Homework 2 Dry

**Due Date: 11/12/2018 23:30**

Teaching assistant in charge:

* Andre Kassis

**Important:** the Q&A for the exercise will take place at a public forum Piazza only. 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

Only the TA in charge can authorize postponements. In case you need a postponement, please fill out the following form: https://goo.gl/forms/D5nxMxf9Uvgej1SL2

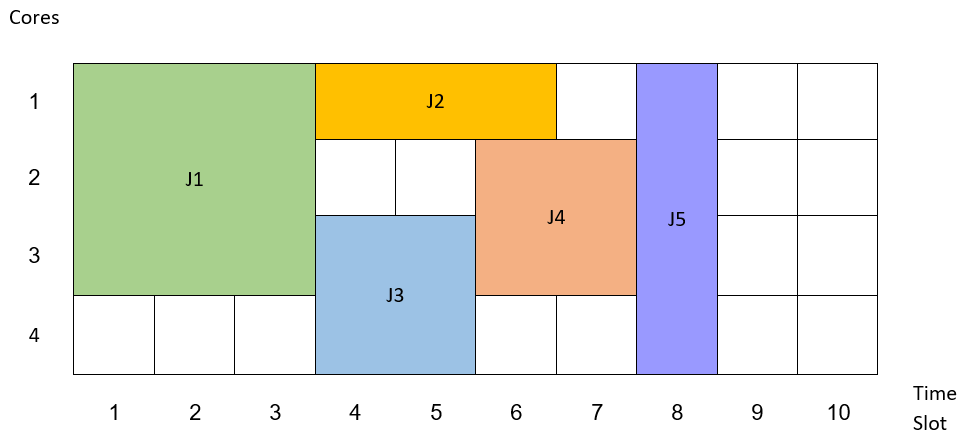
Dry part submission instructions:

1. Please submit the dry part to the electronic submission of the dry part on the course website.
2. The dry part submission must contain a single dry.pdf file containing the following:
   1. The first page should contain the details about the submitters - Name, ID number and email address.
   2. Your answers to the dry part questions.
3. Only typed submissions will be accepted. Scanned handwritten submissions will not be accepted. Only PDF format will be accepted.
4. You do not need to submit anything in the course cell.
5. When you submit, **retain your confirmation code and a copy of the PDF**, in case of technical failure. It is **the only valid proof** of your submission.

**יש לנמק כל תשובה, תשובות ללא נימוק לא יתקבלו.**

# שאלה 1 - זימון תהליכים (50 נק')

שאלה זו עוסקת בנושא batch-scheduling כפי שנלמד בהרצאות. שימו לב שבכל טבלאות התזמון הבאות המספר המייצג time-slot כלשהו מייצג את הזמן שבו החלון הנתון **מסתיים** (כלומר - אם time slot = 3 אזי החלון **מתחיל** ב-t=2 ומסתיים ב- t=3).

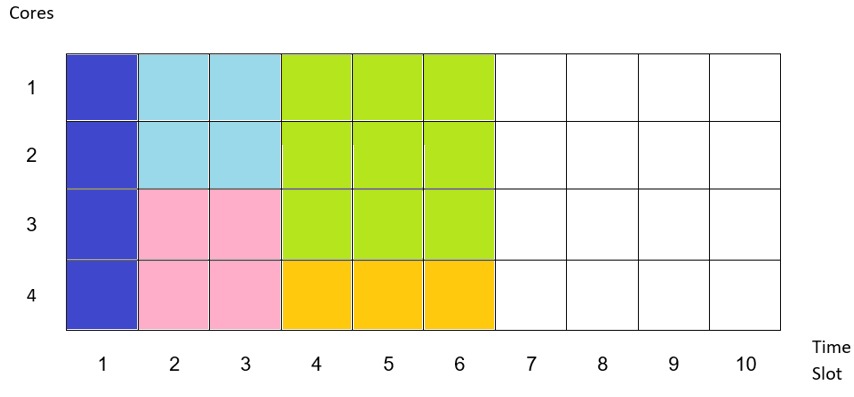


1. (18 נק') בהינתן התזמון הנתון למעלה, חשבו את המדדים הבאים (הראו את דרך החישוב). הניחו שכל התהליכים בנתונים למעלה הגיעו באותו הזמן (t=0):
   1. (6 נק') מדד זמן ההמתנה הממוצע (average wait-time):
   2. (6 נק') מדד זמן התגובה הממוצע (average response-time):
   3. (6 נק') מדד הניצולת (utilization):

