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

11 לספטמבר 2015

פרופ"ח ארז פטרנק
נחשון כהן

אלגוריתמים לניהול זיכרון - 236780
אביב תשע"ה – מועד ב'

הנחיות:

- הבחינה עם חומר סגור.
- אסור להחזיק פלאפונים ברשות הנבחנים בזמן הבחינה.
- כתבו בצורה מסודרת ונקייה ובכתב ברור.
- בבחינה 4 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: (41 נקודות) שאלה זו מנסה לשפר את אלגוריתם הincremntal copying של Baker

א. (5 נק) כאשר מנסים להריץ אלגוריתם copying כמו שהוא (כפי שתוכנן לריצה לא מקבילית) באופן מקבילי לריצת התוכנית נוצרות בעיות. תארו דוגמא לריצה שבה יש בעיה כזו.

ב. (4 נק) תארו כיצד האלגוריתם של Baker מתגבר על הבעיה בעזרת read barrier. מה השמורה שהאלגוריתם שומר במהלך הריצה?

ג. (4 נק) כיצד עובד אלגוריתם incremental? מי מבצע את העבודה של הקולקטור? מתי הוא מבצע אותה? מה היתרון של ביצוע כזה?

ד. (5 נק) האם כמות העבודה הנדרשת להעתיק אובייקטים במהלך האיסוף בשיטת בייקר תלויה באיזה קריאות מבצעת התוכנית במהלך האיסוף או שכמות העבודה קבועה? כלומר, האם תוכנית שמבצעת הרבה קריאות (read barriers) במהלך האיסוף תבצע יותר עבודת העתקה מתוכנית שמבצעת מעט קריאות? או שכל התוכניות תבצענה אותה עבודת העתקה? (למען הפשטות, נניח שאין הקצאות אובייקטים חדשים בזמן ביצוע האיסוף.)

ה. (5 נק) ה- read-barrier באלגוריתם של Baker הוא יקר וגורם לירידה בביצועים. לכן רצו למצוא פיתרון שמשתמש רק ב write barrier. נניח שבאלגוריתם המקורי היו משתמשים רק ב write barrier (כלומר בזמן read של פוינטר לא עושים כלום. רק בזמן write של פוינטר מפעילים את הפעולה שבייקר הפעיל בזמן קריאה.) תנו דוגמא שבה האלגוריתם לא עובד טוב.

ו. (5 נק) ננסה לבנות אלגוריתם שמצליח לעבוד עם write barrier. הרעיון הוא שבמקביל לעבודת התוכנית המקורית על איזור ה-from space, הקולקטור מעתיק את כל האובייקטים ל to space. אחרי שהוא גומר להעתיק את הכל הוא עוצר את התוכנית, מתקן את השורשים של התוכנית להצביע למקום המועתק ב to space ואז נותן לתוכנית להמשיך לעבוד על to space. תנו דוגמא למה הצעה נאיבית כזו לא תעבוד ללא תיקונים.

ז. (8 נק) כדי שההצעה בסעיף הקודם תעבוד נכון, הציעו להוסיף write barrier שבו התוכנית תרשום יומן של כל שינוי שהיא עושה. לאחר שהקולקטור יגמור להעתיק את כל האובייקטים ל to space, הוא יעצור את החוטים, יבצע את כל התיקונים הדרושים ב to space לפי היומן שרשמה התוכנית ואז יתקן את השורשים וייתן לתוכנית לרוץ על to space.

   a. (3 נק) האם מספיק לרשום ביומן שינויים של מחוונים? או שצריך לרשום שינוי לכל שדה כולל למשל לשלמים? נמקו!

   b. (5 נק) כאשר רשום ביומן שמחוון הנמצא בכתובת p עודכן להצביע אל כתובת q, אז גם המחוון p וגם הכתובת שלו q נמצאים ב from space. הסבירו כיצד ניתן לעדכן את to space. איך מוצאים את המחוון שצריך לעדכן ב to space ואיך מוצאים את הכתובת אליה הוא צריך להצביע ב to space?

ח. (5 נק) הועלתה הצעה לשיפור המקביליות של האלגוריתם: לאחר שיועתקו כל האובייקטים ל to space אז הקולקטור יתחיל לתקן אותו לפי היומן במקביל לריצת התוכנית. הסבירו חיסרון ויתרון של שיטה זו. האם לדעתכם כדאי להשתמש בה? נמקו.

שאלה 2: (30 נקודות) שאלה זו מתייחסת לאלגוריתם הרכבת

א. (2 נק) למה משמש אלגוריתם הרכבת?

