

אוניברסיטת בן-גוריון בנגב, המחלקה למדעי המחשב

בוחן אמצע במערכות הפעלה (25 במאי 2011)

מרצים: דני הנדלר ורועי זיוון; מתרגלים: אלון גרובשטיין, איליה מירסקי, אמיר מנצ'ל ודולב פומרנץ

משך המבחן: שלוש שעות
חומר עזר: אסור
ענו על כל השאלות: סה"כ 100 נקודות.

1. (30 נקודות) תזמון (scheduling)

א. (5 נ') . תארו בקצרה אך באופן מדויק וברור את אופן פעולתם של שני האלגוריתמים הבאים לתזמון: Round robin ו-Guaranteed scheduling. בפרט, תארו איזו "הגינות" (fairness) בתזמון מבטיחים שני האלגוריתמים.

ב. (5 נ') . נניח כי במערכת קיימים מספר חוטי קרנל (kernel threads) שהם CPU-bound ומספר חוטי קרנל שהם I/O-bound. איזה מבין שני האלגוריתמים מתאים יותר עבור מערכת כזו? נמקו בקיצור אך באופן מדויק וברור.

ג. (8 נ') . כאשר עסקנו במוניטורים, תיארו את המימושים הבאים. 1) מימוש מוניטור בסמנטיקה של Hoare (בו מובטח כי החוט המקבל את הסיגנל הוא הבא לרוץ במוניטור), ו-2) מוניטור של Java, בו לא מתקיימת הבטחה שכזו ובכל פעם החוטים (threads) המעוניינים להיכנס אליו ונמצאים במצב ready מתחרים על הכניסה.

נניח כי במערכת ישנם מספר חוטים המשתמשים במוניטור. האם הבחירה בין שני האלגוריתמים הנ"ל לתזמון תשפיע יותר על מימוש המשתמש במוניטור מטיפוס Hoare או על מוניטור של Java? נמקו תשובתכם בקיצור אך באופן מדויק וברור.

ד. (12 נ') . נניח כי במערכת בעלת מעבד (CPU) יחיד ישנה קבוצת חוטים אשר כל אחד מהם רץ תמיד משך זמן T לפני שהוא עובר להמתנה ל-I/O וכי context switch אורך זמן S. נניח גם כי במערכת מופעל אלגוריתם תזמון round robin עם time-quantum באורך Q. ניתן להניח כי תמיד ישנם במערכת חוטים המוכנים לרוץ (אינם ממתנינים ל-I/O). חשבו את ניצולת ה-CPU במערכת כפונקציה של פרמטרים אלו עבור המקרים הבאים:

- (i) $Q > T$
- (ii) $S < Q < T$
- (iii) Q שואף לאפס.

2. (35 נקודות) תהליכים וחוטים

א. (15 נק') . מהם הפלטים האפשריים של התוכנית הבאה, על המסך ובקובץ a.txt? עבור כל פלט, ציינו האם הוא מודפס על ידי התהליך האב, תהליך בן או חוט. הניחו כי התוכנית רצה על מערכת עם מעבד יחיד, וכי פתיחת הקובץ מצליחה. הניחו כי בכל תהליך המזהים של החוטים נקבעים בסדר עולה, החל מ-1. בנוסף, הניחו כי כאשר תכנית מרובת חוטים מבצעת פעולת fork רק החוט אשר ביצע את הקריאה משתכפל. הניחו גם כי כל פעולת הדפסה היא אטומית (כלומר, היא לא תופרע באמצע על ידי context switch).

```
int x = -1;

void foo() {
    printf("x=%d\n", x);
    x = pthread_self(); // get thread ID
}

int main() {
    int fd;
    pthread_t tid[2];

    printf("STARTED\n");
    fd = dup(STDOUT_FILENO);
    close(STDOUT_FILENO);
    open("a.txt", O_WRONLY | O_CREAT | O_TRUNC);

    pthread_create(&tid[0], NULL, foo, NULL);
    fork();
    pthread_create(&tid[1], NULL, foo, NULL);
    sleep(10); // sleep for 10 seconds
    printf("after fork x=%d\n", x);

    close(STDOUT_FILENO);
    dup(fd);
    printf("FINISHED\n");

    return 0;
}
```

ב. (10 נק'). הניחו כי ברשותכם שרת אינטרנט המספק בקשות משתמשים. לשרת מבנה נתונים בו הוא שומר ערכים שהובאו עבור בקשות שהתבצעו לאחרונה. בכל פעם שמתקבלת בקשה, מתבצעת פעולת חיפוש מהירה במבנה הנתונים. במידה והמידע הנדרש אינו מצוי במבנה הנתונים, מתבצע תהליך אחזור מתוך דיסק (I/O). הניחו כי לשרת מעבד יחיד. כמו כן הניחו כי לא ניתן לבצע שתי פעולות I/O לדיסק במקביל. נסמן ב-h את ההסתברות למציאת המידע הנדרש במבנה הנתונים, ב-c את הזמן הנדרש לאחזור מידע ממבנה הנתונים וב-t את משך הזמן הנדרש לאחזור מידע מהדיסק (ברור כי $t > c$). בפרט, כאשר המידע הנדרש לא נמצא במבנה הנתונים, משך האיחזור הכולל הוא $t+c$ (ראשית יש לקרוא את המידע מן הדיסק למבנה הנתונים ואז לקרוא ממבנה הנתונים). נתון כי שתי בקשות בלתי תלויות זו בזו, כל אחת מהן למידע שונה, מגיעות בו זמנית למערכת.

