רשימות מקושרות

מבוא למדעי המחשב - 67101

מרצה: אריה שלזינגר

מתרגלים: יפעת חדד ואורי מאיר

סוכם ע"י: שריה אנסבכר

סמסטר ב' תשפ"ג, האונ' העברית

אשמח לקבל הערות והארות על הסיכומים על מנת לשפרם בעתיד,
כל הערה ולו הפעוטה ביותר (אפילו פסיק שאינו במקום או רווח מיותר) תתקבל בברכה;
אתם מוזמנים לכתוב לי לתיבת הדוא"ל: sraya.ansbacher@mail.huji.ac.il.

לסיכומים נוספים היכנסו לאתר:
אקסיומת השלמות - סיכומי הרצאות במתמטיקה
https://srayaa.wixsite.com/math

1 רשימות מקושרות

רשימה מקושרת (או רשימה משורשרת) היא אוסף של איברים (nodes) שלכל אחד מהם שתי תכונות: המידע שלו ומצביע על האיבר הבא ברשימה, ההצבעה היא זו שיוצרת את ה"סדר" של הרשימה המקושרת (כל איבר בה אחרי זה שמצביע עליו), מלבדה אין שום סדר ברשימה. כדי שנוכל "לאחוז" ברשימה עלינו לשמור מצביע על "ראש" הרשימה[1] (ושוב, המצביע הזה הוא הדבר היחיד שהופך את ראש הרשימה לכזה), סוף הרשימה מוגדר להיות זה שהמצביע שלו מצביע על אובייקט שאינו חלק מהרשימה והוגדר להיות כזה שקובע את סוף הרשימה (בפייתון מקובל להשתמש ב-None).

♣ המחשות טובות וגם סיכום מעולה ניתן למצוא בערך "רשימות מקושרות" בוויקיפדיה.

♣ המצביעים הם הדבר היחיד שהופך את הרשימה מאוסף של פרטים לרשימה, אם נאבד אחד מהם לא נוכל לקבל את המשך הרשימה! אחת הטעויות הנפוצות ביותר בנושא זה הוא איבוד אחד המצביעים כשמבצעים שינוי כלשהו ברשימה (דוגמאות בהמשך), טעות נוספת היא ניסיון לגשת לאיבר שעליו מצביע האיבר האחרון.

♣ כפי שנראה בהמשך רשימות מקושרות אינן יעילות כל כך לעומת רשימות רגילות, יתרונן היחיד נעוץ בכך שהן מפרידות בין הסדר הלוגי של האיברים ברשימה לבין המיקום שבו נמצאים האיברים בזיכרון, ההשלכה העיקרית של הפרדה זו היא שכשמסירים איבר מן הרשימה (או מוסיפים איבר) אין צורך להזיז את כל אלו שאחריו ($O(1)$ לעומת $O(n)$ ברשימות רגילות), לכן רשימות מקושרות יעילות בעיקר במקרים שבהם אנחנו מסירים ומוסיפים איברים לרשימה לעיתים קרובות.

♣ רשימה מקושרת יכולה להיות מעגלית, זה קורה כאשר סוף הרשימה מצביע על תחילתה (ואז בעצם אין לרשימה סוף או התחלה).

הרשימה עצמה היא אובייקט בעל תכונה יחידה שהיא הכתובת של ראש הרשימה בזיכרון, ניתן להוסיף לאובייקט תכונות שימושיות נוספות כגון: אורך הרשימה (יעודכן בכל פונקציה שתשנה את אורך הרשימה), מצביע על סוף הרשימה ו/או מצביע על האיבר שלפניו.

כדי ליצור רשימה ריקה ניצור מצביע שיצביע על ראש הרשימה ונאתחל אותו להצביע על None, כדי ליצור רשימה בעלת איבר יחיד נאתחל את המצביע כך שיצביע על האיבר המתאים.

