Taking Lessons Learned from a Proxy Application to a Full Application for SNAP and PARTISN

Geoff Womeldorff, Joshua Payne, and Ben Bergen

Los Alamos National Laboratory

womeld@lanl.gov, payne@lanl.gov, bergen@lanl.gov

Abstract

SNAP is a proxy application which simulates the computational motion of a neutral particle transport code, PARTISN. In this work, we have adapted parts of SNAP separately; we have re-implemented the iterative shell of SNAP in the task-model runtime Legion, showing an improvement to the original schedule, and we have created multiple Kokkos implementations of the computational kernel of SNAP, displaying similar performance to the native Fortran. We then translate our Kokkos experiments in SNAP to PARTISN, necessitating engineering development, regression testing, and further thought.

Keywords:

Development, regression testing, and further thought.

1 Introduction

One of the paramount tasks in the discipline of scientific computing is in adapting computer codes to new and future computer architectures. This is important in and of itself, but especially in the light of the rapidly changing aspects of modern architectures such as increasingly deep memory hierarchies and exploding core counts, from multi-core to many-core. It is the expectation of many practitioners of scientific computing that due to hardcoded algorithmic choices and optimizations, computer codes that run performantly on current high-performance systems will not on future ones. Thus, if we are to protect the investment that we have in physics codes we use on a daily basis, we must assess how to move them forward into the future.

Of the many challenges in modernizing code, one of the greatest is that the platforms we are wishing to run code performantly on do not exist yet. Thus we turn to research products and libraries that abstract away some of the architecture specific decisions, without compromising performance, or, perhaps, offering a performance penalty for the abstraction that we find acceptable. An additional difficulty is in that the computer codes we are truly interested in optimizing are of the size and age that a refactoring is not a task taken without a clear direction. In order to address this problem, we produce less complex programs which reproduce...
the memory and computational patterns of the programs which spawned them, and we denote these proxy applications. In starting our modernization experiments with proxy apps, we are able to much more rapidly prototype various designs to test them.

Specifically, the path we took to meet this challenge, was two-fold, using Kokkos, an execution model and data management library, and Legion, an asynchronous task-based runtime system. We chose to focus on two aspects of SNAP [4], Los Alamos National Laboratory’s neutral particle transport proxy application, on-node performance and cross-node performance. For the cross-node performance aspect of our study, we re-implemented SNAP in its entirety in Legion, using an abstraction layer, Dragon, that provides a simplified interface to much of the set-up code involved in writing in plain Legion (as opposed to the more recent DSL, Regent, from the Legion developers). When we examined on-node performance, we chose to focus solely on the most representative computational kernel in SNAP, \texttt{dim3\_sweep}, and to reimplement it Kokkos. Since productivity was one of our study’s metrics, we made a series of Kokkos implementations, to mimic the path a user or group of users might take, in modernizing a code using Kokkos.

We felt encouraged by our success with SNAP, and so turned our Kokkos themed on-node efforts towards its parent code PARTISN [1]. As PARTISN is a mature code implemented in FORTRAN, this necessitated language interoperability efforts as we were to ensure that we maintained the numerical behaviour through our changes. These interoperability experiments were quite successful, however, in terms of achieving a performance rise, with a kernel executed on a GPU, we were limited by the expression of concurrency in PARTISN; primarily the interaction between shared and distributed memory parallelism.

The rest of this work is organized as follows: we discuss a small amount of background information on SNAP, Legion, and Kokkos in Sections 2, 3, and 4. We then discuss our experiments and their results, for our Legion implementation in Section 5, and for our Kokkos versions in Section 6. Finally, we discuss our efforts in PARTISN in Section 7, and conclusions in Section 8.

2 SNAP Proxy Application

SNAP serves as a proxy application to model the performance of a modern discrete ordinates neutral particle transport application. SNAP may be considered an update to Sweep3D, intended for hybrid computing architectures. It is modeled from the Los Alamos National Laboratory code PARTISN which solves the linear Boltzmann transport equation (TE), a governing equation for determining the number of neutral particles (e.g., neutrons and gamma rays) in a multi-dimensional phase space. SNAP itself is not a particle transport application; SNAP incorporates no actual physics in its available data, nor does it use numerical operators specifically designed for particle transport. Rather, SNAP mimics the computational workload, memory requirements, and communication patterns of PARTISN. The equation it solves has been composed to use the same number of operations, use the same data layout, and load elements of the arrays in approximately the same order. Although the equation SNAP solves looks similar to the TE, it has no real world relevance.

3 Legion Background

Legion [2] is a data-centric programming model for writing high-performance applications for distributed heterogeneous architectures. Making the programming system aware of the struc-
دریافت فوری متن کامل مقاله

امکان دانلود نسخه تمام متن مقالات انگلیسی
امکان دانلود نسخه ترجمه شده مقالات
پذیرش سفارش ترجمه تخصصی
امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
امکان دانلود رایگان ۲ صفحه اول هر مقاله
امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
دانلود فوری مقاله پس از پرداخت آنلاین
پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات