Report

Part a)

Lines 314 to 517 (a total of 203 lines) in cgol.ptx are related to the simple version of the code that uses a row-based index with only global memory access. There are a lot of branches in this section because of the If-statements. Some of the lines with branches are 494, 473, 463, ...

Part b)

For this part I used nvprof using the following command: `nvprof --metrics ipc,inst_executed,cf_executed,ldst_executed,flops_sp,gld_transactions,gst_transactions,gld_throughput,gst_throughput,shared_load_transactions,shared_store_transactions,shared_load_throughput,shared_store_throughput ./cgol s 256 i 100 p 0 u` and without the `u` for the unpotimized version. These are the results.

Simple kernel

Device "GeForce GT 640 (0)" Kernel: play_with_row_based_index(int, int*, int)*

Invocations	Metric Name	Metric Description	Min	Max	Avg
100	ipc	Executed IPC	2.034985	2.068992	2.056328
100	gst_throughput	Global Store Throughput	5.3875GB/s	9.4178GB/s	6.1506GB/s
100	gld_throughput	Global Load Throughput	39.110GB/s	39.667GB/s	39.460GB/s
100	shared_load_transactions	Shared Load Transactions	0	0	0
100	shared_store_transactions	Shared Store Transactions	0	0	0
100	gld_transactions	Global Load Transactions	29108	29108	29108
100	gst_transactions	Global Store Transactions	4282	6066	4909
100	shared_load_throughput	Shared Memory Load Throughput	0.00000B/s	0.00000B/s	0.00000B/s
100	shared_store_throughput	Shared Memory Store Throughput	0.00000B/s	0.00000B/s	0.00000B/s
100	cf_executed	Executed Control-Flow Instruct	114530	118935	116070
100	ldst_executed	Executed Load/Store Instructio	23524	24528	23865

100	flops_sp	FLOPS(Single)	0	0	0
100	inst_executed	Instructions Executed	258662	262063	259861

Optimized kernel (using shared memory)

Device "GeForce GT 640 (0)" Kernel: play_with_shared_memory(int, int*, int)*

Invocations	Metric Name	Metric Description	Min	Max	Avg
100	ipc	Executed IPC	3.079007	3.127835	3.110917
100	gst_throughput	Global Store Throughput	4.4189GB/s	4.4560GB/s	4.4448GB/s
100	gld_throughput	Global Load Throughput	13.222GB/s	13.333GB/s	13.300GB/s
100	shared_load_transactions	Shared Load Transactions	20432	20432	20432
100	shared_store_transactions	Shared Store Transactions	6132	6267	6168
100	gld_transactions	Global Load Transactions	6128	6128	6127
100	gst_transactions	Global Store Transactions	2048	2048	2047
100	shared_load_throughput	Shared Memory Load Throughput	88.171GB/s	88.910GB/s	88.688GB/s
100	shared_store_throughput	Shared Memory Store Throughput	26.528GB/s	27.193GB/s	26.787GB/s
100	cf_executed	Executed Control-Flow Instructions	141098	141463	141267
100	ldst_executed	Executed Load/Store Instructions	34742	34870	34776
100	flops_sp	FLOPS(Single)	0	0	0
100	inst_executed	Instructions Executed	337434	337799	337603

The number of global memory accesses has been dramatically reduced in the optimized version and the amount of shared memory used has increased.

The Number of instructions executed, IPC and number of control instructions are clear from the above tables. The number of single precision floating point operations are 0. Memory accesses are analyzed with more depth. The global load/store throughput and transactions as well as shared memory load/store throughput and transactions have been queried. The total number of load/store instructions can also be seen by the `ldst_executed` variable.

There were no shared memory conflicts in the simple version.

The off-chip memory bandwidth can be seen by the global load/store throughputs.

Part c)

The improved version uses the kernel called `play_with_shared_memory`. This mode is the default mode when you run the program. The unoptimized version can be run using the `u` argument. The following runtime comparison is based on a 256x256 board with 100 iterations.

Simple version:

Total time in kernel = 0.015742 seconds

Optimized version:

Total time in kernel = 0.014867 seconds

Although the optimized version runs faster but it's not very much. The difference is more clear for a 512x512 board and 2000 iterations:

Simple version:

Total time in kernel = 1.229905 seconds

Optimized version:

Total time in kernel = 0.978994 seconds

One extra possible improvement can be to eliminate the IF statements and add paddings to the shared memory tiles that are copied from global memory.