ב. (2 נק) מדוע צריכים להשתמש ברכבות ולא מספיק להשתמש בקרונות?

ג. (4 נק) תארו את הסיטואציה שבה נוצר הבאג של זליגמן וגראורופ.

ד. (6 נק) תכננו תוכנית שמגבירה את הסיכוי שהבאג של זליגמן וגראורופ יקרה. יש להציג את התוכנית (מספיק פסאודו קוד) ואז לומר מה מבנה ה-heap שנוצר ולנמק מדוע הסיכוי עולה. (שימו לב שהתוכנית לא יודעת מתי יתחיל האיסוף.)

ה. (4 נק) תארו תוכנית שבה הבאג של זליגמן וגראורופ לא יכול להופיע. תארו את התוכנית, את מבנה ה-heap ונמקו כמו בסעיף הקודם. על התוכנה לייצר מספיק אובייקטים ולאפשר להם לעבור לדור הישן ליצירת כמה קרונות. כלומר, לא יתקבל פיתרון בו כל אובייקט מת מיד לאחר שהוא נוצר.

ו. (8 נק) באלגוריתם הרכבת נשמרות remembered sets בין הקרונות. כאשר יש מחוון מקרון המצביע לקרון אחר, וצריך לשמור את המחוון ב remembered sets, אנו רוצים להשוות בין האופציות הבאות:
   i. לשמור את כתובת האובייקט המוצבע,
   ii. לשמור את כתובת האובייקט המצביע,
   iii. לשמור את כתובת המצביע עצמו.
   b. (2 נק) ציינו יתרון של שיטה iii על שיטה ii.
   c. (2 נק) ציינו יתרון של שיטה ii על שיטה iii.
   d. (4 נק) למה שיטה i היא בעייתית ואין נוהגים להשתמש בה?

ז. (4 נק) האם התשובות שנתתם בסעיף הקודם מתאימות גם לשמירת remembered sets עבור מחוונים בין דוריים רגילים? או שלגביהם המצב שונה? נמקו.

שאלה 3: (27 נקודות) שאלה זו מתייחסת לאלגוריתמי ה-Compaction

א. (5 נק) בשפת תכנות מסויימת כל האובייקטים באותו גודל. נשאלת השאלה האם עבור ריצה של תוכנית בשפה הזו יש צורך ב- compaction. תארו צורה אחת שבה compaction יעזור ותארו דרך אחרת שבה compaction מועיל לשפות אחרות (עם אובייקטים בגדלים משתנים) אבל לא לשפה הזו.

ב. (4 נק) אלגוריתם ה-2 fingers הוא הפשוט ביותר לדחיסה אבל יש לו שני חסרונות מרכזיים. ציינו שני חסרונות שלו יחסית לאלגוריתמים אחרים.

ג. (18 נק) באלגוריתם של IBM מחלקים את ה heap לשטחים (areas) וכל שטח מחולק לבלוקים קטנים.
   a. (3 נק) הסבירו את השימוש של areas באלגוריתם הדחיסה של IBM. במה מועילה חלוקת ה-heap לשטחים? מדוע אי אפשר פשוט להתייחס ל heap כשטח אחד גדול?
   b. (3 נק) מה היתרון בשימוש בבלוקים? מדוע לא מספיק שטחים?
   c. (3 נק) הסבירו איך מחשבים כתובת של אובייקט לאחר הזזתו. כלומר, בהינתן כתובת A של אובייקט לפני ההזזה, איך נדע מה כתובתו לאחר ההזזה?
   d. (3 נק) מה היתרון בשמירת טבלה חיצונית ל heap שבתוכה מידע על תזוזת אובייקטים, יחסית לשמירת כל המידע ב heap ב forwarding pointers כפי שנעשה באלגוריתם של LISP2?
   e. (3 נק) במהלך השימוש באלגוריתם, הסתבר שיש הרבה בלוקים שיש בהם רק אובייקט אחד. הציעו דרך לייעל את חישוב הכתובת החדשה עבור בלוקים מסוג כזה.
   f. (5 נק) כמו כן הסתבר שיש בלוקים שבהם יש יותר מאובייקט אחד, אבל כל האובייקטים בבלוק צמודים אחד לשני (ללא "חורים" בין האובייקטים החיים). הציעו דרך לייעל את החישוב עבור בלוקים כאלה. בהצעה שלכם ניתן לשנות את המידע הנשמר בטבלת הבלוקים.