Ben-Gurion University of the Negev

SOLUTION

By

Avi Ferdman

שאלה 1

סעיף א'

ניתן דוגמא נגדית שבה האלגוריתם לא מחזיר פתרון עם מספר מטבעות מינימאלי. נניח כאשר n=30, לפי הגדרת האלגוריתם כל המטבעות אינם גדולים מ- 30 ולכן האלגוריתם יבחר תחילה את המטבע הכי גדול כלומר 25. כעת לאחר הבחירה M=5 ולא נותר לאלגוריתם ברירה אלא לבחור כל פעם מטבע בשווי 1 מכיוון שכל שאר המטבעות חורגים בגודלם. לכן האלגוריתם מחזיר את הקבוצה: {25,1,1,1,1,1} שגודלה הוא 6 לעומת הפתרון האופטימלי במקרה זה שהוא הקבוצה {10,10,10} שגודלה 3.

סעיף ב'

משפט: החמדן מחזיר קבוצה חוקית של מטבעות עם מס' מינימאלי של מטבעות.

טענת העזר: בכל שלב קיים פתרון אופטימלי O המכיל את S.

טענת עזר 2: יהא n סכום חיובי, יהי x_i המטבע הגדול ביותר כך ש: $x_i \le n$ ותהי קבוצה A קבוצה מינימאלית בגודלה כך שסכום האיברים בה גדול או שווה ל- n. אז: $x_i \in A$.

הוכחת המשפט בעזרת טענת העזר:

ריצת החמדן מסתיימת כאשר $M = 0$ ובכל שלב של האלגוריתם אנחנו בוחרים מטבע ומקטינים את M, בנוסף לא יתכן כי M שלילי מכיוון שיש בידינו מטבעות ששויים הוא 1 ומכיוון ש-M תמיד מספר שלם ועפי הגדרת האלגוריתם לא נבחר מטבעות שגודלם גדול מ- M לכן תמיד יהיה לנו מטבע כלשהו חוקי שיקטין את M עד שגודלה יהיה 0, לכן האלגוריתם יעצר. האלגוריתם מחזיר פלט חוקי מכיוון שבכל צעד הוא בוחר מטבע כלשהו ומוסיף אותו לקבוצת המטבעות ולכן בסוף יחזיר קבוצת של מטבעות. על פי טענת העזר, קיים פתרון אופטימלי O המכיל את S בסוף ריצת החמדן. נניח בשלילה כי $S \neq O$ אם כך קיים מטבע ששייך לO ואינו שייך לS. ולכן הסכום של O בהכרח גדול ממש מהסכום של S ולכן נובע כי הסכום ההתחלתי גדול מהסכום של S ובמקרה הזה אנחנו יכולים להוסיף לS לפחות עוד מטבע אחד (שערכו 1) אבל זו בסתירה להנחה שהאלגוריתם סיים את ריצתו כי הם מסיים את רימתו רק כאשר: סכום האיברים בS שווה ל $- n$.

הוכחת טענת העזר: באינדוקציה על מספר המטבעות שהחמדן בחר.

בסיס: בתחילת האלגו', $s = \emptyset$ ולכן כל פתרון אופטימלי מקיים את הטענה.

הנחה: נניח שאחרי $i - 1$ בחירות של מטבעות קיים פתרון אופטימלי O כך שמתקיים: $S_{i-1} \subseteq O$

צעד: נוכיח שאחרי הצעד ה-i קיים פתרון אופטימלי 'O שמכיל את הפתרון שנבנה ב-i צעדים הראשונים.

מקרה 1: המטבע שנבחר בבחירה ה- i נסמנו בx_i שייך לקבוצה $O \backslash S_{i-1}$ אז מצאנו קבוצה 'O שמכילה את S_i והיא הקבוצה O.

מקרה 2: המטבע שנבחר בבחירה ה-i נסמנו בx_i **אינו** שייך לקבוצה $O \backslash S_{i-1}$. לפי הגדרת בחירת האלגוריתם נובע כי הסכום של הקבוצה $O \backslash S_{i-1}$ גדול או שווה ל-x_i (כי אחרת האלגוריתם לא היה בוחר את x_i). בנוסף נובע כי $O \backslash S_{i-1}$ מינימאלית בגודלה, מכיוון שאלמלא כן, נניח בשלילה כי קיימת קבוצה שונה P כך שסכום איבריה שווה לסכום איברי הקבוצה $O \backslash S_{i-1}$ וגודל P קטן מגודל $O \backslash S_{i-1}$ אז היינו יוצרים פתרון אופטימלי קטן יותר מהמינימאלי: $S_{i-1} \cup P$ אבל זו סתירה למינימאליות הפתרון האופטימלי. ולכן לפי טענת העזר 2 נובע כי: $x_i \in O \backslash S_{i-1}$ בסתירה להנחה ולכן מקרה זה אינו אפשרי.

