Repurposing Domain-specific Hardware Accelerators for Sparse and Irregular High-Performance Computation

Luk Burchard, Simula

simula

Introduction

High performance accelerators are becoming more popular in datacenters, with the initial goal of helping with Machine Learning tasks. However, we also want to use them for:

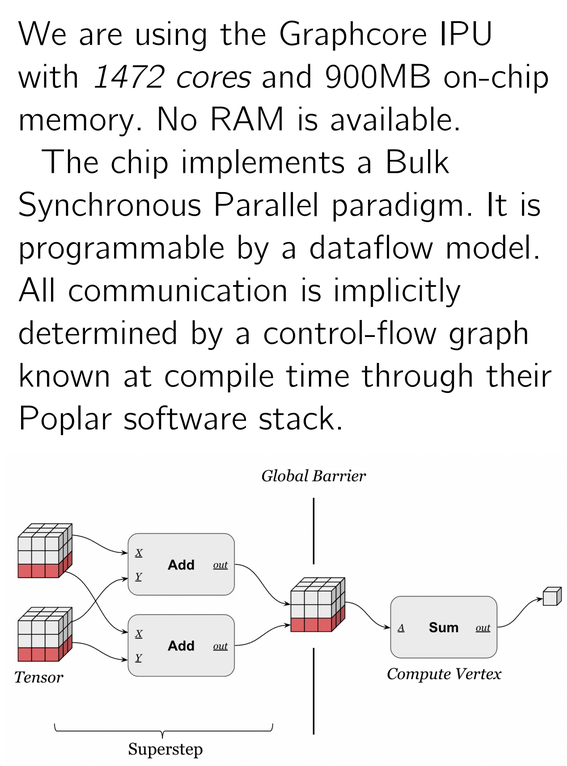
Breadth First Search (BFS)
A fundamental level synchronous graph algorithm which is irregular in its computation starting from a given root node.

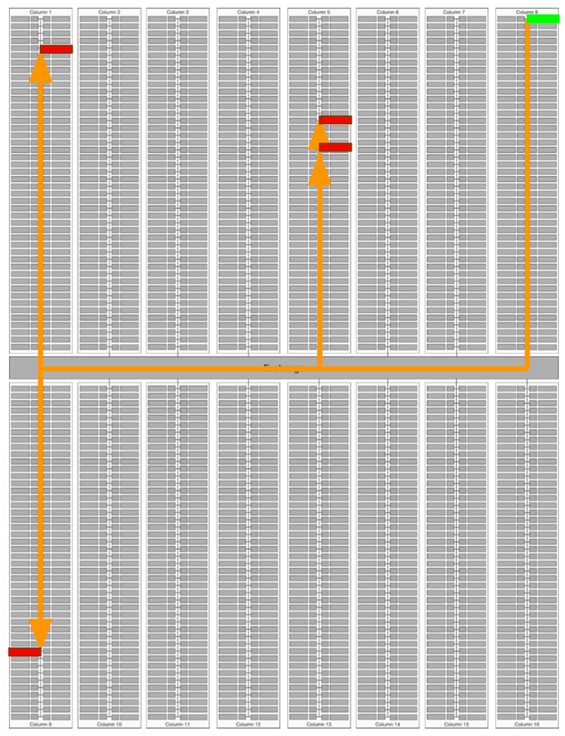
Genomic Sequence Alignment
Dynamic programming problem, which is still faster to solve on CPUs in general applications. GPUs only provide fast special purpose codes.

Cardiac Simulation
Deterministic SpMV operation which can be optimized ahead of runtime. For thousands of processors on a single chip.

