

פתרון (מקוצר) לתרגיל מס' 2 באופטימיזציה גאומטרית

* הפתרונות המוצגים כאן נועדו לתת קווים מנחים לפתרון, אם כי הם בכל זאת מפורטים במידת מה. יש לציין כי במבחן או בהגשה של תרגיל, לעיתים יש לפרט/לנמק מעט יותר.

1. נמיין את נקודות A ואת נקודות B לפי ערכי ה-y שלהן. בעיית ההכרעה: בהינתן פרמטר d, נרצה לדעת האם קיימים פחות מ-k זוגות של נקודות מקבוצות שונות במרחק לכל היותר d. כדי לפתור את בעיית ההכרעה נעבור על הנקודות של A לפי הסדר, ולכל נקודה נבדוק כמה נקודות מ-B נמצאות במרחק לכל היותר d ממנה. לאחר מכן, נסכם את הערכים שהתקבלו ונשווה את התוצאה ל-k. נשים לב שעבור כל נקודה מ-A, קבוצת הנקודות ה"קרובות" אליה ב-B רציפה ביחס לסדר שלהן. כאשר נרצה למצוא את רצף האיברים הקרובים לנקודה a_1 הראשונה מ-A, פשוט נעבור על כל B על מנת למצוא את האיבר הראשון והאחרון של הרצף הקרוב ל- a_1 . כאשר נרצה למצוא את הרצף המתאים לאיבר הבא, נשתמש בגבולות הרצף של a_1 ונטייל מהם לאורך B עד שנגיע לגבולות החדשים, וכן הלאה. (נשים לב שכל הרצפים מוכלים בקטעים מקוננים זה בזה. לכן גבולות רצף הנקודות עבור הנקודה הבאה מ-A תמיד יתחילו מגבולות הרצף הקודם "כלפי פנימה"). מציאת כל הרצפים תיקח $O(n)$ זמן. כלומר לאחר מיון התחלתי בזמן $O(n \log n)$, ניתן לבצע את פרוצדורת ההכרעה בזמן לינארי.

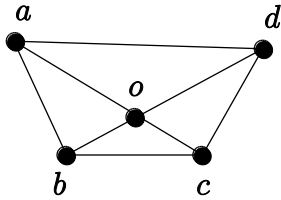
(1) חיפוש פרמטרי - נשתמש ב-n מעבדים, שכל אחד מהם ימצא את גודל הרצף המתאים עבור איבר אחר מ-A. כל מעבד יבצע שני חיפושים בינאריים – אחד למציאת האיבר הראשון ברצף ואחד למציאת האיבר האחרון ברצף. לכן, הפרוצדורה תשתמש ב-n מעבדים ותרוץ ב- $O(\log n)$ זמן.

ניתן להסתכל על חיפוש בינארי יחיד כעל רצף של $\log n$ פעולות השוואה, כך שניתן לבצע השוואה אמ"מ ההשוואה ישירות לפניה כבר בוצעה. כלומר, נוכל להסתכל על כל החיפושים הבינאריים כעל רשת מיון בעומק $O(\log n)$. נשתמש ברשת זו על מנת לבצע וריאנט של האלג' של Cole. נקבל אלג' בסיבוכיות זמן $O(n \log n)$ (מספר לוגריתמי של צעדים, שבכל אחד מהם מריצים פרוצדורת הכרעה אחת בזמן לינארי).

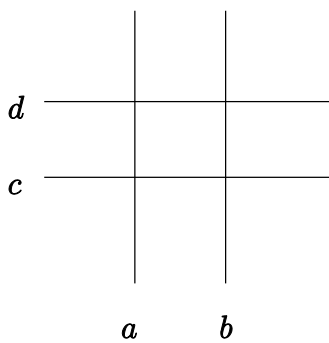
(2) המטריצה M שהאיבר M_{ij} שלה הינו המרחק בין הנקודות a_i, b_j אינה מונוטונית. אך נוכל לפרק אותה לארבע מטריצות מונוטוניות – אחת לכל רביע. כל ערך קריטי נמצא לפחות באחת מארבע המטריצות (הערכים קריטיים הינם מרחקים בין זוגות של נקודות). נריץ על ארבע המטריצות את האלגוריתם לחיפוש מטריצות מונוטוניות בזמן $O(n \log n)$.