ניצולת מחושבת כאחוז מס' הtime slots התפוסים בטבלת תזמון מכלל הtime slots בטבלה (עד אשר התהליך האחרון הסתיים). במקרה זה, מס' התפוסים הוא 24 והמס' הכולל עד לסיום התהליך האחרון הוא 32. לכן, הניצולת היא: .

1. (8 נק') בהנחה שכל התהליכים למעלה מגיעים בזמן 0. איזו מדיניות זימון יכולה לשפר את הניצולת כפי שחושבה בסעיף הקודם? **נמקו** (ראו מצורפת טבלת זימון ריקה על מנת להמחיש את הזימון החדש לפי האלגוריתם, **עליכם למלא את הטבלה בהתאם**)

נשתמש במדיניות SJF ונסדר את התהליכים לפי אורך זמן הריצה שלהם. מצורפת טבלה של סידור התהליכים במדיניות זו.



J3

J1

J5

J4

J2

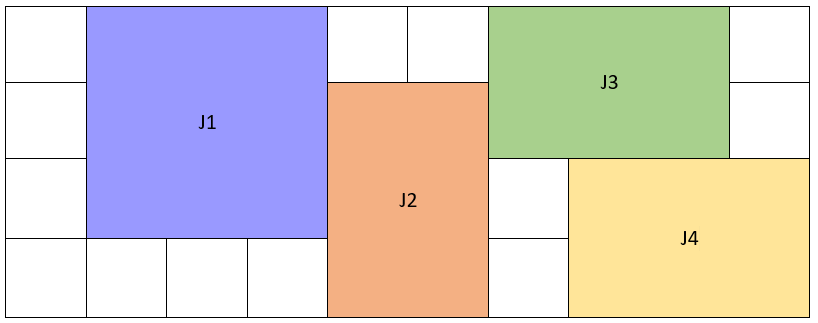
1. (8 נק') מנו 2 יתרונות ו-2 חסרונות של מדיניות זימון מסוג batch-scheduling על פני מדיניות Round-robin. נמקו.  
   **יתרונות:**
   1. פחות החלפות הקשר- כל תהליך שמתחיל לרוץ, ימשיך עד לסיום. זה יתרון כי בכל החלפת הקשר יש תקורה והמעטת החלפות הקשר תמעיט את התקורה.
2. זמן התגובה הממוצע של batch-scheduling נמוך יותר מזמן התגובה הממוצע של RR .

**חסרונות:**

1. ב- batch-scheduling, ה-wait time הממוצע גדול יותר מאשר בRR משום שתהליכים חייבים לחכות שהתהליכים העדיפים עליהם ירוצו קודם.
2. batch-scheduling דורש ידע מקדים של אורכי המשימות ומספר המעבדים עבור כל משימה.

נתונה מדיניות זימון חדשה (BSAF (Biggest Surface-Area First לפיה בהינתן 2 תהליכים זה שנבחר לרוץ קודם הוא זה **שהשטח שלו הגדול יותר** (שטח = זמן \* מספר-מעבדים). שימו לב שמדיניות זימון זו תומכת ב-backfilling - כלומר שאם התהליך העדיף ביותר לפי BSAF לא יכול להיות מתוזמן בחלון כלשהו, יתוזמן התהליך הטוב ביותר שמתאים לחלון אחריו (זה שמתאים לחלון הפנוי והוא הטוב ביותר לפי BSAF). בהינתן 2 תהליכים בעלי אותו שטח נבחר בזה שיש לו את זמן הריצה הקצר ביותר. בהינתן 2 תהליכים בעלי שטח זהה וזמן ריצה זהה, נבחר בזה עם ה-ID (מספר) הנמוך יותר.

