

Parrallel Proccessing – HW 2

Parallel Computations with MPI **Goal: Synchronous computing ‘Game Of Life’**

Dvir Zaguri 315602284
Yehonatan Arama 207938903

1. הקדמה

נדרשנו לבצע חישוב של התוכנית 'Game of Life' באמצעות חישובי MPI סינכרוניים של תוכנית עתירת תקשורת בין תאים שכנים. המטרה היא לבחון את יכולות החישוב המקבילי שמבצע מעבד בעל מספר ליבות עבור חישובים ארוכים עם המון איטרציות באמצעות MPI ומדידת זמנים וניתוח הפרשי הביצועים תוך היעזרות במחשבי ה-hobbit. בעבודה הנ"ל השתמשנו בשני כלים profiling כנדרש בעבודה 'Scalasca' ו-'Jumpshot'. Scalasca היא כלי ראשי לניתוח ביצועים של אפליקציות מקביליות בעוד Jumpshot נותן ויזואליזציה של ביצוע המכונה ב-execution של תוכנית מסוימת ומאפשר למשתמש לנתח בעצמו.

2. מהלך הניסוי

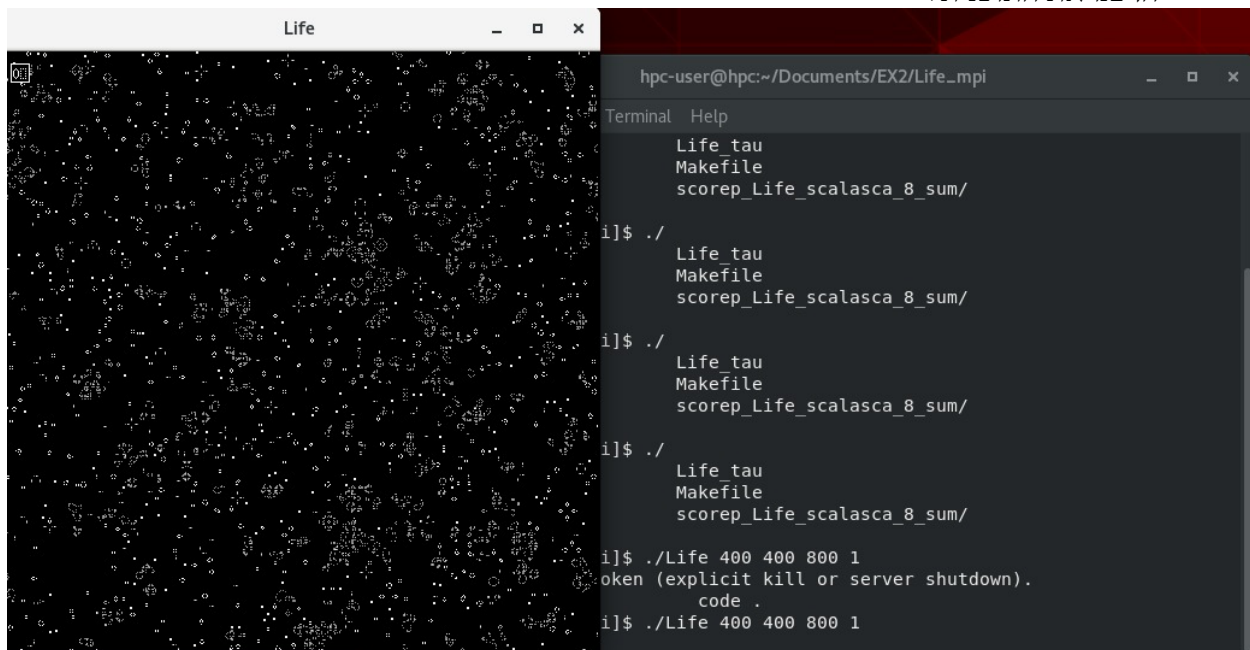
השיטה

הרצת התוכנית הכתובה שקיבלנו באמצעות מחשבי ה-hobbit ומדידת זמני הריצה באמצעות פונקציות טיימינג והדפסות המדידות. לאחר מכן נתח את המדידות באמצעות כלי ה-profiling שהצגנו בהקדמה (scalasca jumpshot) ונשווה להרצה במכונה הווירטואלית.

