רדוקציה – כלי לפתרון בעיות אלגוריתמיות
(כתיבה-ד"ר מיכל ארמוני)

פתרון בעיה נתונה על-ידי **רדוקציה** משמעותה 'תרגום' הבעיה הנתונה לכמה בעיות (אולי אחת) קלות יותר לפתרון, או שפתרונן ידוע כבר, ושימוש בפתרונן של בעיות אלו בתהליך פתרונה של הבעיה הנתונה.

נתבונן למשל בבעיה הבאה:

הציעו אלגוריתם המחשב באופן יעיל את סכום המספרים השלמים בין 1 ל-101 שאינן מתחלקים ב-3.

הפתרון הישיר המתבקש הוא מעבר בלולאה על כל המספרים בין 1 ל-101, בדיקה עבור כל אחד מהם האם הוא מתחלק ב- 3 ואם לא, הוספתו לסכום המצטבר. פתרון זה אינו יעיל ביותר – עלותו ליניארית בתחום המספרים. כלומר, אם היינו נדרשים לפתור את הבעיה על תחום המספרים בין 1 ל- n סיבוכיות הפתרון הזה היא $O(n)$. ניתן כמובן לשפרו מעט. למשל, להתחיל מ- 1, ולבצע רק 33 סיבובים בלולאה, כשבכל סיבוב אנו מוסיפים לסכום המצטבר את מספר הסיבוב כשהוא מוכפל ב- 3, ואת המספר העוקב לו (וודאו כי הנכם מבינים את הפתרון השני). עם זאת, השיפור שמושג ע"י הפתרון השני אינו שיפור בסדר גודל: עדיין עלות הפתרון היא $O(n)$.

אבל, התבוננות מעמיקה יותר בבעיה יכולה להוביל אותנו לפתרון יעיל יותר משמעותית. את סכום המספרים בין 1 ל-101 שאינם מתחלקים ב-3 ניתן להציג כהפרש בין סכום המספרים בין 1 ל-101 ובין סכום המספרים בין 1 ל- 101 אשר מתחלקים ב- 3. הן סידרת המספרים בין 1 ל- 101 והן סידרת המספרים בין 1 ל-101 שמתחלקים ב- 3 (3, 6, 9, ..., 99) הן סדרות חשבוניות. לפיכך ניתן לחשב כל אחד מהסכומים ע"י הצבה בנוסחת סכום סידרה חשבונית, חישוב שעלותו קבועה, כלומר, $O(1)$.

תרגמנו את הבעיה הנתונה לבעיה של חישוב סכום סידרה חשבונית, או כמו שמקובל לומר, ביצענו רדוקציה של הבעיה הנתונה אל בעיית סכום סידרה חשבונית. אם נניח שיש בידינו אלגוריתם ArithmeticSeries(first, diff, num) אשר מקבל כקלט איבר ראשון בסידרה חשבונית, את ההפרש בין איבר לאיבר בסידרה ואת מספר האיברים בסידרה שאת סכומם אנו מעוניינים לחשב, ונותן כפלט את סכום האיברים בסידרה המבוקשת, נוכל בעזרתו לפתור את הבעיה הנתונה באופן הבא:

הפעל את האלגוריתם ArithmeticSeries פעם אחת עם הקלטים first=1, diff=1, num=101 ופעם אחת עם הקלטים first=3, diff=3, num=33 והחזר את הפרש הערכים שקיבלת בשתי ההפעלות.

זו אינה הרדוקציה האפשרית היחידה. האם תוכלו לחשוב על פתרון שונה מעט שמשתמש אף הוא ברדוקציה לבעיית חישוב סכום סידרה חשבונית?

שימו לב שהפתרון שכתבנו משתמש באלגוריתם ArithmeticSeries כקופסה שחורה. הפתרון שלנו מניח שהאלגוריתם ArithmeticSeries קיים ושהוא נותן את התשובה הנכונה. ייתכן שאכן ידוע לנו

כבר שקיים אלגוריתם כזה (שמציב פשוט את הערכים שקיבל בנוסחה המתימטית המתאימה, בסיבוכיות קבועה) וידועה לנו סיבוכיותו. במקרה זה אנו יכולים פשוט להסתמך על כך ולנתח את סיבוכיות הפתרון שלנו כתלות בסיבוכיות האלגוריתם ArithmeticSeries. ייתכן כי איננו מכירים אלגוריתם כזה, או כי איננו זוכרים אותו ואין לנו דרך לשחזרו. גם במקרה זה הרדוקציה מועילה, משום שהיא מאפשרת לנו להתמודד עם הבעיה בשלבים: ראשית אנו פותרים את הבעיה הנתונה מתוך הנחה שכבר פתרנו את בעיית חישוב סכום סידרה חשבונית, ואחר-כך אנו פונים להתמודד עם הבעיה של חישוב סכום סידרה חשבונית.

למעשה, סביר מאוד להניח שזו אינה הפעם הראשונה בה אתם נתקלים באסטרטגיית פתרון רדוקטיבית, אם כי ייתכן שלא נתקלתם בשמה מפורשות. בכל פעם שפתרתם בעיה אלגוריתמית בצורה מודולרית (בין אם בפרדיגמה פרוצדורלית, על-ידי פירוק לתת-משימות שמומשו בעזרת פרוצדורות או פונקציות, ובין אם בפרדיגמה מונחית עצמים, על ידי פירוק העולם למחלקות), ביצעתם תרגום של הבעיה (תרגום שהתבטא בפירוק) לבעיות קטנות יותר, קלות יותר לפתרון (משום תחומן המצומצם) או שפתרונן כבר ידוע (למשל, כאשר השתמשתם במחלקות ספריה או בפונקציות ספריה)

מאחר שפתרון המשתמש ברדוקציה ('פתרון רדוקטיבי') משתמש באבני בניין ידועות, פתרונות רדוקטיביים משרים בדרך כלל פתרונות מורכבים פחות מפתרונות ישירים. למשל, אם במטרה לבנות אלגוריתם הפותר בעיה אלגוריתמית נתונה A מבצעים רדוקציה לבעיה אלגוריתמית אחרת B, ומשתמשים באלגוריתם הפותר את בעיה B כקופסה שחורה, הרי שניתן להסתמך על נכונותו של האלגוריתם הפותר את B. לעומת זאת, אם נבנה אלגוריתם חדש עבור A, יש להוכיח את נכונותם של כל מרכיבי האלגוריתם החדש (גם אם הוא מבוסס במידה רבה על האלגוריתם הידוע הפותר את B, ורק משנה אותו).

נדגים זאת בעזרת בעיה נוספת. נניח שאנו מעוניינים למצוא, בהינתן לנו רשימת מספרים טבעיים שונים זה מזה, את המספר השני בגודלו ברשימה. נוכל כמובן למצוא זאת בעזרת שני משתנים – max ו-second_max – שאת ערכם נעדכן תוך כדי מעבר על הרשימה. פתרון אחר, רדוקטיבי, יסתמך על ההנחה שאנו כבר יודעים למצוא איבר מקסימלי ברשימה, ויפתור את הבעיה הנתונה ע"י רדוקציה לבעיית מציאת איבר מקסימלי ברשימת מספרים. האלגוריתם למציאת מקסימום שני ייתן את הרשימה הנתונה כקלט לאלגוריתם שמוצא איבר מקסימלי ברשימה, יאפס זמנית את ערכו של האיבר המקסימלי שנמצא, ואת הרשימה החדשה ייתן שוב כקלט לאלגוריתם שמוצא איבר מקסימלי ברשימה. הפלט של ההפעלה השנייה יהיה הפלט של האלגוריתם כולו.

הפתרון הרדוקטיבי שלהלן אינו יעיל יותר מהפתרון הישיר, אך הוא מאפשר לנו לפתור את הבעיה בלי להתייחס לפרטים של מציאת מקסימום ברשימה, איתם ניתן להתמודד, אם יש בכך צורך, בשלב מאוחר יותר. כמובן, ניתן גם לבחון את האלגוריתם של מציאת מקסימום ברשימה (שמאתחל משתנה max לאיבר הראשון ברשימה, ומעדכן אותו תוך מעבר על הרשימה עד סופה), ולשנות אותו כך שיחשב גם את second_max (ע"י הוספת משתנה נוסף, אתחול שלו, ועדכון שלו תוך כדי מעבר על

הרשימה), אלא שבמקרה זה אין אנו יכולים כבר להסתמך על נכונות האלגוריתם לחישוב איבר מקסימלי, משום ששינינו אותו, אלא עלינו להוכיח באופן מלא את נכונות האלגוריתם החדש שהתקבל. בנוסף, ייתכן שנטעה בחלק משינויים שערכנו (למשל, לא נבצע אתחול נכון, או שלא נטפל נכוון במקרה שבו האיבר התורן שנבדק קטן מהערך השמור במשתנה max אך גדול מהערך השמור במשתנה second_max). ככל שהאלגוריתם אותו אנו משנים מורכב יותר, או שהשינויים שאנו עורכים בו מורכבים יותר, כך תקשה עלינו מלאכת הוכחת הנכונות המחודשת.

אם כך, פתרון רדוקטיבי נכון כולל כמה מרכיבים:

- זיהוי הבעיה (או הבעיות) אליהן מבצעים רדוקציה. לשם פשטות הניסוח בהמשך נניח שמדובר בבעיה אחת. זהו שלב לא פשוט, שלפעמים מקשר בין בעיות שבמבט ראשון נראות שונות לחלוטין זו מזו.
- תרגום של הקלט הנתון לבעיה לקלט לבעיה שאליה מבצעים רדוקציה.
- תרגום הפלט של הבעיה שאליה אנו מבצעים רדוקציה לפלט לבעיה הנתונה.
- הוכחת הנכונות תוך הסתמכות על נכונותו של פתרון לבעיה אליה ביצענו רדוקציה. הוכחת הנכונות צריכה להתמקד בנכונות התרגומים.
- ניתוח סיבוכיות תוך הסתמכות על סיבוכיותו הידועה של פתרון לבעיה אליה ביצענו רדוקציה.

מושג שעומד בבסיסה של אסטרטגיית פתרון רדוקטיבית הוא **קופסה שחורה**. הפתרון הרדוקטיבי משתמש בפתרון לבעיה אליה בוצעה הרדוקציה בלי להתייחס לפרטיו, בדיוק כמו שבתכנית מודולרית אנו משתמשים בפונקציות בלי להסתמך על המימוש שלהן.

נדגים את השימוש ברדוקציה לפתרון בעיות אלגוריתמיות שונות. הבעיות מחולקות לפי פרקי הקורס, ומומלץ כי תקראו כל סעיף אחרי שסיימתם לקרוא את הפרקים אליהם הוא מתייחס בחומר הלימוד, אך לפני שאתם פונים לפתרון המטלה המתאימה.

פרקים א'+ב' וממ"ן 11

שאלה 1

כתבו אלגוריתם יעיל ככל שתוכלו אשר מקבל כקלט גרף לא מכוון $G=(V,E)$ עם פונקציית משקל $w:E\to\{1,2\}$ ושני צמתים $s,t\in V$ ומוצא משקל מסלול קצר ביותר בין s ל- t ב- G.

הוכיחו את נכונות האלגוריתם ונתח את סיבוכיותו.

תשובה

נפתור את הבעיה על ידי רדוקציה לבעיית משקל מסלול קצר ביותר בגרף חסר משקלות.

ראשית, נבנה מהגרף הנתון G גרף G' באופן הבא: עבור כל קשת $e=(u, v)$ שמשקלה 2 נוסיף לקבוצת הצמתים של הגרף צומת v^e ונוסיף לקבוצת הקשתות את הקשתות (u,v^e) ו- (v^e,v). משקל הקשתות החדשות יהיה 1. כעת קיבלנו גרף שלכל קשתותיו משקל 1, ולכן ניתן להתעלם ממשקלות הקשתות. עתה נמצא בגרף החדש משקל מסלול קצר ביותר מ-s ונחזיר את $d(t)$. כידוע, ניתן לעשות זאת (ע"י BFS) בסיבוכיות $O(|E'|+|V'|)$, כאשר V' ו-E' הן קבוצת הצמתים וקבוצת הקשתות של הגרף החדש, בהתאמה. נשים לב שגודל קבוצת הצמתים בגרף החדש הוא לכל היותר $|V|+|E|$ וגודל קבוצת הקשתות הוא לכל היותר $2\cdot|E|$ ולכן בסה"כ <u>סיבוכיות</u> האלגוריתם היא $O(|E|+|V|)$.

