מאיצים חישוביים ומערכות מואצות
046278

דו"ח הגשה – תרגיל בית רטוב 2

מתן צחורי 208936989

עדי צחורי 315374066

חלק 1

a) ממומש ב-ex2.cu

b) בהרצת השרת במצב Streams עם $load = 0$ קיבלנו:

$$throughput = 438.6 \left[\frac{req}{sec}\right]$$

נסמנו כ-maxLoad.

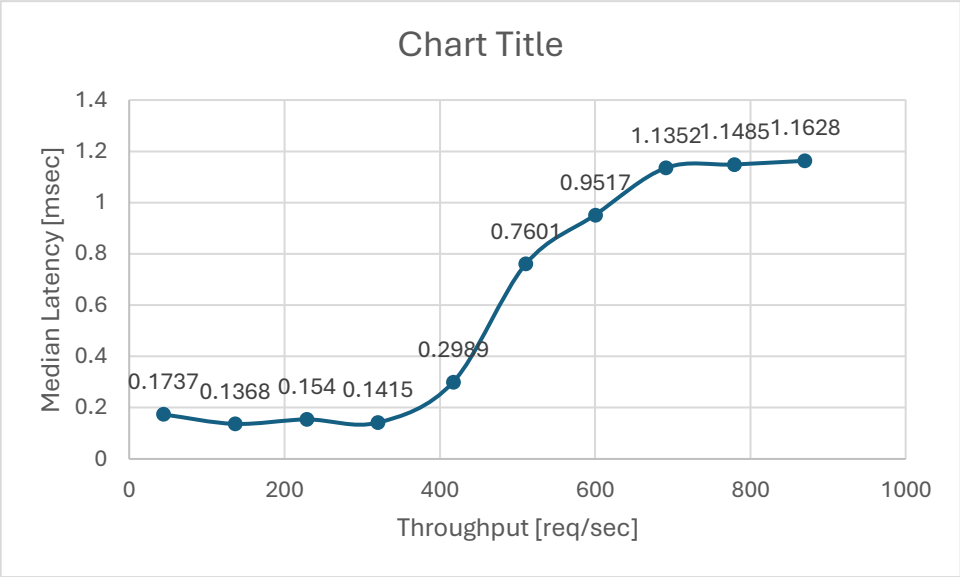
c) נריץ את השרת כעת עם $load$ משתנה בקפיצות קבועות בטווח:
$\left[\frac{maxLoad}{10}, 2 \cdot maxLoad\right]$, נעגל למספר שלם לצורך נוחות.

$$load \in \{43, 136, 229, 321, 414, 506, 599, 692, 784, 877\}$$

להלן התוצאות המתקבלות מההרצות השונות:

Load $\left[\frac{Req}{sec}\right]$	**Throughput** $\left[\frac{Req}{sec}\right]$	**Latency** $[msec]$				
		Average	Minimum	Median	99th Percent	Maximum
43	43.8	0.5406	0.0599	0.1737	4.3726	124.3338
136	135.9	0.2759	0.1241	0.1368	3.1474	123.6057
229	228.7	0.4467	0.0597	0.1540	2.4609	130.0971
321	320.0	0.3471	0.0532	0.1415	2.4350	4.9791
414	416.9	0.5925	0.0428	0.2989	2.8770	4.9230
506	510.6	0.8746	0.0392	0.7601	2.4287	5.1120
599	599.9	0.9785	0.0463	0.9517	2.3254	4.9685
692	690.9	1.1501	0.0483	1.1352	2.5169	4.9347
784	779.2	1.1559	0.0444	1.1485	2.7071	5.0443
877	870.0	1.1338	0.0390	1.1628	2.3212	4.8677

d) להלן מצורף גרף המציג את ההשהיה החציונית עבור קצבי התמונות לשנייה כפי שהגדרנו בסעיף קודם. כאשר ציר ה-x הוא ה-$Throughput$ וציר ה-y הוא ה-$Median\ Latency$:



מגרף זה ניתן לראות כי עבור עומס הקטן מהעומס שמצאנו בסעיף ב, ההשהיה לכל בקשת תמונה נמשכת כמות זמן קטנה יחסית ובפרט נמוכה מההשהיה עבור $maxLoad$.
אך ככל שהעומס על השרת עולה כך ההשהיה הולכת וגדלה. כלומר ככל שמעמיסים יותר על השרת, כך קצב מילוי הבקשות קטן.