פירוט הפתרון

סעיף א':

הרצנו את התוכנית:



```
hpc-user@hpc:~/Documents/EX2/Life_mpi
Terminal Help
Life tau
Makefile
scorep_Life_scalasca_8_sum/

i1]$ ./
Life tau
Makefile
scorep_Life_scalasca_8_sum/

i1]$ ./
Life tau
Makefile
scorep_Life_scalasca_8_sum/

i1]$ ./
Life tau
Makefile
scorep_Life_scalasca_8_sum/

i1]$ ./Life 400 400 800 1
oken (explicit kill or server shutdown).
code .
i1]$ ./Life 400 400 800 1
```

סעיף ב':

ראשית, בעיית game of life היא בעיה דו-מימדית, שניתנת לחלוקה בכמה דרכים: חלוקת הבעיה לפי עמודות, שורות, בלוקים או דרך אחרת. במקרה שלנו ניתן להבחין במקבול בפונקציה `copy_bounds()`. בפונקציה הזו יש מעבר מידע בין ליבות/הוביטים וניתן לראות שבצורה שבה הוגדרה התקשורת בין הליבות, הפונקציה משתמשת ב-`MPI_Sendrecv()` לפי שתי ההזנות של הארגומנטים, `TOLEFT` ו-`TORIGHT`. זאת אומרת שה-`Domain decomposition` של התוכנית היא טורית, חלוקת הריצה לפי עמודות בין ה-ranks השונים אשר נמצאים מימין/משמאל אחד לשני.

סעיף ג':

weak scaling – הוא הגדלת גודל החישוב הכולל בהתאמה להגדלת כוח העיבוד העומד לרשות התוכנית.

ניתן לראות שהתוצאות מתאימות ל-weak scaling לו אנחנו מצפים, זמני הריצה גדלים ככול שהמיקבול גדל. זאת בגלל והתקשורת בין rank-ים לוקחת זמן גדול יותר משמעותית מהזמן שחוסך לנו המקבול וזה מה שגורם לריצה שלנו להתארך כ"כ מפעם לפעם.

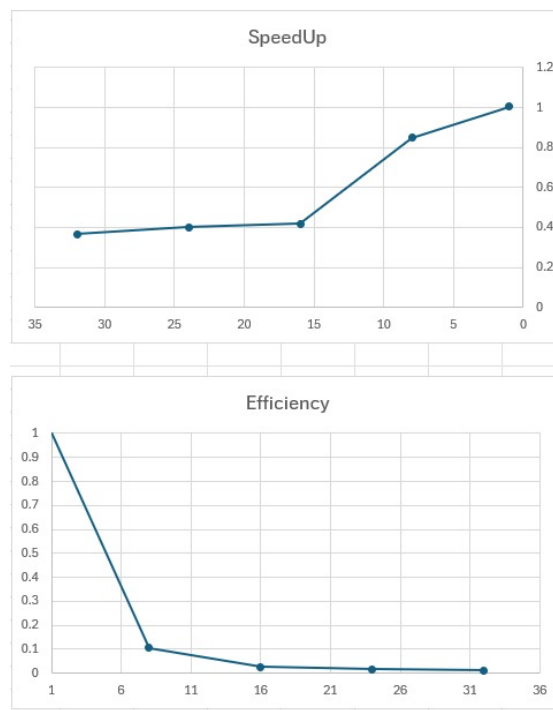
להלן תוצאות ההרצות לפי הטבלה:

# of nodes used	Total # of cores	Program arguments for rows and columns	Serial time	Parallel time
1	1	400	0: 04.4	
1	8	400		0: 05.2
2	16	400		0: 10.6
3	24	400		0: 11
4	32	400		0: 12.05

```

hpc-user@hpc:~/Documents/EX2/Life_mpi
File Edit View Search Terminal Help
hobbit8.ee.bgu.ac.il> time mpirun -np 32 -machinefile ../machinefile ./Life 400 400 800 0
83.152u 9.973s 0:12.03 774.0% 0+0k 0+0io 0pf+0w
hobbit8.ee.bgu.ac.il> vim ../machinefile
hobbit8.ee.bgu.ac.il> time mpirun -np 24 -machinefile ../machinefile ./Life 400 400 800 0
76.739u 8.722s 0:10.94 781.0% 0+0k 0+0io 0pf+0w
hobbit8.ee.bgu.ac.il> vim ../machinefile
hobbit8.ee.bgu.ac.il> time mpirun -np 16 -machinefile ../machinefile ./Life 400 400 800 0
74.258u 8.316s 0:10.59 779.6% 0+0k 0+0io 0pf+0w
hobbit8.ee.bgu.ac.il> time mpirun -np 8 ./Life 400 400 800 0
41.165u 0.211s 0:05.21 794.0% 0+0k 0+0io 0pf+0w
hobbit8.ee.bgu.ac.il> time mpirun -np 1 ./Life 400 400 800 0
4.412u 0.015s 0:04.42 100.0% 0+0k 0+0io 0pf+0w
hobbit8.ee.bgu.ac.il>

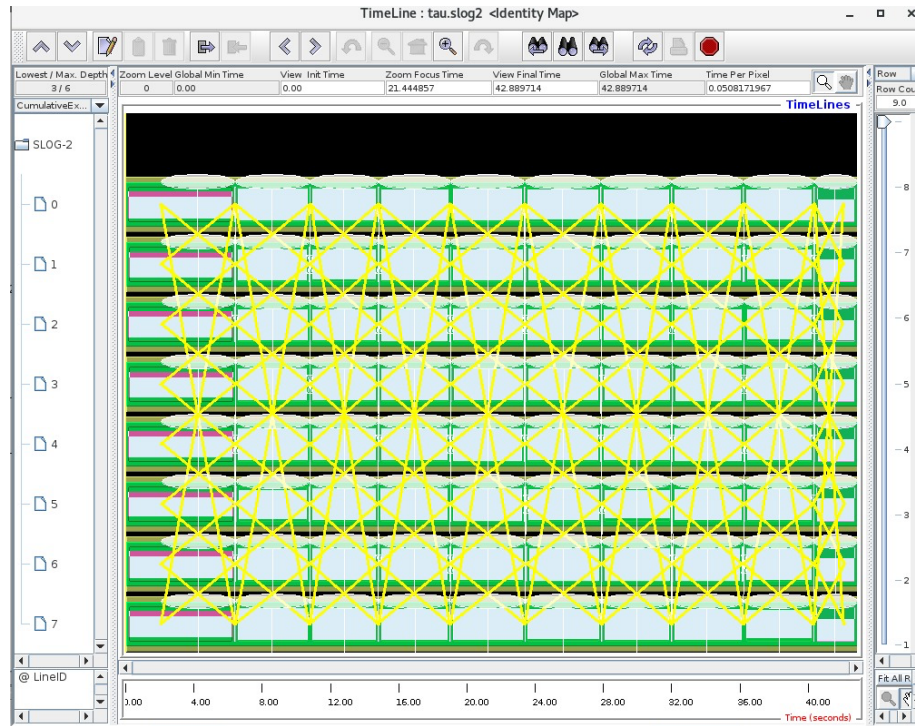
```



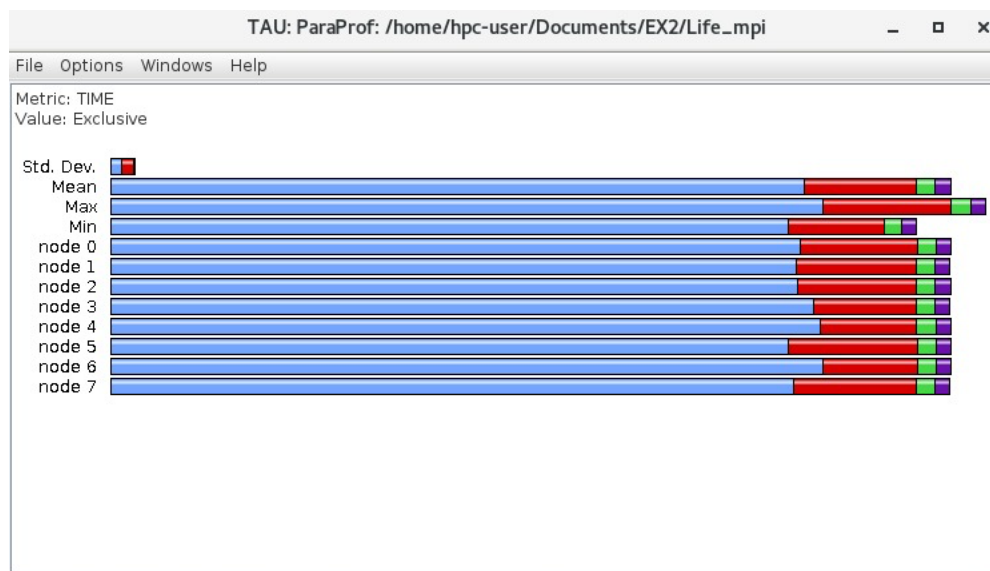
סעיף ה':

