ятовуємо її,

$$\exists k \in K : F_t(k) = 1, \implies k_t^* = k.$$

Якщо ж пройшовшись по всіх мітках в об'єкті $t \in T$ такої мітки не знайшлося,

то це означає, що задача не є супермодулярною, і розв'язок неможливий, тому відповідь „відмова від розпізнавання".

3) Якщо для кожного об'єкту знайшлася мітка, при якій існує допустима розмітка, завершуємо алгоритм, і повертаємо найкращу розмітку, тобто якщо виконується

$$\forall t \in T, \exists k_t^* \in K : F_t(k_t^*) = 1,$$

то набір $k^* = (k_1^*, \ldots, k_{|T|}^*)$ повертаємо як відповідь. Якщо для кожного об'єкта знайшли мітку, то в сукупності ці мітки будуть складати розмітку.

Висновки до розділу 2

В розділі було розглянуто методи отримання оптимальної розмітки за допомогою субградієнтного методу, а також подальше відновлення найкращої розмітки після здійснення оптимізації цільової функції. Було показано, що при певних умовах, метод субградієнтного спуску збігається за скінченну кількість кроків до прийнятного відхилення від глобального мінімуму. Також розглянуто алгоритм викреслювання другого порядку, який дозволяє для репараметризованих якостей задачі перевіряти, чи існує допустима розмітка. В свою чергу, самоконтроль дозволяє знайти допустиму розмітку, якщо вона існує, або ж просигналізує про те, що якась із початкових умов задачі не виконується.

3 ЗАПРОПОНОВАНИЙ АЛГОРИТМ ТОЧНОГО РОЗВ'ЯЗКУ

Третій розділ присвячено запропонованому алгоритму пошуку найкращої розмітки. Також показано, що для супермодулярних задач та при умові цілих значень якостей вершин та ребер завжди за скінченний час знайдеться ε-допустима розмітка, також показано спосіб прискорення алгоритму.

3.1 Алгоритм розв'язку $(\max, +)$ задач розмітки

Використовуючи приведені алгоритми у якості блоків, побудуємо покращений end-to-end алгоритм розв'язку $(\max, +)$ задач розмітки із цілочисельними якостями.

Нехай T — множина об'єктів, K — множина міток, $\Gamma \subset T^2$ — структура сусідства, N_t — множина сусідів об'єкту $t \in T$. Маємо цілочисельні якості вершин та ребер $q_t(k) \in \mathbb{Z}$, $t \in T$, $k \in K$, і $g_{tt'}(k,k') \in \mathbb{Z}$, $tt' \in \Gamma$, $k \in K$, $k' \in K$. Відмітимо, що цілочисельність якостей — не є великим обмеженням на практиці, так як всі обчислювальні машини мають граничну точність. Візьмемо константу ε, причому $\varepsilon \in \mathbb{R}_+$ — міра допустимого відхилення від оптимального значення цільової функції, а також поелементне відхилення розмітки від оптимальної. Спочатку методом субградієнтного спуску виконуємо оптимізацію цільової функції двоїстої задачі $G(\varphi)$ до тих пір, поки алгоритм викреслення другого порядку не надасть не нульові мітки, тобто буде відомо, що для даної репараметризації існує допустима розмітка. Застосовуємо субградієнтний спуск до цільової функції за аргументом φ та отримаємо послідовність репараметризацій $(\varphi^j : j \in \mathbb{N}_0)$. Робимо мінімальну кількість кроків до виконання умови існування розмітки. Зупиняємо роботу субградієнтний спуску на найменшому номері $i \in \mathbb{N}_0$ кроку, за якого для q^{φ^i} і g^{φ^i} існує ε-узгоджений набір вершин та ребер. Результат застосування цієї процедури для обраних q, g і ε

позначимо

$$\phi(q,g;\varepsilon) = \varphi^i.$$

Такий номер i обов'язково знайдеться, адже відомо, що для будь-якого $\varepsilon \in \mathbb{R}_+$ субградієнтний спуск за скінченну кількість $i \in \mathbb{N}_0$ кроків дійде до такого значення φ^i, за якого існує ε-узгоджений набір вершин та ребер [14].

