Name : David Meriin

ID: 304472038

Name: Yossef Yakobi

ID: 301752267

# Report

1. עבור טסט 1 קיבלנו את התוצאות

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Serial | TAS | Beckoff | CLH | MCS |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Avrage throughput |  | 148277.3 | 39199 | 37968.67 | 12631.33333 | 24513.66667 |
| speedUp |  |  | 0.264363 | 0.256065 | 0.085187217 | 0.165323088 |

התוצאות הללו הגיוניות כיוון שבטסט זה רץ רק חוט אחד בכל מקרה והשימוש במנעולים מסובכים גורם לביצועים הרבה פחות טובים כי מתבזבז זמן רב בטיפול במנגנוני הנעילה והשיחרור של המנעולים דבר שגורם לתפוקה הרבה יותר נמוכה מאשר לריצה הסדרתית שבה הזמן הזה מנוצל לביצוע הפעולה הרצויה במקרה זה הגדלת המונה.  
ניתן גם לראות שככל שמנגנון הנעילה מסובך יותר ב CLH וב MCS התפוקה שלהם הכי קטנה לעומת מנעולים פשוטים יותר כמו TAS ו BECKOFF שלהם ביצועים יותר טובים.

1. עבור טסט 2 קיבלנו את התוצאות

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Serial | TAS | Beckoff | CLH | MCS | n |
| Throughput | 148277.3 | 39299.67 | 39192.33 | 12722 | 24549.33 | 1 |
| SpeedUp | 1 | 0.265042 | 0.264318 | 0.085799 | 0.165564 | 1 |
|  |  | 548.6667 | 22603.33 | 1244.333 | 1456 | 8 |
|  |  | 0.0037 | 0.15244 | 0.008392 | 0.009819 | 8 |
|  |  | 123.6667 | 9859.667 | 1226.667 | 1097 | 64 |
|  |  | 0.000834 | 0.066495 | 0.008273 | 0.007398 | 64 |

|  |
| --- |
| אנו רואים כי עבור חוט אחד , המימוש הכי פשוט - TAS הוא גם הכי יעיל ( ביחד עם Backoff), עובדה שמצביעה על כך כי המימושים המסובכים יותר יעילים במספר חוטים גדול. נצפה גם לראות במימושים המסובכים יותר של CLH ו MCS שמנהלים תור שיהיה יותר הגינות בין החוטים. |
| עבור 8,64 חוטים אנו רואים כי היעילות של MCS,CLH קרובה בין המימושים. בנוסף אנו רואים כי במימושים אלו אין ירידה משמעותית בביצועים בין 8 ל64 חוטים. עבור 8 ו64 חוטים הביצועים של TAS הכי נמוכים , מה שמצביע לנו כי המימוש הפשוט של TAS טוב לשימוש רק במספר קטן של חוטים. |
| כלומר אנו מסיקים כי כאשר יש מנעול אחד שפונים אילו הרבה חוטים, כדאי לבחור במימושים המורכבים יותר. בנוסף, הרנדומיזציה באלגוריתם Backoff נראת כפתרון הכי יעיל במקרה של מנעול אחד והרבה חוטים. |

|  |  |
| --- | --- |
| BECKOFF | |
| Inc/ms | deviation |
| 11589 | 297189.53 |
| 13940 | 324153.3 |
| 14665 | 286809.2 |
| 15205 | 324723.4 |
| 16921 | 259793.3 |
| 15790 | 352250.1 |
| 12731 | 302717.5 |

|  |  |
| --- | --- |
| CLH | |
| Inc/ms | deviation |
| 1153 | 116.568 |
| 1118 | 116.241 |
| 892 | 658.612 |
| 892 | 149.205 |
| 1196 | 159.962 |
| 1145 | 133.085 |
| 1054 | 127.265 |

|  |  |
| --- | --- |
| MCS | |
| Inc/ms | deviation |
| 938 | 104.032 |
| 1097 | 93.751 |
| 1027 | 80.041 |
| 958 | 118.060 |
| 951 | 96.403 |
| 1027 | 94.632 |
| 1020 | 92.786 |

1. התוצאות שקיבלנו עבור טסט 3 :

|  |  |
| --- | --- |
| TAS | |
| Inc/ms | deviation |
| 237 | 8542.900 |
| 252 | 8299.754 |
| 237 | 7631.034 |
| 233 | 8641.944 |
| 252 | 6806.455 |
| 237 | 7828.278 |
| 242 | 8491.234 |