<u>נכונות</u>: יש להראות (זוהי הוכחה דו-כיוונית) כי כל מסלול p בגרף המקורי מתאים למסלול p' בגרף החדש כך שאורכו של p' שווה למשקלו של p, ולהיפך. בכיוון הראשון: יהי $p=v_1,...,v_k$ כאשר $v_1=s$ ו- $v_k=t$ מסלול בין s ל- t בגרף המקורי. נבנה ממנו מסלול בגרף החדש: לכל i ($1\le i<k$) כך ש- $w(e_i=(v_i,v_{i+1}))=2$ נחליף את הקשת $e_i=(v_i,v_{i+1})$ ב סידרת הקשתות $(v_i,v^{e_i}),(v^{e_i},v_{i+1})$. קל לראות שהמסלול המתקבל הוא מסלול ב-G' שאורכו כמשקל המסלול המקורי p והוא מסלול מ-s ל-t.

בכיוון השני: יהי $p'=v_1,...,v_k$ כאשר $v_1=s$ ו- $v_k=t$ מסלול בין s ל-t ב-G'. לכל זוג קשתות סמוכות $(v_i,v^{e_i}),(v^{e_i},v_{i+1})$, ($1\le i<k$) קיימת בגרף המקורי, מבניית הגרף G', הקשת $e_i=(v_i,v_{i+1})$ שמשקלה 2. נחליף את שתי הקשתות הללו בקשת $e_i=(v_i,v_{i+1})$. קל לראות שהמסלול המתקבל הוא מסלול ב- G' שאורכו כ משקל המסלול המקורי p והוא מסלול מ- s ל- t. מההתאמה החח"ע בין המסלולים שהוכחנו כעת נובע כי אורכו של מסלול קצר ביותר בין s ל- t בגרף החדש שווה ל משקלו של מסלול קצר ביותר בין s ל-t בגרף המקורי.

שאלה 2

נתון גרף מכוון $G = (V, E)$ עם משקלות אי-שליליים ל **צמתים** (כלומר, לכל צומת $v\in V$ יש משקל $w(v)\ge 0$). אורכו של מסלול $s\to v_1\to v_2\to\dots\to v_\ell$ הוא $\sum_{i=1}^{\ell} w(v_i)$.

תארו אלגוריתם יעיל ככל שתוכלו, אשר מקבל כקלט $G = (V, E)$ כנ"ל ושני צמתים $s, t \in V$, ומחשב אורך מסלול קצר ביותר בין s ל-t בגרף. נתחו את סיבוכיות האלגוריתם והוכיחו את נכונותו.

<u>תשובה</u>

נראה שני פתרונות רדוקטיביים, שניהם מתבססים על רדוקציה לבעיית חישוב אורך מסלול קצר ביותר בגרף בעל משקלות אי-שליליים לקשתות. הפתרון הראשון משאיר את הגרף ללא שינוי אך מגדיר בו פונקציית משקל על הקשתות:

א. לכל $(u,v) \in E$ נגדיר $w'(u,v) = w(v)$.

ב. חשב בגרף עם פונקציית המשקל w' לכל צומת v את משקלו של מסלול קצר ביותר מ-s ל-v.

<u>סיבוכיות</u>: נבצע את (א) בזמן $O(|V|+|E|)$ ואת (ב) בזמן $O(|E|+|V|\log|V|)$ (בעזרת האלגוריתם של דייקסטרא). סה"כ הסבוכיות היא $O(|E|+|V|\log|V|)$.

<u>נכונות</u>: לכל מסלול $p = s(=v_0) \to v_1 \to \ldots \to v_\ell$ מתקיים: $w(p) = \sum_{i=1}^{\ell} w(v_i) = \sum_{i=1}^{\ell} w'(v_{i-1}, v_i) = w'(p)$.

כלומר, משקלי המסלולים זהים לפי שתי פונקציות המשקל. האלגוריתם של דייקסטרא ימצא מסלול קצר ביותר ע"פ w' וזהו גם מסלול קצר ביותר ע"פ w.

נתאר בקצרה פתרון רדוקטיבי שני במהלכו נבנה גרף חדש בו כל צומת v מהגרף המקורי מתפצל לשני צמתים חדשים v_{in} ו- v_{out} וביניהם קשת שמשקלה $w(v)$. כל קשת (v_i, v_j) בגרף המקורי מיתרגמת בגרף החדש לקשת $(v_{i_{out}}, v_{j_{in}})$ שמשקלה 0. בגרף החדש נחשב לכל צומת v את משקלו של מסלול קצר ביותר מ- s_{out} ל- v_{in} <u>בסיבוכיות</u> $O(|E|+|V|\log|V|)$ (בעזרת האלגוריתם של דיקסטרא) וזהו גם משקלו של מסלול קצר ביותר מ- s ל-v בגרף המקורי. גם עבור פתרון זה יש להוכיח <u>נכונות</u> בעזרת הוכחה דו-כיוונית המראה (בדומה למה שנעשה בהוכחת הפתרון לשאלה 1) התאמה חד-חד-ערכית, המשמרת משקל, בין מסלול בין s ל-v בגרף המקורי לבין מסלול בין s_{out} ל- v_{in} בגרף החדש.

<u>שאלה 3</u>

גרף **דו-צבעי** הוא גרף $G=(V,E)$ שבו קבוצת הקשתות E מורכבת משתי קבוצות של קשתות: קשתות אדומות וקשתות כחולות.

כתבו אלגוריתם המקבל כקלט גרף לא מכוון דו-צבעי $G=(V,E)$, ושני צמתים s ו-t ב- V ומוצא ב- G מסלול בין s ל-t שאורכו קצר ביותר מבין כל המסלולים העוברים לכל היותר בקשת אדומה אחת. הוכיחו את נכונות האלגוריתמים ונתחו את סיבוכיותו.

<u>תשובה</u>

נפתור את הבעיה על ידי רדוקציה לבעיית מציאת מסלול קצר ביותר בגרף מכוון ללא צביעה של הקשתות. נבנה מ-G גרף $G'=(V', E')$ ללא פונקצית צביעה באופן הבא: ראשית נהפוך את G לגרף מכוון ע"י כך שנחליף כל קשת (u, v) ב-G בשתי קשתות מכוונות אנטי-מקבילות (u, v) ו-(v, u) שצבען כצבע הקשת המקורית (u, v). מעתה, כל התייחסות לגרף $G=(V, E)$ כוונתה היא לגרף המכוון. נבנה גרף $G^1=(V^1,E^1)$ כך ש- $V^1=V$ ו- E^1 מכילה את כל הקשתות הכחולות מ-E ורק אותן. נבנה גרף נוסף $G^2=(V^2,E^2)$ שהוא שכפול של G^1. נגדיר: $V'=V^1\cup V^2$, ו- $E'=E^1\cup E^2$ ובנוסף, אם בגרף G יש קשת (u, v) אדומה נוסיף ל-E' את הקשת (u^1,v^2).

העלות של כל אחת מהבניות שתוארו כאן היא $O(|V|+|E|)$.

כעת נמצא את אורכו של מסלול קצר ביותר מ- s^1 לכל הצמתים בגרף החדש (s^1 הוא הצומת המתאים ל-s ב-V^1). אורך המסלול הקצר ביותר מ-s ל-t מבין כל המסלולים העוברים לכל היותר בקשת אדומה אחת הוא $\min(d(t^1),d(t^2))$. את המסלול עצמו נשחזר מ- t^1 או מ- t^2, בהתאמה לתוצאת המינימום.

<u>סיבוכיות:</u> עלות מציאת אורכו של מסלול קצר ביותר מ- s^1 לכל הצמתים בגרף החדש היא כמובן $O(|V|+|E|)$ (בעזרת BFS) ועלות שחזור המסלול היא $O(|V|)$. עלות האלגוריתם כולו היא לכן $O(|V|+|E|)$.

<u>נכונות:</u> נראה כי לכל מסלול חוקי בין s ל-t בגרף המקורי מתאים מסלול באותו אורך בין s^1 ו- t^1 או t^2 בגרף החדש, ולהיפך. בכיוון הראשון: יהי $p=v_1,...,v_k$ כאשר $v_1=s$ ו- $v_k=t$ מסלול חוקי בין s ל-t בגרף המקורי, כלומר, במסלול זה אין יותר מקשת אדומה אחת. נבנה ממנו מסלול בגרף החדש: אם קיים i $(1\le i<k)$ כך ש- (v_i,v_{i+1}) היא קשת אדומה (ברור שיש לכל היותר i אחד כזה), אז את הקשת (v_i,v_{i+1}) נחליף ב- (v_i^1,v_{i+1}^2). את כל הקשתות (v_j,v_{j+1}) כך ש- $j<i$ נחליף ב- (v_j^1,v_{j+1}^1) ואת כל הקשתות (v_j,v_{j+1}) כך ש- $j>i$ נחליף ב- (v_j^2,v_{j+1}^2). אם במסלול המקורי אין אף קשת אדומה אז נחליף בו כל קשת (v_i,v_{i+1}) בקשת (v_i^1,v_{i+1}^1). קל לראות שהמסלול המתקבל הוא מסלול ב-G' שאורכו כאורך המסלול המקורי p והוא מסלול מ- s^1 ל- t^1 או t^2. בכיוון השני: יהי $p'=v_1^{\ell_1},...,v_k^{\ell_k}$ ($\ell_i\in\{1,2\}$, $v_1=s$, $\ell_1=1$ ו- $v_k=t$) מסלול בין s^1 ל- t^{ℓ_k} ב-G'. ממבנה הגרף G' נובע שישנן שתי אפשרויות: או שבמסלול זה לכל i $\ell_i=1$ או שקיים $i<k$ אחד בדיוק כך ש- $\ell_i=1$ ולכל $i<j$ $\ell_j=2$. זאת משום שאין בגרף G' קשתות מ- V^2 אל V^1, ולכן או שהמסלול נשאר כל הזמן ב- V^1 או שהוא עובר פעם אחת ל- V^2 ונשאר שם. מבניית הגרף G' נובע שבמסלול המתאים ב-G $p=v_1,...,v_k$ יש לכל היותר קשת אדומה אחת וברור שהוא מסלול מ-s ל-t בגרף המקורי ואורכו שווה לאורך p'.

מההתאמה החח"ע בין המסלולים שהוכחנו כעת נובע כי אורכו של מסלול קצר ביותר בין s^1 ל- t^1 או t^2 בגרף החדש שווה לאורכו של מסלול חוקי קצר ביותר בין s ל-t בגרף המקורי.

שאלה 4

כתבו אלגוריתם יעיל ככל שתוכלו אשר מקבל כקלט גרף לא מכוון וקשיר $G=(V,E)$ בו כל קשת $e\in E$ מסומנת ב-A, B או C, ומקבל גם שני צמתים $s, t\in V$. האלגוריתם צריך לתת כפלט מסלול בין s ל-t, אשר מתחיל בקשת המסומנת ב- B ומסתיים בקשת המסומנת ב- C, ואשר מספר הקשתות בו המסומנות ב-A הוא מינימלי ביחס לכל המסלולים בין s ל-t ב-G אשר מתחילים בקשת המסומנת ב-B ומסתיימים בקשת המסומנת ב-C.
הוכיחו את נכונות האלגוריתם ונתחו את סיבוכיותו.