Пошук однієї з найкращих розміток здійснюють у такий спосіб.

1) Ініціалізуємо початкові значення якостей вершин, введемо функцію $q^0 = q$ та номер кроку $j = 0$.
2) Якщо кожен об'єкт $t \in T$ має лише по одній мітці зі скінченною якістю, алгоритм завершується, а розмітку k^* будують у такий спосіб, що

$$q_t^j(k_t^*) > -\infty, \forall t \in T.$$

3) Якщо ж тривіального рішення не існує, продовжуємо алгоритм. Для довільного об'єкта $t \in T$, у якому залишилося більше однієї мітки зі скінченною вагою, фіксуємо довільну мітку $k \in K$. Функцію якостей ребер залишаємо, а функцію якостей вершин змінимо таким чином, щоб в об'єкті $t \in T$ залишилася тільки обрана мітка k. Позначимо функцію $q' : T \times K \to \mathbb{R} \cup \{-\infty\}$, яка в об'єкті t забороняє усі мітки окрім k

$$q'_{t'}(k') = \begin{cases} -\infty, & t' = t, k' \neq k, \\ q_{t'}^j(k'), & \text{інакше.} \end{cases}$$

Якщо подана на вхід алгоритму задача була супермодулярною, то така модифікація якостей вершин залишить задачу супермодулярною, адже визначення супермодулярності не накладає обмежень на якості вершин.

4) Виконуємо субградієнтний спуск, доки не отримаємо ε-узгоджений набір вершин та ребер, для пошуку значення $E(q',g;\phi(q',g,\varepsilon))$. Якщо виконується рів-

ність

$$\lfloor E(q',g;\phi(q',g,\varepsilon))\rfloor = \lfloor E(q,g;\phi(q,g,\varepsilon))\rfloor,$$

то переходимо на наступну ітерацію $j = j + 1$, створюємо функцію $q^j = q'$ (яка забороняє в об'єкті $t \in T$ усі мітки окрім обраної мітки $k \in K$) та повертаємося до другого кроку (перевірка на те, чи завершився алгоритм). Якщо отриманий результат менший за той, що був отриманий до початку роботи алгоритму пошуку розмітки, тобто

$$\lfloor E(q',g;\phi(q',g,\varepsilon))\rfloor < \lfloor E(q,g;\phi(q,g,\varepsilon))\rfloor,$$

завершуємо алгоритм із відповіддю 'задача не є супермодулярною', тому що було доведено, що у випадку супермодулярної задачі значення $E(q,g;\phi(q,g,\varepsilon))$ дорівнює вазі $G(k^*)$ оптимальної розмітки k^*, а наведена нерівність означатиме, що під час роботи алгоритму може бути знайдена розмітка із меншою вагою. Якщо отриманий результат більший за той, що був отриманий до початку роботи алгоритму пошуку розмітки

$$\lfloor E(q',g;\phi(q',g,\varepsilon))\rfloor > \lfloor E(q,g;\phi(q,g,\varepsilon))\rfloor,$$

виконуємо третій крок алгоритму для іншої мітки $k \in K$ (яку не було обрано раніше у даному об'єкті). Якщо ця нерівність виконується для всіх міток в об'єкті, то завершуємо алгоритм із відповіддю 'задача не є супермодулярною', тому що за умови супермодулярності задачі обов'язково можна знайти мітку, що належить оптимальній розмітці, якість якої дорівнює $\lfloor E(q,g;\phi(q,g,\varepsilon))\rfloor$.

При такій постановці, даний алгоритм потребує не більше ніж $|T| \cdot |K|$ розрахунків величин $\lfloor E(q,g;\phi(q,g,\varepsilon))\rfloor$ (які ми здійснюємо за допомогою субградієнтного спуску) для різних значень q'. Наведену оцінку складності можна покращити, якщо на етапі пошуку мітки використати алгоритм двійкового пошуку.

3.1.1 Алгоритм двійкового пошуку

Двійковий пошук — класичний та ефективний алгоритм пошуку елементів у відсортованому масиві. Основний принцип роботи полягає у багаторазовому поділу половини масиву, яка може містити потрібний елемент, до тих пір, поки список можливих місць не звузиться лише до одного. Двійковий пошук є суттєво швидшим за лінійний, може використовуватися для широкого ряду задач, також він відносно простий у реалізації і на сьогодні є загальновживаним. В найгіршому випадку двійковий пошук працює за логарифмічний час, роблячи $\mathcal{O}(\log n)$ порівнянь елементів, де n — розмір вхідного масиву.