כיוון שסטיית התקן של מנעול Beckoff גדולה מאוד ללעומת סטיית התקן של שאר המנעולים וגם של TAS לעומת CLH ו MCS נציג את התוצאות בכמה גרפים כדי לקבל זום יותר גדול ולראות יותר את פיזור הנקודות בגרף :  
גרף שמציג את כל המנעולים ניתן לראות בו את פיזור התוצאות של Beckoff בצורה ברורה:

גרף שמציג את המנעולים ללא Beckoff ניתן לראות בו את פיזור התוצאות עבור מנעול TAS בצורה ברורה:

גרף שמציג את המנעולים CLH ו MCS בלבד וניתן לראות את פיזור התוצאות שלהם בצורה ברורה :

לכל מנעול ביצענו 7 מדידות וראינו סוג של התכנסות בתוצאות של מספר הגדלות המונה למילי שניות והתכנסות עבור הסטיית תקן.  
התוצאות תאמו את ההנחה שלנו שבמספר חוטים גדול יותר המימושים היותר מסובכים כגון CLH MCS ו Beckoff יעילים יותר מאשר המימוש הפשוט יותר של TAS.  
בנוסף המימושים המסובכים יותר של CLH ו MCS שומרים על הגינות בהרצה של החוטים כמו שלמדנו ולכן הסטיית תקן בהרבה יותר קטנה במימושים של CLH ו MCS מאשר במימושים של Beckoff ו TAS שבהם מי שתופס ראשון תופס וחוטים מסויימים יכולים כמעט אף פעם לא להצליח לנעול את המנעול ולכן בהם הסטיית תקן מאוד גדולה.

1. עבור טסט 1 בPacket Tests, קיבלנו :

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| w | S | Lock | throughput | | SpeedUp |  | **Data For Graph** | |  |
| 25 | LockFree | Tas | 1456 |  | 0.629121 |  | **SpeedUp** | **W** | **Lock** |
| 25 | LockFree | BackOff | 1347 |  | 0.713437 |  | 0.629121 | 25 | TAS |
| 25 | LockFree | CLH | 1079 |  | 0.969416 |  | 0.754902 | 200 | TAS |
| 25 | LockFree | MCS | 1489 |  | 0.684352 |  | 0.936508 | 800 | TAS |
| 25 | HomeQueue | Tas | 916 |  |  |  |  |  |  |
| 25 | HomeQueue | BackOff | 961 |  |  |  | 0.713437 | 25 | BackOff |
| 25 | HomeQueue | CLH | 1046 |  |  |  | 0.941889 | 200 | BackOff |
| 25 | HomeQueue | MCS | 1019 |  |  |  | 0.918812 | 800 | BackOff |
| 200 | LockFree | Tas | 1326 |  | 0.754902 |  |  |  |  |
| 200 | LockFree | BackOff | 1239 |  | 0.941889 |  | 0.969416 | 25 | CLH |
| 200 | LockFree | CLH | 1224 |  | 0.680556 |  | 0.680556 | 200 | CLH |
| 200 | LockFree | MCS | 1309 |  | 0.665393 |  | 0.750503 | 800 | CLH |
| 200 | HomeQueue | Tas | 1001 |  |  |  |  |  |  |
| 200 | HomeQueue | BackOff | 1167 |  |  |  | 0.684352 | 25 | MCS |
| 200 | HomeQueue | CLH | 833 |  |  |  | 0.665393 | 200 | MCS |
| 200 | HomeQueue | MCS | 871 |  |  |  | 0.926295 | 800 | MCS |
| 800 | LockFree | Tas | 504 |  | 0.936508 |  |  |  |  |
| 800 | LockFree | BackOff | 505 |  | 0.918812 |  |  |  |  |
| 800 | LockFree | CLH | 497 |  | 0.750503 |  |  |  |  |
| 800 | LockFree | MCS | 502 |  | 0.926295 |  |  |  |  |
| 800 | HomeQueue | Tas | 472 |  |  |  |  |  |  |
| 800 | HomeQueue | BackOff | 464 |  |  |  |  |  |  |
| 800 | HomeQueue | CLH | 373 |  |  |  |  |  |  |
| 800 | HomeQueue | MCS | 465 |  |  |  |  |  |  |

התוצאות בטסט זה תואמות את התובנות שלנו מהטסטים של המונה שכאשר יש חוט אחד שרץ ככל שיש יותר טיפול במנעולים יש פגיעה בתפוקה והביצועים פחות טובים ורואים זאת כאן בעובדה שהתוצאות של כל המנעולים בשיטת HomeQueue שמשתמש בנעילה של התור פחות טובים מאשר בשיטה של LockFree שבו אין נעילה של התור.