תשובה

נראה שני פתרונות, המשתמשים שניהם ברדוקציה. האחד "מתבקש" יותר ונעזר ברדוקציה לבעיה של מציאת מסלולים קצרים בגרף עם משקלות אי-שליליים. השני מבצע רדוקציה מעט יותר מסובכת לבעיה של מציאת מסלולים קצרים בגרף חסר משקלות, ומניב אלגוריתם יעיל יותר.
השלב הראשון ברדוקציה יהיה משותף לשני הפתרונות ומטרתו להתמודד עם הדרישה שהמסלול המבוקש חייב להתחיל בקשת המסומנת ב- B ולהסתיים בקשת המסומנת ב- C. שימו לב כי לא יהיה זה נכון למחוק קשתות היוצאות מ-s שאינן מסומנות ב-B וקשתות הנכנסות ל-t שאינן מסומנות ב-C. בכך אנו עלולים לאבד מסלולים חוקיים. למשל, בגרף המכיל ארבעה צמתים s, u, w ו-t ואת הקשתות (s, u) המסומנת ב-B, (s, w) המסומנת ב-A, ו-(w, t) המסומנת ב-C, יש מסלול חוקי באורך 4 מ-s ל-t: $suswt$, אבל אם נמחק את הקשת (s, w) לא יהיה בכלל מסלול מ-s ל-t. אנו רואים כי יש להיזהר בתרגום הקלט לבעיה הנתונה לקלט לבעיה אליה אנו מבצעים את הרדוקציה.
אם כך, נבנה גרף חדש $G'=(V', E')$ באופן הבא:
נוסיף צומת חדש s' וצומת חדש t', כלומר, $V'=V\cup\{s', t'\}$. כעת נוסיף קשתות המחברות את הצמתים החדשים:

$$E'=E\cup\{(s', v) \mid v\in V, (s, v)\in E \text{ ומסומנת ב-} B\} \cup\{(v, t') \mid v\in V, (v, t)\in E \text{ ומסומנת ב-} C\}$$

נגדיר על הגרף G' פונקצית משקל w: לכל קשת e שנמצאת בגרף המקורי ומסומנת בו ב-A $w(e)=1$. לכל קשת אחרת e $w(e)=0$. כעת נמצא בגרף G' מסלול קצר ביותר מ- s' ל-t'. נחליף במסלול זה את s' ב-s ואת t' ב-t, ונחזיר את התוצאה.

<u>סיבוכיות:</u> עלות הבנייה היא $O(|V|+|E|)$. מציאת מסלול קצר ביותר ב- G' ניתן לבצע על ידי האלגוריתם של דיקסטרא בעלות $O(|V|\log|V|+|E|)$. סה"כ הסיבוכיות היא $O(|V|\log|V|+|E|)$.

<u>נכונות:</u> בכיוון הראשון נראה כי לכל מסלול חוקי p בגרף המקורי מתאים בגרף החדש מסלול p' מ-s' ל-t' כך ש-$w(p')$ הוא בדיוק מספר הקשתות המסומנות ב-A ב-p. יהי p מסלול חוקי בגרף המקורי.

מאחר ש-p חוקי הוא בהכרח מתחיל בקשת (s, v) המסומנת ב-B ולכן בגרף החדש קיימת הקשת (s', v) ומשקלה 0. בדומה, p מסתיים בקשת (u, t) המסומנת ב-C ולכן בגרף החדש קיימת הקשת (u, t') ומשקלה 0. מאחר שכל קשתות E נמצאות גם ב-E' אז כל התת-מסלול של p מ-v ל-u נמצא גם ב-G' ועל״פ הגדרת w משקלו ב-G' שווה למספר הקשתות המסומנות בו ב-A ב-G. לכן בסך-הכל יש ב-G' מסלול p' מ-s' ל-t' שמשקלו כמספר הקשתות המסומנות ב-A ב-p.

בכיוון השני, יהי p' מסלול מ-s' ל-t' בגרף החדש. מבניית הגרף החדש בהכרח p' מתחיל בקשת (s', v) ומסתיים בקשת (u, t') כך ש-$u, v\in V$, יש ב-E קשת (s, v) המסומנת ב-B וקשת (u, t) המסומנת ב-C ויש ב-G מסלול מ-v ל-u שמתקבל בדיוק מ-p' ע״י הסרת הקשת הראשונה והאחרונה. לכן יש ב-G מסלול מ-s ל-t כך שמשקל הקשתות בו שמסומנות ב-A שווה בדיוק ל-$w(p')$.

ממה שהוכחנו לעיל נובע כי מספר הקשתות המסומנות ב-A במסלול חוקי מ-s ל-t ב-G עם מספר מינימלי של קשתות המסומנות ב-A שווה בדיוק למשקלו של מסלול קצר ביותר מ-s' ל-t' בגרף החדש.

כעת נראה כיצד ניתן להגיע לפתרון יעיל יותר, המבצע רדוקציה לבעיה של מציאת מסלולים קצרים ביותר ממקור נתון בגרף לא ממושקל. היינו רוצים להגיע לגרף שבו מסלולים מ-s' ל-t' מכילים בדיוק רק את הקשתות המסומנות ב-A, ולכן מסלול קצר ביותר בגרף כזה מותאם למסלולים בעלי מספר מינימלי של קשתות המסומנות ב-A בגרף המקורי. נשיג זאת ע״י התרגום הבא:

ראשית, בדיוק כמו קודם, נוסיף לגרף צמתים חדשים s' ו-t' וקשתות המחברות אותן לגרף המקורי. כעת נסיר מהגרף את כל הקשתות שמסומנות ב-A בגרף המקורי. אם בגרף החדש יש מסלול מ-s'ל-t' (כלומר, הגרף החדש קשיר, או ש-s' ו-t' באותו רכיב קשירות) אז ברור שיש בגרף המקורי מסלול חוקי מ-s ל-t שכלל אינו מכיל קשתות המסומנות ב-A ולכן ניתן פשוט למצוא מסלול מ-s' ל-t' ולהחזיר את המסלול המתקבל ממנו אחרי שמחליפים את s' ב-s ואת t' ב-t. ברור שמשקל הקשתות המסומנות ב-A במסלול הזה (ששווה ל-0) הוא מינימלי. אם בגרף החדש s' ו-t' אינם באותו רכיב קשירות נמשיך את התרגום באופן הבא:

יהיו $V_1, \ldots, V_n$ רכיבי הקשירות המתקבלים מהגרף המקורי ע״י הסרת כל הקשתות המסומנות ב-A. הגרף החדש יכיל את הצמתים s', t' וצומת v_i עבור כל רכיב V_i. יהיו v_i ו-v_j $(i\neq j)$ שני צמתים בגרף החדש. אם בגרף המקורי קיימת קשת (u, v) המסומנת ב-A כך ש-$u\in V_i$ ו-$v\in V_j$ נוסיף בגרף החדש את הקשת (v_i, v_j). החיבור בין s' ו-t' לשאר צמתי הגרף החדש ייעשה באופן הבא: עבור כל קשת (s, v) המסומנת ב-B בגרף המקורי, נוסיף קשת (s', v_i) כך ש-V_i הוא רכיב הקשירות המכיל את v, ועבור כל קשת (u, t) המסומנת ב-C בגרף המקורי, נוסיף קשת (v_j, t') כך ש-V_j הוא רכיב הקשירות המכיל את u. בגרף החדש נמצא מסלול קצר ביותר בין s' ל-t'. את המסלול שקיבלנו נתרגם למסלול בגרף המקורי באופן הבא:

את הקשת הראשונה (s', v_i) נחליף בקשת (s, v) מהגרף המקורי, כך ש-$v\in V_i$ ו-(s, v) מסומנת ב-B. את הקשת האחרונה (v_j, t') נחליף בקשת (u, t) מהגרף המקורי, כך ש-$u\in V_j$ ו-(u, t) מסומנת ב-C. כל

קשת (v_i, v_j) תוחלף בקשת (u, v) המסומנת ב-A כך ש-$u\in V_i$ ו-$v\in V_j$. כעת יהיו (u, v) ו-(w, y) שתי קשתות שכנות בסידרה שהתקבלה. בהכרח v ו-w נמצאים באותו רכיב קשירות (וודאו כי הנכם מבינים מדוע) ולכן קיים מסלול המחבר אותם שכולל רק קשתות המסומנות ב-B ו-C (ניתן למצוא אותו למשל ע״י BFS). ״נשתול״ את המסלול הזה בסידרת הקשתות בין (u, v) ו-(w, y). את המסלול המתקבל נחזיר כפלט.

<u>סיבוכיות:</u> המרכיב המשמעותי בכל אחד משלבי הרדוקציה (תרגום קלט לקלט, פתרון הבעיה החדשה ותרגום פלט לפלט) הוא מציאת מסלול קצר ביותר בגרף חסר משקלות ולכן עלותו ($O(|V|+|E|)$, ע״י BFS) היא גם סיבוכיות הפתרון כולו, כלומר, $O(|V|+|E|)$.

<u>נכונות:</u> נשאיר לכם להוכיח את הנכונות עבור המקרה בו יש מסלול בין s ל-t שלא משתמש כלל בקשתות המסומנות ב-A. במקרה השני: נראה התאמה חד-חד ערכית בין מסלולים חוקיים בגרף המקורי בעלי מספר מינימלי של קשתות המסומנות ב-A מ-s ל-t לבין מסלולים קצרים ביותר מ-s' ל-t' בגרף החדש, כך שמספר הקשתות המסומנות ב-A במסלול מהגרף המקורי שווה לאורך המסלול בגרף החדש פחות 2. בכיוון הראשון, יהי p' מסלול קצר ביותר מ-s' ל-t' בגרף החדש. המסלול הזה ניתן לתרגום למסלול בגרף המקורי, כפי שהראינו קודם, בעת תיאור הרדוקציה. וודאו כי הנכם יודעים להראות כי אכן התרגום נותן מסלול בגרף המקורי וכי מספר הקשתות שבו המסומנות ב-A הוא בדיוק כאורך המסלול p' פחות 2. בכיוון השני, יהי p מסלול חוקי בין s ל-t בגרף המקורי, שמספר הקשתות המסומנות בו ב-A הוא מינימלי. בבירור p מתחיל בקשת (s, v) המסומנת ב-B, ומסתיים בקשת (u, t) המסתיימת ב-C, וניתן לחלוקה לתתי-מסלולים (אולי חלקם ריקים) שקשתותיהם מסומנות כולן ב-B או ב-C, ומחברות בין התתי-מסלולים קשתות המסומנות ב-A. הנה מסלול p' בגרף החדש המתאים ל-p: כל תת-מסלול המורכב כולו מקשתות המסומנות ב-B או C (לא כולל הקשת הראשונה והקשת האחרונה) משוכן בשלמותו בתוך אחד מרכיבי הקשירות ולכן פשוט נחליפו בצומת המסמן את הרכיב. את הצומת s נחליף ב-s' ואת הצומת t נחליף ב-t'. וודאו כי הנכם יודעים להראות כי אכן התקבל מסלול בגרף החדש וכי אורכו פחות 2 שווה בדיוק למספר הקשתות המסומנות ב-A במסלול המקורי (היכן השתמשתם בעובדה שמספר הקשתות המסומנות ב-A ב-p הוא מינימלי?). כדי להשלים את ההוכחה, נשים לב שהמסלול שבנינו בכיוון הראשון הוא בהכרח בעל מספר מינימלי של קשתות המסומנות ב-A: אחרת, ניתן לקחת מסלול אחר בעל מספר מינימלי של קשתות המסומנות ב-A, לתרגם אותו כפי שנעשה בהוכחת הכיוון השני ולקבל מסלול קצר יותר מזה שהתחלנו עימו, בסתירה לכך שהתחלנו עם מסלול קצר ביותר מ-s' ל-t'. בדומה, המסלול שקיבלנו בתרגום שנעשה בכיוון השני הוא בהכרח קצר ביותר, כי אחרת ניקח מסלול קצר ביותר, נתרגם אותו כפי שנעשה בכיוון הראשון ונקבל מסלול מ-s ל-t עם פחות קשתות המסומנות ב-A מזה שהתחלנו עימו, בסתירה למינימליות שלו.

שאלה 5

כתבו אלגוריתם ליניארי אשר מקבל כקלט גרף מכוון $G = (V, E)$, שני צמתים $s, t \in V$ וקבוצת צמתים $U \subseteq V$, ועונה "כן" אם קיים מסלול קצר ביותר מ- s ל-t אשר מכיל את כל צמתי U, ו-"לא" אחרת.

נמקו את נכונות האלגוריתם.

תשובה

נציג שלושה פתרונות. הראשון יהיה פתרון שאינו רדוקטיבי במלואו. הפתרון השני והשלישי יהיו פתרונות רדוקטיביים לגמרי.

פתרון ראשון:

1. נמצא את מרחקם של כל צמתי הגרף מ-s.
2. כעת נבצע מיון על צמתי U לפי מרחקם מ-s (ערכי d) כפי שחושב בצעד 1. אם יש שניים שערכם זהה נחזיר תשובה שלילית.
3. אחרת, בלי הגבלת הכלליות תהי $d(u_1), d(u_2), \ldots, d(u_k)$ הסדרה הממויינת של ערכי d של איברי U.