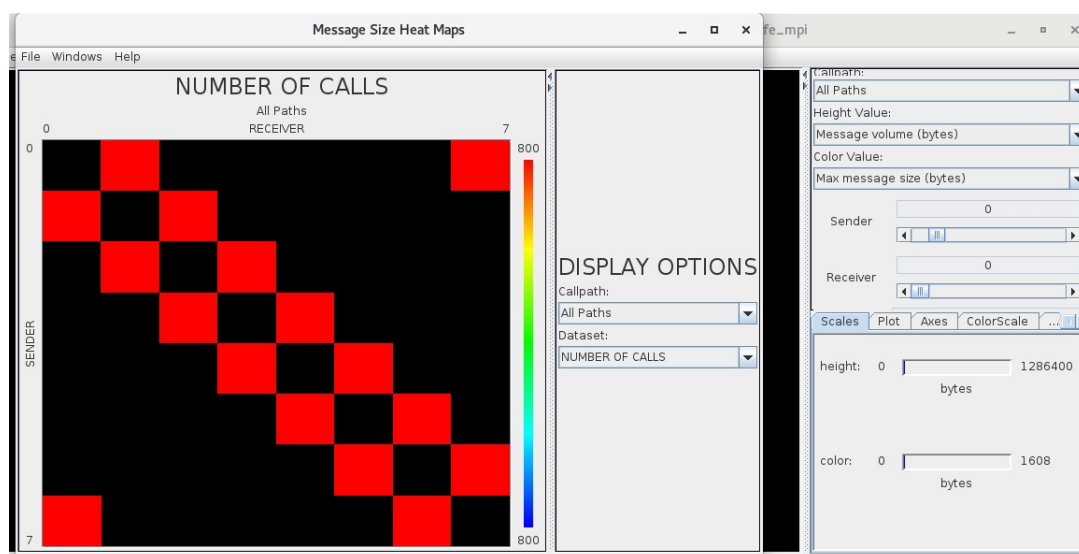
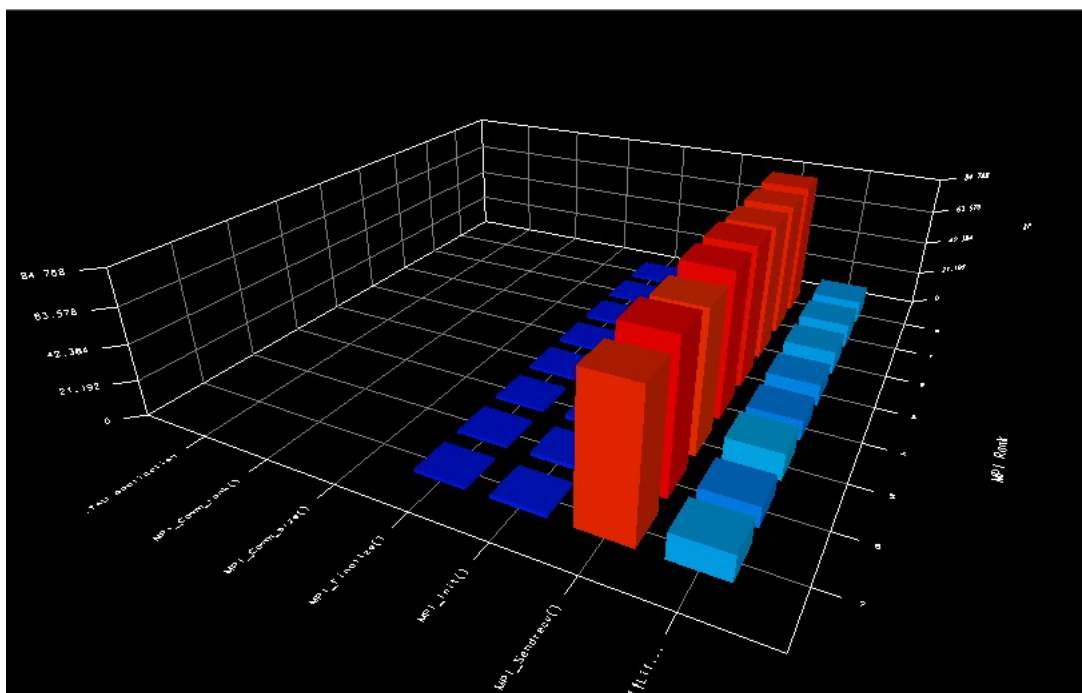
עתה, נריץ את הקודים האלו עם כלי הפרופילינג *Jumpshot* ו-*Scalasca* :

