CUPL : A Compile-time Uncoalesced Memory Access Pattern Locator for CUDA

Madhur Amilkanthwar, Shankar Balachandran
Department of Computer Science and Engineering, Indian Institute of Technology, Madras.
{ madhur, shankar } @ cse.iitm.ac.in

ACM SRC, ICS 2013

1. The problem
How to locate uncoalesced memory access patterns in CUDA programs at compile-time?

2. Background
Uncoalesced vs. Coalesced

3. Motivation
Better DRAM bandwidth utilization.
Less memory transactions.

4. CUPL
Input: A valid CUDA kernel + kernel launch configuration.
Output: Warnings if an array is accessed in an uncoalesced manner.

5. An example

Global void kernel(int *A)
{
    for(int i=0;i<32;i++)
        A[32*threadIdx.x + i] *= 10;
}

Optimized code runs 3.5X faster!!

6. Benefits
CUPL has two-fold use:
- It can help the programmer to locate the regions of the code to optimize.
- It can help a compiler to locate an opportunity to perform efficient data layout transformations.

7. Results

8. Future work
- Handle all cases of uncoalesced accesses e.g. cases where difference, D, is not constant.
- Use CUPL as a preprocessor to perform data layout transformation.
- Integrate CUPL in state-of-the-art C to CUDA code translators.

9. References