מגיש: בן חג'ג'

תז: 311452254

ממן 11

שאלה 1

בעיית השידוך היציב:

בהינתן לוח זמנים עבור כל ספינה, עלי להראות כי ניתן למצוא לוח זמנים מקוצץ שיקיים את התנאי שאין שתי אוניות בו זמנית(באותו יום) באותו נמל.

פתרון : נשתמש בבעיית הזיווג היציב שמתוארת בתחילת הספר ונראה איך היא מקבילה לבעיה זו (נבצע רדוקציה).

הרעיון

נתונה קבוצה של n ספינות= S וn נמלים=P . נסמן בSxP את קבוצת הזוגות הסדורים האפשריים מן הצורה: (s,p) כאשר s הוא ספינה כלשהי וp נמל כלשהו.

נתאר אלגוריתם בוא נמצא זיווג מושלם – כלומר לכל ספינה ולכל נמל יש בדיוק התאמה אחת.

נראה שיש הקבלה לכך שכל גבר ואישה נשואים רק לאדם אחד בסוף התהליך ואף גבר/אישה לא נשאר/ת רווק/ה.

נשתמש ברעיון של הזיווג המושלם המוצע בספר , ונראה איך בעיית הספינות/נמלים מתאימה לבעיית הזיווג המושלם של גייל-שייפלי על גברים/נשים.

נגדיר נמלים באופן דומה לנשים וספינות לגברים.

רדוקציה

רשימת העדפות:

* לכל ספינה נגדיר את רשימת העדפות שלה בצורה הבאה:
  + לוח הזמנים של הספינה כולל את זמני העגינה שלה בכל נמל. ככל שהנמל מופיע מוקדם יותר בלוח הזמנים אנחנו נתייחס לנמל זה כמועדף יותר.
* לכל נמל נגדיר את רשימת העדפות שלו בצורה דומה:
  + נקח את כל לוחות הזמנים של הספינות ונבנה לוח זמנים לנמל באופן שייצג באיזה יום כל ספינה אמורה להגיע אליו. במידה ויש שתי ספינות שאמורות להגיע באותו יום נשמור אותן ביחד.
  + רשימת ההעדפות תהיינה כך שספינה שמגיעה "מאוחר" יותר לנמל , מועדפת יותר.

אם יש שתי ספינות באותו יום, העדפה תהיינה זהה.

תיאור האלגוריתם:

אנחנו נשתמש בטרמינולוגיה דומה לזוגות כמו "מאורסים" ו"זיווג" וכמובן הדימיון בין ספינות/נמלים לגברים/נשים ברור מאליו.

1. בניית רשימות העדפות:
   1. ספינות:
      1. עבור כל ספינה s נעבור על לוח הזמנים שלה מ1 עד m , עבור כל נמל שאנחנו רואים נכניס את הנמל לסוף הרשימה.
   2. נמלים:
      1. עבור כל יום d בחודש (נתחיל מסוף החודש לתחילת החודש) נעבור על לוחות הזמנים של כל ספינה :
         1. נבדוק אם ביום d ספינה s עוגנת בנמל p:
            1. אם כן,נכניס לסוף רשימת העדפות של p את s.
            2. אחרת – נמשיך לאיטרציה הבאה
2. ריצת אלג' הזיווג-היציב :
   1. בתחילת הריצה אף נמל לא משויך לאף ספינה ולהפך.
   2. כל עוד יש ספינה שלא משויכת לאף נמל ועדיין לא הציע לכל הנמלים להיות משודך אליו:
      1. תבחר ספינה כזו:
      2. בדוק ברשימת העדפות של הספינה מי הנמל שמדורג הכי גבוה אצלה ועדיין לא "ביקש" להיות "מזווג" אליו.
      3. אם הנמל פנוי , הספינה s והנמל p הנל "מאורסים" .
      4. אחרת הנמל "מאורס" לספינה אחרת s’:
         1. אם הנמל p מעדיף את s’ על גבי s
            1. s נשאר ללא זיווג
         2. אחרת , הנמל p מעדיף את s לכן:
            1. s וp מתארסים
            2. s’ חופשי ללא זיווג לנמל.
   3. החזר את רשימת ה"מאורסים", זו רשימת הזיווגים הסופית.

רשימת הזיווגים מתארת באיזה נמל כל ספינה עוצרת,ועוגנת לצורך התיקון.

אלג' הזיווג היציב ימומש כמתואר בסעיף 2.3 בספר כדי לוודא ריצה בזמן אופטימלי.