הוכחת טענת עזר 2:

יהא n סכום חיובי, יהי x_i המטבע הגדול ביותר כך ש: $x_i \le n$ ותהי קבוצה A קבוצה מינימאלית בגודלה כך שסכום האיברים בה גדול או שווה ל- n. צ"ל: $x_i \in A$.

נחלק למקרים:

<u>אם $n < 5$:</u> אז כל איברי הקבוצה חייבים להיות מסוג המטבע בשווי 1.

<u>אם $5 \le n < 10$</u>: אז נניח בשלילה שהמטבע 5 אינו שייך לA. אם כך לפי הגדרת הסכום n חייבים להיות בA לפחות 5 מטבעות שוויים הוא 1, ואז ניתן להחליף את חמשתם במטבע אחד ששווי 5 ובכך הקטנו את גודלה של A ב-4 ונקבל סתירה למינימאליות של גודלה של A.

<u>אם $10 \le n < 25$</u>: אז נניח בשלילה שהמטבע 10 אינו שייך לA. ונחלק למקרים:

אם המטבע 5 מופיע לפחות פעמיים: אז נחליף שני מופעים של המטבע 5 במטבע 10 אחד ונקבל סתירה למינימאליות של A.

אם המטבע 5 מופיע לכל היותר פעם אחת: אז שאר המטבעות שיכולים להיות אלה רק מטבעות בשווי 1 וקיימים לפחות 5. לכן נחליף 5 מטבעות שסכומם 1 ועוד מטבע של 5 במטבע אחד של 10 ונקבל סתירה למינימאליות של A.

<u>אם $25 < n$</u>: נחלק למקרים: ובכל מקרה נניח בשלילה שהמטבע 25 אינו שייך לקבוצה A.

אם 10 מופיע לפחות פעמיים: אז אם 5 יופיע נחליף 2 מטבעות של 10 ומטבע אחד של 5 במטבע של 25 טנקבל סתירה למינימאליות. אם 5 לא מופיע אז או שמופיעים לפחות 5 מטבעות של 1 ואז נחליף 5 מטבעות של 1 ושני מטבעות של 10 במטבע אחד של 25 ונקבל סתירה למינימאליות. או שקיימים לפחות 3 מטבעות של 10 ובמקרה כזה נחליף 3 מטבעות של 10 במטבע אחד של 25 ואחד של 5 ונקבל סתירה למינימאליות של A.

אם 10 מופיע לכל היותר פעם אחת: אז או ש5 מופיע לפחות 3 פעמים ואז במקרה כזה נחליף 3 מטבעות של 5 ומטבע 1 של 10 במטבע 1 של 25 ונקבל סתירה למינימאליות. או ש5 מופיע לכל היותר פעמיים. ואז יכולים להיות או 15 או 10 או 5 מטבעות של 1 ובכל מקרה ניתן להחליף כמות של מטבעות (הכמות הזו גדולה מ1) במטבע 1 של 25 ונקבל סתירה למינימאליות של A.

זמן ריצה:

מאתחלים רשימה ריקה שתכיל את המטבעות שנבחרו. בכל איטרציה אנחנו אנחנו מקטינים את הסכום שנדרש להגיע אליו לכל הפחות ב1. בכל איטרציה מבצעים מספר קבוע של פעולות. לבסוף מחזירים את רשימת המטבעות. לכן במקרה הגרוע נעבור n פעמים על האיטרציה ולכן סה"כ זמן הריצה הוא: $O(n)$.

שאלה 2

נתאר אלגוריתם: (S- רשימה מקושרת לפלט, A- רשימה מקושרת דו כיוונית ממויינת לפרוייקטים פוטנציאלים לשיבוץ, D- רשימה מקושרת של הדד ליינים, N- מספר הפרויקטים שבחרנו, j- ההפרש בין שני דד ליינים סמוכים)

הגדר: $S = \emptyset$, $A = \emptyset$, $D = \emptyset$, $M = \{p_1, p_2, \dots, p_n\}$ $N = 0$, $j = 0$

עבור על קבוצת כל הפרוייקטים והכנס את כל זמני הדדן ליין **השונים** לרשימה D בנוסף הוסף לסוף הרשימה את 0 (זהו זמן ההתחלה).