   בשלב זה נבדוק אם קיים לכל $1 \le i < k$ מסלול קצר ביותר מ- u_i אל u_{i+1} שאורכו בדיוק $d(u_{i+1})-d(u_i)$, ואם קיים מסלול קצר ביותר מ-u_k אל t שאורכו בדיוק $d(t)-d(u_k)$.

   נבצע זאת ע"י ביצוע BFS, תוך הכנסת שינויים קלים באלגוריתם: נתחיל מ- s. הצומת הראשון מצמתי U שאליו נגיע הוא בהכרח u_1, הצומת הראשון במיון משלב 2. כשנשלוף את u_1 מהתור, במקום להכניס את כל שכניו שערך d שלהם עדיין אינסופי לאותו תור, נכניס אותם לתור חדש, ויחד איתם נכניס גם את שכניו של u_1 שכבר נמצאים בתור, וערך d שלהם גדול מאחד מזה של u_1. נמשיך את פעולת האלגוריתם על התור השני ולא על הראשון. בכך למעשה אנו מצמצמים את המשך פעולתו של האלגוריתם לעץ ששורשו u_1. אם וכאשר נשלוף את u_2 מהתור החדש, ברור שזיהינו מסלול קצר ביותר מ- u_1 אל u_2, ועלינו לבדוק שאורכו הוא האורך הדרוש, כלומר, עלינו לבדוק שערך d של u_2 כעת זהה לערכו בהרצה המקורית. אם לא, נחזיר תשובה שלילית. אחרת, נמשיך באותו אופן, ע"י הכנסת שכניו הרלבנטיים של u_2 לתור חדש, והמשך עבודה על התור החדש, כך עד שנוציא את u_k מהתור התורן, ואת כל שכניו שערך d שלהם אינסופי או גדול באחד מזה של u_k נכניס לתור חדש. אם בהמשך פעולת האלגוריתם נצליח לשלוף מהתור האחרון את t וערך d שלו יהיה שווה לערכו בהרצה המקורית, נחזיר תשובה חיובית. אחרת, נחזיר תשובה שלילית.

סיבוכיות: צעד 1 ניתן לביצוע ע"י BFS וסיבוכיות הרצת BFS היא כמובן ליניארית בגודל הגרף, כלומר $O(|V| + |E|)$. צעד 2 ניתן לביצוע ע"י מיון דלי שעלותו היא $O(|V|)$. השינויים בהרצה השנייה כוללים השוואת ערכי d, השוואה שמתבצעת $|V|$ פעמים, ולכן עלותה חסומה ע"י $O(|V|)$; הכנסה לתור חדש במקום למקורי, שאין לה השפעה על הסיבוכיות; ועבור חלק מהצמתים גם הכנסה לתור

חדש בנוסף לתור שבו הם כבר נמצאים, אבל גם זה כרוך בעלות קבועה עבור כל צומת כזה ולכן חסום ע״י $O(|V|)$. עדיין, כל צומת **יוצא** מתור רק פעם אחת לכל היותר (ייתכן שהוא נכנס פעמיים, אבל אם הוא נכנס פעמיים, ההכנסה הראשונה כבר לא תגרור אחריה הוצאה כי זו הכנסה לתור שהופסק השימוש בו). לכן, סיבוכיות הגירסה הזו של BFS היא $O(|V| + |E|)$.

שימו לב שסיבוכיות הזיכרון אף היא לא משתנה. אמנם יש הרבה תורים, אבל גודל כל התורים יחד אינו עולה על פעמיים גודל התור היחיד בהרצה הראשונה. בהרצה הרגילה כל צומת נכנס לתור פעם אחת בדיוק וכאן הוא נכנס לכל היותר פעמיים (אולי גם רק פעם אחת ואולי אף פעם).

<u>נימוק לנכונות האלגוריתם</u>: ראשית נסביר את התשובה השלילית בשלב 2: נניח שיש שני צמתים u_i ו-u_j שמרחקם מ-s זהה, והוא שווה ל-d. יהי p מסלול העובר גם דרך u_i וגם דרך u_j. נניח, בלי הגבלת הכלליות, כי הוא עובר קודם ב-u_i. קטע המסלול מ-s ל-u_j בוודאי ארוך מ-d: אם הוא קצר מ-d או שווה לו אז הרישא שלו מ-s ל-u_i בהכרח קצרה מ-d, בסתירה לכך ש-d הוא אורך המסלול הקצר ביותר מ-s ל-u_i. מאחר שקטע המסלול מ-s ל-u_j ארוך מ-d, וידוע שקיים מסלול מ-s ל-u_j שאורכו d, הרי ניתן להחליף ב-p את הקטע מ-s ל-u_j בקטע קצר יותר, ולכן בהכרח p אינו מסלול קצר ביותר.

אם לכל צמתי U ערכי d שונים וניתן למצוא קטעי מסלול כנאמר בתיאור לפני האלגוריתם ברור שניתן לחברם למסלול מ-u_1 ל-t שאורכו $d(t)$-$d(u_1)$. ברור שקיים מסלול מ-s ל-u_1 שאורכו $d(u_1)$ ולכן בסך הכל ניתן ליצור מסלול מ-s ל-t שעובר דרך כל צמתי U ואורכו $d(t)$, ולכן הוא מסלול קצר ביותר.

הפעלת ה-BFS בצעד 3 מאפשרת לבדוק אם קיימים קטעי מסלול כאלו, כי בכל פעם שמתחילים תור חדש מצטמצמים רק לצמתים הנגישים מ-u_i התורן. שימו לב, ההוספה לתור החדש של צמתים שכבר ניתן להם ערך d והוא גדול ב-1 מזה של ה-u_i התורן, נועדה להתמודד עם מצב שבו יש צומת x שהוא שכן גם של ה-u_i התורן וגם של y – צומת אחר בשכבה של ה-u_i התורן, שיצא מהתור לפניו. במקרה זה x צריך להיות בתת-עץ של ה-u_i התורן, למרות שהוא כבר הוכנס לתור, ולכן הוא מוכנס שוב לתור החדש (עם אותו ערך d). זה המקרה היחיד שבו יכול להיות שצומת שערך d שלו גדול מזה של ה-u_i התורן והוא נגיש מ-u_i התורן כבר נמצא בתור, וכבר קיבל ערך סופי, כי ההפרש בין ערכי d של צמתים בתור אינו עולה על 1.

הפיסקה האחרונה, המתייחסת להרצת ה- BFS בצעד 3 היא הסיבה שיש כאן נימוק נכונות ולא הוכחת נכונות (כפי שאכן נדרש בשאלה): מאחר שבהרצה זו שינינו מעט את אלגוריתם BFS המקורי, צריך למעשה להוכיח את נכונותו מחדש, ולא ניתן להסתמך עליו כעל קופסה שחורה.

<u>פתרון שני (הוצע ע״י המנחה שרי שינוולד):</u>

הפתרון דומה מאוד לפתרון הקודם, אלא שהוא משתמש באלגוריתם למציאת מסלולים קצרים בגרף לא ממושקל כקופסה שחורה, בלי לשנותו, ומשום כך קל יותר להוכיח את נכונותו.

1. נמצא את מרחקם של כל צמתי הגרף מ-s.
2. אם קיים צומת $u \in U$ כך ש-$d(u) \geq d(t)$ החזר ״לא״.

3. כעת נבצע מיון על צמתי U לפי מרחקם מ-s (ערכי d) כפי שחושב בצעד 1. אם יש שניים שערכם זהה נחזיר תשובה שלילית. מיון דומה נבצע על צמתי הגרף שאינם ב-U. עתה נעבור על שתי הרשימות הממוינות במקביל (כמו שנעשה באלגוריתם של מיזוג) ועבור כל צומת $v \in V-U$ עבורו מתקיים כי יש צומת $u \in U$ כך ש-$d(u)=d(v)$ נסיר את v מהגרף.
4. בגרף המתקבל מצא את מרחקו של t מ-s. אם הוא שווה למרחקו בגרף המקורי, החזר "כן". אחרת, החזר "לא".

<u>סיבוכיות</u>: צעד 1 ניתן לביצוע ע"י BFS וסיבוכיות הרצת BFS היא כמובן ליניארית בגודל הגרף, כלומר $O(|V|+|E|)$. עלות צעד 2 אף היא ליניארית. המיונים בצעד 3 ניתנים לביצוע ע"י מיון דלי שעלותו היא $O(|V|)$, הזיהוי אלה צמתים יש להסיר מהגרף נעשה אף הוא ע"י בזמן ליניארי, ע"י מעבר סדרתי על שתי הרשימות הממויינות במקביל. הסרת צומת ניתנת לביצוע בזמן קבוע, אם נבצע עיבוד מקדים על הגרף, שעלותו ליניארית, ובו נשמור לכל צומת פרט לרשימת הקשתות היוצאות גם את רשימת הקשתות הנכנסות, כרשימת הצבעות אל תוך רשימות השכנויות. כלומר, הקשת (u, v) תופיע כרגיל ברשימת השכנים של צומת u, ובנוסף, ברשימת הקשתות הנכנסות אל הצומת v תהיה הצבעה אל הקשת (u, v) ברשימה של u. צעד 4 ניתן אף הוא לביצוע ע"י BFS בעלות $O(|V|+|E|)$ וזוהי גם הסיבוכיות הכוללת של הפתרון כולו.

<u>הוכחת נכונות</u>:

ראשית נראה את נכונות התשובה המוחזרת בצעד 1: נניח שיש צומת $u \in U$ כך ש-$d(u) \geq d(t)=d$. נניח בשלילה שקיים מסלול p קצר ביותר מ-s ל-t העובר דרך u. מאחר שזהו מסלול קצר ביותר אל t אורכו d. תת-המסלול שלו מ-s אל u אף הוא מסלול קצר ביותר, אך מאחר שהוא תת-מסלול של p אורכו קטן ממש מאורכו של p, כלומר, קטן מ-d. קיבלנו אם כך שאורכו של מסלול קצר ביותר מ-s ל-u קטן ממש מ-d, בסתירה לכך ש-$d(u) \geq d$.

את נכונות התשובה השלילית המוחזרת בצעד 2 כבר הצדקנו בפתרון הקודם.

נראה כעת את נכונות התשובה המוחזרת בצעד 4. לשם כך נראה כי אם קיים מסלול כנדרש בגרף המקורי אז הוא קיים גם בגרף החדש, ולהיפך – אם קיים מסלול באורך $d(t)$ בגרף החדש מ-s ל-t אז קיים מסלול כנדרש בגרף המקורי.

בכיוון הראשון: אם קיים מסלול קצר ביותר מ-s ל-t בגרף המקורי שמכיל את כל צמתי U, אז בלי הגבלת הכלליות תהי $d(u_1), d(u_2), \ldots, d(u_k)$ הסדרה הממויינת של ערכי d של איברי U. הראינו כבר קודם שבהכרח כל ערכי d של איברי U שונים זה מזה וקטנים מ-$d(t)$. מאחר שתת-מסלול של מסלול קצר ביותר אף הוא מסלול קצר ביותר, הרי שאיברי U נמצאים על המסלול הזה במקומות $d(u_1), d(u_2), \ldots, d(u_k)$. בהכרח שאר הצמתים, שאינם איברי U, נמצאים במסלול במקומות אחרים, ושוב בגלל שתת-מסלול של מסלול קצר ביותר הוא מסלול קצר ביותר נובע שערכי d של כל הצמתים שעל המסלול שאינם איברי U שונים מ-$d(u_1), d(u_2), \ldots, d(u_k)$. לכן אף אחד מהצמתים שבמסלול לא הוסר

מהגרף ולכן המסלול הזה קיים גם בגרף החדש. ברור שלא קיים בגרף החדש מסלול קצר ממנו מ- s ל-t, ולכן מרחקו של t מ-s בגרף החדש שווה למרחקו של t מ-s בגרף המקורי והאלגוריתם יענה "כן".