חלק 2

a) נחשב את מספר ה-$Threadblocks$ שניתן להריץ עבור כל SM ונכפיל במספר ה-SM שקיימים ב-GPU שלנו.
את המידע אודות מאפייני ה-GPU נשלוף בעזרת המתודה $cudaGetDeviceProperties()$. המידע אותו נרצה לשלוף הוא: מספר הרגיסטרים הכולל עבור כל SM, נסמן $\max\_regs\_SM$, הזכרון המשותף הכולל עבור כל SM, נסמן $\max\_smem\_SM$, ומספר החוטים הכולל עבור כל SM, נסמן $\max\_threads\_SM$. נרצה גם לדעת מה מספר ה-SM הכולל שיש לנו ב-GPU, נסמן $total\_SM$.
נרצה לבדוק איך כל אחד מהמאפיינים הנ"ל מגביל את מספר ה-TBs שניתן להריץ.
עבור רגיסטרים נחשב: $\frac{\max\_regs\_SM}{regs\_per\_thread \cdot threads\_per\_tb}$, כאשר מספר הרגיסטרים לכל חוט נתון והוא 32, ומספר החוטים לכל בלוק גם כן ידוע.
עבור זיכרון משותף נחשב: $\frac{\max\_smem\_SM}{smem\_per\_block}$, כאשר את הזכרון המשותף לכל בלוק נמצא באמצעות הדגל (nvcc --ptxas-options=-v) בנוסף לזיכרון המשותף שמשתמשת הפונקציה $interpolate$.
עבור חוטים נרצה לבדוק שמספר החוטים המבוקש לא חורג ממספר החוטים האפשריים לבלוק (אם כי אף אחת מהבקשות למספר חוטים בבלוק בתרגיל זה לא חורגת ממספר החוטים האפשרי - 1024). נחשב: $\frac{\max\_threads\_SM}{threads\_per\_tb}$.
מתוך החישובים הנ"ל ניקח את הערך המינימלי שהוא המגבלה עבורנו ונכפיל במספר ה-SM ב-GPU. ובסה"כ נקבל את המספר TBs הכולל שנוכל להריץ.

b) ממומש ב-ex2.cu

c) התורים המחברים בין ה-CPU ל-GPU ממומשים עם $RingBuffer$ שנלמד בתרגול.
מכיוון בכל צד כותב לצד אחד ורק קורא מהצד השני, פעולות ההכנסה וההוצאה מהתור אינם נדרשים להיות אטומיים.
בצד של ה-CPU, ה-CPU הוא היחיד שמכניס ומוציא משני התורים ולכן נוכל לבצע את הפעולות ללא הגנה.
בצד של ה-GPU, מספר חוטים שונים יכולים לנסות להוציא/להכניס לתורים באותו זמן, לכן פעולות אלא הן פעולות קריטיות עליהן נדרש להגן עליהן בעזרת מנעול בסקופ ה-GPU. המנעול ממומש בעזרת $cuda :: atomic$ ו-$exchange$ כפי שראינו בתרגול.

d) עבור 1024 חוטים קיבלנו:

I. בהרצת השרת במצב $Queues$ עם $load = 0$ קיבלנו:

$$throughput = 23564.8 \left[\frac{req}{sec}\right]$$

נסמנו כ-maxLoad.

II. נריץ את השרת כעת עם $load$ משתנה בקפיצות קבועות בטווח: $\left[\frac{maxLoad}{10}, 2 \cdot maxLoad\right]$, נעגל למספר שלם לצורך נוחות.

$$load \in \{2356, 7331, 12306, 17280, 22255, 27230, 32205, 37180, 42154, 47129\}$$

להלן התוצאות המתקבלות מההרצות השונות:

Load $\left[\frac{Req}{sec}\right]$	**Throughput** $\left[\frac{Req}{sec}\right]$	**Latency** $[msec]$				
		Average	Minimum	Median	99th Percent	Maximum
2356	2360	1.2221	0.0415	0.6356	13.5978	22.1500
7331	7322.4	1.6421	0.0427	0.8974	18.8720	32.8370
12306	12267.9	1.1807	0.0454	1.0118	3.3889	7.4612
17280	17214.7	1.2836	0.0432	1.1686	3.5187	8.1081
22255	22136.1	3.2151	0.0413	1.7163	18.4067	25.0771
27230	27008.2	1.4989	0.0414	1.4339	4.5534	11.8838
32205	30541.9	7.2534	0.0441	6.7705	15.1227	21.6213
37180	25235.5	64.7553	0.0495	65.6774	124.1765	129.1646
42154	30887.6	42.6014	0.0438	42.5507	83.2830	89.8744
47129	27993.3	80.0443	1.0236	91.2204	144.8387	150.9760

e) עבור 512 חוטים קיבלנו:

I. בהרצת השרת במצב $Queues$ עם $load = 0$ קיבלנו:

$$throughput = 25834.6 \left[\frac{req}{sec}\right]$$

נסמנו כ-maxLoad.

II. נריץ את השרת כעת עם $load$ משתנה בקפיצות קבועות בטווח: $\left[\frac{maxLoad}{10}, 2 \cdot maxLoad\right]$, נעגל למספר שלם לצורך נוחות.

$$load \in \{2583, 8037, 13491, 18945, 24399, 29853, 35307, 40761, 46215, 51669\}$$

להלן התוצאות המתקבלות מההרצות השונות:

Load $\left[\frac{Req}{sec}\right]$	**Throughput** $\left[\frac{Req}{sec}\right]$	**Latency** $[msec]$				
		Average	Minimum	Median	99th Percent	Maximum
2583	2583	1.2755	0.0469	0.8249	10.0307	22.1209
8037	8020.6	1.2279	0.0490	1.0061	3.5298	6.1595
13491	13359.4	2.2044	0.0471	1.3726	18.9037	32.6071
18945	18723.4	2.7532	0.0477	1.6893	18.8057	35.8692
24399	24262.2	1.7014	0.0481	1.5935	5.4561	10.5460
29853	28093	8.3685	0.0517	7.1907	23.4176	34.7279
35307	31080.3	19.8775	0.0788	20.0585	37.4587	46.3057
40761	23637.6	86.0527	0.0583	85.4288	175.2483	182.5065
46215	30716.9	54.2996	0.3360	53.7820	106.3185	115.9771
51669	28262.9	76.7883	0.8303	72.1633	159.5514	166.1539

f) עבור 256 חוטים קיבלנו:

I. בהרצת השרת במצב $Queues$ עם $load = 0$ קיבלנו:

$$throughput = 29805.6 \left[\frac{req}{sec}\right]$$

נסמנו כ-maxLoad.

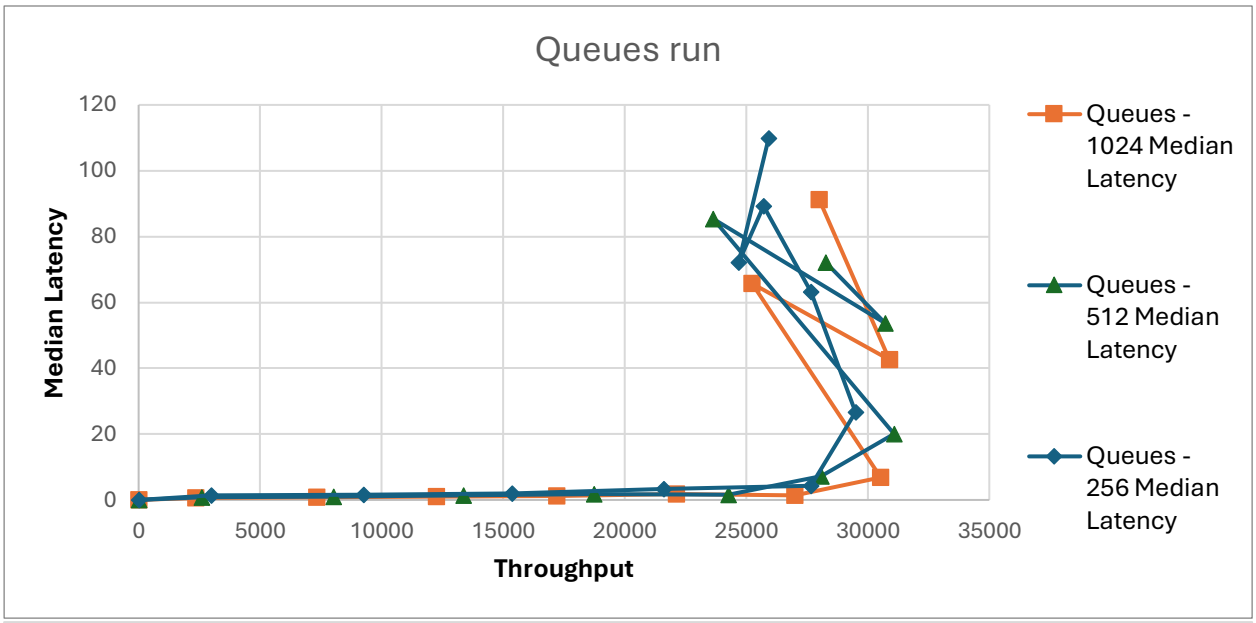
II. נריץ את השרת כעת עם $load$ משתנה בקפיצות קבועות בטווח: $\left[\frac{maxLoad}{10}, 2 \cdot maxLoad\right]$, נעגל למספר שלם לצורך נוחות.