מיין את D מהגדול לקטן.

מיין את M עלפ"י גודל הדדן ליין מהגדול לקטן ובין פרוייקטים עם דדן ליין זהה עפ"י גודל הצעת המחיר מהגדולה לקטנה. (כלומר הראשון ברשימה יהיה זה בעל הדדן ליין הגדול ביותר ומתוך הדד ליינים הגדולים ביותר הזהים הוא בעל העלות הגבוהה ביותר).

עבור באיטרציה על הרשימה D עד האיבר אחד לפני אחרון (לשם נוחות ניעזר באינדקס רץ נסמנו i):

עבור כל הפרוייקטים p המקיימים $p.d = D[i]$ בצע:

אם $p.d > N$, הוסף לרשימה A את p לפי גודל הצעת המחיר מגדול לקטן.

$j = D.get(i) - D.get(i + 1)$

כל עוד $j > 0$ וגם A לא ריקה בצע:

אם $(\mathrm{A}.get(0).d > N)$ בצע:

$\mathrm{S}.add(\mathrm{A}.remove(0))$ (הוסף לתחילת הרשימה)

$N \leftarrow N + 1$

$j = j - 1$

אחרת, בצע:

$\mathrm{A}.remove(0)$

החזר את S

הוכחת נכונות:

משפט: החמדן מחזיר סדרה חוקית של פרוייקטים $\mathrm{P} = (p_{i_1}, \dots, p_{i_l})$ כך ש: $\sum_{j=1}^{l} m_{i_j}$ מקסימלי.

טענת העזר: בכל שלב קיים פתרון אופטימלי O המכיל את S.

הוכחת המשפט ע"י טענת העזר:

בסיום האלגוריתם קיים פתרון אופטימלי O כך ש: $\mathrm{S} \subseteq O$ נראה כי הרווח מS שווה לרווח המקסימלי מ- O. אם $S = O$ סיימנו. לכן נניח כי $S \neq O$ כלומר קיים $p \in O$ כך ש: $p \notin S$. לפי תיאור האלגוריתם נובע כי כל החודשים שנמצאים לפני $p.d$ תפוסים ועלותם גדולה או שווה מעלותו של p כי p נמצא במבנה הנתונים החל מהמעבר על חודש הדד ליין שלו ובכל פעם נשלף הפרוייקט בעל העלות הגבוהה ביותר אז אם הוא לא נשלף לא היה חודש שלא שובץ בו פרוייקט וכל הפרוייקטים ששובצו בחודשים אלה גדולים או שווים בעלותם אליו. בנוסף לאחר הדד ליין p אינו תורם לעלות הכללית ולכן ניתן להסירו מ- O מבלי לפגוע ברווח. הוכחנו זאת על p כלשהו שאינו שייך לS ולכן זה נכון לכל p ששייך ל-O ואינו שייך ל-S ולכן הרווח שלהם שווה.

הוכחת טענת העזר:

הוכחה באינדוקציה על שלבי האלגוריתם, במקרה שלנו: $|S| = \mathrm{l}$ כאשר l זה שלב האלגוריתם שבו אנו נמצאים (נגדיר "שלב" כפקודת ההוספה של החמדן).

בסיס: $S = \emptyset$ כל פתרון אופטימלי O מקיים $S \subseteq O$

הנחה: נניח כי בסיום השלב ה: $l-1$ קיים פתרון אופטימלי O כך ש: $S_{l-1} \subseteq O$

טענת עזר 2: יהי O פתרון אופטימלי המסכים עם החמדן על $l-1$ הבחירות הראשונות של החמדן, יהי p_l הפרוייקט שהחמדן בחר במקום ה-l ויהי t_l פרוייקט בפתרון האופטימלי במקום ה-l מהסוף אזי $p_l.d \le t_l.d$

צעד: צ"ל קיים פתרון אופטימלי O' בסוף השלב ה: l כך ש: $S_l \subseteq O$. יהי p_l הפרוייקט הנבחר בשלב ה-l אזי: $S_l = S_{l-1} \cup \{\mathrm{p}_l\}$. יהי O פתרון אופטימלי כלשהו המסכים עם $l-1$ הבחירות הראשונות של החמדן.

נחלק למקרים:

מקרה א': $\mathrm{p}_l \in O$ וגם p_l נמצא במקום ה-l מהסוף ב-O אזי $S_i \subseteq O$ וניקח $O' = O$.