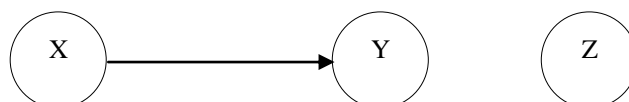
i. מהו משך הזמן הצפוי עד להשלמת הבקשות במערכת מבוססת חוטי משתמש (user space threads)?

ii. מהו משך הזמן הצפוי עד להשלמת הבקשות במערכת מבוססת חוטי קרנל (kernel threads)?

ג. (10 נק'). נתון קטע הקוד הבא:

```
void sigchld_handler(int s) {
    printf("S");
}
int main(){
    signal(SIGCHLD, sigchld_handler);
    signal_block(SIGCHLD);
    if (fork() != 0) {
        printf("A");
        signal_unblock(SIGCHLD);
        printf("B");
        wait ();
        printf("C");
    } else {
        printf("D");
    }
}
```

ידוע כי הפקודות `signal_block` וכן `signal_unblock` חוסמות ומשחררות חסימה לסיגנלים. שרטטו גרף מכוון המתאר את הפלטים האפשריים לקוד זה. כל צומת בגרף תסמל הדפסה וכל קשת מכוונת תייצג יחס סדר מתחייב בין הדפסות. לדוגמא, אם עפ"י קוד מסוים ידוע כי יודפסו "X", "Y" ו-"Z" וכי ההדפסה של "X" תופיע בהכרח לפני ההדפסה של "Y" (אך "Z" יכול להופיע לפני או אחרי כל אחת מן ההדפסות האחרות), יתקבל הגרף הבא:



3. (35 נקודות) סינכרוניזציה – מניעה הדדית

א. (5 נ') . בכיתה הגדרנו את התנאים חופש מקיפאון (deadlock freedom) וחופש מהרעבה (starvation freedom) עבור אלגוריתמים למניעה הדדית. כתבו את ההגדרות המדוייקות של תנאים אלו.

ב. (10 נ') . בכיתה הגדרנו תנאי הוגנות (fairness) עבור בעיית המניעה ההדדית הקרוי (FIFO) first-in-first-out.

i. כתבו את ההגדרה המדוייקת של תנאי זה.

ii. האלגוריתם למניעה הדדית של פטרסון לשני תהליכים מובא בסוף שאלה זו. האם האלגוריתם מקיים תנאי FIFO? נמקו בקצרה אך במדוייק.

ג. (10 נ') . בשאלה זו עליכם לממש אלגוריתם למניעה הדדית עבור שלושה תהליכים p, q, r . על האלגוריתם לקיים מניעה הדדית וחופש מקיפאון. בנוסף, עליו לקיים את הדרישה הבאה המבטיחה עדיפות לתהליך p : אם p מתחיל להמתין למנעול טרם שהתהליך המחזיק במנעול החל לבצע את קוד היציאה (exit section) שלו, אזי p הוא התהליך הבא שיכנס לקטע הקריטי.

לצורך המימוש מותר להשתמש אך ורק במשתנים התומכים בקריאות ובכתיבות. אין להשתמש בסוגים אחרים של משתנים (בפרט, אין להשתמש בסמפורים או בפעולות כגון test-and-set). ניתן גם להשתמש באלגוריתם של פטרסון למניעה הדדית עבור שני תהליכים כאבן בניין. הסבירו בקצרה את האלגוריתם שכתבתם ונמקו בקצרה מדוע הוא מקיים את שנדרש (אין צורך בהוכחה פורמלית).

ד. (10 נ') . שלומציונה טוענת כי אלגוריתם המקיים את הדרישות של סעיף ג. בהכרח אינו מקיים חופש מהרעבה (starvation freedom). האם טענתה צודקת? נמקו תשובתכם בקצרה, אך באופן מדוייק וברור.

להלן האלגוריתם של פטרסון למניעה הדדית עבור שני תהליכים. שני הדגלים $b[0]$ ו- $b[1]$ מאותחלים לערך false. המשתנה turn מאותחל לערך 0 או 1 (אין זה משנה מבחינת נכונות האלגוריתם).

```
Algorithm for  $p_0$ 

b[0]=true
turn=0
await (b[1]=false OR turn=1)
CS
b[0]=false
```

```
Algorithm for  $p_1$ 

b[1]=true
turn=1
await (b[0]=false OR turn=0)
CS
b[1]=false
```

בהצלחה!