TECHNION
Israel Institute of Technology

Computer Science Department

Operating Systems – 234123

Homework Exercise 2 – Dry

מגישים:

יאיר שחר 200431260 **syairsha@campus.technion.ac.il**

שירן סעדה 301731998 **Shiransaada@campus.technion.ac.il**

Teaching Assistant in charge:

Mohammed Dabbah

Assignment Subjects & Relevant Course material

Processes, Scheduler.

Recitations 1-4, Lectures 1-3

שאלה 1

שאלה זו עוסקת במדיניות זימון התהליכים של לינוקס כפי שנלמדה בתרגולים. לנוחיותכם מצורפים מספר macros המשמשים את זמן התהליכים

```
#define MAX_PRIO 140
#define MIN_TIMESLICE (10 * HZ / 1000)
#define MAX_TIMESLICE (300 * HZ / 1000)
#define TASK_TIMESLICE(p)
MIN_TIMESLICE + (MAX_TIMESLICE – MIN_TIMESLICE) * \
(MAX_PRIO – 1 – (p)>static_prio)/39
#define TASK_INTERACTIVE(p) \
((p)>prio <= (p)>static_prio – DELTA(p))
prio = static_prio – bonus
#define EXPIRED_STARVING(rq) \
((rq)>expired_timestamp && \ ((jiffies – (rq)>expired_timestamp)
>=STARVATION_LIMIT * ((rq)>nr_running + 1))
BONUS(p) = 25% × 40 × ( SleepAvg/MaxSleepAvg – 1/2)
DELTA(p) = 5 × 20 TaskNice(p) + 2
```

א. נניח כי תהליך A מסווג כחישובי על ידי אלגוריתם הזימון של לינוקס ובעל עדיפות סטטית x ותהליך B מסווג באינטראקטיבי על ידי אלגוריתם הזימון של לינוקס ובעל אותה עדיפות סטטית x .האם יתכן כי העדיפות הדינמית של A טובה יותר משל B?

לא יתכן שהעדיפות הדינמית של A תהיה טובה משל B, הוכחה:

נתון כי העדיפויות הסטטיות DELTA(A)=DELTA(B) <= nice(A)=nice(B) <= static_prio(A)=static_prio(b)=x

נסמן Delta(A)=Delta

A חישובי ולכן Bonus(A)<Delta

B אינטרקטיבי ולכן Bonus(B)>=Delta

נניח בשלילה של A עדיפות חזקה משל B ולכן prio(A)=x-bonus(a)<prio(B)=x-bonus(B)

ולכן Delta<=Bonus(B)<Bonus(A)

לכן Bonus(A)>Delta בסתירה לכך שA חישובי ולא אינטרקטיבי.

לכן עדיפות הדינמית של A לא יכולה להיות טובה יותר משל B.

ב. נניח כי תהליכים A ו B מסווגים כחישוביים על ידי אלגוריתם הזימון של לינוקס ובעלי עדיפות דינמית שווה, אבל עדיפות הסטטית של A טובה יותר (נמוכה יותר מספרית). איזה יתרון מקבל A על B?

מכיוון שStatic_prio(A)<Static_prio(B) נובע כי המאקרו TASK_TIMESLICE(A)>TASK_TIMESLICE(B) ולכן היתרון שיקבל A הוא שהוא יקבל TimeSlice גדול יותר משל B למרות שהם כרגע בעדיפות דינמית זהה

ג. נניח כי תהליכים A ו B מסווגים כאינטראקטיביים על ידי אלגוריתם הזימון של לינוקס ובעלי עדיפות דינמית שווה. אבל העדיפות הסטטית של A טובה יותר (נמוכה יותר מספרית). איזה יתרון מקבל A על B?

מכיוון שנתון כי Static_prio(A)<Static_prio(B) נובע כי Nice(A)<Nice(B) ולכן Delta(A)<Delta(B) ולכן יותר קל לתהליך A לקיים את האישיוייון Delta(A)<Bonus(A) ועל כן להחשב אינטרקטיבי,

הייתרון הוא שלA יש יותר זמן שהוא יכול להשתמש במעבד ועדיין להחשב אינטרקטיבי לעומת תהליך B שיותר בקלות יכול להחשב כתהליך חישובי.

שימו לב: בסעיף ד, שלושה תת סעיפים (1,2,3)

ד. לפניך קטע מתוך הקוד של פונקציית הגרעין yield_sched_sys אשר מממש את הטיפול בתהליכים עם מדיניות זימון OTHER

```
1.list_del(&current¬>run_list);
2. if(!list_empty(array¬>queue +current¬>prio)){
3.  list_add(&current¬>run_list,array¬>queue[current¬>prio].next);
4.  goto out_unlock;
5. }
6. __clear_bit(current¬>prio, array¬>bitmap);
7. i =sched_find_first_bit(array¬>bitmap); // this would return MAX_PRIO on
fail (in case no set bits found)
8. if(i==MAX_PRIO ||i<=current¬>prio)
9.  i =current¬>prio;
10. else
11.     current¬>prio =i;
12. list_add(&current¬>run_list,array¬>queue[i].next);
13. __set_bit(i,array¬>bitmap);
14. out_unlock:
15. // release locks & call schedule
```

1. בהנחה שקיימים תהליכים נוספים שאינם expired ב runqueue האם יתכן כי ביצוע yield_sched על ידי תהליך עם מדיניות זימון OTHER לא יגרום להחלפת הקשר?
נחלק למקרים לפי הקוד:
נתון כי קיימים תהליכים נוספים בrunqueue, נחלק למקרים:
- ***אם קיימים תהליכים בתור העדיפות של התהליך הנוכחי, נוסיף את התהליך הנוכחי להיות אחד אחרי התהליך הבא בתור (שורות 1-4) ונבצע החלפת הקשר (שורה 14, קריאה ל()schedule)***
- ***אם התהליך הנוכחי היה היחיד בעדיפות שלו:***
  - ***אם קיים תהליך בעדיפות חזקה יותר,נמוכה יותר מספרית, (שורות 8-9) נוסיף את התהליך לתור העדיפות הנוכחי שלנו ונבצע החלפת הקשר (12-14), בהחלפת ההקשר נבחר את התהליך בעדיפות החזקה יותר***
  - ***אם התהליך הראשון שקיים תהליך בעדיפות חלשה יותר, גבוהה יותר מספרית (שורה 8, 11) נוסיף את התהליך להיות אחד אחרי התהליך הראשון בעדיפות זו ונבצע החלפת הקשר (12-14)***

בכל המקרים אכן נבצע החלפת הקשר ולכן לא יתכן שביצוע sched_yield לא יגרום להחלפת הקשר תחת תנאי השאלה.

2. איזו בעיה הייתה נוצרת אם היינו מחליפים את שורה 8 בשורה:

```
if(i<=current>prio)
```

במקרה שכזה אנו מתעלמים מן המקרה קצה שבו התהליך שלנו הוא התהליך היחיד בrunqueue ובמידה וזה היה המצב היינו מנסים להוסיף אותו לתור העדיפות MAX_PRIO שחורג מגבולות מערך תורי העדיפויות והיינו מקבלים שגיאה

3. איזו בעיה הייתה נוצרת אם היינו מחליפים את שורה 8 בשורה:

```
if(i==MAX_PRIO)
```

במקרה כזה היינו מתעלמים מן המקרה בו קיים במערכת תהליך עם עדיפות חזקה יותר מהעדיפות של התהליך שלנו,

והיינו מוסיפים את התהליך שלנו להיות אחד אחרי התהליך החזקה יותר ברשימת העדיפויות, ובכך מחזקים את

העדיפות של התהליך שלנו כתופעת לוואי ולא מכיוון שרצינו לעשות זאת.

שאלה 2

במסגרת הבחינה בקורס, תתבקשו לענות על שאלות הן על החומר הנלמד (בתרגולים ובהרצאות) והן על מערכות שונות ומגוונות אשר לא נלמדו בקורס, דבר הדורש הכללה של החומר ועקרונותיו.

בשאלה זו ננתח מערכת זימון הדומה במהותה ללינוקס, אך במקביל, גם קצת שונה. נתון אלגוריתם SCHED_OTHER במערכת זו:

- עדיפות התהליכים נקבעת על פי השדה **`static_prio`** אשר טווח ערכיו בין 1 ל 5 וערכו נקבע על ידי המשתמש (1 הוא העדיף ביותר ו 5 הכי פחות עדיף).
- בכל רגע נתון מוגדר **זמן ריצה מינימלי** למשימות בעלות עדיפות i אשר נסמנו q_i או q_i , על פי הנוסחה:

$$q_i = \max\left\{\frac{\text{target\_ latency}}{N_i}, \text{min\_ granularity}\right\}$$

כאשר `target_latency`, `min_granulartiy` קבועים המוגדרים במערכת, ו- N_i מספר המשימות עם עדיפות i.

- לכל משימה יש שדה `vruntime` המאותחל ל 0 בעת יצירתה. בכל פסיקת שעון מתעדכן שדה באופן הבא:

$$current \rightarrow vruntime += current \rightarrow static\_prio$$

- בכל פעם שמשימה נבחרת לרוץ נשמר ערכו של השדה `vruntime` כך:

$$current \rightarrow start\_vruntime = current \rightarrow vruntime$$

- בכל פסיקת שעון נבדקים שני התנאים הבאים:
  - קיימת משימה בעלת `vruntime` קטן יותר משל המשימה הנוכחית
  - q_{current->static_prio} <= current->vruntime - current->start_vruntime
- אם שני התנאים הללו מתקיימים, מתבצעת החלפת הקשר (בעזרת הדגל `need_resched`)
- בהחלפת הקשר המשימה הבאה שנבחרת לרוץ היא זאת בעלת ה `vruntime` המינימלי.

א. בהנחה שהאלגוריתם הנ"ל עושה שימוש בעץ חיפוש מאוזן הממוין לפי `vruntime`.

1. מהי סיבוכיות הזמן של **בחירת** המשימה הבאה? הסבר.

נחזיק מצביע לעלה הכי שמאלי של עץ החיפוש, בכך נוכל לגשת ב(O(1 למשימה בעלת הvruntime הכי נמוך.

2. מהי סיבוכיות הזמן של **הוספת** משימה חדשה? הסבר.

סיבוכיות הזמן תהיה כשל הוספת איבר לעץ חיפוש מאוזן ז"א (O(logn כאשר n הוא מספר המשימות בעץ

3. מהי סיבוכיות הזמן של **הסרת** משימה? הסבר.

סיבוכיות הזמן תהיה כשל הוספת איבר לעץ חיפוש מאוזן ז"א (O(logn כאשר n הוא מספר המשימות בעץ

ב. נניח שבמערכת יש שני תהליכים CPU-Bound בעלי עדיפות זהה ושהשניים הללו הינם התהליכים היחידים במערכת. איזה אחוז מהזמן ירוץ כל אחד מהתהליכים?

נניח תהליכים A וB במערכת בעלי עדיפות static_prio = i

מכיוון שתהליכים אלו בעלי עדיפות זהה A_q=B_q

וכן בכל רגע שתהליך רץ יתבצע current->vruntime += vruntime+i

לכן לאחר k=q/i <= k*i=q , מספר פסיקות השעות שאחד התהליכים ירוץ ויתחיל התהליך השני שנתוניו זהים ולכן ירוץ k=q/i פסיקות שעון גם כן עד שיחליף לתהליך הקודם לו וכך הלאה. ז"א על כל פסיקת שעון שA ירוץ גם B ירוץ פסיקת שעון.

נסמן בTa,Tb זמן ריצה כולל של התהליכים A וB ובT את זמן הריצה הכללי. ומכיוון שעל כל פסיקת שעון שA ירוץ B ירוץ מתקיים Ta=Tb וכן Ta+Tb=T (כי אלו שני התהליכים היחידים במערכת)

לכן Ta=0.5T=Tb ולכן שני התהליכים ינצלו 50% מהזמן כל אחד.

ג. נניח שבמערכת שני תהליכים CPU-Bound, אחד בעל עדיפות 1, והשני בעל עדיפות 3. נניח שהשניים הללו הינם התהליכים היחידים במערכת. איזה אחוז מהזמן ירוץ כל אחד מהתהליכים?

יהי תהליך A: static_prio(A)=1

יהי תהליך B: static_prio(B)=3

לכן בכל פסיקת שעות בה A רץ מתקיים A->vruntime+=1

ובכל פסיקת שעון בה B רץ מתקיים B->vruntime+=3

מכיוון שאלו שני התהליכים היחידים במערכת מתקיים q(a)=q(b) נסמן q.

כאשר תהליך B ירוץ, יצטרך k=q/3 פסיקות שעון עד שיחליט להחליף הקשר לA

כאשר תהליך A ירוץ, יצטרך k=q פסיקות שעון עד שיחליט להחליף הקשר לB

לכן על כל פסיקת שעון בה B רץ, A ירוץ 3 פסיקות שעון.

נסמן את סה"כ הזמן בT כאשר Ta=זמן ריצת A Tb= זמן ריצת B כאשר Ta=3*Tb

Ta+Tb=T

Ta=0.75 ,Tb=0.25T <= 3Tb+Tb=T

לכן תהליך B ירוץ 25% מהזמן ותהליך A ירוץ 75% מהזמן.

ד. באלגוריתמי הזימון SCHED_OTHER שלמדנו (בתרגול ובהרצאה), המערכת מחשבת לכל תהליך עדיפות דינמית כדי להבדיל בין תהליכים שהם IO-Bound לתהליכים שהם CPU-Bound. באלגוריתם הנ"ל, אין הפרדה כזו. כיצד בכל זאת מתעדפת המערכת תהליכי IO-Bound כראוי?

מכיוון שתהליך IO-Bound כמעט ולא משתמש בזמן ריצה, הvruntime שלו ישאר מאוד נמוך ולכן העדיפות שלו תשאר חזקה לאורך זמן

ה. באיזו בעיה היינו עלולים להיתקל אם לא היה נעשה שימוש בקבוע `min_granulartiy`?

במצב בו N, מספר התהליכים מאוד גדול, אנו עלולים להגיע לזמן ריצה מינימלי מאוד קטן ובכך לגרום להמון החלפות הקשר שיגרמו לזמן רב להתבזבז על החלפות ההקשר עצמן ולא בהרצת התהליכים.

שימו לב: הסעיף הבא מתייחס למערכת כפי שנלמדה בתרגולים. בפרט, אלגוריתם זימון המשימות הוא זה שנלמד בביתה ולא האלגוריתם שהוזכר לעיל.

ו. נניח כי תהליך מסויים מודע לזמן בו מתרחשת פסיקת שעון והוא מסוגל לבחור להריץ קוד כרצונו בדיוק *לפני/אחרי שמתקבלת פסיקת שעון. כיצד יכול התהליך לנצל זאת כדי "לרמות" את אלגוריתם הזימון?*

אם תהליך יודע מתי תהיה פסיקת שעון, יוכל התהליך לקרוא לפונקציה yield ובכך לבצע החלפת הקשר לפני שפסיקת השעון תגרום להורדת זמן מהtime_slice שלו,

בכך בעצם יוכל לרמות את המערכת ולרוץ בין פסיקות שעות בלי לפגוע בtime_slice שלא לעולם.

Submission Format

1. Only **typed** submissions in **PDF** format will be accepted. Scanned handwritten submissions will not be graded.
2. The dry part submission must contain a single PDF file named with your student IDs –

   DHW3_123456789_300200100.pdf
3. The submission should contain the following:
   a. The first page should contain the details about the submitters - Name, ID number and email address.
   b. Your answers to the dry part questions.
4. Submission is done electronically via the course website, in the **HW2 – Dry** submission box.

Grading

1. **All** question answers must be supplied with a **full explanation**. Most of the weight of your grade sits on your **explanation** and **evident effort**, and not on the absolute correctness of your answer.
2. Remember – your goal is to communicate. Full credit will be given only to correct solutions which are **clearly** described. Convoluted and obtuse descriptions will receive low marks.

Questions & Answers

- The Q&A for the exercise will take place at a public forum Piazza **only**. Please **DO NOT** send questions to the private email addresses of the TAs.
- Critical updates about the HW will be published in **pinned** notes in the piazza forum. These notes are mandatory and it is your responsibility to be updated.

A number of guidelines to use the forum:

- Read previous Q&A carefully before asking the question; repeated questions will probably go without answers
- Be polite, remember that course staff does this as a service for the students
- You're not allowed to post any kind of solution and/or source code in the forum as a hint for other students; In case you feel that you have to discuss such a matter, please come to the reception hour
- When posting questions regarding **hw2**, put them in the **hw2** folder

Late Days

Please **DO NOT** send postponement requests to the TA responsible for this assignment. Only the **TA in charge** can authorize postponements. In case you need a postponement, please fill out the attached form : https://goo.gl/forms/HDFZz3MMtmZxvgXg2