(3) כמו שראינו בשאלה 3 בתרגיל הראשון, נבחר באופן אקראי קבוצה R של $r = cn \log n$ זוגות של נקודות מ-A ו-B. נמצא את שני המרחקים d_1, d_2 של זוגות מ-R שהמרחק ה-k נמצא ביניהם, ע"י שימוש בפרוצדורת ההכרעה. בהסתברות גבוהה מספר הזוגות שהמרחק ביניהם נמצא בין d_1 לבין d_2 הינו $O(n)$. ניתן למצוא זוגות אלו בזמן לינארי ע"י שינוי קל של פרוצדורת ההכרעה (תוך כדי הרצת פרוצדורת ההכרעה עבור d_2 , נעבור אחד-אחד על המרחקים הגדולים מ- d_1). כעת נריץ את פרוצדורת ההכרעה ונבדוק מהו מספר הזוגות m שהמרחק ביניהם קטן מ- d_1 . כדי למצוא את הזוג ה-k-m בטווח המבוקש נמיין את הזוגות לפי המרחק ביניהם. תוחלת זמן הריצה של האלג' הינה $O(n \log n)$.

2. (a) נמיין את נקודות החלק העליון של העקום משמאל לימין. בהינתן בסיס ab , נסובב את הצירים כך שהבסיס יקביל לציר ה- x (ועדיין יהיה על המעטפת התחתונה). שטח המשולש המקסימאלי יתקבל עבור הנקודה העליונה ביותר. מקמירות העקום, אנו יודעים שערך ה- y של הנקודות (לאחר סיבוב הצירים) יעלה ואחר כך ירד. לכן, נוכל למצוא את הנקודה הגבוהה ביותר בעזרת חיפוש בינארי.



(c) נסמן ב- o את נקודת החיתוך של ac עם bd . (לשם הפשטות נסמן $area(\Delta) = S(\Delta)$). מתקיים לכל u "מעל" הקונפיגורציה:
 $S(adu) + S(bcu) = S(acu) + S(bdu) + (S(abc) - S(abd))$
 נחשב את שטח המשולשים Δabc ו- Δabd (מחצית הבסיס ab כפול הגובה) - הגובה יהיה המרחק של הקודקוד השלישי (c או d) מהישר שמכיל את ab . קל לראות שהקודקוד הרחוק יותר הינו d , ולכן $S(abd) \leq S(abc)$. נקבל $S(adu) + S(bcu) \leq S(acu) + S(bdu)$.



(b) נגדיר מטריצה M כך שכל שורה וכל עמודה מתאימה לנקודה מ- $P \cap C^-$ (לפי סדר הנקודות על P). אם $a < b$, האיבר M_{ab} יכיל את השטח המקסימלי של משולש שבסיסו הינו הקטע ab , אחרת M_{ab} יכיל את $-\infty$. נרצה להראות ש- M הינה מטריצת Monge הפוכה. לצורך כך נבחר שתי שורות ושני טורים של המטריצה, ונסמן אותם בהתאם לציור משמאל. צריך להראות ש-
 $M_{ac} + M_{bd} \geq M_{ad} + M_{bc}$
 אנו יודעים ש- $a < b$ ו- $c < d$ (לפי סדר הנקודות על P), אך זה מותיר אותנו עם שש פרמוטציות אפשריות של הנקודות. עם המקרה $a < b < c < d$ ניתן להתמודד בקלות ע"פ מה שהוכחנו ב-(c). ניתן לבדוק שאר המקרים נפתרים באופן טריויאלי.

כעת, אנו יודעים ש- M היא totally monotone. נשתמש באלגוריתם שראינו בשיעור על מנת למצוא את האיבר המקסימאלי בה ב- $O(n \log n)$ זמן. האלגוריתם לוקח מספר לינארי של פעולות אך כל אחת לוקחת זמן $O(\log n)$.