נכונות האלגוריתם:

* נראה כי מתקיימים התנאים הדרושים:
  + במצב ההתחלתי אף ספינה לא מזווגת לאף נמל לפי תנאי השאלה
  + כל ספינה נמצאת בכל נמל בדיוק פעם אחת – כלומר לכל ספינה יש העדפה ברורה של ספינה כלשהיא על אחרת, אין חוסר בהירות לגבי זה.
  + ברשימת העדפות של כל נמל מופיעה כל ספינה בדיוק פעם אחת > מתקיים מתוך הנתון שכל ספינה עוגנת באותו נמל בדיוק פעם אחת.
  + בסיום הריצה הקשר מחייב והספינה תשאר בנמל עד סיום עבודות התחזוקה.
* נראה כי מתקיים האילוץ שבו שתי ספינות לא ישהו באותו נמל באותו זמן :
  + כלומר שאם s וp נשואים כאשר s מגיעה לp ביום d כלשהו, לאחר יום d זה לא תגיע אף ספינה אחרת.
    - נניח בשלילה שs וp נשואים וספינה s’ מגיעה בימים שאחרי שספינהs הגיעה לp.
    - נובע מכך שהספינה s’ הייתה גבוהה יותר ברשימת העדפות של הנמל p.
    - ועל כן כאשר יצטרך הנמל לבחור ספינה הוא יבחר את s’ ולא את s.
    - הגענו לסתירה לכך s,p נשואים.
    - מ.ש.ל

סיבוכיות זמן הריצה:

לטובת בניית רשימת העדפות :

עבור n ספינות > o(n).

עבור n נמלים נעבור על כל ספינה בכל יום : o(n\*m).

ולכן צורך בניית רשימת העדפות נצטרך o(n\*m).

לפי עמוד 52 זמן הריצה של אלג' הזיווג היציב הוא O(N^2).

ומכיוון שm גדול מn לפי הנתון , זמן הריצה הכולל יהיה o(n\*m).

שאלה 2

הרעיון הכללי של האלגוריתם הוא: גרף מכוון כך שדרגת הכניסה של כל צומת הוא גדול מ0 יתקיים אם בגרף יש לפחות מעגל אחד. נתחיל מצומת כלשהוא ונכוון את הגרף בעזרת סריקת BFS עם שינויים קלים.

תיאור האלגוריתם:

1. נתחיל מצומת s כלשהו בסריקת DFS , נסמן את העץ שקיבלנו בT.
2. אם בT כל הקשתות הן קשתות עץ אזי לא קיים מעגל >
   1. נחזיר FALSE (אי אפשר לקיים את תנאי השאלה).
3. אחרת :
   1. קיימת קשת "חוזרת" (היא לא קשת עץ) נכוון אותה כך : e=(u,v).
   2. ניגש לכל הקשתות הלא-מכוונות היוצאות מv ונכוון אותן החוצה. נבצע את זה בשיטת

סריקת BFS כך שהקשת מכוונת כל פעם לשכבה הi+1.

* + 1. נבצע את התהליך בסריקת BFS, בשיטה רקורסיבית ונכוון החוצה את הקשת
    2. נעצור כאשר חזרנו לv ועברנו על כל הצמתים ונחזיר TRUE

נכונות האלגוריתם:

הפלט של ריצת DFS על G הוא עץ T מושרש ולו שני סוגי קשתות: קשתות עץ וקשתות חוזרות(אחורה).

* אם כל הקשתות הן קשתות עץ, כלומר אין קשתות חוזרות(אין מעגלים) אז : * , ולכן – G הוא עץ.*
  + נסמן מספר הצמתים בG : , ולכן לפי משפט 3.1
  + נניח כי ניתן להגדיר גרף מכוון על פי דרישות השאלה :
  + אז : כלומר סך דרגות הכניסה הוא n-1 אבל יש n צמתים אז בהכרח לצומת אחד אין קשת כניסה אליו ולכן זו סתירה.
  + נחזיר FALSE.
* אם ישנן קשתות חוזרות נסמן אותה e=(u,v) ונכוון קשת החוצה מu לv.
  + לv בהכרח יש קשתות יוצאות אחרת e לא הייתה קשת חוזרת.
  + הכוונת הקשתות מתבצעת על בסיס אלגוריתם BFS לפי שכבות כאשר v הוא השורש(כל הקשתות של צמתים בשכבה i מצביעים על קשתות בשכבה i+1).
  + בצורה זו, כאשר אנחנו סורקים את כל הצמתים עד שחוזרים חזרה לv(לוודא שסרקנו את כל הצצמתים) אנחנו אוכפים שדרגת הכניסה של כל צומת בגרף היא לפחות 1.

סיבוכיות האלגוריתם:

אלגוריתם DFS הוכח בספר שסיבוכיות זמן הריצה שלו היא : O(m+n).

