

## תרגיל 1 תאריך הגשה: 14.04.2013

### שאלה 1

בכל אחד משני הסעיפים, האלגוריתם צריך להיות בעל יחס תחרותיות נמוך ככל שתוכלו למצוא. אדם עומד בצומת של **ארבעה** רחובות (ראו איור). הוא מחפש ארנק שאיבד באחד מארבעת הרחובות,



במרחק של לפחות יחידת מרחק אחת מהצומת (אך איננו יודע באיזה מהרחובות איבד את הארנק). עלות אלגוריתם הינה מרחק ההליכה הכולל בחיפוש אחר הארנק, עד להגעה לנקודה בה נמצא הארנק. תכננו אלגוריתם לבעיה הזו, כלומר אלגוריתם הסורק את הרחובות ומוצא את הארנק.

ב. הכלילו את האלגוריתם של סעיף א' למקרה שבו בצומת יש  $t > 2$  רחובות.

### שאלה 2

בעיית הניחוש המקוונת **החסומה** מוגדרת כדלקמן:

נתון פרמטר טבעי  $k$ . יש לנחש מספר טבעי  $T$  על-ידי הצגת סדרת ניחושים עולה, הסדרה נפסקת כאשר הניחוש הנוכחי הוא גדול או שווה ל- $T$ .

האלגוריתם ישלם את סכום  $k$  המספרים האחרונים שניחש (או את סכום כל המספרים שניחש, אם הוא ניחש פחות מ- $k$  מספרים).

לדוגמה, אם  $k=3$  וסדרת הניחושים היא  $2,4,7,9,14,20,34,\dots$ , אז אם  $T=3$  אז האלגוריתם ישלם 6 ואילו אם  $T=15$  אז האלגוריתם ישלם 43.

א. עבור המקרה  $k=2$ : תכננו אלגוריתם בעל יחס תחרותיות 2 לכל היותר (כדי לתאר את האלגוריתם יש לתאר את סדרת הניחושים). הוכיחו חסם תחתון המוכיח שיחס התחרותיות של כל אלגוריתם הוא לפחות 2.

ב. עבור המקרה  $k=4$ : תכננו אלגוריתם בעל יחס תחרותיות נמוך ככל שתוכלו (יחס התחרותיות כאן צריך להיות **נמוך ממש מ-4**).

### שאלה 3

נגדיר את האלגוריתם הבא לבעיית הכיסוי המקוונת (הקמת צבירים על הישר הממשי). האלגוריתם נקרא **AlmostCenter (AC)**. כאשר מגיעה בקשה חדשה לנקודה  $p$ , אם  $p$  כבר מכוסה על-ידי צביר קיים אז לא עושים דבר. אחרת מקימים את הצביר  $[p-0.4, p+0.6]$ . הוכיחו שיחס התחרותיות של האלגוריתם **AC** הוא 3. לשם כך:

א. הראו דוגמה לקלט שעבורו יחס התחרותיות הוא 3 (או סדרת קלטים שעבורם יחס התחרותיות שואף ל-3).

ב. הוכיחו שיחס התחרותיות הוא 3 לכל היותר.

### שאלה 4

בעיית  $k$ -מרכז על ישר מוגדרת באופן הבא. זוהי בעיית **offline** בה נתונות  $n$  נקודות על הישר הממשי:  $V = \{v_1, v_2, \dots, v_n\}$ , ויש לבחור קבוצה  $U$  של  $k$  נקודות על הישר הממשי (לאו דווקא מתוך  $V$ ). כמו

בעיית המקורית, נגדיר לכל  $x \in V$ :  $D(x) = \min_{u \in U} d(x, u)$  ונרצה להביא את  $\max_{x \in V} D(x)$

למינימום.

א. תארו אלגוריתם הפותר את בעיית 1-מרכז על ישר בזמן הריצה  $O(n)$ .

ב. תארו אלגוריתם המשתמש בתכנות דינמי ומחשב פתרון אופטימלי לבעיית  $k$ -מרכז (כך שזמן הריצה

שלו הוא פולינומיאלי ב- $n$  וב- $k$ ). רמז: השתמשו בסעיף א' לתתי סדרות רצופות של נקודות מ- $V$ .