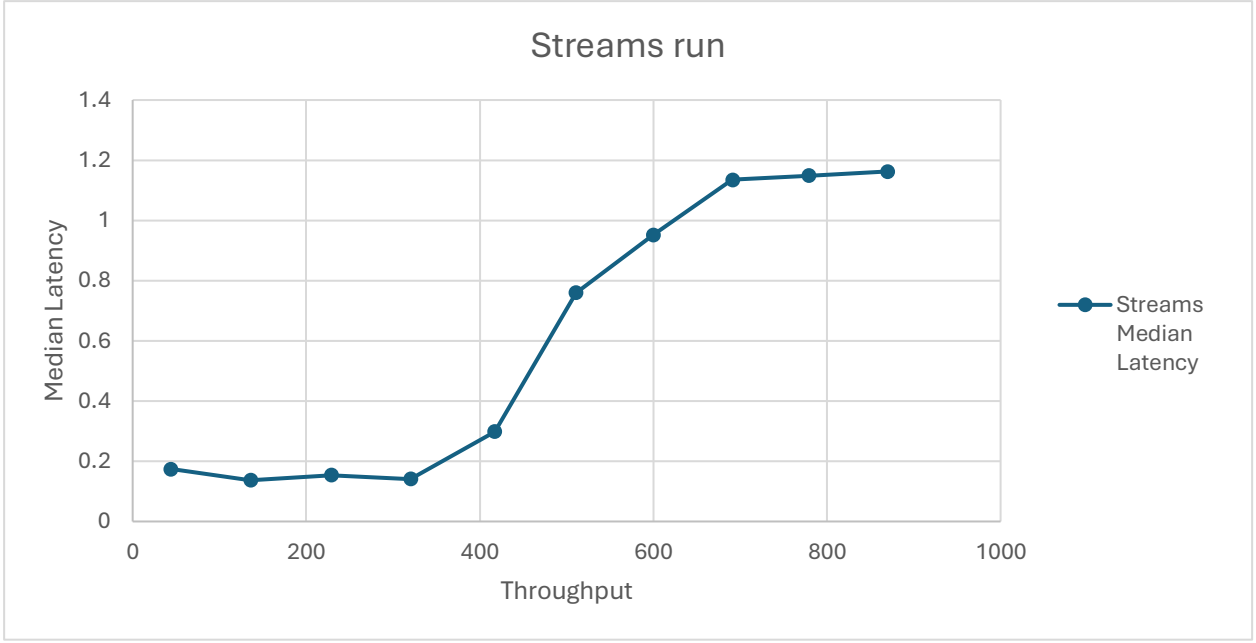
$$load \in \{2980, 9272, 15565, 21857, 28149, 34442, 40734, 47026, 53318, 59611\}$$

להלן התוצאות המתקבלות מההרצות השונות:

Load $\left[\frac{Req}{sec}\right]$	**Throughput** $\left[\frac{Req}{sec}\right]$	**Latency** $[msec]$				
		Average	Minimum	Median	99th Percent	Maximum
2980	2986.7	1.7354	0.0695	1.3006	13.0232	28.4818
9272	9260.9	2.1569	0.0759	1.6722	15.2046	31.0827
15565	15352.1	3.0714	0.0685	1.9927	22.6278	40.7089
21857	21610	6.3231	0.0695	3.3755	25.5577	40.8086
28149	27654.6	5.9098	0.0726	4.3808	20.1226	37.8714
34442	29493.4	25.5279	0.0817	26.7479	48.5357	64.2646
40734	27673.6	61.0745	0.1009	63.1935	115.4722	121.6248
47026	25715	88.7153	0.1009	89.3396	171.5398	186.8136
53318	24687.8	72.0962	0.6006	72.2073	141.9914	219.0242
59611	25915.2	109.6600	1.0582	109.9643	214.0395	224.0965

g) להלן מצורפים הגרפים המציגים את ההשהיה החציונית עבור קצבי התמונות לשנייה כפי שהגדרנו בסעיפים קודמים. כאשר ציר ה-x הוא ה-$Throughput$ וציר ה-y הוא ה-$Median\ Latency$: (שני גרפים שונים בגלל סקלות שונות).





h) ניתן ללמוד מתוך הגרפים שכאשר עוברים את נק' ה-maxLoad עבור כל אחד מתצורות ההפעלה זמני ההשהיה קופצים בצורה דרמתית, והמערכת לא מסוגלת לעמוד בדרישת העומס במצבים אלו.
בנוסף, אם נתמקד באזורים בגרף שבהם המערכת כן עומדת בדרישת העומס, ניתן לראות שעבור מספר רק יותר של חוטים ההשהיה קטנה יותר בממוצע.

i) החלפת התורים מזוג תורים לכל המערכת, לזוג תורים לכל $Threadblock$ יכולה לשפר את ביצועי השרת בהנחה שה-CPU מספיק למלא/לרוקן את התורים בזמן וכן מסוגל לפזר את הבקשות בצורה אחידה בין ה-$Threadblocks$. כעת, מכיוון שמספר $Threadblocks$ עלולים לגשת לתור להוצאה/הכנסה של בקשות בו זמנית, הגישה לתור מוגנת על ידי מנעול. הפרדת התור לתורים פרטיים יאפשר גישה ללא הגנה עם מנעול (בגלל אופן מימוש התור), ועשוי בכך להאיץ את קצב הטיפול בבקשות – אין חסימת גישה.

j) העברת התור $CPU - GPU$ לזכרון ה-GPU עשויה לשפר ביצועים בכך שיהיה חסכון בקריאות מהתור מתוך ה-GPU לזיכרון שנמצא ב-CPU.
הגרעין שרץ על כל $Threadblock$ מנסה בלולאה אינסופית לבדוק אם קיים איבר בתור $CPU - GPU$, ואם כן לשלוף אותו. כלומר מתבצעות מספר גדול מאוד של בקשות גישה לזיכרון הנמצא ב-CPU ובכך מועברות הרבה בקשות זיכרון שעוברות ב-$PCIe$ ולוקחות זמן, לעומת זאת ה-CPU ניגש לתור זה בכל פעם שהוא מוסיף בקשה.
בהעברת התור לזיכרון ה-GPU רוב הבקשות יגיעו מתוך ה-GPU ולא יצטרכו להמשך מתוך זיכרון ה-CPU.

k) על מנת להעביר את התור $CPU - GPU$ לזיכרון של ה-GPU, נצטרך לתת ל-CPU גישה לזיכרון זה.
ניתן לבצע זאת באופן סימטרי לאופן שבו אפשרנו ל-GPU לגשת לזכרון של ה-CPU.
בעזרת $Memory\ Map\ I/O$ נבקש למפות מרחב כתובות זיכרון ב-CPU לכתובות ב-GPU (מרחב הכתובות יהיה מחוץ לטווח הכתובות הפיזיות של ה-CPU). בבקשת גישה לזיכרון שנמצא ב-GPU, הכתובות יעברו תרגום דרך ה-$PCIe$ ונקבל את הכתובת המתאימה ב-GPU.
ב-$NVIDIA$ למשל ניתן לבצע זאת בעזרת טכנולוגיית $GPUDirectRDMA$.