בכיוון השני, נוכיח קודם כל טענת עזר כללית: כל מסלול מצומת s אל צומת x שמרחקו מ-s הוא d חייב לעבור עבור כל $i<d$ בצומת שמרחקו מ-s הוא i. נסתכל על המופע הראשון במסלול של צומת v שמרחקו מ-s גדול מ-i או שווה לו (ברור שיש לפחות צומת אחד כזה, כי מרחקו של x מ-s הוא $d>i$). אם מרחקו של v מ-s הוא i סיימנו. אחרת, מרחקו של v מ-s גדול ממש מ-i. ע"פ בחירת v, מרחקו מ-s של הצומת w שקודם ל-v במסלול הוא d' כך ש-$d' \leq i-1$. אבל מכאן אנו מקבלים שיש מסלול מ-s ל-w שאורכו d' ואם נוסיף לו את הקשת (w, v) נקבל מסלול מ-s ל-v שאורכו $d'+1 \leq (i-1)+1$, כלומר שאורכו $d'+1 \leq i$ בסתירה לכך שמרחקו של v מ-s גדול ממש מ-i.

ובכן, נניח שהאלגוריתם עונה "כן" בצעד 4, ולכן מרחקו של t מ-s בגרף החדש שווה למרחקו של t מ-s בגרף המקורי. יהיו $d(u_1), d(u_2), \ldots, d(u_k)$ הסדרה הממויינת של ערכי d של איברי U כפי שחושבו בצעד 1. יהי p המסלול הקצר ביותר שנמצא בגרף החדש מ-s ל-t. מסלול זה קיים גם בגרף המקורי, ולכן, ע"פ טענת העזר, המסלול הזה בהכרח עובר עבור כל $i<d(t)$ בצומת שמרחקו מ-s הוא i. בפרט, מאחר ש-$d(t)>d(u_k)$, עבור כל $1 \leq i \leq k$ המסלול עובר בצומת שמרחקו מ-s הוא $d(u_i)$. אבל מאחר שלכל $1 \leq i \leq k$ הצומת היחיד בגרף החדש שמרחקו מ-s הוא $d(u_i)$ הוא u_i עצמו, נקבל ש-p הוא מסלול מ-s ל-t שעובר דרך כל צמתי U. בנוסף, מאחר שמרחקו של t מ-s בגרף החדש שווה למרחקו מ-s בגרף המקורי, הרי ש-p הוא גם מסלול קצר ביותר בגרף המקורי. לכן, אם האלגוריתם עונה "כן" בצעד 4 אכן קיים בגרף המקורי מסלול כנדרש.

<u>פתרון שלישי (הוצע ע"י המנחה סיימון קורמן):</u>

יתרונו של הפתרון הזה הוא שהוא משתמש ברדוקציה לבעיית מציאת אורך מסלולים קצרים ביותר ממקור יחיד בגרף לא ממושקל, ולכן מאפשר להשתמש ב- BFS בקופסה שחורה, בלי לשנות בו דבר. משום כך, הוכחת הנכונות שלו קלה יותר מזו של הפתרון הראשון.

אינטואיטיבית, נבנה מתוך הגרף הנתון G גרף חדש G', שבו מסלולים העוברים דרך צמתים מ- U יועדפו על פני מסלולים באותו אורך המכילים פחות צמתים מ- U. נעשה זאת ע"י כך שבכל מסלול קצר ביותר העלות בגין קשתות שלא נכנסות לצמתים מ-U תהיה כפולה.

הבנייה תתבצע באופן הבא: כל צומת v שאינו ב-U יוחלף בשני צמתים חדשים v_{in} ו-v_{out}, שביניהם תחבר קשת מ-v_{in} אל v_{out}. כל קשת שנכנסה בגרף המקורי ל-v תיכנס בגרף החדש ל-v_{in}, וכל קשת שיצאה בגרף המקורי מ-v תצא בגרף החדש מ-v_{out}.

באופן פורמלי: $G'=(V', E')$ כאשר

$$V'=U \cup \{v_{in} \mid v \in V-U\} \cup \{v_{out} \mid v \in V-U\}$$

$$E'=\{(u, v) \mid u, v \in U\} \cup \{(v_{in}, v_{out}) \mid v \in V-U\} \cup \{(u, v_{in}) \mid u \in U, v \in V-U, (u, v) \in E\} \cup$$

$$\{(u_{out}, v_{in}) \mid u, v \in V-U, (u, v) \in E\} \cup \{(u_{out}, v) \mid u \in V-U, v \in U, (u, v) \in E\}$$

טענת עזר: יהי p מסלול כלשהו ב-G ויהי p' המסלול המתאים לו ב-G', המתקבל ע"י החלפת הצמתים שאינם ב-U והקשתות הנוגעות בהם כפי שתואר בבנייה לעיל. יהי x מספר הצמתים ב-p שאינם מ-U. אז $|p'| = |p| + x$.

הוכחת הטענה מיידית מכך שכל צומת שאינו ב-U הוחלף בשני צמתים ונוספה קשת המחברת ביניהם.

האלגוריתם:

1. חשב את מרחקו של t מ-s ב-G.
2. יהי $\delta(s, t)$ הערך $d[t]$ כפי שחושב בשורה 1.
3. בנה את הגרף G' כפי שתואר לעיל.
4. חשב את מרחקו של t_{out} מ-s ב-G'.
5. אם מתקיים $d[t_{out}] \le 2 * \delta(s,t) - |U| + 1$ החזר "כן", אחרת, החזר "לא".

<u>סיבוכיות:</u> מספר הצמתים בגרף החדש חסום ע"י פעמיים מספר הצמתים בגרף המקורי, ובדומה, מספר הקשתות בגרף החדש חסום ע"י פעמיים מספר הקשתות בגרף המקורי. לכן בסה"כ $|V'| = O(|V|)$ ו- $|E'| = O(|E|)$. אם כך, הן הבנייה והן צעדים 1 ו-4 ניתנים לביצוע בזמן ליניארי בגודל הקלט לבעיה, כלומר העלות היא $O(|V| + |E|)$.

<u>נכונות:</u> יש להראות את תקפות התרגום של הרדוקציה ולכן ההוכחה היא דו-כיוונית:

טענה 1: אם קיים מסלול כנדרש בשאלה אז בהכרח $d[t_{out}] \le 2 * \delta(s,t) - |U| + 1$.

הוכחת טענה 1: יהי p מסלול קצר ביותר מ-s ל-t העובר בכל צמתי U. ברור שהוא פשוט (אחרת לא היה קצר ביותר). מנכונות האלגוריתם שבוצע בשלב 1 (BFS) מתקיים $\delta(s,t) = d[t] = |p|$. מאחר ש-p עובר בכל צמתי U, ואורכו של מסלול פשוט תמיד גדול ב-1 ממספר צמתיו אז מספר הצמתים ב-p שאינם ב-U הוא $|p| - |U| + 1 = \delta(s,t) - |U| + 1$. יהי p' המסלול המתאים ל-p ב-G'. מטענת העזר נובע כי $|p'| = |p| + \delta(s,t) - |U| + 1 = 2 * \delta(s,t) - |U| + 1$. מאחר שקיים ב-$G'$ מסלול מ-s ל-t שאורכו $2 * \delta(s,t) - |U| + 1$ ברור שזהו חסם לאורכו של מסלול קצר ביותר מ-s ל-t ב-G' ולכן בהכרח $d[t_{out}] \le 2 * \delta(s,t) - |U| + 1$.

טענה 2: אם $d[t_{out}] \le 2 * \delta(s,t) - |U| + 1$ אז קיים מסלול כנדרש בשאלה.

הוכחת טענה 2: יהי p' מסלול קצר ביותר מ-s ל-t ב-G'. מנכונות האלגוריתם שבוצע בצעד 4 (BFS), ומכך ש-$d[t_{out}] \le 2 * \delta(s,t) - |U| + 1$ נובע כי אורכו של מסלול קצר ביותר מ-s ל-t ב-G' אינו עולה על $2 * \delta(s,t) - |U| + 1$, ובפרט $|p'| \le 2 * \delta(s,t) - |U| + 1$. נסמן את אי-השוויון האחרון ב-(*). נביט במסלול p המתאים ל-p' ב-G. נניח בשלילה ש-p אינו עובר בכל צמתי U. לכן, מספר צמתי p שהם מתוך U קטן ממש מ-$|U|$, ומספר צמתי p שאינם מ-U גדול ממש מ-$|p| - |U| + 1$. מאחר ש-p הוא

מסלול מ-s ל-t ב-G ברור שאורכו אינו קטן מאורך המסלול הקצר ביותר מ-s ל-t ב-G, כלומר $|p| \geq \delta(s,t)$. לכן נקבל כי מספר צמתי p שאינם מ-U גדול ממש מ-$\delta(s,t)-|U|+1$. מכך ומטענת העזר נובע כי $|p'| > |p| + \delta(s,t) - |U| + 1 \geq 2*\delta(s,t) - |U| + 1$, בסתירה ל-(*). אם כך, הראינו עד כה ש-p עובר בכל צמתי U. עדיין יש להראות ש-p מסלול קצר ביותר, כלומר $|p| = \delta(s,t)$. נניח בשלילה כי $|p| > \delta(s,t)$. בדומה לטיעון בהוכחת טענה 1, מאחר ש-p עובר בכל צמתי U, אז מספר הצמתים ב-p שאינם ב-U הוא $|p|-|U|+1 = \delta(s,t)-|U|+1$. מכאן, מטענת העזר ומהנחת השלילה האחרונה נקבל כי $|p'| = |p| + \delta(s,t) - |U| + 1 > 2*\delta(s,t) - |U| + 1$, שוב בסתירה ל-(*). לכן p הוא המסלול הנדרש בשאלה.

פרק ג', פרק 17 וממ"ן 12

שאלה 1

כתבו אלגוריתם יעיל ככל שתוכלו המקבל כקלט גרף לא מכוון $G=(V,E)$ עם פונקציית משקל $w:E\rightarrow R^{+}$ ועץ פורש מינימלי $T=(V,E')$ של G ומוצא עץ פורש מינימלי $T_1=(V,E'')$ של G כך שמספר הקשתות הנמצאות גם ב-T וגם ב-T_1 הוא מינימלי.

הוכיחו את נכונות האלגוריתם ונתחו את סיבוכיותו.

תשובה

נפתור את הבעיה על ידי רדוקציה לבעיית מציאת עץ פורש מינימלי.

נגדיר פונקציית משקל חדשה על קשתות הגרף: $w'(e)=\begin{cases}(w(e),0) & e\notin T\\ (w(e),1) & e\in T\end{cases}$.

נגדיר $(w(e_1),\ b_1)=w'(e_1)<w'(e_2)=(w(e_1),\ b_1)$ אם ורק אם $w(e_1)<w(e_2)$ או $w(e_1)=w(e_2)$ ו-$b_1<b_2$. אינטואיטיבית, המשמעות היא שבמיון לפי קשתות, קשתות ששייכות לעץ T נמצאות אחרי קשתות בעל אותו משקל שאינן שייכות ל-T.

תחת פונקציית המשקל החדשה משקלו של עץ פורש T_1 הוא $(\ |T\cap T_1|$ $w'(T_1)=(w(T_1),$ ועץ T_1 הוא קל מעץ T_2 אם ורק אם $w\ (T_1)<w\ (T_2)$ או $w\ (T_1)=w\ (T_2)$ ו- $|T\cap T_2|<|T\cap T_1|$.

נמצא עץ פורש מינימלי T_1, ע"פ פונקציית המשקל החדשה וזה יהיה התשובה של האלגוריתם כולו.

<u>סיבוכיות:</u> לא שינינו את גודל הגרף ולכן העלות היא בדיוק כעלות מציאת עץ פורש מינימלי (בעזרת פרים או קרוסקל, ע"פ בחירתנו).

<u>נכונות:</u> ברור ש-T_1 הוא עץ פורש של הגרף. נראה שהוא עץ פורש מינימלי ע"פ פונקציית המשקל המקורית. נניח בשלילה שקיים עץ אחר T_2 כך ש-$w(T_2)<w(T_1)$. מכאן נובע גם $w'(T_2)<w'(T_1)$ בסתירה למינימליות T_1 ע"פ w'. עתה נראה שמספר הקשתות הנמצאות גם ב-T וגם ב-T_1 הוא מינימלי ביחס לכל העצים הפורשים המינימליים של הגרף. נניח בשלילה שקיים עץ פורש מינימלי אחר T_2 כך ש-$|T\cap T_2|<|T\cap T_1|$. מאחר שגם T_1 וגם T_2 הם עצים פורשים מינימליים של הגרף אז $w(T_2)=w(T_1)$. לכן, נקבל בהכרח $w'(T_2)<w'(T_1)$, שוב בסתירה למינימליות T_1 ע"פ w'.