בהשוואה לתוצאות של Worker Rate מהתרגיל הקודם אנחנו רואים שיש ירידה כאן לעומת התרגיל הקודם וזאת כתוצאה מהמימושים היותר מסובכים של המנעולים שמגיעים עם overhead של טיפול בנעילה ושיחרור המנעול. יש ירידה גדולה יותר עבור W=25 (מ 4850 לפחות מ 1500) שכן הזמן לטיפול בחבילה הוא קטן יותר ולכן יש יותר נעילות ושיחרורים של המנעול דבר שגורם להרעה בביצועים.

1. עבור טסט 2 ב Packet Tests, קיבלנו :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Serial** |  |  |  |  |
|  | n | W | throughput | |  |
|  | 1 | 1000 | 445 |  |  |
|  | 4 | 1000 | 449 |  |  |
|  | 10 | 1000 | 450 |  |  |
|  |  |  |  |  |  |
|  | 1 | 6000 | 78 |  |  |
|  | 4 | 6000 | 78 |  |  |
|  | 10 | 6000 | 78 |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| **Parallel** |  |  |  |  |  |
| n | W | throughput | speedup | Lock | Strategy |
| 1 | 1000 | 377 | 0.847191 | TAS | LockFree |
| 1 | 1000 | 340 | 0.764045 | TAS | RandomQueue |
| 1 | 1000 | 366 | 0.822472 | TAS | LastQueue |
| 1 | 1000 | 410 | 0.921348 | BackOff | LockFree |
| 1 | 1000 | 349 | 0.78427 | BackOff | RandomQueue |
| 1 | 1000 | 399 | 0.896629 | BackOff | LastQueue |
| 4 | 1000 | 948 | 2.111359 | TAS | LockFree |
| 4 | 1000 | 891 | 1.98441 | TAS | RandomQueue |
| 4 | 1000 | 901 | 2.006682 | TAS | LastQueue |
| 4 | 1000 | 912 | 2.03118 | BackOff | LockFree |
| 4 | 1000 | 844 | 1.879733 | BackOff | RandomQueue |
| 4 | 1000 | 864 | 1.924276 | BackOff | LastQueue |
| 10 | 1000 | 1242 | 2.76 | TAS | LockFree |
| 10 | 1000 | 655 | 1.455556 | TAS | RandomQueue |
| 10 | 1000 | 396 | 0.88 | TAS | LastQueue |
| 10 | 1000 | 1730 | 3.844444 | BackOff | LockFree |
| 10 | 1000 | 1232 | 2.737778 | BackOff | RandomQueue |
| 10 | 1000 | 1207 | 2.682222 | BackOff | LastQueue |
|  |  |  |  |  |  |
| 1 | 6000 | 77 | 0.987179 | TAS | LockFree |
| 1 | 6000 | 74 | 0.948718 | TAS | RandomQueue |
| 1 | 6000 | 70 | 0.897436 | TAS | LastQueue |
| 1 | 6000 | 77 | 0.987179 | BackOff | LockFree |
| 1 | 6000 | 76 | 0.974359 | BackOff | RandomQueue |
| 1 | 6000 | 77 | 0.987179 | BackOff | LastQueue |
| 4 | 6000 | 175 | 2.24359 | TAS | LockFree |
| 4 | 6000 | 296 | 3.794872 | TAS | RandomQueue |
| 4 | 6000 | 281 | 3.602564 | TAS | LastQueue |
| 4 | 6000 | 176 | 2.25641 | BackOff | LockFree |
| 4 | 6000 | 288 | 3.692308 | BackOff | RandomQueue |
| 4 | 6000 | 270 | 3.461538 | BackOff | LastQueue |
| 10 | 6000 | 436 | 5.589744 | TAS | LockFree |
| 10 | 6000 | 703 | 9.012821 | TAS | RandomQueue |
| 10 | 6000 | 656 | 8.410256 | TAS | LastQueue |
| 10 | 6000 | 433 | 5.551282 | BackOff | LockFree |
| 10 | 6000 | 696 | 8.923077 | BackOff | RandomQueue |
| 10 | 6000 | 605 | 7.75641 | BackOff | LastQueue |

