הטכניון - מכון טכנולוגי לישראל **הפקולטה למדעי המחשב**

17 לפברואר 2011

פרופ"ח ארז פטרנק
שחר תמנת

אלגוריתמים לניהול זיכרון - 236780
חורף תשע"ב – מועד א'

הנחיות:

- הבחינה עם חומר סגור.
- אסור להחזיק פלאפונים ברשות הנבחנים בזמן הבחינה.
- כתבו בצורה מסודרת ונקייה ובכתב ברור.
- בבחינה 3 שאלות, משך הבחינה שלוש שעות.
- נמקו את כל תשובותיכם.
- כאשר מבקשים להציע פיתרון אלגוריתמי לאיזושהי בעיה, יש לנסות לתת פיתרון יעיל ככל היותר במקום ובזמן.

בהצלחה !

שאלה 1: (22 נקודות) שאלה זו מתייחסת לאלגוריתם ה- Mostly Concurrent.

א. (4 נק) תארו כל פאזה באלגוריתם: scanning, card cleaning, ו- stop-the-world.
ב. (3 נק) מתי ואיך card הופך ל-dirty?
ג. (3 נק) מתי ואיך card מתנקה? (כלומר סימן ה-dirty שלו נכבה?)
ד. (4 נק) תאר סיטואציה בה אובייקט ייסרק פעמיים: פעם בעת הסריקה הרגילה ופעם בעת ניקוי ה-cards.
ה. (8 נק) האם יש מצב שבו ניתן לדעת בזמן הסריקה הראשונה (בשלב ה-scanning) שהאובייקט שאותו אנו סורקים כעת ייסרק שוב בזמן ניקוי ה-cards? אם לא – הסבירו מדוע זה לא ייתכן. אם כן – הציעו שיטה למנוע (או לפחות לצמצם) סריקות כפולות כאלה וכך לייעל את עבודת האיסוף.

שאלה 2: (53 נקודות) שאלה זו מתייחסת לאלגוריתם ה-Compaction.

א. (3 נק) לאילו מהאלגוריתמים הקלאסיים (mark&sweep, reference counting, copying) יכול השימוש ב-compaction לעזור? נמקו!
ב. (4 נק) אלגוריתמי הדחיסה המוכרים לנו מסדרים את האובייקטים לפי אחד משלושה סידורים: arbitrary, sliding, and linnearizing. הסבירו את טבעו של כל סידור ולכל סידור ציינו אם יש בו יתרון עבור תוכניות מסויימות. הסבירו מה היתרון ואיזו סוג של תוכניות ירוויחו ממנו.
ג. (4 נק) ציינו יתרון של האלגוריתם של Jonkers על האלגוריתם Lisp2, ציינו יתרון של ה-Compressor על האלגוריתם של Jonkers.
ד. (3 נק) איזו תכונה נדרשת מהאובייקטים ע"מ שיהיה אפשר להשתמש באלגוריתם השירשור של Jonkers?
ה. (8 נק) במעבר הראשון באלגוריתם של Jonkers מבצעים עדכון של מצביעים קדימה וגם threading של מצביעים אחורה. במעבר השני מבצעים עדכון של מצביעים אחורה וגם הזזה של האובייקטים.
מה יקרה אם נשנה את הסדר: במעבר הראשון רק נבצע threading של כל המצביעים אל האובייקט (גם קדימה וגם אחורה) ובמעבר שני נבצע לכל אובייקט: עידכון כל המצביעים שלו ואז הזזת האובייקט.
אם האלגוריתם ממשיך לעבוד נכון, איזו מן הגרסאות שלו יותר יעילה?
אם האלגוריתם לאחר השינוי יכול לבצע טעות, הסבירו (ע"י דוגמה) איזו טעות יכולה לקרות.
ו. (4 נק) שלושת השלבים באלגוריתם Lisp2 של SUN הם: install forwarding pointers, fix pointers, move objects. הסבירו מה נעשה בכל שלב.

במיקבול של SUN ממקבלים כל שלב בנפרד כך שיתאים לחלוקת ה-heap ל-n איזורים. הסעיפים הבאים מתייחסים למיקבול זה.

ז. (4 נק) כזכור המיקבול של האלגוריתם ממקבל כל פאזה בנפרד. תארו את המיקבול של כל פאזה.
ח. (3 נק) באופן כללי כשמבצעים מיקבול, יש לחלק את העבודה לחלקים אשר מתבצעים במקביל ע"י חוטים שונים. הסבירו מה השיקולים בחלוקת העבודה. תנו לפחות שיקול אחד מדוע כדאי לחלק את העבודה לחלקים קטנים ולפחות שיקול אחד מדוע כדאי לחלק לחלקים גדולים.
ט. (3 נק) נתבונן במיקבול של <u>השלב השני</u>. במיקבול המקורי, כל חוט לוקח על עצמו איזור שלם ומבצע בו את מה שדרוש לעשות בשלב השני. האם ניתן לחלק את העבודה למנות קטנות יותר, כך שכל חוט ייקח חלק מאיזור ולא את כולו? אם לא, הסבירו את הבעיה הנוצרת. אם כן, הסבירו מדוע זה עובד.
י. (3 נק) עתה נתבונן במיקבול של <u>השלב הראשון</u>. האם ניתן לעשות את אותו דבר גם בשלב הזה? כלומר, לתת לחוטים לעבוד במקביל על תתי-איזורים, מבלי לשנות את גדלי האיזורים שבתוכם מזיזים את האובייקטים? אם כן, הסבירו מדוע הדברים מסתדרים. אם לא, הסבירו את הבעיה.