שאלה 2

נתון גרף לא מכוון $G=(V,E)$, עם פונקציית משקל $w:E\rightarrow R^{+}$. יש למצוא תת-קבוצה $E'\subseteq E$ של קשתות, כך שהקבוצה E' תכיל לפחות קשת אחת מכל מעגל בגרף והמשקל הכולל של הקשתות ב-E' יהיה מינימלי.

תארו אלגוריתם יעיל ככל שתוכלו לפתרון הבעיה.

הוכיחו את נכונות האלגוריתם ונתחו את סיבוכיותו.

<u>תשובה</u>

נפתור את הבעיה על ידי רדוקציה לבעיית מציאת עץ פורש מקסימלי.

<u>האלגוריתם:</u> נמצא בגרף עץ פורש מקסימלי. פלט האלגוריתם יהיה קבוצת הקשתות E' ש**אינה** שייכת לעץ שמצאנו.

<u>סיבוכיות:</u> לשם מציאת עץ פורש מקסימלי ניתן להשתמש באלגוריתם של קרוסקל כאשר מיון הקשתות הוא בסדר יורד במקום בסדר עולה (גם את האלגוריתם של פרים ניתן להתאים בקלות כך שימצא עץ פורש מקסימלי במקום מינימלי – חשבו כיצד). לכן הסיבוכיות הכוללת היא כסיבוכיות מציאת עץ פורש מינימלי.

<u>נכונות:</u> בבירור הקבוצה E' שמצאנו מכילה לפחות קשת אחת מכל מעגל בגרף: אחרת קיים מעגל בגרף שאין ממנו אף קשת ב-E', כלומר – כל קשתותיו הן ב-E-E'. אבל ע"פ האלגוריתם E-E' הוא עץ ולכן לא יתכן שהוא מכיל מעגל.

נראה כעת שמשקלה של E' מינימלי. נניח בשלילה שקיימת קבוצה אחרת E'' המכילה לפחות קשת אחת מכל מעגל ומשקלה קטן משל E'. מאחר ש-E'' מכילה לפחות קשת אחת מכל מעגל הרי ש-E-E'' אינה מכילה מעגלים ולכן ניתן להרחיבה לעץ פורש T של הגרף ובבירור מתקיים $w(T)\geq w(E\text{-}E'')$ (כי T מכיל את E-E''). מצד שני, ע"פ ההנחה $w(E'')<w(E')$ ולכן $w(E\text{-}E'')>w(E\text{-}E')$. בסה"כ מתקבל כי $w(T)>w(E\text{-}E')$ בסתירה להיות E-E' עץ פורש מקסימלי.

<u>שאלה 3</u>

נניח שנתונות לנו k רשימות של מספרים – $L_1,\ldots,L_k$. כל אחת מהרשימות ממויינות ומטרתנו למזג את כל k הרשימות לרשימה ממויינת אחת.

לצורך כך נתון לנו אלגוריתם A הממזג שתי רשימות ממוינות נתונות לרשימה ממוינת אחת ואנו יכולים להשתמש בו כקופסה שחורה. עלות הפעלת האלגוריתם על שתי רשימות היא בדיוק סכום אורכי הרשימות.

כתבו אלגוריתם יעיל ככל שתוכלו המקבל כקלט את k הרשימות ומחליט על אילו קלטים להפעיל את האלגוריתם A בכל שלב, כך שעלות המיזוג הכוללת תהיה מינימלית.

למשל – עבור שלוש רשימות L_1,L_2,L_3 ניתן למזג קודם את L_1 ו- L_2 ואחר-כך למזג את הרשימה הממוזגת עם L_3. במקרה זה העלות הכוללת של המיזוג היא $2|L_1|+2|L_2|+|L_3|$ אבל ניתן גם למזג קודם את L_1 ו- L_3 ואחר-כך למזג את הרשימה הממוזגת עם L_2 ובמקרה זה העלות הכוללת של המיזוג היא $2|L_1|+2|L_3|+|L_2|$. האלגוריתם צריך לבחור בדרך שעבורה הערך המתקבל הוא מינימלי.

נתחו את סיבוכיות האלגוריתם והוכיחו את נכונותו.

<u>תשובה</u>

נבצע רדוקציה לבעיית קוד הופמן: כל רשימה תהיה אות בא"ב. השכיחות של כל אות תהיה אורך הרשימה המתאימה לה. מיזוג של שתי רשימות נותן רשימה שאורכה הוא סכום אורכי הרשימות, ולכן, כמו בבעיית קוד הופמן, השכיחות של הרשימה הממוזגת הוא סכום שכיחויות הרשימות. אחרי תרגום הקלט לקלט לבעיית קוד הופמן נפעיל את האלגוריתם של הופמן ונקבל עץ עם עלות מינימלית, כאשר עלות העץ סוכמת עבור כל אותיות הא"ב את מכפלת שכיחות האות בעומק האות בעץ. לכן, עלות העץ למעשה סוכמת עבור כל הרשימות את מכפלת אורכן במספר המיזוגים שהן עוברות, כלומר, את עלות המיזוג. לכן, מנכונות האלגוריתם של הופמן, התקבל עץ המייצג מיזוגים שעלותם מינימלית.

<u>הסיבוכיות</u> היא כמובן כמו סיבוכיות האלגוריתם של הופמן, כלומר: $O(k\log k)$. נשים לב, שאם לא נתון אורכה של כל רשימה, יש קודם לחשב זאת, אך זה ניתן לביצוע בזמן ליניארי באורך הקלט, כי הקלט כולל את הרשימות עצמן.

<u>פרקים ד׳+ה׳ וממ״ן 13</u>

<u>שאלה 1</u>

גרף מכוון נקרא **חצי-קשיר היטב** אם לכל $u,v \in V$ יש בגרף מסלול מכוון מ-u ל-v, או מ-v ל-u, או שניהם. הציעו אלגוריתם, יעיל ככל שתוכלו, המקבל גרף מכוון, ובודק אם הוא חצי קשיר היטב.

<u>תשובה</u>

נפתור ע״י רדוקציה לבעיית מיון טופולוגי בגרף מכוון חסר מעגלים.

<u>האלגוריתם</u>: חשב את גרף רכיבי קשירות היטב $G^{SCC}=(V^{SCC},E^{SCC})$. חשב מיון טופולוגי של צמתי G^{SCC}. בדוק שיש קשת ב-G^{SCC} מכל צומת אל הצומת העוקב לו ע״פ המיון. אם כן, הגרף חצי קשיר, אחרת הוא לא חצי קשיר.

<u>סיבוכיות</u> האלגוריתם היא $O(|V|+|E|)$ כי זוהי סיבוכיות מציאת G^{SCC} וגם סיבוכיות ביצוע המיון הטופולוגי על G^{SCC} (גודלו של G^{SCC} בודאי חסום ע״י גודלו של G). את התנאי ניתן לבדוק גם כן בזמן $O(|V|+|E|)$.

<u>נכונות</u>: כפי שהוכח בספר, גרף ה-G^{SCC} הינו גמ״ל, ולכן ניתן להריץ עליו מיון טופולוגי.

עבור צומת $x \in V$ נסמן ב-$x' \in V^{SCC}$ את הרכיב הקשיר היטב של x.

נראה תחילה שאם האלגוריתם מודיע ״כן״ (כלומר, יש קשת מכל צומת אל העוקב לו במיון), אז הגרף חצי קשיר. יהיו $u, v \in V$. אם $u'=v'$ אז הם באותו רכיב קשירות היטב ולכן יש גם מסלול מ-u ל-v וגם מסלול מ-v ל-u. אחרת $u' \neq v'$. נניח בלי הגבלת הכלליות ש-u' מופיע לפני v' במיון הטופולוגי. לכן המשמעות היא שיש מסלול ב-G^{SCC} מ-u' ל-v', כי בין כל שני צמתים עוקבים יש קשת, וקל לראות שניתן לתרגם מסלול זה למסלול מ-u ל-v.

בכיוון ההפוך, נניח ש $u', v' \in V^{SCC}$ שני צמתים עוקבים במיון הטופולוגי שאין ביניהם קשת (בלי הגבלת הכלליות v' מופיע אחרי u'). כל מסלול ב-G^{SCC} שמתחיל ב-v' אינו יכול להגיע ל-u' כיוון שאז היה מכיל קשת אחורה. גם כל מסלול ב-G^{SCC} שמתחיל ב-u', יתחיל בקשת לצומת שנמצאת אחרי u' במיון, וצומת זו היא גם אחרי v' (כי הרי אין קשת מ-u' ל-v') וכיוון שהמסלול אינו יכול ״לחזור אחורה״ במיון הרי הוא לא יוכל להגיע ל-v'. לכן ב-G^{SCC} אין מסלול מ-u' ל-v' וגם ההפך. מזה נובע שאין מסלול ב-G בין צומת ב-u' לבין צומת ב-v' וההפך (כי אם היה מסלול כזה, הוא היה ״משרה״ מסלול בין רכיבי הקשירות ב-G^{SCC}). לכן הגרף איננו חצי-קשיר.

פרק ו' וממ"ן 14

שאלה 1

א. הראו כיצד ניתן להשתמש ברשת זרימה כדי לפתור את הבעיה הבאה:

נתונים גרף לא מכוון $G=(V,E)$ ושני צמתים $u,v \in V$. האם u ו-v נמצאים על מעגל פשוט?

הוכיחו את נכונות האלגוריתם ונתחו את סיבוכיותו.

ב. כתבו אלגוריתם המקבל כקלט גרף לא מכוון $G=(V,E)$ ושתי קשתות $e_1,e_2 \in E$ וקובע האם יש ב-G מעגל פשוט המכיל את e_1 ו-e_2.

הוכיחו את נכונות האלגוריתם ונתחו את סיבוכיותו.

תשובה

א. נפתור את הבעיה על ידי רדוקציה לבעיית מציאת זרימה מקסימלית.

u ו-v נמצאים על מעגל פשוט, אם קיימים ביניהם שני מסלולים זרים בצמתים.

נהפוך את הגרף לגרף מכוון (ע"י החלפת כל קשת לא מכוונת בשתי קשתות אנטי-מקבילות). נוסיף מקור s וקשת (s, u) בקיבול 2. נוסיף בור t וקשת (v, t) בקיבול 2. כדי להשלים את הגדרת הרשת עלינו להגדיר עוד את הקיבולים בגרף המקור. במקרה זה נרצה לאפשר קיבול לצמתים ולא לקשתות, כלומר, היינו רוצים להגביל את כמות הזרימה שיכולה לעבור דרך כל צומת מצמתי הגרף המקורי, ולא להגביל את כמות הזרימה שיכול לעבור דרך קשת מקשתות הגרף המקורי. נחלק אם כך את הבעיה לשני חלקים. בשלב ראשון נניח שאנו יודעים למצוא זרימה מקסימלית ברשת שבה יש קיבולים גם לצמתים ולא רק לקשתות, ונשתמש באלגוריתם זה כקופסה שחורה. בשלב שני נראה כיצד פותרים את הבעיה של מציאת זרימה מקסימלית ברשת שבה יש קיבולים גם לצמתים וגם לקשתות. למעשה, את הבעיה המקורית אנו פותרים על ידי רדוקציה לבעיית מציאת זרימה מקסימלית ברשת שבה יש קיבולים לצמתים ולקשתות, ואת זו אנו נפתור בתורה גם כן על ידי רדוקציה, הפעם לבעיית מציאת זרימה מקסימלית ברשת זרימה סטנדרטית.

ובכן, ניתן לכל אחת מקשתות הגרף המקורי קיבול אינסופי, ולכל אחד מצמתי הגרף המקורי קיבול 1. לצמתים החדשים, s ו-t, ניתן גם כן קיבול אינסופי. עכשיו נפעיל את האלגוריתם שאת קיומו הנחנו, למציאת זרימה מקסימלית ברשת מ-s ל-t. אם הצלחנו למצוא זרימה שערכה 2 נחזיר תשובה חיובית. אחרת, נחזיר תשובה שלילית.

