**ממן 11**

**1)**

**הרעיון המרכזי של האלגוריתם**:

נשתמש באלגוריתם זיווג היציב יהי S קבוצת הספינות ו-P קבוצת הנמלים.

נתאים כל ספינה לנמל, כך שאם ספינה מותאמת לנמל פירוש הדבר הוא שכאשר הספינה תגיע לנמל אליו היא מותאמת, היא תישאר בו עד לסוף החודש לצורך תחזוקה.

כמו באלגוריתם המוצג בספר מספר הספינות והנמלים זהה, ואסור ש2 ספינות יבצעו תחזוקה באותו נמל.

נגדיר כעת את העדפות:

**ההעדפות של ספינה:**

ספינה s תעדיף את נמל p על פני נמל p’ אם נמל p נמצא בלוח הזמנים של s לפני נמל p’.

בדרך זו, הנמל הראשון בלוח הזמנים של ספינה מסוימת יהיה בעדיפות הראשונה שלה, הנמל השני בעדיפות השנייה, וכן הלאה.

**ההעדפות של נמל:**

נמל p יעדיף את ספינה s על פני ספינה s’ אם ספינה s מגיעה לנמל, לפי לוח הזמנים המקורי, אחרי ספינה s’.

בדרך זו, הספינה שעתידה להגיע אחרונה לנמל תהיה בעדיפות הראשונה שלו, הספינה שעתידה להגיע אחת-לפני-אחרונה תהיה בעדיפות השנייה, וכן הלאה.

**פירוט האלגוריתם**:

נבנה את רשימת ההעדפות של הספינות, ע"י ריצה על כל הספינות. לכל ספינה נסרוק את לוח הזמנים שלה, ובכל פעם שנתקל בנמל כלשהו, נכניס אותו לסוף רשימת ההעדפות של הספינה.

כעת נבנה את רשימת ההעדפות של הנמלים ע"י לולאה על היום ה-i בחודש,מהסוף עד להתחלה (m עד 1).

בתוך הלולאה תהיה לולאה פנימית, שתרוץ על הספינה ה-j, החל מn ועד 1. בגוף הלולאה הפנימית נבדוק האם הספינה ה-j עוגנת בנמל כלשהו ביום ה-i בחודש. אם כן, נקרא לנמל זה p, ונכניס את הספינה ה-j לסוף רשימת ההעדפות של הנמל p. כעת ירוץ אלגוריתם הזיווג היציב.

במצב ההתחלתי, כל הספינות והנמלים הינם פנויים, כל עוד קיימת ספינה s פנויה שטרם הציעה נישואין לנמל אחר, ספינה s כזאת תציע נישואין לנמל p הראשון ברשימת ההעדפות שלה שהיא עוד לא הציעה לו נישואין בעבר. אם נמל p הוא פנוי אז הנמל p והספינה s מתארסים. אחרת, אם נמל p כבר מאורס לs’, אז הוא יבחן מי מדורג גבוה יותר ברשימת ההעדפות שלו מבין הספינות s וs’, ויתארס אליו, בדיוק כמו באלגוריתם המקורי. בסוף, כשלא נשארו ספינה או נמל פנויים, כולם מתחתנים והאלגוריתם עוצר. את אלגוריתם זיווג יציב נממש בעזרת מבני הנתונים המתוארים בסעיף 2.3 בספר הלימוד.