3. עבור נקודה x , נסמן ב- x^* את הישר הדואלי ל- x . עבור צלע ab נסמן ב- $l(a,b)$ את הישר העובר דרך ab . במישור הפרימלי, שטח מינימלי של משולש שאחת מצלעותיו היא ab מתקבל ע"י נקודה c שהישר העובר דרכה ומקביל ל- ab הוא הקרוב ביותר ל- $l(a,b)$. $l(a,b)$ עובר, במישור הדואלי, לנקודת החיתוך בין a^* ו- b^* . ישר מקביל ל- ab עובר לנקודה עם אותו שיעור x כמו נקודת חיתוך זו. עבור מספר ישרים המקבילים ל- $l(a,b)$, זה שהכי קרוב ל- $l(a,b)$ מתאים לנקודה (עם אותו שיעור x) שהכי קרובה לנקודת החיתוך. כמו כן, עבור נקודה c , הישר c^* עובר דרך הנקודה שדואלית לישר המקביל ל- $l(a,b)$ שעובר דרך c . לכן, נבנה את מערך הישרים הדואלים לנקודות מ- A, B, C . נעבור על המערך ב-sweep כאשר ב-y-structure נחזיק 3 מבנים שכל אחד מתאים לקבוצה. כאשר ניתקל בנקודת חיתוך של ישרים מ-2 קבוצות שונות, נחפש ב-y-structure המתאים לקבוצה השלישית את הישר שהכי קרוב בכיוון האנכי לנקודת החיתוך. נחשב את שטח המשולש המתאים ונשמור את שטח המשולש המינימלי שמצאנו עד כה. זמן ריצה: $O(n^2 \log n)$. כדי להשתמש בשיטה של chan נחלק כ"א מהקבוצות ל-2 תתי קבוצות בגודל שווה. נקבל שמונה תתי בעיות שבכל אחת $3n/2$ נקודות ופתרון הבעיה המקורית הוא המינימום מבין פתרונות תתי הבעיות. נפתור את הבעיה בתוחלת זמן של $O(n^2 \log n)$.

4. נפתור את הבעיה עבור שלושה מקרים, ולבסוף נמצא את המקסימום מבין הפתרונות. מקרה ראשון: כל אחת מהנקודות שתומכות במלבן נמצאת ברביע שונה.

טענה: נניח שנתונה לנו נקודה ברביע הראשון (רביע ימני עליון) שתומכת במלבן מלמעלה. אזי המלבן הריק המכיל את הראשית הנקבע ע"י נקודה זו הוא יחיד. סקיצת ההוכחה: נניח בשלילה שקיימים שני מלבנים מועמדים R', R'' המקיימים את התנאים. נסמן ב- p', p'' בהתאמה את הנקודה התומכת במלבן משמאל עבור כל אחד מהמלבנים. p', p'' חייבות להיות ברביע השמאלי עליון כיוון שהוא אינו ריק. נניח ש- $x(p') > x(p'')$. כיוון שהנקודה התומכת במלבן מלמטה נמצאת ברביע השמאלי התחתון, p' תהיה מוכלת ב- R'' . סתירה. (כנ"ל לגבי שאר הנקודות התומכות).

לכן במקרה זה נבחן כל נקודה ברביע הראשון כמועמדת להיות תומכת במלבן מלמעלה. עבור נקודה נתונה נמצא את הנקודה התומכת במלבן משמאל ברביע שני ע"י מציאת הנקודה הימנית ביותר שנמצאת מתחת לנקודה זו. ניתן לממש ע"י עץ חיפוש. (וכן הלאה עבור שאר הנקודות התומכות).

באופן דומה יש לבדוק כל נקודה ברביע הראשון כמועמדת להיות נקודה תומכת ימנית. המקרה השני בו 2 מהנקודות התומכות נמצאות באותו רביע ו-2 הנקודות התומכות האחרות נמצאות ברביעים שונים נפתר באופן דומה.

מקרה שלישי: 2 מהנקודות התומכות נמצאות באותו רביע ו-2 הנקודות התומכות האחרות נמצאות ברביע מנוגד.

נניח בה"כ שמדובר ברביע הראשון (הימני העליון) וברביע השמאלי התחתון. נעבור על נקודות הרביע הראשון משמאל לימין. עבור כל נקודה שנעבור בדרך, נבחר אותה אם היא נמצאת מתחת לנקודה התחתונה ביותר שבחרנו עד עכשיו. נעבור על נקודות הרביע השמאלי התחתון מימין לשמאל. עבור כל נקודה שנעבור בדרך, נבחר אותה אם היא נמצאת מעל לנקודה העליונה ביותר שבחרנו עד עכשיו. כעת 2 הקבוצות הממויינות שבחרנו מקיימות את התנאים עבור אלגוריתם ה-SMAWK שראינו בכיתה. יש לנפות מלבנים מועמדים אשר מכילים נקודות מהרביע השמאלי עליון והימני תחתון.

כל אחד מהמקרים נפתר בזמן $O(n \log n)$.