Якщо розглядати двійковий пошук для пошуку числа у відсортованому масиві, то схематично алгоритм буде виглядати наступним чином:

Вхід: A — масив впорядкованих чисел довжиною $n \in \mathbb{N}_+$, x — шуканий елемент.

Вихід: індекс $i : 0 \leq i < n$, такий що $A[i] = x$ або -1, якщо елемента x в масиві немає.

1) Порівнюємо x із середнім елементом масиву A, якщо не співпадає і $len(A) = 1$, то повертаємо -1.
2) Якщо співпадає, то завершуємо алгоритм, повертаємо індекс середини масиву.
3) Якщо x більше за середній елемент, то x може лежати лише в правій частині масиву A. Рекурсивно запускаємо процедуру для правої частини.
4) Якщо x менше за середній елемент, то повторюємо процедуру рекурсивно для лівої частини масиву A.

Двійковий пошук може робити набагато більше, ніж просто знайти значення в масиві чисел. Його можна використовувати скрізь, де існує монотонна функція, яка приймає значення істинна або хиба. Припустимо, що $f(x)$ — булева функція, яка приймає значення 0 або 1. Двійковий пошук можна використовувати, коли булева предикатна функція $f(x)$ є монотонною за змінною x, тоді двійковий пошук буде

сигналізувати про перехід від нулів до одиничок чи навпаки.

У контексті нашої задачі, таке застосування може бути корисним. Роль предиката буде виконувати алгоритм викреслювання другого порядку, а пошук буде відбуватися на скінченному масиві міток K.

3.1.2 Застосування двійкового пошуку у самоконтролі

На етапі пошуку мітки $k_t \in K$ для об'єкта $t \in T$ на кожній ітерації j використаємо двійковий пошук: розіб'ємо множину K на такі дві підмножини K_1 та K_2, що

$$\begin{cases} K_1 \cup K_2 = K, \\ K_1 \cap K_2 = \emptyset, \\ ||K_1| - |K_2|| \leq 1, \end{cases}$$

та введемо функцію q'

$$q'_{t'}(k') = \begin{cases} -\infty, & t' = t, k' \in K_1, \\ q^j_{t'}(k'), & \text{інакше.} \end{cases}$$

Якщо $\lfloor E(q',g;\phi(q',g,\varepsilon))\rfloor = \lfloor E(q,g;\phi(q,g,\varepsilon))\rfloor$, то шукане k може знаходитися у K_2, і його треба шукати у цій множині. Якщо $\lfloor E(q',g;\phi(q',g,\varepsilon))\rfloor < \lfloor E(q,g;\phi(q,g,\varepsilon))\rfloor$, задача не є супермодулярною. Якщо $\lfloor E(q',g;\phi(q',g,\varepsilon))\rfloor > \lfloor E(q,g;\phi(q,g,\varepsilon))\rfloor$, шукаємо відповідь у множині K_1. Пошук у вибраній множині $K_i, i \in \{1,2\}$ виконують за тим самим принципом. Наприкінці, коли залишається лише одне k, потрібно перевірити рівність $\lfloor E(q',g;\phi(q',g,\varepsilon))\rfloor = \lfloor E(q,g;\phi(q,g,\varepsilon))\rfloor$, заборонивши усі інші мітки в об'єкті, тому що, коли множина K_1 не підходила, ми одразу вибирали K_2, але не перевіряли, чи вона підходить. Отже, для наведеного алгоритму розмітки потрібно не більше ніж $|T| \cdot \log_2 |K| + 1$ розрахунків величин $\lfloor E(q',g;\phi(q',g,\varepsilon))\rfloor$ для різних q'.