<u>נכונות:</u> ברור שכל זרימה העוברת ברשת מ-s ל-t עוברת קודם בקשת (s, u) ומסיימת בקשת (v, t).

ברור שפרט לשתי קשתות אלו הזרימה הזאת לא עוברת דרך s או t, כי אין ברשת הזאת קשתות שנכנסות ל-s או קשתות שיוצאות מ-t. לכן, אם הצלחנו להזרים ברשת זרימה בגודל 2, אז ניתן להזרים ברשת מ-u ל-v זרימה בגודל 2, שאינה עוברת דרך s ו-t. לכן, הזרימה הזו עוברת רק בצמתים שקיבולם 1. מכאן נובע שבהכרח קיימים ברשת שני מסלולים בין u ל-v, שעוברים רק

דרך צמתי הגרף המקורי, והם זרים בצמתים. מכאן נובע שבגרף המקורי (ללא כיוונים לקשתות), ניתן למצוא בין u ל-v שני מסלולים זרים בצמתים ולכן u ו-v על מעגל פשוט. בכיוון השני: אם u ו-v על מעגל פשוט וקיימים ביניהם שני מסלולים זרים בצמתים אז ניתן להזרים 1 בכל מסלול כזה ובסה"כ ניתן להזרים 2 מ-u ל-v ולכן גם מ-s ל-t.

עלינו עוד להראות כיצד מוצאים זרימה מקסימלית ברשת שבה יש קיבולים גם לצמתים וגם לקשתות, ומה סיבוכיות הפתרון. ובכן, בהינתן רשת זרימה עם מקור s ובור t, שבה יש קיבולים גם לצמתים וגם לקשתות, נבצע רדוקציה לבעיית מציאת זרימה מקסימלית ברשת שבה יש קיבולים רק לקשתות, באופן הבא: נפצל כל צומת w לשני צמתים w_{in} ו-w_{out}, ולכל צומת w נוסיף את הקשת (w_{in}, w_{out}), שקיבולה יהיה כקיבול הצומת w. כל קשת (x, y) נחליף בקשת (x_{out}, y_{in}) שקיבולה יהיה כקיבול הקשת (x, y). נמצא בגרף זרימה מקסימלית מהצומת s_{in} אל הצומת t_{out}.

<u>נכונות:</u> כל זרימה חוקית ברשת החדשה מתארת זרימה חוקית ברשת המקורית, כאשר הזרימה בקשת מהצורה (x_{out}, y_{in}) מתאימה לזרימה בקשת (x, y), והזרימה בקשת מהצורה (w_{in}, w_{out}) מבטיחה שהזרימה שעוברת דרך הצומת לא תעלה על קיבולו. מצד שני, כל זרימה חוקית ברשת המקורית מתארת זרימה חוקית ברשת החדשה, כאשר זרימה דרך קשת (x, y) מתאימה לזרימה בקשת מהצורה (x_{out}, y_{in}), וזרימה דרך צומת w מתאימה לזרימה דרך הקשת (w_{in}, w_{out}). לכן ברור שערך זרימה מקסימלית בשתי הרשתות הוא זהה.

<u>סיבוכיות</u> האלגוריתם היא בדיוק כמו סיבוכיות מציאת זרימה מקסימלית על הרשת המקורית, כי גודלה של הרשת החדשה הוא באותו סדר גודל כמו הרשת המקורית: מספר הצמתים ברשת הוא $2\cdot|V|$, ומספר הקשתות הוא $|E|+|V|$.

כעת ניתן לנתח את <u>סיבוכיות</u> הפתרון הכולל: מספר הצמתים ברשת שבנינו (עם קיבולים לצמתים) הוא $|V|+2$, ומספר הקשתות הוא $|E|+2$. לכן, ברשת שנבנתה אחרי הרדוקציה מספר הצמתים הוא $O(|V|)$ ומספר הקשתות הוא $O(|E|+|V|)$. מאחר שהקיבולים הם שלמים (ונשארים כך גם אחרי הפעלת הרדוקציה לרשת סטנדרטית), והזרימה המקסימלית בבירור חסומה ע"י 2 (כי זה קיבולו של החתך שמצידן האחד s ומצידו השני שאר צמתי הגרף), אז ניתן למצוא זרימה מקסימלית (ע"י האלגוריתם של פורד ופלקרסון) בסיבוכיות $O(|V|+|E|)$.

ב. הפתרון יהיה דומה לזה של סעיף א', שוב על ידי רדוקציה לבעיית מציאת זרימה מקסימלית ברשת עם קיבולים לצמתים ולקשתות.

תהי $e_1=(u, v)$ ותהי $e_2=(x, y)$. אם יש בגרף מעגל פשוט המכיל את $e_1=(u, v)$ ואת $e_2=(x, y)$, אז יש מסלולים זרים בצמתים בין u ל-x ובין v ל-y או שיש מסלולים זרים בצמתים בין u ל-y ובין v ל-x. בדומה לסעיף א', נוסיף לגרף מקור s ובור t, נוסיף קשתות (s, u), (s, v), (x, t), (y, t) שקיבולן 1. לקשתות הגרף המקורי ניתן קיבול אינסופי. לצמתי הגרף המקורי ניתן קיבול 1, ולצמתים החדשים, s ו-t ניתן קיבול אינסופי. בדומה לסעיף א', נמצא זרימה מקסימלית בין s ל-t (כבר ידוע

לנו כי ניתן למצוא זרימה מקסימלית בגרף שבו יש קיבולים גם לצמתים וגם לקשתות). אם הזרימה המקסימלית שווה ל-2, נחזיר תשובה חיובית. אחרת, נחזיר תשובה שלילית.

<u>סיבוכיות:</u> מספר הצמתים בגרף החדש הוא $|V|+2$ ומספר הקשתות הוא $|E|+4$. לכן, גם במקרה זה, ברשת שנבנתה אחרי הרדוקציה מספר הצמתים הוא $O(|V|)$ ומספר הקשתות הוא $O(|E|+|V|)$. גם הפעם הקיבולים שלמים, והזרימה המקסימלית חסומה ע"י 2 (שוב, זהו קיבולו של החתך שמצידו האחד s ומצידו השני שאר צמתי הגרף). לכן גם הפעם ניתן לחסום את סיבוכיות מציאת זרימה מקסימלית ע"י $O(|V|+|E|)$.

<u>נכונות:</u> אם ניתן להזרים זרימה בערך 2 מ-s ל-t, אז הקשתות (s, u), (s, v), (x, t) ו-(y, t) רוויות. לכן, ניתן להזרים (בלי לעבור שוב דרך s ו-t) יחידת זרימה מ-u ויחידת זרימה מ-v ולהזרים יחידת זרימה אל x ויחידת זרימה אל y. מאחר שקיבולי הצמתים פרט ל-s ו-t שווים ל-1 נקבל מכך כי בהכרח קיימים בגרף המקורי מסלולים זרים בצמתים מ-u אל x ומ-v אל y או מ-u אל y ומ-v אל x. בכל מקרה נובע מכך ש-e_1 ו-e_2 על מעגל פשוט. בכיוון השני, אם e_1 ו-e_2 על מעגל פשוט, כלומר יש בגרף המקורי מסלולים זרים בצמתים בין u ל-x ובין v ל-y או שיש מסלולים זרים בצמתים בין u ל-y ובין v ל-x, אז ניתן להזרים ברשת יחידת זרימה מ-u אל x ויחידת זרימה מ-v אל y או להזרים יחידת זרימה מ-u אל y ויחידת זרימה מ-v אל x. בכל מקרה, ניתן להזרים ערך 2 מ-s ל-t.

<u>שאלה 2</u>

יהי $G=(V,E)$ גרף מכוון עם קיבולים אי-שליליים על הקשתות המייצג את מערכת הצינורות של המוביל הארצי. כמו כן, נתונים מיקומים ברשת של k תחנות שאיבה, כל אחת עם יכולת השאיבה המירבית שלה, ומיקומים ברשת של r ישובים, כל אחד עם דרישת המים שלו.

כתבו אלגוריתם יעיל ככל שתוכלו המחליט אם ניתן לספק את הדרישה של כל הישובים, ואם כן, כיצד.

הוכיחו את נכונות האלגוריתם ונתחו את סיבוכיותו.

<u>תשובה</u>

נפתור את הבעיה על ידי רדוקציה לבעיית מציאת זרימה מקסימלית. נוסיף לגרף מקור s ונחברו בקשת לכל אחת מתחנות השאיבה, כאשר קיבול הקשת הוא יכולת השאיבה המירבית של התחנה אליה מובילה הקשת. בדומה, נוסיף לגרף בור t ונחבר אליו בקשת כל ישוב, כאשר קיבול הקשת הוא דרישת המים של הישוב ממנו מגיעה הקשת. בידינו כעת רשת זרימה עם מקור, בור ופונקציית קיבול. נמצא זרימה מקסימלית ברשת. אם ערכה שווה לסכום דרישות הישובים נחזיר תשובה חיובית. אחרת, נחזיר תשובה שלילית.

<u>סיבוכיות</u>: מספר הצמתים ברשת הוא $|V|+2$ ומספר הקשתות חסום ע"י $|E|+k+r \le |E|+2\cdot|V|$.
ננתח את סיבוכיות מציאת זרימה מקסימלית לפי אדמונדס-קארפ ונקבל כי סיבוכיות האלגוריתם כולו היא $O(|V|\cdot|E|^2)$ (לא ידוע חסם על הזרימה המקסימלית, ובל מקרה הזרימה אינה בהכרח בשלמים ולכן ניתחנו לפי אדמונדס-קארפ.

<u>נכונות</u>: מאחר שהזרימה שמצאנו מספקת את אילוצי הקיבול ומחוק שימור הזרימה ברור כי כל תחנה מזרימה לכל היותר את מה שיכול להיכנס אליה, כלומר, לא יותר מיכולת השאיבה שלה. אם ערך הזרימה המקסימלית שמצאנו שווה לסכום דרישות הישובים הרי שכל הקשתות הנכנסות ל-t רוויות, ולכן, מחוק שימור הזרימה, הזרימה שנכנסת לכל ישוב שווה לדרישת הישוב. כלומר, אכן סופקו דרישות כל הישובים. אם ערך הזרימה המקסימלית קטן מזה, הרי ממקסימליות הזרימה נובע שאין זרימה שערכה גדול יותר, ולכן לא ניתן לספק את דרישות כל הישובים.

<u>שאלה 3</u>

תהי $U=\{1, 2, 3, ..., n\}$. נתונות קבוצות $S_1,S_2,...,S_k$ שכל אחת מהן מוכלת ב-U. תנו אלגוריתם המוצא קבוצת נציגים $r_1,r_2,...,r_k$ כך שכל הנציגים שונים זה מזה ולכל i $r_i \in S_i$ (או מודיע שאין כזו).
נתחו את סיבוכיות האלגוריתם והוכיחו את נכונותו.

<u>תשובה</u>

נפתור ע"י רדוקציה לבעיית מציאת זיווג מקסימלי בגרף דו-צדדי.
נבנה גרף ובו n קודקודים $1, 2, ..., n$ המייצגים את איברי U, ו-k קודקודים $s_1,\ldots s_k$ המייצגים את הקבוצות. הקשת (i, s_j) קיימת בגרף אם ורק אם $i\in s_j$. נמצא בגרף הזה זיווג מקסימלי ונסמן את גודלו ב-m. אם $m<k$ אז אין נציגים $r_1,\ldots r_k$ כנדרש. אחרת, אם $m=k$ אז יהיו $r_1,\ldots r_k$ הקודקודים המשתתפים בזיווג מבין איברי U.
<u>סיבוכיות</u>: בגרף שבנינו $k+n$ צמתים ולכל היותר $O(nk)$ קשתות. סיבוכיות מציאת זיווג מקסימלי בגרף היא $O((|V|+|E|)\cdot|V|)=O(((k+n)+(nk))\cdot(k+n))=O(nk\cdot(k+n))$.
<u>נכונות</u>: אם יש נציגים כנדרש אז הזיווג המקשר כל נציג לקבוצה אליה הוא שייך הוא זיווג חוקי בגרף שגודלו k ולכן בוודאי נקבל $m=k$. בכיוון השני, אם קיבלנו $m=k$ אז יש k איברים שונים מ-U $r_1,r_2,...,r_k$ המשתתפים בזיווג ולכל i $r_i \in S_i$.