**W=1000**

**W=6000**

באופן כללי עבור w=1000 יש יותר "מאבק" על מנעולים לעומת w=6000, זאת מכיוון שכאשר w=6000 החוט משקיע יותר זמן ב"עבודה" ובעצם ניגש פחות פעמים לנסות להוציא איבר מהתור. כלומר כל חוט מנסה לנעול פחות פעמים. אנו רואים שכאשר w=1000 נראה שלפתרון הlock-free הביצועים הטובים ביותר בכל החוטים , זאת מכיוון שכל שאר החוטים מבזבזים משאבים על נעילת התור לעצמם, וכן הרצת הלוגיקה של התור כרוכה בתקורה מסוימת גם כן. כפי שראינו בסעיף 2 לTAS אין ביצועים טובים כאשר הרבה חוטים מנסים לנעול את אותו המנעול. מכיוון שיש "מעט" עבודה, החוטים ינסו לנעול כל הזמן מנעולים וחלק גדול הזמן לא יצליחו. כאשר משתמשים באסטרטגיית רנדומית בבחירת תור מקבלים ביצועים יותר טוב מאלו של Last-Queue. זאת מכיוון שבLast-Queue מנסים לנעול הרבה מנעולים ולא מנעול אחד ספציפי , זה גורם לתחרות עוד יותר גדולה בין החוטים על המנעולים מאשר בRandom-Queue.

לעומת זאת , כאשר w=6000 לLock-Free הביצועים הכי נמוכים, זאת מכיוון שכל התורים מקבלים כמות עבודה שווה , אך חלק עובדים הרבה יותר מאחרים ולא עומדים בעומס . מצב זה גורם לכך שהdispatcher יכול להתקע זמן מסוים על לנסות ולהכניס לתור, כי הוא מנסה להכניס פקטה לתור כל עוד זה מלא. עבור אסטרטגיית הrandom queue אנו רואים שאלו הביצועים הטובים ביותר . גם עבור מנעול TAS וגם Backoff. תוצאה זו מצביעה על כך שעבור בקרת עומסים ניתן להשתמש באלגוריתם פשוט כמו רנדומיזציה . אלגוריתם זה תורם בכך שכל חוט בוחר תור ממנו הוא יוציא פקטה , בכל פעם שהוא מוכן לעבוד על פקטה. גישה זו תגרום לכך שתורים יתרוקנו בממוצע יותר מאשר בLock-Free , כי כאשר תור פנוי לdequeue יש סיכוי שחוט כלשהו יבחר בו ויקח איבר ממנו . זאת בניגוד לLock-Free , שם התור יאלץ לחכות עד שהחוט יסיים לעבוד על הפקטה ( שכנראה העבודה "ארוכה" ביחס לעבודה שהחוט היה עובד בw=1000).

Last-Queue מניב ביצועים פחות טובים מRandomQueue מכיוון שיש יותר מאבקים על מנעולים , גם בזמן החיפוש של המנעול הזמין וגם כאשר מוצאים תור פנוי לנעילה. יכול להיות ששני חוטים מצאו תור פנוי לנעילה, וכעת שניהם רצים עליו וכל פעם כל חוט מחכה שהחוט השני ישחרר את המנעול על אותו התור.

