Parallel Implementation of 0/1 Knapsack Problem

Tejaskumar K
Computer Science and Engineering
National Institute of Technology
Surathkal, Karnataka-575025
Email: tejaskumark11@gmail.com

Tejas R
Computer Science and Engineering
National Institute of Technology
Surathkal, Karnataka-575025.
Email: tejas1908@gmail.com

I. Problem Statement

The project aims at implementing the 0/1 knapsack problem which is a NP- hard problem using dynamic programming in both sequential manner using CPU and parallel manner using CUDA and GPU support.The computational tests will be performed on varying input sizes,and the processing times obtained with the sequential and parallel codes will be compared .

A. Description of 0/1 Knapsack Problem

The Knapsack problem is a combinatorial optimization problem where the objective is to maximize the benefit of objects in a knapsack without exceeding its capacity. Given a knapsack of capacity C and a set of n items, each with a weight w_i and value p_i, objective is to find optimal packing of a knapsack. Optimal packing is the one in which the total weight of items in the knapsack is less than or equal to the capacity C and in which value of items is maximal among other feasible packings.

B. Dynamic Programming

Dynamic Programming is a technique for solving problems whose solutions satisfy recurrence relations with overlapping subproblems. Dynamic Programming solves each of the smaller subproblems only once and records the results in a table rather than solving overlapping subproblems over and over again. The table is then used to obtain a solution to the original problem. The classical dynamic programming approach works bottom-up. To design a dynamic programming algorithm for the 0/1 Knapsack problem, we first need to derive a recurrence relation that expresses a solution to an instance of the knapsack problem in terms of solutions to its smaller instances.

II. Project Execution Plan

A. Implementation details

The Serial execution of the dynamic programming approach of 0/1 Knapsack problem is implemented in C language and run on a CPU, then the parallel execution is done with the NVIDIA CUDA compatible GPU. On CUDA compatible NVIDIA cards, the threads are seperated in blocks and these blocks are distributed on the multiprocessors.A multiprocessor processes one block at a time. When the threads of a block terminate, a new block is launched on the idle multiprocessor.

B. Inputs

The inputs to the serial and parallel codes include arrays w(weights of the items) and p (the corresponding profits),the capacity of the sack C and the no of items(n).

C. Outputs

The outputs are the sequential execution time and parallel execution time for different values of n and the corresponding speedup which is the ratio of the two.

D. Measurement metrics

The computational time is measured in seconds for both the sequential and parallel executions.Speedup value serves as a metric for comparison of the performance of the serial versus the parallel models.

III. Project Timeline

The planned timeline for the project is as follows:
26/10/2018 - Submission of Project proposal
2/11/2018 - Reading Paper and Serial Execution of Knapsack
9/11/2018 - Parallel execution of Knapsack algorithm
12/11/2018 - Comparison of results and documenting into a report
15/11/2018 - Final demonstration of the code

IV. Work Distribution

Serial code execution - Teammate 1
Parallel code execution - Both the teammates
Comparison of results and report making - Teammate 2

References

[1] Vincent Boyer, Didier El Baz,Mousaa Elkihel. Solving Knapsack problems on GPU , Computers and Operations Research,Elsevier, 2012,39 (1), pp.42-47. <10.1016/j.cor.2011.03.014>. <hal-01152223>