Homework Dry 2

**שאלה 1:**

בכיתה ובתרגיל הבית ראיתם את אלגוריתם הזימון של לינוקס גרסה 2.6.

1. באילו אירועים יכולה להשתנות העדיפות הדינאמית של תהליך?

* כאשר תהליך מסיים את ה Time slice הנוכחי שלו ב scheduler\_tick()
* כאשר חוזרים מהמתנה בפונקציה wakeup\_process

1. מתי משתנה ה-Time slice של תהליך?

* כאשר עושים fork ה TS - מוקטן בחצי מה TS של האב .
* בכל פעימת שעון – קטן ב 1 ב scheduler\_tick.
* כאשר תהליך מסיים את ה TS שלו – מקבל TS חדש מלא .
* כשבן מת – היתרה חוזרת לאב.

כעת נגדיר אלגוריתם זימון חדש. האלגוריתם זהה לאלגוריתם המופיע בגרסה 2.6 של לינוקס, אותו למדתם בתרגולים, עם השינויים הבאים:

* נגדיר תהליך אינטראקטיבי שקיבל Time slice נוסף במהלך Epoch מסוים כתהליך **עתיר** למשך ה-Epoch הנ"ל. תהליך עתיר, נשאר כזה עד סוף ה-Epoch.
* מהרגע שתהליך הופך לעתיר ב-Epoch מסוים, עדיפותו הדינמית במהלך כל אותו ה-Epoch משתנה בצורה הבאה: בכל פעם שנגמר לו ה-Time slice, הערך המספרי של עדיפותו הדינמית גדל ב-1 והוא מקבל Time slice נוסף, עד שערך העדיפות הדינמית מגיע ל-139. במקרה זה, אם נגמר ה-Time slice פעם נוספת, התהליך מועבר לתור ה-expired.
* כאשר תהליך עתיר מועבר לתור ה-expired, הוא מפסיק להיות עתיר, ועדיפותו הדינמית מחושבת כמו באלגוריתם הזימון המקורי של לינוקס (2.6).
* במידה ותהליך עתיר יוצר בן, ה-Time slice של האב מתחלק בין האב והבן (מעוגל כלפי מטה). תהליך הבן במקרה זה גם כן יהיה עתיר למשך אותו ה-Epoch.
* בעת מתן Time slice נוסף, לא יבדק סף ההרעבה (כלומר, לא נעשה שימוש במאקרו EXPIRED\_STARVING(rq)).

הנח:

מספר התהליכים במערכת אינו מוגבל.

1. האם תיתכן הרעבה באלגוריתם הזימון החדש? הסבירו

לא. לא תיתכן הרעבה באלגוריתם הזימון החדש כי מספר חידושי ה TS שתהליך יכול לקבל הינו חסום וקבוע בכל Epoch – לכל היותר 40, כי חידוש ה TS תלוי בעדיפות של התהליך, ורואים שערך העדיפות הדינמית של תהליך הינו חסום . בנוסף, היות ומספר התהליכים הוא גם כן חסום ע"י 32K אז כל התהליכים ירוצו באותו Epoch ללא חשש להרעבה.

גם במקרה של יצירת המון בנים ע"י fork אין חשש מהרעבה נוכח העובדה ש ה TS מתחלק בין האב לבן (בעיגול כלפי מטה) , ולכן מספר הבנים שניתן ליצור עבור כל עדיפות הוא סופי, ולכן כולם יסיימו באיזשהו שלב ויאפשרו לתהליכים אמרים גם לרוץ.

1. אם נוסיף לאלגוריתם החדש את הבדיקה של המאקרו EXPIRED\_STARVING(rq) בכל פעם שמוסיפים Time Slice בדומה לגירסה שנלמדה בכיתה, האם תיתכן הרעבה? הסבירו.

לא, כי ממילא לא תיתכן הרעבה מאותה סיבה של הסעיף הקודם, והוספת בדיקה זו לא תשפיע על זה.

1. האם יתכן מצב שבו תהליך יקבל מספר גדול יותר של חידושי Time slice במהלך אותו ה-Epoch באלגוריתם החדש לעומת המקורי? הסבירו

כן. למשל המצב שבו תהליך עם עדיפות סטטית 120, באלגוריתם הישן, תהליך זה יכול להיחשב כאינטראקטיבי אחרי שהוא מסיים את ה TS שהוא קיבל בפעם ראשונה (דבר זה אפשרי כי עבור nice=0 ההחלטה האם תהליך מסוים חישובי או אנטרקטיבי היא החלטה דינמית של ה scheduler) אבל הוא לא ייחשב כך אחרי שהוא מסיים את ה TS שלו בפעם השנייה, ולכן הוא קיבל בעצם חידוש אחד של ה TS אותו ה-Epoch .

לעומת זאת, באלגוריתם החדש, תהליך זה יהפוך לתהליך עתיר אחרי שייחשב כאנטרקטיבי בפעם הראשונה, לכן הוא יקבל 19 חידושי TS במהלך אותו ה-Epoch לעומת פעם אחת לפי האלגוריתם הישן.

1. האם יתכן מצב שבו תהליך יקבל מספר גדול יותר של חידושי Time slice במהלך אותו ה-Epoch באלגוריתם המקורי לעומת החדש? הסבירו

כן. מספר חידושי ה TS במהלך אותו ה-Epoch הוא קבוע באלגוריתם החדש, ואילו באלגוריתם הישן הוא איננו. לדוגמא, עבור תהליך אינטראקטיבי עם עדיפות סטטית 120 שסיים את ה TS שלו בפעם הראשונה, לפי האלגוריתם החדש, הוא ייחשב אחר כך לעתיר, ויקבל 19 חידושי TS במהלך ה-Epoch הנוכחי.לעומת זאת, באלגוריתם הישן, אותו תהליך יוכל להיחשב כאנטרקטיבי כל פעם ה TS שלו נגמר, ולכן הTS שלו יכול להיות גדול מ 19 חידושי TS במהלך אותו ה-Epoch .

**שאלה 2:**

בשאלה זאת עליכם לסמן את התשובה/תשובות הנכונה/נכונות ולנמק היטב את בחירתכם. תשובה לא מנומקת תקבל ניקוד חלקי בלבד. שימו לב: בכל סעיף יכולות להיות כמה תשובות נכונות. סימון תשובות לא נכונות יגרע מהציון.

א. מדיניות תזמון FCFS (First Come First Served) יכולה להיות אטרקטיבית כי:

1. היא פשוטה למימוש
2. היא הוגנת כלפי כל התהליכים
3. היא מקטינה את הזמן ההמתנה הממוצע במערכת
4. היא מקטינה את סה"כ זמן ההמתנה במערכת
5. היא מקטינה את הזמן התגובה הממוצע במערכת
6. כי ניתן למימוש ע"י תור תהליכים המוכנים הוא FIFO.
7. יש הרעבה של תהליכים.

ג,ד,ה. אינה מקטינה את הזמנים האלו כי התהליכים יכולים להגיע בסדר כלשהו, והמדיניות הזו תטפל בהם בסדר הגעתם.

ב. מדיניות תזמון SRTF (Shortest Remaining Time First) יכולה להיות אטרקטיבית כי:

1. היא פשוטה למימוש
2. היא הוגנת כלפי כל התהליכים
3. היא מקטינה את הזמן ההמתנה הממוצע במערכת
4. היא מקטינה את סה"כ זמן ההמתנה במערכת
5. היא מקטינה את הזמן התגובה הממוצע במערכת

לפי ההרצאה מקטינה את זמן ההשהייה הממוצע, (זמן המתנה+זמן ריצה) אבל זמן הריצה של התהליכים ידוע מראש ולכן קבוע, לכן ממזער את זמן ההמתנה.

אם זמן תגובה = זמן ההשהייה, נכון מההרצאה.

ג. אילו משיטות התזמון הבאות עלולות לגרום לאפקט השיירה (convoy effect)?

1. FCFS
2. RR
3. SRTF
4. כל מדיניות זימון עם הפקעה (preemption)
5. כל מדיניות זימון בלי הפקעה (preemption)
6. אף תשובה אינה נכונה

באפקט השיירה יש תהליך ארוך עתיר חישובים שתופס את המעבד בעוד שתהליכים עתירי input/output מתבטלים. מצב זה יכול לקרות במדיניות שהן ללא הפקעה בלבד.

FCFS היא מדיניות ללא הפקעה, והמדיניות בסעיפים ב ו-ג הן עם הפקעה.

Did it all? Good work! Print and submit your assignment. ☺

**Good Luck,**

The course staff