Hardware

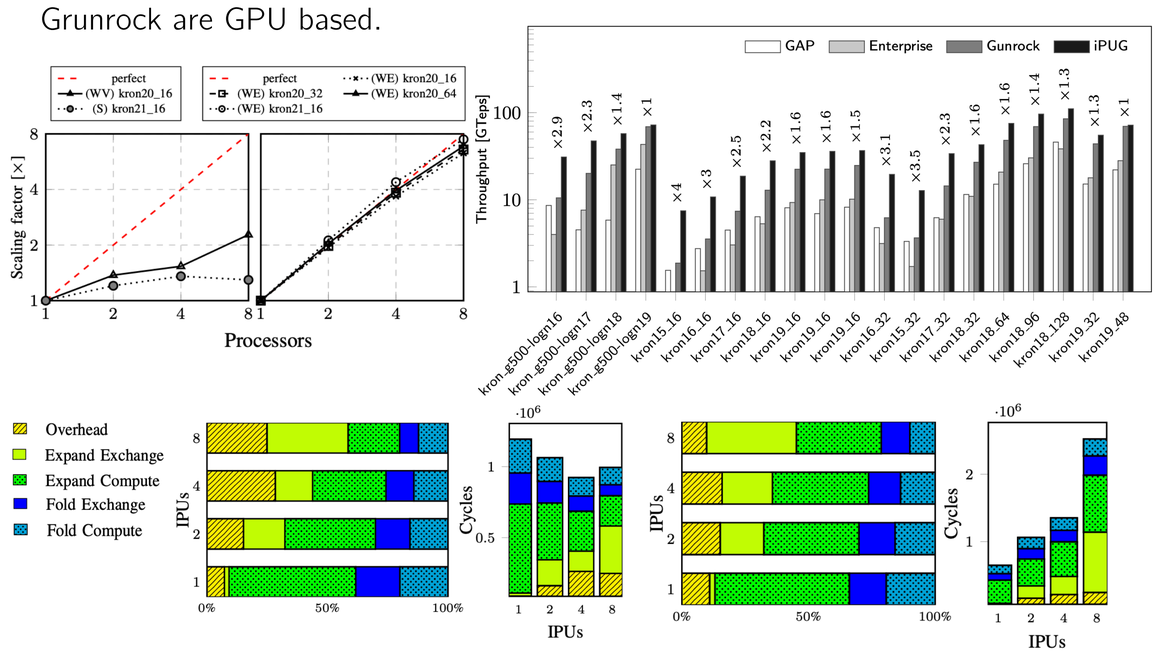
We are using the Graphcore IPU with *1472 cores* and 900MB on-chip memory. No RAM is available.
The chip implements a Bulk Synchronous Parallel paradigm. It is programmable by a dataflow model. All communication is implicitly determined by a control-flow graph known at compile time through their Poplar software stack.





Breadth First Search

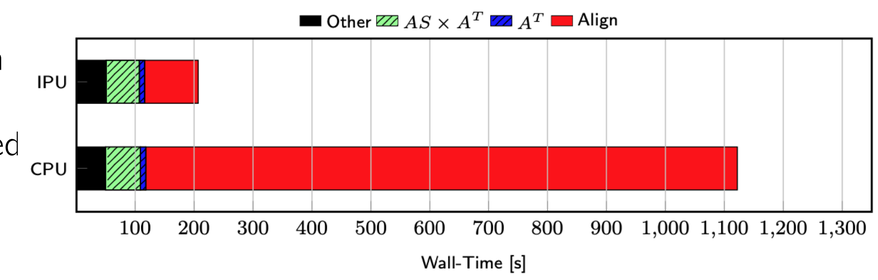
Reformulation to the SpMV approach by Jeremy Kepner and John Gilbert, with A the adjacency matrix and x the frontier $x_{t+1} = A \cdot x_t$. The partitioning of A is 2D on the IPU and 1D across IPUs. ***kron{scale}_{avg(edge)}*** vertex scaled (WV), edges (WE) and strong (S). GAP is CPU based, Enterprise and Grunrock are GPU based.