**נכונות האלגוריתם:**

ראשית נראה שרשימת ההעדפות שבנינו מקיימות את תנאי אלגוריתם הזיווג היציב. עלינו להראות שבכל רשימת העדפה של ספינה מופיע כל נמל בדיוק פעם אחת. הנ"ל נובע מכך שכל ספינה עוגנת בכל נמל פעם אחת בדיוק, ולכן בלוח הזמנים של כל ספינה נמצא כל נמל פעם אחת בדיוק. את רשימת ההעדפה אנו מחשבים ישירות מכל נמל בלוח הזמנים, ולכן ברשימת ההעדפה נמצא כל נמל בדיוק פעם אחת. עלינו גם להראות שבכל רשימת העדפה של נמל מופיעה כל ספינה בדיוק פעם אחת. כפי שהראנו קודם לכן, כל ספינה עוגנת בכל נמל בדיוק פעם אחת, ולכן גם בכל נמל עוגנות כל הספינות (בימים שונים, ע"פ לוח הזמנים שלהם). את רשימת ההעדפה של כל נמל אנו מחשבים מן הספינות שעוגנות באותו נמל, לפי סדר עגינתן, לכן רשימת ההעדפה של הנמל יורכב מכל הספינות. כעת, לפי אלגוריתם הזיווג היציב המקורי, כל ספינה משויכת לנמל בקשר חד-חד ערכי ועל. כל ספינה עובדת לפי לוח הזמנים המקורי שלה, אך כאשר היא מגיעה לנמל אליו היא "נשואה", היא עוצרת בו לתחזוקה עד לסוף החודש. עלינו להראות שכאשר ספינה עוצרת לתחזוקה בנמל מסוים, מאותו רגע והלאה, לא מבקרת שום ספינה אחרת באותו נמל (שכן אסור ל2 ספינות להיות יחדיו באותו נמל ממניעי בטיחות). במילים אחרות, עלינו להראות שאם ספינה s "נשואה" לנמל p, והיא מגיעה אליו ביום t, אז לא קיים  שעוגן בנמל p בזמן  .

נניח בשלילה שקיים s' כזה. אם קיים s' כזה, אז פירוש הדבר הוא שבלוח הזמנים המקורי הספינה s' אמור לעגון בנמל p בזמן מאוחר יותר מאשר *הספינה s. אם כך הדבר, אז הדירוג של* s' *הוא גבוה יותר משל s ברשימת ההעדפות של הנמל p, לפי הגדרת רשימת ההעדפות של הנמלים. לפיכך, כאשר הנמל יצטרך לברור מבין s ל* s'*, הוא יעדיף את* s'*, שכן הוא מדורג גבוה יותר. הנמל p יעדיף להתארס ל*s' *במקום לs. אך ידוע לנו שp וs "נשואים", בסתירה להנחה.*

*לכן – לא* קיים  שעוגן בנמל p בזמן  בזאת הוכחנו שהאלגוריתם שהצענו עובד.

**ניתוח זמן הריצה:**

רשימת ההעדפות של הספינות מחושבת ע"י ריצה על n ספינות, כאשר בגוף הלולאה אנו סורקים את m ימי החודש. כל שאר הפעולות רצות בזמן קבוע. לכן רשימת ההעדפות רצה ב 

רשימת ההעדפות של הנמלים מחושבת ע"י ריצה על m ימי החודש, כאשר בגוף הלולאה אנו רצים בלולאה פנימית על n הספינות, ובכל פעם מבצעים פעולות בזמן קבוע. לכן רשימת ההעדפות רצה ב 

זמן הריצה של אלגוריתם הזיווג היציב במימוש היעיל ביותר הוא, כפי שהוכח בעמודים 50-52 בספר הלימוד.

מכיוון שנתון ש , ניתן להסיק שזמן הריצה של כל האלגוריתם הוא:

**2)**

**הרעיון המרכזי של האלגוריתם**:

*על מנת לכוון את הגרף, עלינו שיהיה בו מעגל אחד לפחות. אם יש מעגל, נמצא קודקוד במעגל, נראה אותו כמעין "שורש" ונכוון בסריקת* BFS.

**האלגוריתם**:

*ראשית, נניח שהגרף מכוון ניקח קודקוד כלשהו, ונריץ עליו סריקת DFS נקבל עץ T בחזרה, אם כל הקשתות בT הן קשתות עץ, אז האלגוריתם יחזיר "לא", אחרת, קיימת קשת חוזרת . נכוון אותה כ. כעת נפנה אל כל הקשתות היוצאות מv שאינן קשתות מכוונות ונכוון אותן אל הקשתות הבאות בתור, וכן הלאה עד שלא נמצא קשתות לא מכוונות. נדאג להדפיס למסך כל קשת שכוונה. בנוסף, האלגוריתם מחזיר "כן".*