מקרה ב': $p_l \in O$ וגם p_l **לא** נמצא במקום ה-l כלומר נבחר אחרי הבחירה הl. אם כך בסדרה O נמצא פרוייקט אחר אחרת במקום הl נסמנו p_l' אזי ניצור סדרה חדשה O' ע"י החלפת המיקומים של הפרוייקטים p_l' ו- p_l וזו בהכרח סדרה חוקית כי בסה"כ שיבצנו את p_l במקום שיועד לו ע"י האלגוריתם שזה מקום חוקי בשבילו ואת p_l' הקדמנו את זמן הביצוע שלו. ובסך הכל לא שינינו את העלות הכוללת ולכן היא נשארת מקסימלית וקיבלנו פתרון אופטימלי $S_i \subseteq O'$

מקרה ג': $\mathrm{p}_l \notin O$: נסמן את סדרת הבחירות שלנו ב-$S_l = (p_l, p_{l-1}, \dots, p_1)$ ואת הסייפא ה-l של הפתרון האופטימלי $\mathrm{O}_l = (t_l, p_{l-1}, \dots, p_1)$ נעזר בטענת עזר 2, ונסיק כי
$t_l.d \ge p_l.d$ אם כך לפי תיאור האלגוריתם כאשר בחרנו את הפעילות ה-p_l נובע כי t_l בהכרח היתה במבנה הנתונים שלנו של הפרוייקטים הפוטנציאלים ולכן אם האלגוריתם בחר את p_l נובע כי $t_l.m \le p_l.m$ ובפרט $t_l.m = p_l.m$ כי אחרת היינו מקבלים סתירה לאופטימליות של O ע"י החלפה ביניהם. לכן נובע כי ניתן לשבץ במקום הl ב-O_l את p_l במקום t_l והפתרון $S_i \subseteq \mathrm{O}' = \mathrm{O} \backslash \{t_l\} \cup \{p_l\}$ חוקי (כי הפרוייקט ניתן לשיבוץ שם) ואופטימלי (כי בסה"כ לא שינינו את העלות הכוללת).

הוכחת טענת עזר 2:

יהי O פתרון אופטימלי המסכים עם החמדן על $l-1$, יהי p_l הפרוייקט שהחמדן בחר במקום ה-l ויהי t_l פרוייקט בפתרון האופטימלי במקום ה-l מהסוף נניח בשלילה כי $p_l.d > t_l.d$. נסמן את הסייפא ה-l של הפתרון האופטימלי ב-$\mathrm{O}_l = (t_l, p_{l-1}, \dots, p_1)$. מכיוון שכל פעילות לוקחת חודש בדיוק ב-S נובע כי לכל $1 \le j < i \le l$ מתקיים: $j - i \le p_j.d - p_i.d$. ומההנחה: $p_l.d \ge t_l.d + 1$ לכן נובע שניתן לבצע את הפרוייקט p_l מיד אחרי t_l ואחר כך את $l-1$ הפעילויות בסייפא של הפתרון האופטימלי, ונקבל פתרון רווחי יותר מהאופטימלי בסתירה.

זמן ריצה

נשתמש במבנה נתונים של רשימה מקושרת ומערך של מצביעים לרשימה הזו. כדי שנוכל לבצע הכנסה של איבר ב$O(logn)$ ושליפת האיבר המקסימלי ב-$O(1)$.

סה"כ במהלך האלגוריתם אנחנו ממיינים את רשימת כל הפרוייקטים: $\mathrm{O}(\mathrm{n}\log n)$

כל פרוייקט נכנס לכל היותר פעם אחת לרשימה המקושרת$O(\mathrm{n}\log n)$

כל פרוייקט נשלף לכל היותר פעם אחת מהרשימה המקושרת: $O(1)$

סה"כ ריצת האלגוריתם: $O(n \log n)$

שאלה 3

סעיף א:

יהיו $G = (V, E)$ גרף, פונקציית משקל $W: E \cup \{e\} \longrightarrow R$, $T = (V,\ E_T)$ MST של G ותהי $e \notin E$.

צריך לבנות אלגוריתם שיקבע האם T הוא עפ"מ של הגרף: $G' = (V, E \cup \{e\})$.

הוסף את הצלע e לעץ T.

עבור על המעגל שנוצר בעץ עם הוספת הצלע e.

אם קיימת צלע במשקל גדול ממש מהמשקל של e החזר "שקר".