למשל, בהינתן 4 התהליכים הבאים (האיור להמחשה של מימדי התהליכים בלבד):



האלגוריתם BSAF יתעדף אותם בסדר הבא:

J1 (3\*3=9 surface area)

J2 (3\*2=6, shorter runtime than J3 and J4)

J3 (2\*3=6, same as J4, but lower ID)

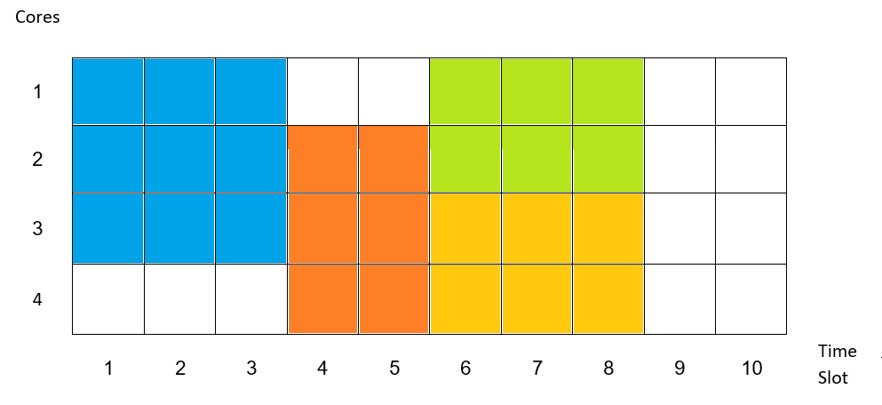
J4

1. (8 נק') האם BSAF סובל מ-convoy effect? נמקו (עליכם להמחיש בעזרת הטבלה הנתונה).

כן, בתמונה הראשונה (למטה) ניתן לראות את טבלת התזמון של התהליכים הנ"ל, לפי מדיניות BSAF. בתמונה השנייה (למטה) ניתן לראות את טבלת התזמון של התהליכים הנ"ל לפי מדיניות SJF (עבור תהליכים באותו אורך, נבחר את התהליך בעל השטח הקטן יותר). עבור המדיניות הראשונה מתקיים-

ועבור המדיניות השנייה מתקיים-

לכן, BSAF סובלת מ- convoy effect.