Jumpshot:

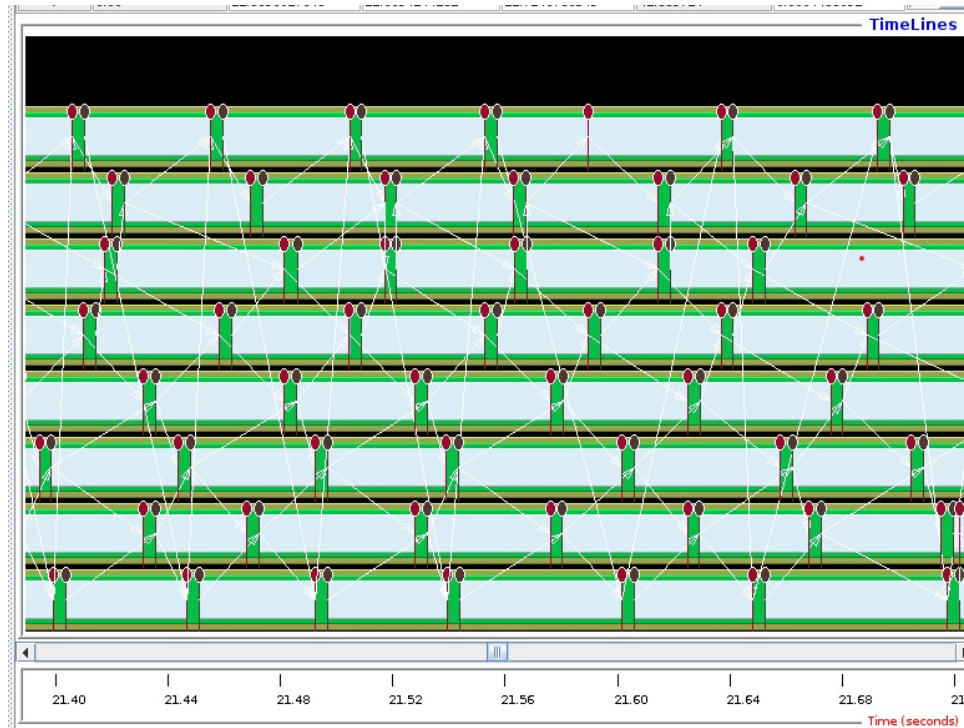


Paraproof:

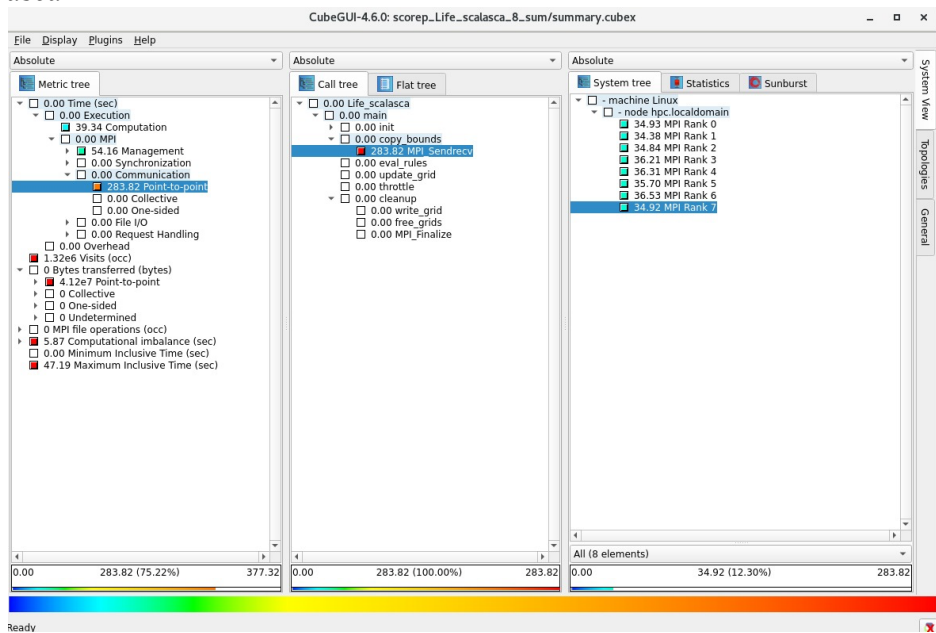




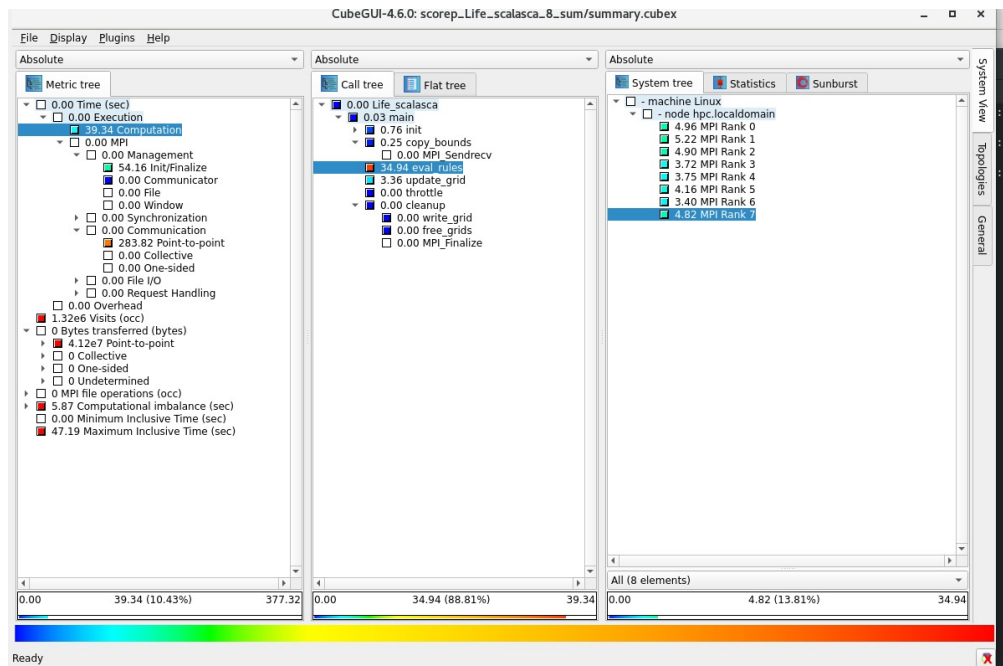
בתמונה זו ניתן לראות שכל rank מתקשר רק עם המעבדים שמימנו/משמאלו



Scalasca:



ניתן לראות בניתוח זה שהתבזבזו 283 יחידות זמן עבור התקשורת בין הליבות והמחשבים בעוד שאר הזמן, הרבה פחות זמן, לקחו החישובים עצמם.



3. סיכום ומסקנות

לסיכום, ניתן להבחין כי כאשר אנחנו כותבים תוכנית בצורת *weak scaling* נצפה לראות עליה בזמן הריצה. לדעתנו את התוכנית הזאת היה ניתן גם לכתוב כ- *strong scaling* אם היינו רוצים לבצע את החישוב של המשחק בצורה מהירה יותר. הכל תלוי במטרות התוכנית.

בנוסף, ניתן לראות שחלוקת העבודה למעבדים התבצעה בצורה של עמודות במטריצת הבעיה. אנחנו חושבים שזו חלוקה טובה (אם כי שוות ערך לחלוקה של שורות), זאת מכיוון שאם היינו מחלקים נגיד לקוביות אז היינו מקבלים תקשורת ל-4 צדדים, כלומר ליותר תהליכים ויתכן והיינו 'מבזבזים' יותר זמן על תקשורת, שגם ככה מסתמן שלוקחת הרבה זמן מהתוכנית. נהננו מהעבודה, תודה.