**נכונות האלגוריתם:**

*ב*T *יש 2 סוגי קשתות: קשתות עץ, קשתות חוזרות, אם כל הקשתות הן קשתות עץ, הרי שמתקיים ולכן גם G המקורי הוא עץ. לכן, אם בG יש n צמתים, אז יש בG בדיוק קשתות (משפט 3.1). אם ננסה להגדיר גרף מכוון G’ נקבל ש . אם סך דרגות הכניסה הוא n-1 אז בהכרח קיים קודקוד שדרגת הכניסה שלו היא 0. במצב זו אנו דוחים.*

*אחרת, קיימת קשת חוזרת . נכוון אותה כ. נבחין שלv בהכרח יש קשתות היוצאות ממנו, שכן אחרת לא היתה קשת חוזרת, אלא קשת עץ. אנו מבצעים את הכוונות הקשתות במעין סריקת BFS לפי שכבות כאשר v הוא השורש. אנו מכוונים את כל הקשתות כך שכל צומת ברמה מצביע על צומת ברמה . כך דרגת הכניסה של כל צומת היא לפחות 1, והקודקוד v עצמו גם דרגתו לפחות 1 שכן קיימת קשת .*

**סיבוכיות זמן הריצה:**

את רכיבי הקשירות השונים ניתן למצוא בעזרת DFS או BFS בזמן של , כנל זמן הריצה של בניית DFS וסריקת עץ BFS לכן סה"כ *זמן ריצה* 

**3)**

**הרעיון המרכזי של האלגוריתם**:

*נתרגם את הפסוקית הלוגית לקשרי גרירה בדרך זו נוכל לתרגם את הפסוקיות לגרף, ולבצע חיפוש מסלול בגרף.*

**האלגוריתם**:

*נבנה גרף מכוון באופן הבא:*

*1) לכל ליטרל ניצור את הקודקודים ו- .*

*2) לכל פסוקית בנוסחה המקורית ניצור 2 קשתות: *

*כל עוד קיים קודקוד שלא עבר השמה:*

*נבחר קודקוד כזה.*

*נריץ סריקת DFS ממנו.*

*אם לא נמצא בקבוצה שמצאנו בסריקה, נבצע השמה: נגדיר T (אמת) לכל קודקוד בקבוצה שנמצאה, ו-F (שקר) לכל הקודקודים הנגדיים שלהם.*

*אחרת, נריץ סריקת DFS על במקום.*

*אם לא נמצא בקבוצה שמצאנו בסריקה, נבצע השמה: נגדיר T (אמת) לכל קודקוד בקבוצה שנמצאה, ו-F (שקר) לכל הקודקודים הנגדיים שלהם.*

*אחרת, האלגוריתם יחזיר כישלון.*

*בסיום האלגוריתם יחזיר הצלחה.*

**נכונות האלגוריתם:**

*לפי כללי הלוגיקה, מתקיים לכל בוליאניים:*

אנו מריצים סריקת DFS החל מv ובכך מקבלים את רכיב-הקשירות שלו. כל הקודקודים שאנו מקבלים חזרה מהסריקה הם צאצאים של v. אם נמצא בהם את , הרי שאי אפשר להציב אמת בv שכן אז נקבל סתירה במסלול הבוליאני. במקרה זה ננסה לראות האם בסריקת DFS החל מ נמצא את v. אם כן, אז ו- נגישים הדדית, ולכן שום הצבה לא תגרום לנוסחה להיות ספיקה, והאלגוריתם דוחה. אחרת, פירוש הדבר שקיים איזשהו מסלול שלא גורם, פירוש הדבר שנוכל להגדיר אמת לכל הליטרלים בו, וזה לא יגרום לשום סתירה. *בסוף הריצה האלגוריתם הצליח להגדיר ערכי אמת ושקר לכל הליטרלים, ולא נמצאו שום סתירות.* לפי האופן בו בנינו את הגרף, *המסלולים שקולים לתנאים הבולינאיים המקוריים, ולכן ערכי הליטרלים מספקים את הנוסחה המקורית.*

**סיבוכיות זמן הריצה:**

בניית הגרף רצה ב אנו רצים בלולאה על קודקודים שעוד לא ביצעו להם השמה, ובכל פעם מריצים, במקרה הגרוע, סריקת DFS כפולה על קודקודים, ובמהלך סריקה זו אנו מבצעים השמה ל. אם נסמן ב את מספר הקודקודים שעשינו להם השמה בסריקה הכפולה , נקבל ש  כמובן שגם עברנו על  קשתות, ולכן גוף הלולאה כולו רץ בעלות לינארית.