3.2 Властивості розв'язку

Якщо для супермодулярної $(\max, +)$ задачі розмітки із впорядкованістю n (яка може бути невідомою) знайшовся ε-узгоджений набір вершин та ребер, для кожної пари сусідніх об'єктів $tt' \in \Gamma$ можна обрати таке ребро $((t,k_t),(t',l_{t'})) \in A$ (необов'язково різні), що

$$\begin{cases} n_t(k_t) \geq n_t(l_t), \\ n_{t'}(k_{t'}) \geq n_{t'}(l_{t'}), \end{cases}$$

для яких справджується нерівність

$$g_{tt'}(k_t,k_{t'}) + g_{tt'}(l_t,l_{t'}) \geq g_{tt'}(k_t,l_{t'}) + g_{tt'}(l_t,k_{t'}).$$

З визначення ε-узгодженості маємо нерівності

$$g_{tt'}(k_t,l_{t'}) \geq \max_{l \in K, l' \in K} g_{tt'}(l,l') - \varepsilon,$$

$$g_{tt'}(l_t,k_{t'}) \geq \max_{l \in K, l' \in K} g_{tt'}(l,l') - \varepsilon,$$

отже

$$g_{tt'}(k_t,k_{t'}) + g_{tt'}(l_t,l_{t'}) \geq g_{tt'}(k_t,l_{t'}) + g_{tt'}(l_t,k_{t'}) \geq 2 \cdot \max_{l \in K, l' \in K} g_{tt'}(l,l') - 2 \cdot \varepsilon.$$

З визначення максимального елементу маємо нерівність

$$-g_{tt'}(l_t,l_{t'}) \geq - \max_{l \in K, l' \in K} g_{tt'}(l,l'),$$

тому

$$g_{tt'}(k_t,k_{t'}) \geq \max_{l \in K, l' \in K} g_{tt'}(l,l') - 2 \cdot \varepsilon.$$

Зауважимо, що за умови

$$\max_{l\in K,l'\in K} g_{tt'}(l,l') - \varepsilon > g_{tt'}(k_t,k_{t'}) \geq \max_{l\in K,l'\in K} g_{tt'}(l,l') - 2\cdot\varepsilon,$$

ребро $((t,k_t),(t',k'_t))$ не належить ε-узгодженому набору A ребер, проте може належати оптимальній розмітці.

Якщо задача є супермодулярною і знайшовся ε-узгоджений набір вершин та ребер, існує така розмітка $k : T \to K$, для якої справджується вираз

$$\begin{aligned}
&\sum_{t\in T} q_t(k_t) + \sum_{tt'\in\Gamma} g_{tt'}(k,k') = \sum_{t\in T} q^{\varphi}_t(k_t) + \sum_{tt'\in\Gamma} g^{\varphi}_{tt'}(k,k') \geq \\
&\geq \sum_{t\in T}\left(\max_{l\in K} q^{\varphi}_t(l) - \varepsilon\right) + \sum_{tt'\in\Gamma} \max_{l\in K,l'\in K} (g^{\varphi}_{tt'}(l,l') - 2\cdot\varepsilon) = \\
&= \sum_{t\in T} \max_{l\in K} q^{\varphi}_t(l) - |T|\cdot\varepsilon + \sum_{tt'\in\Gamma} \max_{l\in K,l'\in K} g^{\varphi}_{tt'}(l,l') - 2\cdot\Gamma\cdot\varepsilon = \\
&= \sum_{t\in T} \max_{l\in K} q^{\varphi}_t(l) + \sum_{tt'\in\Gamma} \max_{l\in K,l'\in K} g^{\varphi}_{tt'}(l,l') - (|T| + 2\cdot\Gamma)\cdot\varepsilon.
\end{aligned}$$

Найкраща розмітка не може мати більшу якість, ніж сума найбільших значень q та g. Це означає, що за умови

$$(|T| + 2\cdot\Gamma)\cdot\varepsilon < 1,$$

якість розмітки k дорівнює вазі найкращої розмітки, бо функції q і g є цілочисельними [12].