אחרת, החזר "אמת"

הוכחת נכונות האלגוריתם:

משפט: האלגוריתם מחזיר "אמת" אם T הוא עפ"מ של הגרף $G' = (V, E \cup \{e\})$ אחרת מחזיר "שקר".

טענת עזר:

יהא T עפ"מ של הגרף $G = (V, E)$. T עפ"מ ב$G' = (V, E \cup \{e\})$ אמ"מ e צלע הכבדה ביותר במעגל שיצרה עם צירופה לקבוצת הצלעות של T.

הוכחת המשפט באמצעות טענת העזר:

יהא T עפ"מ של הגרף $G = (V, E)$. אם האלגוריתם מצא כי e היא אינה הכבדה ביותר במעגל שנוצר עם הוספתה, לפי טענת העזר T אינו עץ פורס מינימאלי של $G' = (V, E \cup \{e\})$ ולכן האלגוריתם מחזיר שקר. אחרת אם e היא הכבדה ביותר במעגל שנוצר עם הוספתה עפ"י טענת העזר T עץ פורס מינימאלי של $G' = (V, E \cup \{e\})$ והאלגוריתם מחזיר "אמת".

הוכחת טענת העזר:

יהא T עפ"מ של הגרף $G = (V, E)$.

$\Longleftarrow$נניח כי T עפ"מ ב$G' = (V, E \cup \{e\})$ צ"ל e צלע הכבדה ביותר במעגל שיצרה עם צירופה לקבוצת הצלעות של T. נניח בשלילה כי e אינה הצלע הכבדה ביותר במעגל שיצרה עם צירופה לקבוצת הצלעות של T. נסמן את הצלע הכבדה ביותר ב-e'. כעת לפי טענה שנלמדה בכיתה (יהי $T = (V, E_T)$ עפ"מ של $G = (V, E)$ ותהי $e \in E \backslash E_T$ אז בגרף $H = (V, E_T \cup \{e\})$ קיים מעגל C, $e \in C$ ולכל $e' \in C$ הגרף $T' = (V, (E_T \cup \{e\})\{e'\})$ הינו עץ פורס של G) נוצר עץ פורש ב-$G' = (V, E \cup \{e\})$ נסמנו: $T' = (V, (E_t \cup \{e\}) \backslash \{e'\})$. ונובע:
$\mathrm{W(T') = W(T) + W(e) - W(e') < W(T)}$ בסתירה למינימאליות של T.

$\Longrightarrow$ נניח כי e צלע הכבדה ביותר במעגל שיצרה עם צירופה לקבוצת הצלעות של T צ"ל T עפ"מ ב $G' = (V, E \cup \{e\})$. לפי משפט שנלמד בכיתה (יהי $G = (V, E)$ גרף קשיר, לא מכוון וממושקל. תהי e צלע כבדה ביותר במעגל C כלשהו ב-G אזי קיים עפ"מ T של G שלא מכיל את e) קיים עפ"מ T' ב-$G' = (V, E \cup \{e\})$ שאינו מכיל את e. לכן T' עפ"מ ב-G לכן $W(T) = W(T')$ והם חלים על אותה קבוצת קודקודים V ולכן T עפ"מ ב-$\mathrm{G'}$.

זמן ריצה:

נתחזק שדה max שיחזיק את קשת ואת משקלה. נאתחל אותו בערך דיפולטי. נרוץ בסריקת DFS על הגרף החל מאחד הקודקודים של הקשת e, עם אילוץ יחיד נוסף בהתחלת הסריקה – עליו להתחיל ראשית עם הקודקוד השני שאליו מתחברת הקשת e. באופן כללי האילוץ אינו משנה את זמן הריצה של DFS וגם תחזוק השדה max מכיוון שהם מוסיפים מספר פעולות קבוע בכל איטרציה. נעצור את סריקת ה- DFS ברגע שנגלה מעגל.

האלגוריתם ממשיך עם מספר קבוע של פעולות אחרי סריקת ה-DFS הזו. לכן זמן ריצת האלגוריתם יהיה זמן הריצה של ה- DFS במקרה שלנו הסריקה תיעצר לכל היותר לאחר מעבר על V סדר גודל של V קשתות כי לאחר מעבר על V קשתות לכל היותר ימצא מעגל. לכן זמן הריצה של האלגוריתם הוא: $O(V)$

סעיף ב':

יהי $G = (V, E)$ גרף קשיר פונקציית משקל $W: E \longrightarrow R$. בנוסף, תהי $U \subseteq V$ תת-קבוצה של קודקודים.