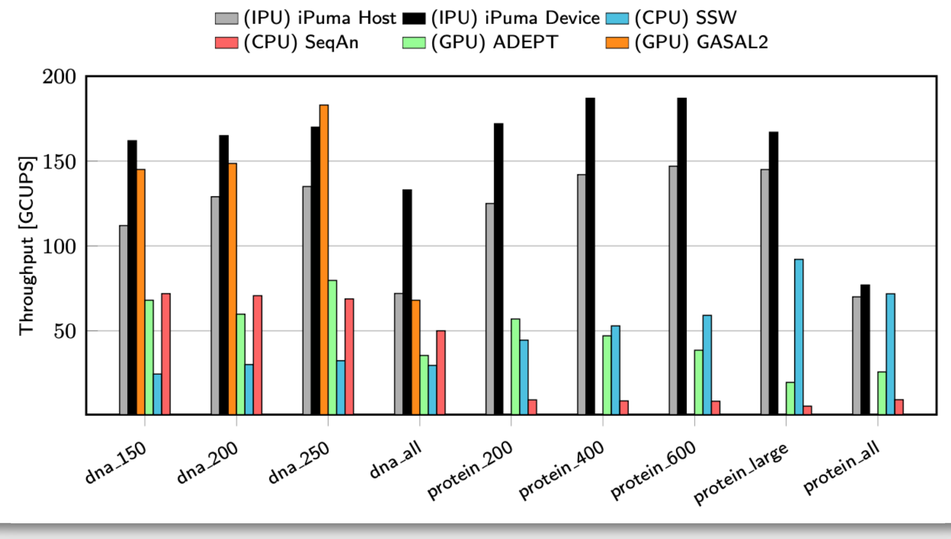
ISC21, iPUG: Accelerating breadth-first graph traversals using manycore Graphcore IPUs
HiPC21, iPUG for Multiple Graphcore IPUs: Optimizing Performance and Scalability of Parallel Breadth-First Search

Sequence Alignment

Smith-Waterman with affine gap panelties was used We are able to process short to medium length sequences in any combination. We are not limited by other factors, and can process inhomogenous inputs fast. iPuma is a drop-in library with upcoming CLI tooling. We are still limited by long to long sequence alignments.



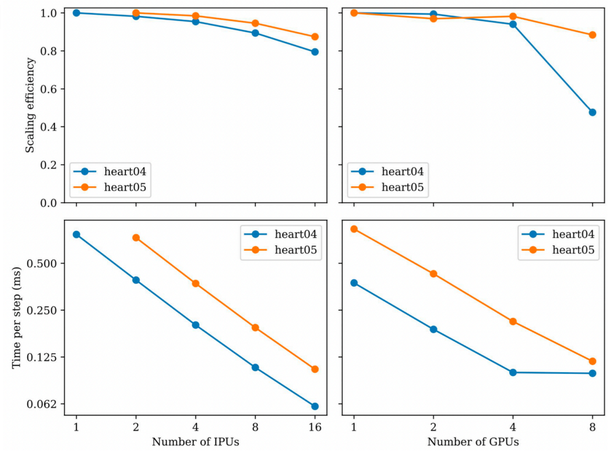
CPUs traditionally perform dominantly well, GPU codes are specialized and are not fast on non-prefiltered input.



Cardiac Simulation

We showed a speedup to both A100 GPUs, using Lynx, a FVM monodomain simulator using 3D unstructured tetrahedral meshes, on the IPU. The PDE step uses less time than the ODE step and is faster than the GPUs. The ODE step is 8x slower and takes more time on the IPU. This is due to the relatively low instruction throughput. The mapping and division to thousands of cores and iregular hardware topologies is a target for optimization.

The PDE phase even with communication benefits from the fast on-chip interconnect and the tile-local memory bandwidth of one random memory load instruction per cycle. The results used a 3M and 7M tetrahedral model.



Conclusion

We showed, the benefits of the IPU are available when the problem size fits onto the chip. Off chip communication and access to a shared memory instance are unlikely parallel or take away a bigger portion of the processor. Whe showed some instances where the computational complexity is higher than external transmission costs, thus this overhead is amortized. It is feasible to use IPUs practically applied in these cases.

We are looking more into:

- repeated SpMV
- combinatorial
- bioinformatic workloads

, such as backtracking and k-mer counting