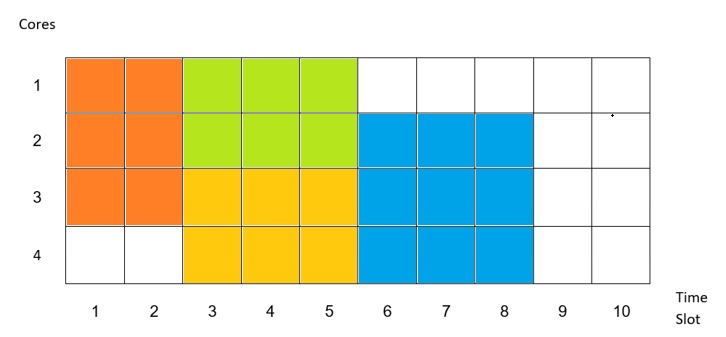


J3

J1

J4

J2



J3

J2

J4

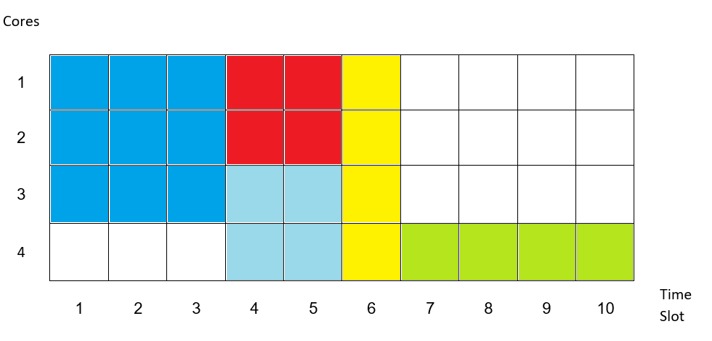
J1

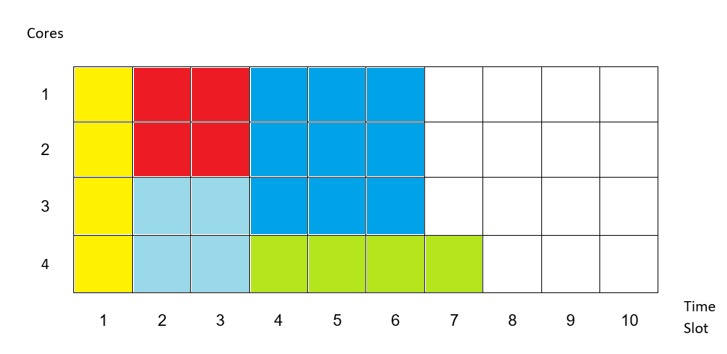
1. (8 נק') האם אלגוריתם הזימון BSAF הינו אופטימלי מבחינת מדד הניצולת? הוכיחו או הפריכו (עליכם להמחיש בעזרת הטבלה הנתונה)

לא. ניתן לראות בתמונה הראשונה (מדיניות BSAF) שהניצולת היא (חישוב כפי שביצענו בסעיף 1):

ובתמונה השנייה (מדיניות SJF כאשר עבור תהליכים באותו אורך, נבחר את התהליך בעל השטח הקטן יותר) הניצולת היא (חישוב כפי שביצענו בסעיף 1):

לכן, BSAF לא אופטימלי מבחינת ניצולת.





# שאלה 2 - החלפת הקשר (20 נקודות)

.1נתון הקטע הבא מתוך קוד הגרעין להחלפת הקשר:

**01.** movl prev, %eax

**02.** movl next, %edx

**03.** pushl %esi

**04.** pushl %edi

**05.** pushl %ebp

**06.** movl %esp, prev->**thread**.esp

**07.** movl next->**thread**.esp, %esp

**08.** movl $**1f**, prev->**thread**.eip

**09.** pushl next->**thread**.eip

**10.** jmp \_\_switch\_to

**11.** **1**:

**12.** popl %ebp

**13.** popl %edi

**14.** popl %esi

א. מה יקרה אם נחליף את שורות 10-8 בשורה **call \_\_switch\_to** ?

eip שומר את הכתובת של הפקודה הבאה לביצוע. בשורה 8 אנו שומרים ב-eip את הפקודה הבאה לביצוע בעת חזרת ריצת התהליך. בשורה 9 אנו שומרים את ה-eip במחסנית. אם נבצע את חילוף השורות התוכנית תרוץ לפקודה הבאה לביצוע שהיא שורה 11. עבור המקרה הסטנדרטי, אנו מעוניינים ששורה 11 תרוץ, אך עבור המקרה שבו התהליך ש"חוזר" לרוץ הוא בן שהגיע מfork אנו רוצים שהשורה הבאה שתרוץ תהיה ret\_from\_fork כפי שראינו בתרגול. לכן נקבל התנהגות לא צפוייה.

ב. נניח כי ביצענו קריאה למאקרו current לפני שורה 1 ,לאחר איזה שורה קריאה נוספת למאקרו תחזיר תשובה שונה?

לאחר שורה 7. המאקרו מחזיר את הpcb הנוכחי לפי מחסנית הגרעין הנוכחית. בשורה 7 אנו מחליפים מחסנית גרעין, לכן לאחר שורה זו נקבל תשובה שונה במאקרו.

ג. מתרגל בקורס רצה לשפר את זמן הביצוע של החלפת ההקשר. מכיוון ששם לב כי המאקרו **switch\_to** אינו עושה כלל שימוש ברגיסטר ecx ומאחר וידוע כי פעולות על רגיסטרים מהירות משמעותית מפקודות המערבות את הזיכרון, הציע להחליף את שורה **05** בשורה **mov %ebp, %ecx** ואת שורה 12 בשורה **mov %ecx, %ebp** האם הקוד יעבוד כראוי לאחר השינוי המוצע?

בשורה 5 נשמר ebp של prev.

בשורה 12, לאחר החלפת המחסניות, מבוצע שחזור של ebp של next.

אם נחליף את השורות כפי שהוצע, בecx ישמר ebp של prev בשורה 5, החלפת המחסניות לא תשפיע על ecx (וגם לא \_\_switch\_to). לכן, בשורה 12 ישוחזר אותו ערך של ecx כלומר ebp של prev ולא של next כפי שרצינו.

2. כפי שראינו בתרגול מצביע לבסיס מחסנית הגרעין של התהליך הנוכחי נשמר גם ב **thread->esp0** וגם ב – **tss->esp0**האם ניתן לוותר על אחד מהם? אם לא הסבר מדוע ואם כן פרט על מי וכיצד נשיג את המידע הדרוש.

לא ניתן לוותר על **tss->esp0** כי הוא מקושר לרכיב חומרתי במעבד. על **thread->esp0** ניתן לוותר על חשבון זמן חישוב. ניתן לחשב אותו בעזרת **tss->esp0** בכל פעם שנצטרך להשתמש בו.

# שאלה 3 – זימון תהליכים (30 נקודות)

שאלה זו עוסקת במדיניות זימון התהליכים של לינוקס כפי שנלמדה בתרגולים.

לנוחיותכם מצורף חלק מהקוד כפי שנלמד בתרגולים:

#define MAX\_PRIO 140

#define MIN\_TIMESLICE (10 \* HZ / 1000) /\* 10 msec \*/

#define MAX\_TIMESLICE (300 \* HZ / 1000) /\* 300 msec \*/

#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

**if** (prio < MAX\_RT\_PRIO)

prio = MAX\_RT\_PRIO;

**if** (prio > MAX\_PRIO - **1**)

prio = MAX\_PRIO – **1**;

BONUS(p) = **10**\*(SleepAvg/MaxSleepAvg-**1**/**2**)

DELTA(p) = **5**\*TaskNice(p)/**20**+**2**

1. (5 נק') האם ייתכן מצב בו תהליך אינטראקטיבי **A** יהיה באותה העדיפות הדינאמית כמו תהליך חישובי ? **B** אם כן תארו מצב כזה (תארו 2 תהליכים, את עדיפותם הסטטית, ואת עדיפותם הדינאמית - והוכיחו מספרית​ בעזרת חישובי **bonus** ו-**DELTA** שהתהליכים אכן מקיימים את הדרישות), אם לא נמקו מדוע.

|  |  |  |
| --- | --- | --- |
| B-חישובי | A - אינטראקטיבי |  |
| -10 | 8- | Nice |
| -1 | 1 | Bonus |
| 120 + Nice = 120 -10 = 110 | 120 + Nice = 120 – 8 = 112 | Static |
| (Nice/4) + 2 = 0 | (Nice/4) + 2 = 0 | Delta |
| Static – Bonus = 110+1 = 111 | Static – Bonus = 112-1 = 111 | Prio |
| No | Yes | Interactive? |

1. (2 נק') בתרגול מצוינות **4** סיבות בגינן עשוי תהליך להגיע לפונקציה .schedule בחנו את הקוד של**sys\_sched\_yield** ו- **interruptible\_sleep\_on**. בכל אחת מהפונקציות הנ"ל ישנה קריאה ישירה לפונקציה **schedule** . אם היינו משנים קריאה זו ל - **set\_tsk\_need\_resched(current)** כיצד זה היה משפיע אם השינוי היה מתבצע רק ב- **sys\_sched\_yield**? כיצד זה היה משפיע אם השינוי היה מתבצע רק ב- interruptible\_sleep\_on ? נמקו (בתשובתכם התייחסו לנכונות הביצוע - כלומר האם התהליך אכן מוצא מהקשר? האם הקרנל ימשיך לעבוד בצורה תקינה? וכו')

עבור **sys\_sched\_yield**: במצב הנוכחי, sys\_sched\_yield מבצעת החלפת הקשר באופן מיידי, אך במידה ונחליף לשורה המצויינת בסעיף - במידה ויש קוד נוסף לביצוע בקרנל מוד, הוא יבוצע לפני שהתהליך הנוכחי יוותר על המעבד (בעת חזרה ליוזר מוד). במידה ואין קוד נוסף לביצוע בקרנל מוד, התהליך יבדוק את הדגל טרם חזרתו ליוזר מוד ויבצע את החלפת ההקשר.

עבור **interruptible\_sleep\_on**: במצב הנוכחי, interruptible\_sleep\_on משנה את מצב התהליך הנוכחי ומכניסה אותו לתור המתנה, לאחר מכן מבצעת החלפת הקשר ע"י קריאה ל-schedule, בעת חזרה מהמתנה, בפונקצייה זו התהליך יצא מתור ההמתנה. במידה ונחליף את השורה המתבקשת, התהליך יכנס לתור המתנה ומיד יצא ממנו. כלומר התהליך לא יהיה בהמתנה כלל ולכן נקבל התנהגות בלתי צפויה. אם התהליך יחזור ליוזר מוד, ויצטרך לבצע החלפת הקשר (בגלל הדגל שהדלקנו) הוא עשוי להימחק מתור הריצה ב-schedule (כי שינינו את מצבו בפונקציה interruptible\_sleep\_on) ויווצר מצב שהוא לא בתור ריצה ולא באף תור המתנה.

1. לצורך שאלה זו נזכיר כיצד מחושב **sleep\_avg** של תהליך בגרעין:

#define MAX\_SLEEP\_AVG (2\*HZ)

sleep\_time = jiffies – p->sleep\_timestamp;

p->sleep\_avg += sleep\_time;

**if** (p->sleep\_avg > MAX\_SLEEP\_AVG)

p->sleep\_avg = MAX\_SLEEP\_AVG;

#define EXPIRED\_STARVING(rq) \

((rq)->expired\_timestamp && \

(jiffies – (rq)->expired\_timestamp >= \

STARVATION\_LIMIT \* ((rq)->nr\_running + **1**))

1. נניח שקיים תהליך אינטראקטיבי **A** בעדיפות סטטית 100 ובעל **sleep\_avg=MAX\_SLEEP\_AVG**.

בנקודת זמן t=0התהליך החל לבצע משימה חישובית ארוכה. מהו הזמן המקסימלי (במילישניות) שהתהליך ירוץ לפני שיעבו ר ל **expired** בהנחה שהוא התהליך היחיד ב?runqueue

A אינטראקטיבי לכן הוא ירוץ מספר time slices עד אשר הוא יחשב חישובי ולאחר סיום הtime slice הנוכחי שלו יעבור ל-expired. זה יקרה לאחר **1800ms**.

2. כיצד תשובתכם לסעיף הקודם הייתה משתנה אם נתון שקיים תהליך **B** ב **expired** והחל מהרגע t=0 שתואר קודם, נותרו 1000 מילישניות עד כילוי ה ? **timestamp\_expired**כלומר עד שהמאקרו **EXPIRED\_STARVING** מתחיל להחזיר **TRUE**).

לאחר 1000 מילישניות המאקרו יחזיר אמת ו-scheduler\_tick לא תחזיר תהליכים אינטראקטיבים לאקטיב, בשלב זה, A טרם סיים את הtime slice הנוכחי שלו. לאחר 1200 מילישניות, A יסיים וכאשר הוא יבצע בדיקה ל- **EXPIRED\_STARVING** הוא יועבר ל-expired. כלומר **1200ms** A יועבר לexpired.

1. מהו פרק הזמן המינימלי שיכול לעבור מהרגע שהודלק הדגל **need\_resched** בתהליך שרץ ועד שהוא מגלה את הצורך בהחלפת הקשר (במערכת מרובת מעבדים)?

כן.

1. האם אפשר לתת דוגמה בה תהליך יכול להתחיל לרוץ קצת אחרי פסיקת שעון ולעזוב את ה CPU קצת לפני פסיקת השעון הבאה (פחות מ ?(tickאם הדבר בלתי אפשרי הסבירו למה, אחרת תנו דוגמה מפורטת.

כן, במידה והתהליך שהחל לרוץ מעט אחיר פסיקת השעון, יבצע קריאת מערכת חוסמת, למשל wait, שאיננה תלויה בשעון.