נתאר אלגוריתם המוצא MST_U נסמנו T כך שכל הקוד' מהקבוצה U הם עלים ב-T.

עבור המקרים הלא מנוונים (מקרה מנוון הוא $|V| = |U| = 2$ עבור מקרה זה הגרף עצמו הוא MST_U)

1. הגדר: $E' = \emptyset$
2. עבור כל קוד' $u \epsilon U$ בצע:
   2.1 מבין כל הקשתות של u שאינן מקשרות אותו לקודקוד אחר בU בחר את הקלה ביותר נסמנה e. (אם אין כזו החזר לא קיים MST_U)
   2.2 $E' \longleftarrow E' \cup \{e\}$
   2.3 הסר את כל הצלעות ששייכות ל- u
3. נסמן את קבוצת הצלעות החדשות ב- E_2.
4. נריץ DFS לבדיקת קשירות של $G' = (V \backslash U, E_2)$ אם אינו קשיר, החזר לא קיים MST_U.
5. מצא MST בגרף $G' = (V \backslash U, E_2)$ נסמנו ב - $T' = (V', E'_T)$. ע"י שימוש באלגוריתם של Prim.
6. החזר את העץ הפורש על קבוצת הקודקודים כולה $T = (V,\ E'_T \cup E')$.

הוכחת נכונות:

האלגוריתם נכון פרט למקרה מנוון שבו הגרף מורכב מ2 קוד' של U שמחוברים ביניהם בקשת.

סימונים:

יהי G גרף קשיר, $w: E \to R$ פונקציית משקל על צלעות הגרף. תהי $U \subseteq V$,
תהי $E_{V \backslash U} = \{(u, v) | (u, v) \in E \land u, v \notin U\}$
יהי $T' = (V \backslash U, E_2)$ עפ"מ בגרף $G' = (V \backslash U, E_{V \backslash U})$
תהי E_U קבוצת הצלעות שמאגדת את אוסף כל הצלעות כך שלכל קוד' $u \in U$ קיימת קשת אחת ויחידה ב- E_U והיא הקשת המינימאלית מבין הקשתות שמחברות את u לקוד' כלשהו ב-V\U.

משפט:

האלגוריתם מחזיר MST_U של הגרף G אם קיים, אחרת מחזיר כי לא קיים.

טענת עזר:

קיים $T = (V, E_T)$ שהוא MST_U ב-G אם"ם הגרף $G' = (V \backslash U, E_{V \backslash U})$ קשיר, וגם לכל קוד' $u \epsilon U$ קיימת קשת שמחברת אותו לקוד $v \in V \backslash U$.

הבחנה:

לכל עץ פורס $T = (V, E_T)$ ב- $G = (V, E)$ נסמן את קבוצת העלים שלו ב-A, תהי קבוצה $B \subseteq A$ אזי $T' = (V \backslash B, E_{T \backslash B})$ הוא עץ פורס ב-$G' = (V \backslash B, E \backslash \{(u, v) | u \in B\})$. (מפאת חוסר שורות לא נוכיח את הטענה אבל היא יחסית טריוויאלית ההוכחה תראה שהגרף שקיבלנו קשיר וקיימות בו $|V \backslash B| - 1$ צלעות)

<u>הוכחת המשפט עפ"י טענת העזר:</u>

האלגוריתם עוצר כי הוא עובר פעמיים על סדר גודל של קבוצת הקודקודים U ומבצע את האלגוריתם של Prim, שניהם תהליכים סופיים.
נוכיח כי האלגוריתם מחזיר פתרון חוקי כלומר עץ פורס:
<u>גודל קבוצת הצלעות שקיבלנו מההרצה של Prim שווה ל: $|V| |U| - 1$:</u> מכיוון שהאלגוריתם של Prim מחזיר עפ"מ חוקי על קבוצת הקודקודים $V \backslash U$ (והקבוצות זרות) ולאחר מכן אנחנו מוסיפים את קבוצת הקודקודים U ולכל קודקוד ב- U מוסיפים צלע אחת ממנו לקודקוד מהקבוצה $V \backslash U$ (צלע זו אינה קיימת בגרף עדיין) ולכן גודל קבוצת הצלעות שקיבלנו שווה לגודל קבוצת הקודקודים פחות אחד.
<u>כעת נראה כי הגרף קשיר:</u> יהיו $u, v \in V$ נראה כי קיים מסלול ביניהם.