Якщо задача є супермодулярною та має цілочисельні якості, то наведений алгоритм пошуку розмітки надасть оптимальну розмітку для

$$\varepsilon \leq \frac{1}{|T| + 2\cdot|\Gamma| + 1}.$$

У загальному випадку $(\max,+)$ задачі розмітки належать класу складності $EXP - APX$ — класу задач, для яких за поліноміальний від розміру вхідний да-

них час можна знайти наближення, похибка якого обмежена експонентою від розміру даних [21]. Проте зауважимо, що у тому разі, коли задача не є супермодулярною, проте має цілочисельні якості і алгоритм надав розмітку k^*, ця розмітка буде оптимальною, оскільки за побудовою алгоритму ціла частина значення функції E не змінюється і дорівнює $G(k^*)$.

3.2.1 Спосіб перевірки отриманого розв'язку

Важливою властивістю отриманого розв'язку є те, що його коректність можна перевірити за скінченний час. Функція φ є так званим сертифікатом цієї задачі [22]. Це набір чисел, розмір якого має поліноміальну залежність від розміру вхідних даних задачі, та за допомогою якого можна перевірити коректність розв'язку за кількість операцій, що має поліноміальну залежність від розміру вхідних даних задачі. Маючи $\varphi = \phi(q,g;\varepsilon)$ та k^*, можна легко перевірити нерівність

$$G(k^*) > E(q,g;\varphi) - 1.$$

Справедливість цієї нерівності означає, що знайдена розмітка k^* дійсно є розв'язком поданої на вхід алгоритму $(\max,+)$ задачі розмітки з цілочисельними якостями ребер та вершин.

Висновки до розділу 3

Було розглянуто узагальнений метод розв'язування $(\max,+)$ задач розмітки із цілочисельними якостями на основі субградієнтного спуску. Показано, що для супермодулярних задач (з відомою або невідомою впорядкованістю міток) алгоритм за скінченний час обов'язково знайде ε-допустиму розмітку. Якщо задача не є супермодулярною, алгоритм може повернути або ε-допустиму розмітку або відповідь

„задача не є супермодулярною” і відповідь обов’язково буде правильною. Також надано спосіб перевірки отриманого розв’язку.

4 РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТІВ

Четвертий розділ присвячено практичному застосуванню алгоритму розв'язку супермодулярних $(\max,+)$ задач розмітки із самоконтролем на базі субградієнтного спуску.

4.1 Приклад застосування для не супермодулярної задачі

Наразі невідомо, яким є співвідношення між кількістю супермодулярних задач та кількістю тих задач, які може розв'язувати наведений алгоритм. Водночас цей приклад свідчить про те, що що множина не супермодулярних задач, які наведений алгоритм може розв'язувати, не є пустою.

Використаємо представлений алгоритм для невеликої $(\max,+)$ задачі розмітки для ілюстрації. Візьмемо довільне додатне парне число $x \geq 10$. Нехай задачу задано множиною об'єктів $T = \{t_1,t_2,t_3,t_4\}$, структурою сусідства
$\Gamma = \{(t_1,t_2),(t_1,t_3),(t_2,t_4),(t_3,t_4)\}$, множиною міток $K = \{0,1,2\}$, якостями вершин

$$\begin{aligned} q_{t_1}(0) = q_{t_2}(0) = q_{t_3}(0) &= 0, \\ q_{t_1}(1) = q_{t_2}(1) = q_{t_3}(1) &= -\frac{x}{2}, \\ q_{t_1}(2) = q_{t_2}(2) = q_{t_3}(2) &= -x, \\ q_{t_4}(0) &= -x+1, \\ q_{t_4}(1) &= -\frac{x}{2}-1, \\ q_{t_4}(2) &= -1, \end{aligned}$$

і якостями ребер

$$g_{tt'}(k,k') = -\left(\frac{x}{2}+1\right)\cdot[\![k \neq k']\!],$$

для яких задача не є супермодулярною.