אלגוריתם BFS גם הוא סיבוכיות זמן הריצה היא O(m+n). (התוספת שלנו לכיוונון תוך כדי ריצה של BFS לא משפיעה על זמן הריצה בסדר גודל ).

ולכן סה"כ זמן הריצה של האלגוריתם המוצע הוא O(m+n).

שאלה 3

רעיון האלגוריתם:

1. כל פסוקית תתורגם בעזרת זהויות לוגיות כך: ול :
2. כל ליטרל והנגדי שלו יהיו חלק מהגרף שנבנה.
3. נבצע השמה לכל רכיב קשירות נפרד ונוודא שאין סתירה.
   1. הרעיון הוא לוודא שאין מסלול בין ליטרל והנגדי שלו . במידה ויש נחזיר FALSE במידה ואין נחזיר TRUE.

תיאור האלגוריתם:

1. נבנה גרף G(v,e) כך:
   1. לכל ליטרל x ניצור את הקודקודים x ו.
   2. נחבר בקשת מכוונת כל קודקוד עם קודקוד אחר כך:
      1. אם קיים הקשר .

נחבר בקשת מכוונת בין ל ובין ל.

1. נעבור על הגרף כל עוד יש קודקוד v שלא עבר השמה:
   1. נבחר קודקוד v כזה ונריץ DFS :
      1. אם לא נמצא במסלול נבצע השמה לכל הקודקודים בקבוצה T ולכל הקודקודים הנגדיים F .
      2. אחרת - אם נמצא קודקוד : נריץ DFS על :
         1. אם v לא נמצא נבצע השמה לכל הקודוקדים בקבוצה T ולכל הקודקודים הנגדיים נמצא השמה F .
         2. אם v נמצא נחזיר FALSE .
   2. בסיום השמה מוצלחת, שלכל הקודוקדים בגרף יש השמה. נחזיר TRUE.

נכונות האלגוריתם

בכל איטרציה לכל צומת v שעדיין אין לו השמה נריץ עליו DFS כדי למצוא את כל הקודקודים ברכיב הקשירות שלו. אם אין את הקודקוד הנגדי של v בקבוצת הצמתים הנ"ל אז אפשר להמשיך לבדוק את שאר רכיבי הקשירות. אם כן קיים הקודקוד הנגדי של v בקבוצת הצמתים אז נקבל סתירה ומצב זה לא אפשרי(אין השמה אפשרית).

**אבל**  - צריך ואפשר לבדוק את הצד הנגדי , כלומר לבדוק את קבוצת הצמתים של בעזרת DFS. אם v לא בקבוצת הצמתים נקבל את ההשמה אחרת נקבל סתירה ונחזיר FALSE.

אם לא נקבל FALSE ולא הצמתים תהיה השמה נחזיר TRUE.

סיבוכיות האלגוריתם

בניית הגרף G : O(n+m). (עבור כל ליטרל יש קשת)

מהלך האלגוריתם:

* ריצה על כל קודקוד שלא בוצע לא השמה :
  + במקרה הגרוע( בן של v) , נריץ DFS פעמיים על כל v באיטרציה הראשית.
  + נסמן בk את מס' הקודקודים שנרוץ עליהם בלולאה החיצונית k בהכרח קטן מv.
* מס' הקודקודים שמבצעים עליהם השמה הוא קבוע בכל איטרציה חיצונית ולכן במהלך האלגוריתם נשאר בזמן ריצה של O(n).

לכן סיבוכיות זמן הריצה של האלגוריתם הינו O(n+m).

שאלה 4

רעיון האלגוריתם

נבנה גרף חדש G’ המורכב משלושה עותקים ערוכים של G.

נבצע שינויים בG כך שנבנה גרף חדש G’ שיקיים את הנדרש .

הרעיון הוא שבכל "פניה"/מעבר לקודקוד מועדף אנחנו "עוברים" לגרף הבא. משפנינו לשני קודקודים מועדפים אנחנו מגיעים לגרף השלישי ובכך "מסיימים".

בניית הגרף

בהינתן גרף G ,נבנה את G’ כך: נסיר את כל הצלעות המכוונות לקודקודים מועדפים.

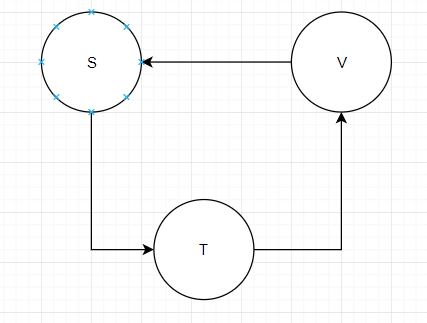
גרף זה בלי הצלעות שהורדנו יהיה הגרף G1,G2,G3.

