Azrieli
College of Engineering
Jerusalem
עזריאלי
מכללה אקדמית להנדסה
ירושלים

המחלקה להנדסת תוכנה

פרויקט גמר – תשפ"ג

פיתוח אלגוריתם למציאת מרוץ קדימויות

Algorithm for race detection

מאת

אלה רוזנברג

318323086

מנחה אקדמי: דר' צור לוריא אישור: דר' צור לוריא תאריך:1.12.2022

אחראי תעשייתי: מר יוסי קריינין אישור: תאריך: 1.12.2022

Ramat Bet Hakerem, P.O.B 3566, Jerusalem 9103501 רמת בית הכרם, ת.ד. 3566, ירושלים מיקוד
Tel. +972-2-6588000 | Fax. +972-2-6588001 | Email: info@jce.ac.il | Web: www.jce.ac.il

מערכות ניהול הפרויקט:

#	מערכת	מיקום
1	מאגר קוד	github.com/ellaro/BSc
2	יומן	github.com/ellaro/BSc/issues

מידע נוסף (מחק את המיותר)

סוג הפרויקט	תעשייתי - Mobileye
פרויקט מח"ר	לא
פרויקט ממשיך	זה פרויקט חדש

מבוא

חישוב מקבילי הוא כאשר מריצים אלגוריתם על כמה מעבדים במקביל.
כל מעבד מריץ תת משימה מתוך האלגוריתם ובסוף הפלט הוא שילוב כל התוצאות.
יש לשיטה זו יתרון משום שזה מקטין את זמן הריצה משמעותית.
אחד החסרונות הבולטים הוא race condition (מרוץ קדימויות) מצב ששני מעבדים צריכים לגשת לאותו תא בזיכרון באותו הזמן.
קשה לאבחן מצב כזה בשיטות בדיקה סטנדרטיות (testing).
זה משפיע על האלגוריתם להיות לא דטרמיניסטי כי כל קלט לא בהכרח יביא לפלט אחד, הסדר לא ידוע מראש, לכן זאת בעיה NP קשה .
יש צורך לתזמן בין שתי המשימות האלה כדי לא לאפשר מצב כזה.

תיאור הבעיה

בחישוב מקבילי רוצים להריץ אלגוריתם על כמה מעבדים במקביל ובכך להקטין את זמן הריצה.
בהינתן אלגוריתם אנחנו רוצים לפרק אותו לתתי משימות T1…Tn כל תת משימה כזאת חייבת להתבצע באופן רציף על מעבד אחד.
נגדיר DAG , גרף תלויות על T1…Tn כאשר יש חץ מ Ti ל Tj בגרף אם Ti מתבצע בהכרח ליפני Tj (גרף מכוון).
גרף התלויות DAG הוא גם יחס סדר חלקי מכיוון שהוא רפלקסיבי, טרנזיטיבי ואנטי-סימטרי .
מרוץ קדימויות (race condition)מצב שיכול להיווצר כאשר 2 מעבדים צריכים לגשת לאותו תא בזיכרון באותו זמן.
זה יכול ליצור באגים שמאוד קשה למצוא ע"י testing כי הם יכולים לקרות רק לעיתים מאוד נדירות.
בנוסף אם יש שתי תתי משימות Ti ו Tj שאין ביניהם חץ בגרף התלויות והם נוגעים באותו תא בזיכרון אז תוצאת האלגוריתם עלול להיות תלוי בסדר הביצוע Ti ו Tj .
זה גם יכול ליצור באג, כי תוצאת האלגוריתם לא דטרמניסטית.
לחברה כמו Mobileye, אין מקום לטעויות או שגיאה מינימלית, שכן החברה מפתחת מערכות עזר לנהגים שצריכים לקבל המון מידע, בו זמנית, לבצע אנליזה, ולקבל החלטות מהירות ונכונות.
קל וחומר שנושא זה הינו עניין בטיחותי מוחלט ויכול בקלות להשפיע על הסביבה שלו.
לכן לא ניתן להתעלם מבעיות כאלה כמו שלפעמים מפתחים עושים.