*סה"כ זמן ריצה* 

**4)**

**הרעיון המרכזי של האלגוריתם**:

*נסמן (X היא קבוצת קודקודי V שאינם קודקודים מועדפים, בלי s ו-t).*

*נגדיר גרף מכוון חדש G’ בו נגדיר 5 חלקים שונים, כאשר כל חלק יהיה מחובר להבא בתור:*

*אם קיים מסלול שמגיע מs לt בגרף , הרי שהוא מקיים את תנאי השאלה. נוכל לסרוק את הגרף בסריקת BFS ואז נקבל את המסלול הקצר ביותר ב בין s לt, אם קיים.*

**האלגוריתם**:

*נסמן .*

*נבנה מ גרף מכוון חדש באופן הבא:*

*לכל קודקוד מועדף ניצור בגרף החדש את .*

*את נעתיק כמו שהם לגרף החדש.*

*לכל קודקוד ניצור בגרף החדש את .*

*לכל קשת ניצור בגרף החדש את*

*לכל קשת ניצור בגרף החדש את*

*לכל קשת ניצור בגרף החדש את*

*לכל קשת ניצור בגרף החדש את*

*לכל קשת ניצור בגרף החדש את*

*לכל קשת ניצור בגרף החדש את*

*לכל קשת ניצור בגרף החדש את*

*כעת על הגרף החדש נריץ סריקת BFS החל מקודקוד s. כאשר נתקל בt בסריקה, המסלול מs עד t הוא המסלול הקצר ביותר שמקיים את תנאי השאלה. אם לא נמצא t בסריקה, הרי שלא קיים מסלול כזה.*

**נכונות האלגוריתם:**

*נסביר מדוע המסלול שאנו מוצאים מגיע מs אל t דרך בדיוק שני קודקודים מועדפים מ :*

*בגרף המכוון החדש שכפלנו חלק מהקודקודים והקשתות כדי שנוכל לזהות את המסלול הדרוש.*

*מוגדר בתצורה של 5 חלקים שונים, כאשר כל חלק יהיה מחובר להבא בתור:*

*כללי הקשתות שנגדיר בגרף החדש מתוארים בסעיפים 1-7, כאשר הסבר הסעיפים להלן:*

*מאפשר מעבר מקודקוד התחלתי s אל איבר אחר כלשהו בשלב 1*

*מאפשר מעבר מקודקוד התחלתי s ישירות לשלב 2*

*מאפשר מעברים משלב 1 ל2 ומשלב 3 ל4*

*מאפשר מעבר משלב 2 ל3 ומשלב 4 ל5*

*מאפשר לעבר משלב 2 ל4*

*מאפשר מעבר משלב 4 לקודקוד הסופי t*

*מאפשר מעבר משלב 5 לקודקוד הסופי t*

*כל מסלול בגרף החדש שיגיע בהצלחה לs לt יקיים את תנאי השאלה.*

*נסביר מדוע המסלול שנמצא הוא בהכרח הקצר ביותר:*

*אנו מריצים סריקת BFS ולכן נקבל תמיד את המסלול הקצר ביותר מכל כל קבוצת המסלולים האפשריים שיקיימו את תנאי השאלה. זאת כי BFS סורק את העץ לפי שכבות (ז"א, בהינתן שכבות , BFS יסרוק את שכבה i לפני שכבה j).*

*נסביר מדוע המסלול שמצאנו ב אכן קיים בG:*

*נבחין שבבניית הקשתות הוספנו לקשתות הקיימות סימנים מיוחדים (מספרי אינדקסים וסימן טאג). כמובן שמחיקת הסימנים המיוחדים ישחזר את אותן קשתות מקוריות שקיימות בG המקורית. לכן המסלול תקין עבור G.*

**ננתח את סיבוכיות זמן הריצה:**

בניית הגרף רצה ב, שכן אנו רצים פעם אחת על כל הקשתות והקודקודים של גרף G ומהם בונים את .

זמן הריצה של סריקת העץ לפי BFS הוא .

*סה"כ זמן ריצה .*