<u>שאלה 4</u>

נתונים n נשים, n גברים ו- m שדכנים. לכל שדכן נתונה קבוצה חלקית של נשים וגברים אותם הוא מכיר. כל שדכן יכול לשדך כל גבר שהוא מכיר לכל אשה שהוא מכיר. כתבו אלגוריתם המוצא את

המספר המקסימלי של זוגות שניתן ליצור, בעזרת כל השדכנים יחד, בהינתן שלכל שדכן מותר לשדך לכל היותר k זוגות (k הוא קבוע של הבעיה). הוכיחו את נכונות האלגוריתם ונתחו את סיבוכיותו.

<u>תשובה</u>

נפתור על ידי רדוקציה לבעיית מציאת זרימה מקסימלית. נבנה לבעיה רשת זרימה: הרשת תכיל צומת לכל אישה, צומת לכל גבר ושני צמתים לכל שדכן - עבור הנשים שהוא מכיר, ועבור הגברים שהוא מכיר. בנוסף, כמובן, הרשת תכיל מקור ובור. מהמקור יש קשת לכל אישה, שקיבולה 1. מכל אישה יש קשת בקיבול 1 אל ה"צומת הנשי" של כל שדכן שמכיר אותה. באופן דומה, יש קשתות שקיבולן 1 מכל הגברים אל הבור, ומ"הצומת הגברי" של כל שדכן יש קשת בקיבול 1 אל כל הגברים שהוא מכיר. לבסוף, לכל שדכן יש קשת המחברת את שני הצמתים שלו, וקיבולה k. ניתן <u>קווים כלליים להוכחת הנכונות</u> (וודאו שאתם יודעים להשלימה): זרימה בשלמים בגרף מגדירה שידוך: כל מסלול זרימה מחבר בין אישה לגבר, דרך השדכן שביצע את השידוך. כמובן שכל שידוך מגדיר זרימה. זרימה מקסימלית תיתן שידוך מקסימלי. ברור ששדכן יכול לשדך רק גברים ונשים שהוא מכיר, וברור שהוא יכול לשדך לכל היותר k זוגות, בגלל מגבלת הזרימה "דרכו". מספר הצמתים ברשת שבנינו הוא $O(n+m)$, ומספר הקשתות הוא $O(n \cdot m)$. ניתן לחסום את הזרימה המקסימלית על ידי n, ולכן <u>סיבוכיות</u> האלגוריתם כולו היא $O(n^2 \cdot m)$.

<u>פרקים ז'+ח' וממ"ן 15</u>

<u>שאלה 1</u>

מטריצה מעגלית A היא מטריצה מגודל $n \times n$ כך שכל שורה של A מתקבלת על-ידי הזזה מעגלית של השורה שמעליה במקום אחד ימינה. כלומר, אם השורה הראשונה היא $(a_0, a_1, ..., a_{n-1})$, אז השורה השנייה היא $(a_{n-1}, a_0, ..., a_{n-2})$, השורה השלישית היא $(a_{n-2}, a_{n-1}, a_0, ..., a_{n-3})$ וכך הלאה, עד השורה האחרונה שהיא $(a_1, ..., a_{n-2}, a_{n-1}, a_0)$.

כתבו אלגוריתם יעיל ככל שתוכלו אשר מקבל כקלט מטריצה מעגלית A מגודל $n \times n$ ווקטור עמודה v באורך n ומחשב את המכפלה Av. הוכיחו את נכונות האלגוריתם ונתחו את סיבוכיותו.

<u>תשובה</u>

נפתור את הבעיה על ידי רדוקציה לבעיית חישוב כפל פולניומים.

תוצאת המכפלה המבוקשת היא וקטור עמודה $(r_0,\ r_1,\ \ldots,\ r_{n-1})$ כאשר $r_i = \sum_{j=0}^{n-1} A_{ij} \cdot v_j$. אבל אם השורה הראשונה של A היא $(a_0, a_1, ..., a_{n-1})$ אז $A_{ij} = a_{(j-i) \bmod n}$. יהי p פולינום שמקדמיו הם $(a_0, a_1, ..., a_{n-1})$. יהי q פולינום שמקדמיו הם $(v_{n-1},\ v_{n-2},\ \ldots,\ v_0)$ (איברי v בסדר הפוך). בשלב ראשון נחשב, את הוקטור C, מכפלת p ו-q. מהגדרת כפל פולינומים נקבל כי $C_i = \sum_{k=0}^{i} p_k q_{i-k} = \sum_{k=0}^{i} p_{i-k} q_k = \sum_{k=0}^{\min(i,n-1)} a_{i-k} v_{n-1-k}$.

לכן:

$$C_{n-1-i} + C_{2n-i-1} = \sum_{k=0}^{n-i-1} a_{n-i-1-k} v_{n-1-k} + \sum_{k=0}^{2n-i-1} a_{2n-i-1-k} v_{n-1-k} = \sum_{k=0}^{n-i-1} a_{n-i-1-k} v_{n-1-k} + \sum_{k=n-i}^{n-1} a_{2n-i-1-k} v_{n-1-k}$$

המעבר האחרון נובע מכך שעבור ערכי k הקטנים מ-n-i איברי a המתאימים שווים ל-0 ועבור ערכי k גדולים יותר מ-n-1 איברי v המתאימים שווים ל-0. נמשיך לפתח את השוויון ונקבל:

$$\sum_{k=0}^{n-i-1} a_{n-i-1-k} v_{n-1-k} + \sum_{k=n-i}^{n-1} a_{2n-i-1-k} v_{n-1-k} = \sum_{k=0}^{n-i-1} a_{(-i-1-k) \bmod n} v_{n-1-k} + \sum_{k=n-i}^{n-1} a_{(-i-1-k) \bmod n} v_{n-1-k} =$$

$$= \sum_{k=0}^{n-1} a_{(-i-1-k) \bmod n} v_{n-1-k} = \sum_{k=0}^{n-1} a_{(k-i) \bmod n} v_k = r_i$$

<u>סיבוכיות:</u> עלות חישוב C היא $O(n\log n)$ (ע"י FFT). עלות חישוב הוקטור r בעזרת ערכי הוקטור C, כפי שנובע מהשוויון שלעיל, היא $O(n)$. לכן עלות האלגוריתם כולו היא $O(n\log n)$.

שאלה 2

כתבו אלגוריתם יעיל ככל שתוכלו אשר מקבל כקלט גרף מכוון וחסר מעגלים $G=(V,E)$, קבוצה $V'\subseteq V$ וקשת $e\in E$ ומוצא את מספר המסלולים ב-G שמתחילים בצומת מ-V' **או** עוברים דרך e, אך לא מקיימים את שני התנאים ביחד, ושאורכם הוא חזקה של 2.

נתחו את סיבוכיות האלגוריתם והוכיחו את נכונותו.

תשובה

נפתור ע"י רדוקציה לכפל מטריצות. תהי A מטריצת הסמיכויות של הגרף. A_e תהיה המטריצה A בה מחוקה הכניסה שמתאימה לקשת e. A^k היא כידוע המטריצה שמכילה את מספרי המסלולים השונים באורך k, ו-A_e^k היא המטריצה שמכילה את מספרי המסלולים השונים באורך k שאינם עוברים דרך e. עבור k מסויים, מספר המסלולים הדרושים באורך k הוא:

$$\sum_{v\in V',u\in V} A_e^k[v,u] + \sum_{v\notin V',u\in V} A^k[v,u] - \sum_{v\notin V',u\in V} A_e^k[v,u]$$

המחובר הראשון סוכם מסלולים שמתחילים ב- V' ולא עוברים דרך e. ההפרש בין האיבר השני לשלישי נותן את המסלולים שלא מתחילים ב- V' ועוברים דרך e: האיבר השני סוכם את המסלולים שלא מתחילים ב-V' (חלקם כנראה עוברים דרך e וחלקם לא), והאיבר השלישי סוכם את המסלולים שלא מתחילים ב-V' ולא עוברים דרך e.

מאחר שהגרף חסר מעגלים, אורכו של כל מסלול בו חסום על ידי n-1, כאשר $|V|=n$, ולכן ערכי k שמעניינים אותנו הם רק עבור $k\leq n$-1 שהם גם חזקה שלמה של 2. לכן ניתן לחשב את המטריצות השונות על ידי $O(\log(n-1))$ מכפלות מטריצות. מכפלת כל מטריצה נעשית בזמן $O(n^{\log 7})$ (ע"י האלגוריתם של שטראסן) ועלות החישובים הנוספים בכל שלב היא $O(n^2)$, ולכן בסך הכל <u>הסיבוכיות</u> היא $O(\log(n-1)\cdot n^{\log 7})$.

שאלה 3

תהי A מטריצה בגודל $n\times n$. יהי $S(n)$ הזמן הלוקח לחשב את המטריצה A^2. הראו כי ניתן להכפיל שתי מטריצות X,Y בגודל $n\times n$ בזמן של לכל היותר $S(2n)$.

תשובה

נראה את הדרוש על ידי שנראה רדוקציה מבעיית כפל מטריצות אל בעיית העלאת מטריצה בריבוע. כלומר, נראה כי בהינתן שתי מטריצות, קלט לבעיית כפל מטריצות, ניתן לחשב את מכפלתן בעזרת אלגוריתם שיודע להעלות מטריצה נתונה בריבוע. ובכן, בהינתן שתי מטריצות X,Y בגודל $n\times n$

נבנה מהן את המטריצה $A_{2n\times 2n}=\begin{pmatrix}0 & X\\ Y & 0\end{pmatrix}$. ע״פ הנתון בשאלה ניתן לחשב ב-$S(2n)$ את $A^2=\begin{pmatrix}XY & 0\\ 0 & YX\end{pmatrix}$. מתוכה ניתן לקבל את תוצאת מכפלת שתי המטריצות הנתונות, ע״י העתקת הרבע השמאלי העליון. עלות בניית המטריצה A היא $O(n^2)$, וכך גם עלות הוצאת מכפלת המטריצות X,Y לפלט. ברור כי $S(2n)$, עלות חישוב המטריצה A^2, היא לפחות $O(n^2)$ (כי זהו גודל הפלט), ולכן זהו האיבר הדומיננטי והעלות הכוללת היא $O(S(2n))$.

<u>שאלה 4</u>

נדון בבעיה הבאה. בהינתן שתי קבוצות A,B שכל אחת מהן היא תת-קבוצה בגודל n של הקבוצה $U=\{1,2,3,...,10n\}$, יש למצוא את קבוצת כל הסכומים האפשריים של איבר מ- A ואיבר מ-B, כלומר $A+B=\{a+b : a\in A, b\in B\}$.

<u>דוגמה</u>: אם B={3,7,10} A={1,5,8} אזי $A+B$={4,8,11,12,15,18}.

א. הראו כי גודל קבוצת הפלט הוא $O(n)$.

ב. כתבו אלגוריתם הפותר את הבעיה הנ״ל בסיבוכיות $O(n\cdot\log n)$. הוכיחו את נכונות האלגוריתם ונתחו את סיבוכיותו.

<u>תשובה</u>

א. הקבוצה $A+B$ מוכלת בקבוצה $\{2, 3, \ldots, 20n\}$ ולכן גודלה חסום ע״י $20n$-1 והוא לפיכך $O(n)$.

ב. נפתור ע״י רדוקציה לבעיית כפל פולינומים. נגדיר עבור הקבוצה A פולינום $p_A(x)=\sum_{j=1}^{10n} a_j x^j$ כאשר מקדמיו a_j נקבעים כך: אם $j\in A$ אז $a_j=1$, אחרת $a_j=0$. בדומה נגדיר פולינום $p_B(x)$. נחשב את פולינום המכפלה של שני הפולינומים האלו <u>בסיבוכיות</u> $O(n\cdot\log n)$ (ע״י FFT). הפלט יהיה כל האינדקסים j כך של-j מקדם שונה מ-0 בפולינום המכפלה. כדי להוכיח את <u>נכונות</u> האלגוריתם יש להראות כי המקדם של x^j בפולינום המכפלה מתאר את מספר הזוגות של איבר מ-A ואיבר מ-B שסכומם j (וודאו כי הינכם יודעים להוכיח זאת).