מקרה א: $u, v \in V \backslash U$ אם כך לפי הנכונות של Prim הוא מחזיר עץ פורש מינימאלי ובפרט קיים מסלול ביניהם.

מקרה ב: $u \in U, v \in V\backslash U$ אם כך, לפי אופן הוספת הצלעות לאחר קבלת העפ"מ עפ"י Prim אנו מקשרים כל קוד' ששייך ל-U לקוד, כלשהו ב-U\V. בפרט קיימת הקשת (u, v') כך ש-
$v' \in V\backslash U$. לפי מקרה א' קיים מסלול $P = (v, v_1, \dots, v')$ ולכן נוכל ליצור מסלול בין v ל-u באופן הבא: $\mathrm{P}' =$
$.(v, v_1, \dots, v', \mathrm{u})$

מקרה ג: $u, v \in U$ מלבד המקרים המנוונים, לפי אופן הוספת הצלעות לאחר קבלת העפ"מ עפ"י Prim אנו מקשרים כל קוד' ששייך ל-U לקוד כלשהו ב-U\V. בפרט קיימות הקשתות $(u, v'), (v, v'')$ כך ש- $v', v'' \in V\backslash U$. לפי מקרה א' קיים מסלול $P = (v', v_1, \dots, v'')$ ולכן נוכל ליצור מסלול בין v ל-u באופן הבא: $.\mathrm{P}' = (v, v', v_1 \dots, v'', \mathrm{u})$

מהגדרת הקבוצה E_U כל קוד' $u \in U$ דרגתו היא 1 ולכן הוא עלה. לכן האלגוריתם מחזיר עץ פורש שכל קוד מהקבוצה U הוא עלה נסמנו ב-$T = (V, E_T)$. נותר להוכיח כי משקלו הוא מינימאלי. נניח כי קיים עץ אחר $T'(V, E_T)$ שמשקלו קטן ממש ממשקל T.
נסמן ב$E_{T,U}$ ו-$E_{T',U}$ את קבוצת כל הצלעות כך שקוד מ-$u \in U$ נמצא בהן מתוך T ו-T' בהתאמה. מבחירת הצלעות $E_{T,U}$ באלגוריתם נובע כי: $W(E_{T,U}) \leq W(E_{T',U})$. נסמן את הגרפים שמתקבלים ע" הסרת קבוצת כל הקוד שייכים ל- U מהעצים מתוך T ו-T' בהתאמה ב- $G_{T\backslash U}$ ו-$G_{T'\backslash U}$ בהתאמה. עפ"י ההבחנה נובע כי הם עצים פורסים בגרפים החדשים (ללא הקוד שהוסרו והצלעות שקשורות אליהם). ונובע:
$W(T') = W\left(E_{T',U}\right) + W\left(G_{T'\backslash U}\right) < W(T) = W\left(G_{T\backslash U}\right) + W(E_{T,U})$ אבל זו סתירה למינימאליות המשקל של $G_{T\backslash U}$ כעץ פורס מינימאלי ב-$(V\backslash U, E_{V\backslash U})$ מכיוון שכך הוא נבחר ע"י האלגוריתם.

כעת נניח כי לא קיים $T = (V, E_T)$ שהוא MST_U ב-G. עפ"י טענת העזר נובע כי או ש$G' = (V\backslash U, E_{V\backslash U})$ אינו קשיר או שקיים קוד' $u \in U$ כך שלא קיימת קשת שמחברת אותו לקוד $v \in V\backslash U$. ובכל מקרה כזה האלגוריתם שלנו יחזיר שלא קיים MST_U מכיוון שהוא בודק את התנאים האלה בהוראות 2.1 ו-4 בתיאור האלגוריתם.

הוכחת טענת העזר:

$\Leftarrow$ יהי $T = (V, E_T)$ שהוא MST_U ב-G צ"ל הגרף $G' = (V\backslash U, E_{V\backslash U})$ קשיר, וגם לכל קוד' $u \epsilon U$ קיימת קשת שמחברת אותו לקוד $v \in V\backslash U$. נניח בשלילה שקיים קוד $u \in U$ כך שלא קיימת קשת שמחברת אותו לקוד' $v \in V\backslash U$ או ש-$G' =$ $(V\backslash U, E_{V\backslash U})$ אינו קשיר.