יא. (3 נק) עתה נתבונן במיקבול של השלב השלישי. האם ניתן לעשות את אותו דבר גם בשלב הזה? כלומר, לתת לחוטים לעבוד במקביל על תתי-איזורים, מבלי לשנות את גדלי האיזורים שבתוכם מזיזים את האובייקטים? אם כן, הסבירו מדוע הדברים מסתדרים. אם לא, הסבירו את הבעיה.

יב. (3 נק) עתה נניח שרוצים לרכז את כל האובייקטים בתחילת ה-heap. עדיין רוצים למקבל לפחות חלק מהעבודה. נניח שמבצעים את השלב הראשון והשלישי באופן סדרתי. האם ניתן למקבל את השלב השני (בקלות)? נמקו!

יג. (8 נק) שוב, נניח שרוצים לרכז את כל האובייקטים בתחילת ה-heap ועדיין רוצים למקבל לפחות חלק מהעבודה. נניח שמבצעים את השלבים השני והשלישי באופן סידרתי ורוצים למקבל את השלב הראשון. הציעו דרך למקבל את השלב הראשון. נסו לתת לכל חוט לעבוד על איזור (למרות שההזזה עצמה לא תיעשה לפי איזורים) והסבירו כיצד ניתן בסוף השלב הראשון להוסיף שלב סידרתי קצר שמאפשר לשלבים הבאים לעבוד עם הפלט של השלב הראשון ביעילות.

שאלה 3: (25 נקודות) שאלה זו מתייחסת ל- Reference Counting)

מבין האופציות הבאות ל-write barrier עבור reference counting על אפליקציה רב-חוטית, קבעו איזה מהם עובד נכון. נגדיר שאופציה עובדת נכון אם כאשר עוצרים את החוטים במהלך הריצה ואם אף חוט לא נמצא באמצע ביצוע write barrier אז הערכים של ה-reference count נכונים לעת העצירה. נניח שה-write-barrier מתבצע גם על השורשים ולכן הספירה אמורה להיות מדוייקת. עבור אופציות לא טובות תנו דוגמא ל-race שיוצר בעיה. עבור אופציות טובות נמקו מדוע אין races בעייתיים.
לביצוע פעולת כתיבה Update(ptr, obj) נבצע:

א. (5 נקודות)
   a. קרא את הערך של ptr לתור רגיסטר R.
   b. הצב מצביע ל-obj בתוך ptr.
   c. הורד את ה- reference count של האובייקט המוצבע מ-R.
   d. העלה את ה- reference count של obj שהתקבל כפרמטר.

בסעיף זה אף אחת מהפעולות אינה אטומית (כלומר אינה מוגנת בסינכרון כלשהו.)

ב. (5 נקודות)
כמו ב-1 כאשר כל הפעולות a עד d מתבצעות יחד באופן אטומי (ע"י מנעול גלובלי יחיד על עידכוני מכוונים ב-heap.)

ג. (5 נקודות)
כמו ב-1 כאשר פעולות c ו-d מתבצעות כל אחת לחוד באופן אטומי (ע"י מנעול על האובייקט המתאים).

ד. (5 נקודות)
כמו ב-1 כאשר פעולות a ו-b מתבצעות (יחד) באופן אטומי (ע"י מנעול על האובייקט המכיל את ptr).

ה. (5 נקודות)
כמו ב-1 כאשר פעולות a ו-b מתבצעות יחד באופן אטומי ופעולות c ו-d מתבצעות באופן אטומי כל אחת לחוד.

<u>הבהרה</u>: פעולות מתבצעות יחד באופן אטומי כאשר לוקחים מנעול מבצעים את הפעולות כולן ואז משחררים את המנעול. פעולות מתבצעות כל אחת לחוד באופן אטומי כאשר לכל פעולה תופסים את המנעול ומשחררים אותו אחריה.
<u>הבהרה נוספת</u>: האובייקט אותו נועלים עבור שינוי reference count (פעולות c או d) הוא האובייקט שלו משנים את ה-reference count. האובייקט שאותו נועלים בעת שינוי מכוון (פעולות a או b) הוא האובייקט שמכיל את ptr אשר אותו משנים.