1.1 פעולות בסיסיות

אלגוריתם 1 מציאת אורך הרשימה

1. נאתחל $index := 0$ ו-$current := head$.
2. כל עוד המידע של $current$ אינו None:
   - נוסיף ל-$index$ 1.
   - נעדכן את $cuurent$ כך שיהיה שווה למצביע של האיבר הבא.
3. נחזיר את $index$.

[1] מצביע זה יקרא להלן "$head$".

אלגוריתם 2 הוספת איבר

1. ניצור את האיבר המבוקש או נביא אותו מכל מקום שהוא, על האיבר להיות אובייקט בעל שתי תכונות: המידע שלו ומצביע (שמיועד להצביע על האיבר הבא ברשימה).
2. נשנה את המצביע שלו כך שיצביע על האיבר שיהיה אחריו לאחר ההוספה (אם האיבר החדש נוסף בסוף הרשימה המצביע יצטרך להצביע על None).
3. נשנה את המצביע על האיבר שיהיה לפניו כך שיצביע על האיבר החדש (אם האיבר החדש נוסף בתחילת הרשימה מדובר ב-$head$).
4. אם אורך הרשימה הוא אחת התכונות שלה נוסיף לו 1.

♣ נשים לב שיש חשיבות לסדר בין סעיפים 1 ו-2: אם נבצע אותם בסדר הפוך נאבד את המצביע על האיבר שאמור להיות אחרי האיבר החדש.

♣ באותה צורה ניתן להוסיף רשימה אחת לאחרת: משנים את המצביע של סוף הרשימה הנוספת כך שיצביע על האיבר שמיועד להיות אחריו ואת המצביע של האיבר שמיועד להיות לפני הרשימה משנים כך שיצביע על ראש הרשימה הנוספת, אם אורך הרשימה הוא אחת התכונות שלה יש להוסיף לו את אורך הרשימה הנוספת.

♣ כדי להסיר איבר מהרשימה נשנה את המצביע של האיבר שלפניו[2] כך שיצביע על האיבר שאחרי האיבר המוסר מן הרשימה[3] ובאותה צורה ניתן להסיר תת-רשימה, כמובן שגם כאן אם אורך הרשימה הוא אחד מתכונות הרשימה יש לעדכן אותו בהתאם[4].

אלגוריתם 3 היפוך הסדר של רשימה

1. נאתחל $current := head$ ו-$head :=$ None.
2. כל עוד $current$ אינו None:
   - נגדיר את $next$ להיות האיבר שאחרי האיבר $current$.
   - "נוסיף" את $current$ לתחילת הרשימה באמצעות האלגוריתם להוספת איבר שראינו לעיל, כך $current$ יצביע על ראש הרשימה הנוכחי ו-$head$ יצביע על $current$ (כזכור הגדרנו בתחילת האלגוריתם $head :=$ None ולכן הרשימה בעצם ריקה בתחילת הלולאה, א"כ הוספה איבר איבר בתחילת הרשימה של כל האיברים תיצור בדיוק את הפעולה המבוקשת).
   - נעדכן $current := next$.
3. נחזיר את הרשימה.

[2] אם אנו מסירים את ראש הרשימה זהו המצביע שבאמצעותו אנו "אוחזים" ברשימה כולה.
[3] אם האיבר המוסר הוא האיבר האחרון המצביע יצטרך להצביע על None.
[4] נשים לב שלעומת הוספת רשימה לאחרת בהסרת תת-רשימה איננו יודעים את אורך תת-הרשימה שאנו מסירים ולכן עלינו לעבור עליה בלולאה כדי לדעת את האורך ($O(n)$).

אלגוריתם 4 בדיקה אם איבר נמצא ברשימה

1. נאתחל $index := 0$ ו-$current := head$.
2. כל עוד $current$ אינו None:
   - אם המידע של $current$ הוא האיבר המבוקש:
     - נחזיר את $index$ (או רק True).
   - אחרת:
     - נגדיר את $current$ להיות האיבר הבא.
3. נחזיר False.

2 רשימות מקושרות דו-כיווניות

יש להשלים פרק זה.