מקרה א': אם נניח בשלילה שקיים קוד $u \in U$ כך שלא קיימת קשת שמחברת אותו לקוד' $v \in V\backslash U$. אז מכיוון ש- u עלה לפי הגדרת MST_U נובע כי הוא מחובר לקוד $u_2 \in U$. ולכן פרט למקרה המנוון u_2 מחובר לקוד' כלשהו אחר אבל זו סתירה כי אז דרגתו תהיה 2 והוא לא יהיה עלה.

מקרה ב': $G' = (V\backslash U, E_{V\backslash U})$ אינו קשיר. אם כך לפי ההבחנה אם נסיר מT את הקבוצה U ואת הקשתות הקשורות אליהם נקבל עץ פורש של הגרף החדש אבל זו סתירה להנחה.

$\Rightarrow$ נניח כי הגרף $G' = (V\backslash U, E_{V\backslash U})$ קשיר, וגם לכל קוד' $u \epsilon U$ קיימת קשת שמחברת אותו לקוד $v \in V\backslash U$ צ"ל קיים $T =$ (V, E_T) שהוא MST_U ב-G. נבנה עץ פורש חוקי נסמנו T' כך שכל קודקוד מהקבוצה U יהיה עלה ולכן אם קיים בהכרח גרף כזה נוכל מבין כל הגרפים החוקיים לבחור אחד כזה מינימאלי ולכן זה יוכיח כי קיים MST_U ב-G. אם הגרף G' קשיר נוכל למצוא בו עץ כלשהו (אפשר למצוא עפ"מ אבל אין הכרח לכך), כעת נחבר כל קוד' $u \in U$ לאחד מהקודקודים בקבוצה $V\backslash U$ ובהכרח יש כזה לפי ההנחה. דרגתו של כל אחד מהקוד ששייכים לקבוצה U הוא 1 ולכן הוא עלה בגרף. נותר להוכיח שהגרף שקיבלנו הוא עץ פורס:

קשירות:

יהיו $u, v \in V$ נראה כי קיים מסלול ביניהם.

מקרה א: $u, v \in V\backslash U$ אם כך העץ T' פורש ובפרט קיים מסלול ביניהם.

מקרה ב: $u \in U, v \in V\backslash U$ אם כך, לפי אופן הוספת הצלעות לאחר קבלת T' אנו מקשרים כל קוד' ששייך ל-U לקוד, כלשהו ב-U\V. בפרט קיימת הקשת (u, v') כך ש-
$v' \in V\backslash U$. לפי מקרה א' קיים מסלול $P = (v, v_1, \dots, v')$ ולכן נוכל ליצור מסלול בין v ל-u באופן הבא: $\mathrm{P}' =$
$.(v, v_1, \dots, v', \mathrm{u})$

מקרה ג: $u, v \in U$ מלבד המקרים המנוונים, לפי אופן הוספת הצלעות לאחר קבלת T' אנו מקשרים כל קוד' ששייך ל-U לקוד, כלשהו ב-$\mathrm{V} \setminus \mathrm{U}$. בפרט קיימות הקשתות $(u, v'), (v, v'')$ כך ש- $v', v'' \in V \setminus U$. לפי מקרה א' קיים מסלול $P = (v', v_1, \dots, v'')$ ולכן נוכל ליצור מסלול בין v ל-u באופן הבא: $\mathrm{P}' = (v, v', v_1 \dots, v'', \mathrm{u})$.

גודל קבוצת הצלעות היא $|V| - 1$: מכיוון שקיבלנו את העץ T' נובע כי גודל קבוצת הצלעות בו היא $|V \setminus U| - 1 = |\mathrm{V}| - |\mathrm{U}| - 1$. כעת עפ"י אופן הוספת הקודקודים מU והצלעות ששייכות אליהן אנו מוסיפים סה"כ $|U|$ קודקודים ו- $|U|$ צלעות, ולכן הגודל של קבוצת הצלעות היא:
$|V| - |U| - 1 + |U| = |V| - 1$.

זמן ריצה:

עוברים על כל קוד' ב-U ועל כל הקשתות שלו: $O(V + E)$

הרצת DFS : $O(|V| + |E|)$

מבצעים אלגוריתם Prim: $O(|E| \log |V|)$

מוסיפים את הקשת המינימאלית לכל קוד U: $O(|V|)$

(הערה: ניתן לשפר את Prim ע"י שימוש בערימת פיבונצ'י ואז זה יהיה $O(|E| + |V| \log V)$)

ובסך הכל זמן הריצה של האלגוריתם הוא: $O(|E| \log V)$ (אם לא משתמשים בפיבונצ'י אחרת, יהיה כמו בהערה).