Запустимо процедуру субградієнтного спуску. Виберемо початкову репара-

метризацію

$$\varphi_{tt'}(k) = 0, \forall k \in K, t \in T, t' \in N_t,$$

і крок субградієнтного спуску

$$\gamma_i = \left(\frac{x}{2} + 1\right)/i.$$

Виконаємо 2 ітерації субградієнтного спуску й отримаємо значення репараметризації

$$\varphi_{t_3t_4}(0) = \left(\frac{x}{2} + 1\right)/2, \varphi_{t_3t_4}(2) = -\left(\frac{x}{2} + 1\right)/2,$$
$$\varphi_{t_2t_4}(0) = \left(\frac{x}{2} + 1\right)/2, \varphi_{t_2t_4}(2) = -\left(\frac{x}{2} + 1\right)/2,$$
$$\varphi_{t_4t_3}(0) = -\left(\frac{x}{2} + 1\right)/2, \varphi_{t_4t_3}(2) = \left(\frac{x}{2} + 1\right)/2,$$
$$\varphi_{t_4t_2}(0) = -\left(\frac{x}{2} + 1\right)/2, \varphi_{t_4t_2}(2) = \left(\frac{x}{2} + 1\right)/2.$$

Всі інші значення репараметризації φ не змінилися. За такої репараметризації і

$$\varepsilon \leq \frac{1}{|T| + 2 \cdot |\Gamma| + 1} \approx 0.077,$$

задача є ε-узгодженою. Значення цільової функції

$$E(q,g,\varphi) = -x + 1.$$

У кожному об'єкті всі мітки мають скінченну вагу, тому другий крок алгоритму пропустимо. Обхід об'єктів графу здійснюємо у порядку (t_1,t_2,t_3,t_4), а обхід міток в об'єкті виконуємо у порядку $(0,1,2)$. Нехай $K_1 = \{0\}$, $K_2 = \{1,2\}$. Зафіксуємо мітку 0 в об'єкті t_1 та заборонимо інші мітки у цьому об'єкті

$$q'_{t_1}(0) = 0, q'_{t_1}(1) = \infty, q'_{t_1}(2) = \infty,$$
$$q'_t(k) = q_t(k), \forall t \in \{t_2,t_3,t_4\}, \forall k \in K.$$

Використаємо значення φ, отримані на попередньому кроці. Одразу маємо ε-узгоджений набір вершин та ребер, а також

$$E(q',g,\varphi) = E(q,g,\varphi) = -x + 1.$$

В об'єкті t_1 залишимо мітку 0, тобто $k^*_{t_1} = 0$. Проведемо аналогічну процедуру для всіх інших об'єктів. У результаті отримаємо оптимальну розмітку k^*.

$$k^*(t_1) = k^*(t_2) = k^*(t_3) = k^*(t_4) = 0,$$

з

$$G(k^*) = -x + 1 > E(q,g,\varphi) - 1 = -x.$$

4.2 Результати експериментів

Як приклад застосування алгоритму було використано задачу знешумлення бінарного зображення [2, 3, 6, 8]. Зображення з шумом отримується шляхом додавання сильного гаусового шуму до початкового зображення (без шуму). Приклад зображення на 4.1. Нехай T — множина пікселів зображення, $K = \{0,1\}$ — множина міток, де 0 — чорний колір на результуючому зображенні, а 1—білий. Використаємо структуру сусідства із 4 об'єктами. Об'єкти формують граф-решітку із зв'язками, які графічно представлені як знак „+". Для кожного об'єкта його сусідами будуть об'єкти, які знаходяться на одну клітинку вище, нижче, вправо і вліво.

(а) Зображення без шуму

(б) Зображення із накладеним гаусовим шумом

Рисунок 4.1 — Приклад вхідного зображення, $|T| = 4096$

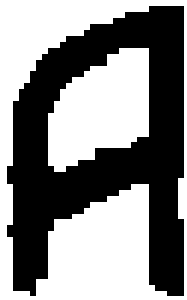
Для задачі знешумлення використаємо якості вершин

$$q_t(k) = \begin{cases} p_t, & k = 0, \\ 255 - p_t, & k = 1, \end{cases}$$

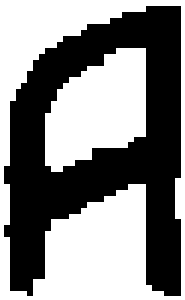
де p_t — інтенсивність пікселя в об'єкті $t \in T$, і якості ребер