תיאור הפתרון

רוצים אלגוריתם שמקבל כקלט את גרף התלויות ורשימה של תאים בזיכרון עבור כל תת משימה
האלגוריתם מוודא שאם אין חץ בין Ti ו Tjאז הם לא נוגעים באותו תא בזיכרון
זה יבטיח שלא יהיו באגים של מרוץ קדימויות או חוסר דטרמיניזם
נניח שיש n משימות ושכל משימה נוגעת לכל היותר k תאים בזיכרון אלגוריתם נאיבי לוקח O(n^2k) \
בשימוש שלנו באלגוריתם של Mobileye k,n ~~O(10^4) ואז n^2k~~ O(10^12) זמן הריצה של
האלגוריתם הנאיבי גדול מידי.
אלגוריתם אלטרנטיבי:

a. בונים את הפרמוטציות P1…Pk שמכבדות את גרף התלויות בונים רשימה כך שלכל i, j יהיה Ti, Tj
b. עבור כל פרמוטציה Pj , מניחים שזה הסדר שבחר המתזמן, ועוברים על המשימות בסדר Pj עבור כל תא m בזיכרון , עוקבים אחרי הסדר שבו תת המשימות נוגעות.
c. מוצאים שלכל תא m בזיכרון כל הסדרים שקיבלנו בשלב b הם זהים.

זמן הריצה של האלגוריתם O(c + t*n*k + a)

לפי החישוב, השאיפה היא ש t יהיה יותר קטן מ n (t<n) וש- a יתבצע ביעילות ואז נקבל שיפור משמעותי ע"פ האלגוריתם הנאיבי.
מציאת אוסף פרמוטציות מינימלי שמקיים את התנאי הפרויקט שלנו מתמקד בביצוע שלב a ביעילות ובהבטחת t קטן ככל האפשר.
באופן כללי מציאת אוסף פרמוטציות מינימלי שמקיים את התנאי נקרא בעיית ה poset dimensions.
אין בספרות האקדמית קישור ב poset dimension לחישוב מקבילי.
בנושא של חישוב מקבילי מדברים הרבה על מערכות של locks שמבטיחות שאין race condition. אך לא מבטיחות ביצוע דטרמיניסטי או כלים אחרים שעוקבים אחרי סדר ביצוע המשימות תוך כדי זמן ריצה dynamic race detection tools
בספרות המתמטית יש הרבה תוצאות של אלגוריתמים לחישוב ה poset dimension
ידוע שזאת בעיה NP קשה וגם לקרב אותה באופן סדיר זאת בעיה NP קשה.
יש מצבים מועטים שיש אלגוריתם פולינומי עבור הבעיה.

על הלקוח

הפרויקט הוא תעשייתי בשיתוף עם חברת Mobileye .
כידוע Mobileye מפתחת מערכות עזר לנהגים והמטרה שעומדת לנגד עיניהם הוא לפתח מערכות אוטונומיות.
במערכות האלה יש הרבה אלגוריתמים מורכבים שמשתמשים בתכנות מקבילי.
כמו שנאמר, באגים מסוג כזה יכולים לסכן חיים ואין מקום לטעויות במוצר ש Mobileye מפתחת.
לכן, אנחנו לא יכולים להתעלם משגיאות כמו חוסר דטרמיניסטיות או race condition אנחנו צריכים לאתר אותם.
כל אלגוריתם שצריך למקבל אותו יכול להשתמש באלגוריתם שאנחנו מפתחים.

סקירת ספרות

בתעשייה הסיטואציה הזאת לא כל כך נפוצה ובדרך כלל מטפלים בה בדרכים הבאות:

- מנעולים שמונעים race condition אבל יכולים ליצר חוסר דטרמניסטיות
- להריץ Testing ולבדוק אם קיים race condition, לרוב לא נראה את זה ולרוב בתעשייה זה לא חשוב שזה קיים ולכן הם יתעלמו מזה.
- מנהל, כלי תוכנה שמלווה ועוקב אחרי האלגוריתם מבחוץ אחד בלתי תלוי בשני ולכן זה מאט מאוד.