נחבר בין הגרפים כך:

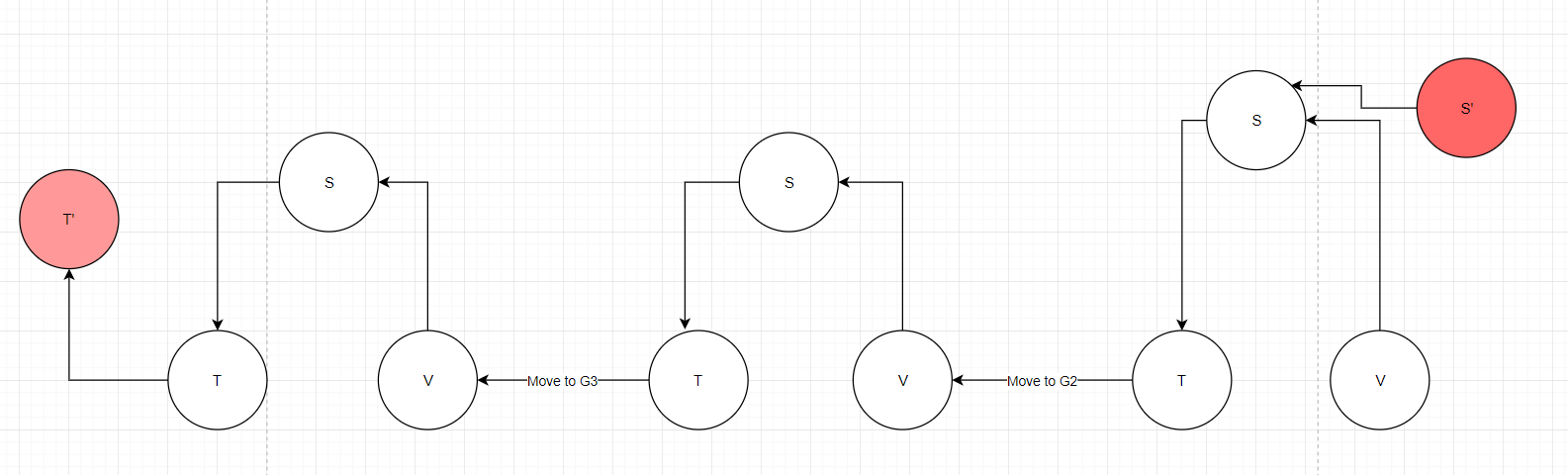
לכל צלע שהייתה מכוונת בגרף המקורי לקודקוד מועדף נייצר צלע שמכוונת מקודקוד זה בגרף G1 לקודקוד המועדף בG2 , באותו אופן עבור קודקוד כלשהו בG2 לקודקוד בG3.

נוסיף שני קודקודים s’ t’ כך : צלע מכוונת מs’ לs בגרף G1. צלע מכוונת מt בגרף G3 לt’.

דוגמא: הגרף לדוגמא מהממ"ן:



לאחר הבניה של הגרף החדש G’ , זה יראה כך:



כאשר v הוא צומת מועדף.

מהלך האלגוריתם

לאחר בניית הגרף החדש G’, נריץ BFS מs’ לt’ וזה יהיה המסלול הקצר ביותר מs’ לt’. אורך מסלול זה גדול ב2 מאורך המסלול המקורי הקצר ביותר בגלל התוספת של שני הצמתים s’ וt’ והקשתות שלהם. לכן מאורך המסלול שאנחנו מקבלים לאחר ריצת הBFS נחסיר 2 וזה אורך המסלול שנקבל.

נכונות האלגוריתם

קל לראות שקיימת התאמה חד חד ערכית ועל בין הגרף שבנינו לגרף המקורי:

לכל צומת יש 3 מופעים בגרף החדש שבנינו ושמרנו על כל הקשתות בגרף המקורי. כל קשר שקיים בגרף המקורי קיים גם בגרף החדש שבנינו. (לא כולל s’ ו-t’ שהוספנו).

לאחר בניית הגרף בצורה זו , הרצת BFS מחזירה לנו את המסלול הקצר ביותר , אם היה מסלול קצר יותר היינו מקבלים אותו בסריקה זו.

סיבוכיות זמן ריצה

בניית הגרף החדש : O(m+n) מעבר על כל הקשתות והקודקודים(תהליך היצירה מוסיף מכפלה בקבוע).

BFS – O(m+n).

לכן סה"כ לאלגוריתם זה: O(m+n)