$$g_{tt'}(k,k') = K \cdot |k - k'|, \forall tt' \in \Gamma,$$

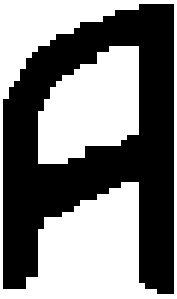
де K—коефіцієнт згладжування. Якості вершин мінімальні, коли присвоюємо мітку 0 темним пікселям або мітку 1 світлим. Якості ребер відповідають за плавність переходів. Зафіксуємо $K = 50$ для прикладів. На 4.2 порівняємо результати роботи алгоритму із алгоритмом пошуку максимального потоку (реалізація [2]). Всі обчислення робилися на комп'ютері з процесором Intel Core i7-10510U і оперативно запам'ятовуючим пристроєм DDR4 2667MHz. Алгоритм пошуку розв'язує задачу точно [2–4, 10], тому значення максимального потоку має співпадати із значенням цільової функції задачі оптимізації.



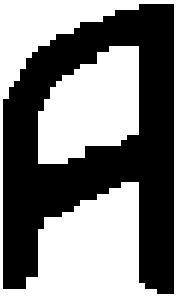
(а) maxflow (0.004s, $|f| = 415612$)



(б) Запропонований алгоритм (41s, $E(q,g,\varphi) = 415612$)



(в) maxflow (0.004s, $|f| = 429022$)



(г) Запропонований алгоритм (41s, $E(q,g,\varphi) = 429022$)

Рисунок 4.2 — Знешумлення, верхній ряд $K = 50$, нижній $K = 100$

Висновки до розділу 4

Показано приклад застосування представленого алгоритму для не супермодулярних задач. Приклад демонструє, що множина не супермодулярних задач, які наведений алгоритм може розв'язувати, не є пустою. Показано застосування алгоритму до задачі знешумлення бінарного зображення. На виході результати запропонованого алгоритму та алгоритму пошуку максимально потоку співпадають. Більше того, значення максимального потоку та значення цільової функції оптимізації також рівні між собою. Алгоритм максимального потоку працює швидше, проте представлений алгоритм є застосовним до значно більшої кількості задач, а також,

не вимагає відомої впорядкованості міток для кожного об’єкта.

ВИСНОВКИ

Розглянуті $(\max, +)$ задачі розмітки є складними задачами дискретної оптимізації. Для часткових випадків алгоритми точного розв'язку є занадто складними обчислювально або вимагають виконання строгих умов для задачі.

Дана робота містить постановку $(\max, +)$ задачі розмітки, огляд основних методів розв'язання. Було представлено новий спосіб розв'язання такого класу задач, заснований на методі субградієнтного спуску. Показано, що при достатньо нестрогих початкових умовах (супермодулярність та цілі значення якостей) алгоритм дозволяє знайти ε-допустиму розмітку за скінченний час. Наведено теоретичне обґрунтування алгоритму. Також було показано спосіб прискорення алгоритму за рахунок використання двійкового пошуку. Дане покращення робить цей алгоритм швидшим за багато існуючих ітеративних методів розв'язку (водночас даючи гарантії на „правильність" відповіді) і, на відміну від алгоритмів точного розв'язку, покриває набагато ширший клас $(\max, +)$ задач розмітки.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1 Шлезингер, М. И. Решение (MAX,+)-задач структурного распознавания с помощью их эквивалентных преобразований. Часть 1 / М. И. Шлезингер, В. В. Гигиняк. — 2007. — no. 1. — Pp. 3–15.

2 Boykov, Y. Fast approximate energy minimization via graph cuts / Y. Boykov, O. Veksler, R. Zabih // *IEEE Transactions on Pattern Analysis and Machine Intelligence*. — 2001. — Vol. 23, no. 11. — Pp. 1222–1239.

3 Boykov, Y. An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision / Y. Boykov, V. Kolmogorov // *IEEE Transactions on Pattern Analysis and Machine Intelligence*. — 2004. — Vol. 26, no. 9. — Pp. 1124–1137.

4 Savchynskyy, Bogdan. Discrete Graphical Models — An Optimization Perspective / Bogdan Savchynskyy // *Foundations and Trends® in Computer Graphics and Vision*. — 2019. — Vol. 11, no. 3-4. — Pp. 160–429. http://dx.doi.org/10.1561/0600000084.

5 Wang, Chaohui. Markov Random Field modeling, inference & learning in computer vision & image understanding: A survey / Chaohui Wang, Nikos Komodakis, Nikos Paragios // *Computer Vision and Image Understanding*. — 2013. — Vol. 117, no. 11. — Pp. 1610–1627. https://www.sciencedirect.com/science/article/pii/S1077314213001343.

6 Szeliski, Richard. Computer Vision: Algorithms and Applications / Richard Szeliski. — 1st edition. — Berlin, Heidelberg: Springer-Verlag, 2010.

7 Michail I. Schlesinger, Václav Hlaváč. Ten Lectures on Statistical and Structural Pattern Recognition / Václav Hlaváč Michail I. Schlesinger. — Springer Dordrecht.

8 Greig, D.M. Exact Maximum A Posteriori Estimation for Binary Images / D.M. Greig, B.T. Porteous, Allan Seheult // *Journal of the Royal Statistical Society, Series B*. — 1989. — 01. — Vol. 51. — P. 271–279.

9 Dash, Protima. Developing algorithm to obtain the maximum flow in a network flow problem / Protima Dash, Md. Mosfiqur Rahman, M.S. Akter // *Journal of Advanced Research in Dynamical and Control Systems*. — 2019. — 01. — Vol. 11. — Pp. 455–459.

10 Ishikawa, H. Exact optimization for Markov random fields with convex priors / H. Ishikawa // *IEEE Transactions on Pattern Analysis and Machine Intelligence*. — 2003. — Vol. 25, no. 10. — Pp. 1333–1336.

11 Schlesinger, Dmitrij. Transforming an Arbitrary Minsum Problem into a Binary One / Dmitrij Schlesinger, BORIS FLACH. — 2006. — 01.

12 Шлезингер, М. И. Анализ алгоритмов диффузии для решения оптимизационных задач структурного распознавания [Текст] / М. И. Шлезингер, К. В. Антонюк // *Кибернетика и системный анализ*. — 2011.

13 Shor, Naun Zuselevich. The Subgradient Method / Naun Zuselevich Shor // Minimization Methods for Non-Differentiable Functions. — Berlin, Heidelberg: Springer Berlin Heidelberg, 1985. — Pp. 22–47. https://doi.org/10.1007/978-3-642-82118-9_3.

14 Schlesinger, Michail. Stop Condition for Subgradient Minimization in Dual Relaxed (Max,+) Problem / Michail Schlesinger, Evgeniy Vodolazskiy, Nikolai Lopatka // Proceedings of the 8th International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition. — EMMCVPR'11. — Berlin, Heidelberg: Springer-Verlag, 2011. — P. 118–131.

15 Shlezinger, M. I. Syntactic analysis of two-dimensional visual signals in the presence of noise / M. I. Shlezinger // *Cybernetics*. — 1976. — Vol. 12, no. 4. — Pp. 622–627.

16 Werner, Tomas. Revisiting the Linear Programming Relaxation Approach to Gibbs Energy Minimization and Weighted Constraint Satisfaction / Tomas Werner // *IEEE*

Transactions on Pattern Analysis and Machine Intelligence. — 2010. — Vol. 32, no. 8. — P. 1474–1488.

17 Chong, E.K.P. An Introduction to Optimization / E.K.P. Chong, S.H. Zak // Wiley Series in Discrete Mathematics and Optimization. — Wiley, 2013. https://books.google.com.ua/books?id=iD5s0iKXHP8C.

18 В. К. Коваль, М. И. Шлезингер. Двумерное программирование в задачах анализа изображений / М. И. Шлезингер В. К. Коваль // *Автоматика и телемеханика*. — 1976.

19 Rossi, Francesca. Handbook of Constraint Programming / Francesca Rossi, Peter van Beek, Toby Walsh. — USA: Elsevier Science Inc., 2006.

20 Водолазский, Е. В. Обобщенные задачи разметки с мажоритарным полиморфизмом для некоторого класса полуколец / Е. В. Водолазский // *Управляющие системы и машины*. — 2015. — no. 6. — Pp. 3–7.

21 Li, Mengtian. Complexity of Discrete Energy Minimization Problems. — 2016.

22 Arora, Sanjeev. Computational Complexity: A Modern Approach / Sanjeev Arora, Boaz Barak. — 1st edition. — USA: Cambridge University Press, 2009.