1. עבור טסט 3 ב Packet Tests, קיבלנו :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | w | S | Lock | throughput | n | Speedup |
| Serial | 1000 | - | - | 444 | 1 |  |
| Serial | 1000 | - | - | 450 | 8 |  |
| Serial | 1000 | - | - | 450 | 64 |  |
| Serial | 6000 | - | - | 78 | 1 |  |
| Serial | 6000 | - | - | 78 | 8 |  |
| Serial | 6000 | - | - | 79 | 64 |  |
| Parallel | 1000 | Lock-Free | 0 | 360 | 1 | 0.810811 |
| Parallel | 1000 | Random-Queue | 0 | 350 | 1 | 0.788288 |
| Parallel | 1000 | Last-Queue | 0 | 389 | 1 | 0.876126 |
| Parallel | 1000 | Lock-Free | 1 | 406 | 1 | 0.914414 |
| Parallel | 1000 | Random-Queue | 1 | 397 | 1 | 0.894144 |
| Parallel | 1000 | Last-Queue | 1 | 361 | 1 | 0.813063 |
| Parallel | 1000 | Lock-Free | 0 | 1409 | 8 | 3.131111 |
| Parallel | 1000 | Random-Queue | 0 | 626 | 8 | 1.391111 |
| Parallel | 1000 | Last-Queue | 0 | 491 | 8 | 1.091111 |
| Parallel | 1000 | Lock-Free | 1 | 1479 | 8 | 3.286667 |
| Parallel | 1000 | Random-Queue | 1 | 1252 | 8 | 2.782222 |
| Parallel | 1000 | Last-Queue | 1 | 1207 | 8 | 2.682222 |
| Parallel | 1000 | Lock-Free | 0 | 209 | 64 | 0.464444 |
| Parallel | 1000 | Random-Queue | 0 | 429 | 64 | 0.953333 |
| Parallel | 1000 | Last-Queue | 0 | 115 | 64 | 0.255556 |
| Parallel | 1000 | Lock-Free | 1 | 501 | 64 | 1.113333 |
| Parallel | 1000 | Random-Queue | 1 | 471 | 64 | 1.046667 |
| Parallel | 1000 | Last-Queue | 1 | 51 | 64 | 0.113333 |
| Parallel | 6000 | Lock-Free | 0 | 72 | 1 | 0.923077 |
| Parallel | 6000 | Random-Queue | 0 | 73 | 1 | 0.935897 |
| Parallel | 6000 | Last-Queue | 0 | 75 | 1 | 0.961538 |
| Parallel | 6000 | Lock-Free | 1 | 77 | 1 | 0.987179 |
| Parallel | 6000 | Random-Queue | 1 | 75 | 1 | 0.961538 |
| Parallel | 6000 | Last-Queue | 1 | 77 | 1 | 0.987179 |
| Parallel | 6000 | Lock-Free | 0 | 335 | 8 | 4.294872 |
| Parallel | 6000 | Random-Queue | 0 | 580 | 8 | 7.435897 |
| Parallel | 6000 | Last-Queue | 0 | 533 | 8 | 6.833333 |
| Parallel | 6000 | Lock-Free | 1 | 336 | 8 | 4.307692 |
| Parallel | 6000 | Random-Queue | 1 | 552 | 8 | 7.076923 |
| Parallel | 6000 | Last-Queue | 1 | 399 | 8 | 5.115385 |
| Parallel | 6000 | Lock-Free | 0 | 420 | 64 | 5.316456 |
| Parallel | 6000 | Random-Queue | 0 | 362 | 64 | 4.582278 |
| Parallel | 6000 | Last-Queue | 0 | 200 | 64 | 2.531646 |
| Parallel | 6000 | Lock-Free | 1 | 537 | 64 | 6.797468 |
| Parallel | 6000 | Random-Queue | 1 | 358 | 64 | 4.531646 |
| Parallel | 6000 | Last-Queue | 1 | 227 | 64 | 2.873418 |

**W=1000**

**W=6000**

גם כאן אנו רואים כי עבור w=1000 Lock-Free הוא פתרון טוב (עד מספר חוטים 8), שכן כמות העבודה נמוכה והפתרונות עם מנעולים בעיקר נאבקים על לתפוס מנעול של תור. ב64 חוטים אנו רואים ירידה משמעותית מכיוון שהdispatcher לא מספיק למלא את התורים בהרבה מן החוטים , דבר שגורם לחוטים לזרוק הרבה שגיאות של תור ריק , פעולות שידועות להיות בזבזניות . פתרון הbackoff\_Random-queue הוא הכי יעיל עבור 64 חוטים שכן הרנדומיזיציה מקטינה את הסיכוי לבחור בתור ריק כאשר אנחנו רוצים להוציא ממנו פקטה , וכן גם אם התור ריק החוט ינסה לבחור שוב תור אחר ולא לנסות להוציא מאותו התור. פתרון הbackoff יותר טוב מזה של הTAS שכן ההמתנה לנעילה נוספת מפחיתה ניסיונות חסרי טעם של נעילת המנעול.

עבור w=6000 , עד 8 חוטים אנו רואים תוצאות דומות ביחסים בין השיטות לניסוי הקודם.

כאן מכיוון שיש לכל חוט יותר עבודה , וכן ישנם פחות דרישות לנעילות והתנגשויות בין החוטים השיפור לעומת הריצה הסדרתית גדול יותר מכאשר w=1000.

ב64 חוטים Lock-Free הוא בעל הביצועים הטובים ביותר. זאת מכיוון שתחרות על מנעול עם מספר רב של חוטים מאטה את הריצה לעומת מספר קטן של חוטים. אמנם כאשר w=1000 עבור 64 חוטים Tas\_Random-Queue ו- Backoff\_Random-queue הראו ביצועים טובים יותר , אך בw=6000 הצורך במנעולים הוא קטן יותר שכן כל חוט עובד יותר זמן על הפקטה אותה הוציא מהתור. מכיוון שיש צורך במנועולים קטן יותר ותקורה גדולה עבור פתרון מאבקי מנעולים , לאורך זמן הפתרון של Lock-Free עדיף.

נסיק כי נעדיף פתרון עם 8 חוטים , שם הביצועים הכי טובים. זאת מכיוון שכמו שציינו הdispatcher לא מספיק מהיר בכדי להשאיר את כל החוטים "עסוקים" מספיק לאורך הריצה.