Enabling Decentralised Machine Learning on RISC-V

Gianluca Mittone, Robert Birke, Iacopo Colonnelli, Marco Aldinucci
Università degli Studi di Torino, Dipartimento di Informatica & CINI HPC Key Tecnologies and Tools lab

Motivations

While tools for Decentralized ML are starting to flourish, many are not flexible and portable enough to experiment with novel systems (e.g., RISC-V), non-fully connected topologies, and asynchronous collaboration schemes. We present a methodology based on the FastFlow parallel programming library, capable of overcoming all these limitations and generating different working DML schemes on two emerging architectures (ARM-v8, RISC-V) and Intel.

Results

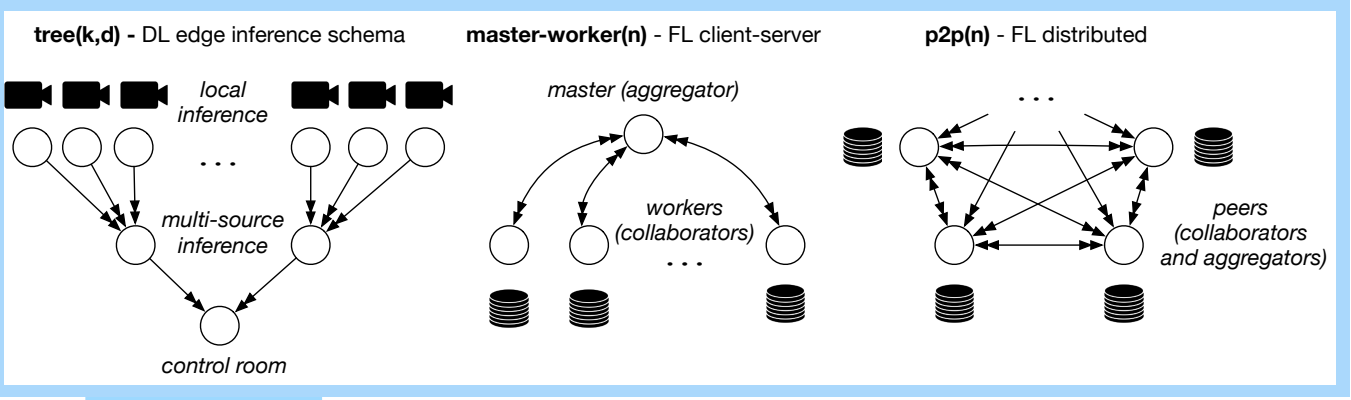
We propose a lightweight middleware for experimenting with FL at the edge. The proposed middleware comprises a run-time (FastFlow C++ header-only library) supporting the pattern-based generation of distributed streaming networks and a methodology to bring ML solutions often developed in Python to a C++ distributed system.

Porting modern FL software on RISC-V

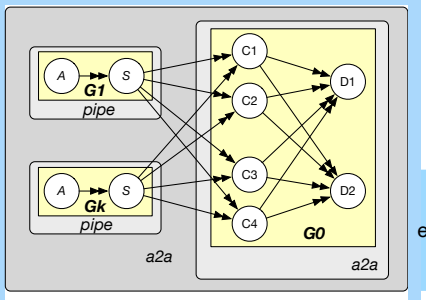
PyTorch (`https://gitlab.di.unito.it/alpha/riscv/torch`)
Some of the PyTorch dependencies are still not compatible with the RISC-V ecosystem. Some of them are mandatory to complete the compilation process (breakpad, SLEEF). Others do not break the compilation process but affect some PyTorch core functionalities (cpuinfo). Others are compatible but not yet optimized.

OpenFL (`https://github.com/alpha-unito/OpenFL-extended`)
OpenFL is an open-source FL framework developed by Intel. Despite officially not supporting RISC-V, we successfully made it work by recompiling ad hoc several Python packages (grpcio, scipy) and the OpenBLAS library.

FastFlow (`https://github.com/fastflow`)
FastFlow, being a vanilla C++20 header-only library, makes it possible to easily experiment with any distributed system having a working C++20 compiler, such as those based on Intel, ARM, or the emerging RISC-V architectures.



Fastflow description

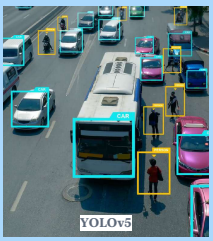


distributed-memory streaming graph with K+1 groups (G0,..., Gk)

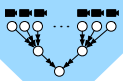
experiment

EPI-TO: heterogeneous cluster: 2 x **RISC-V** 4-core@1.2GHz (U740 SiFive SoC), 4 x **ARM** 80-core@3GHz (Ampere Altra Q80-30) + 2 x NVidia BF-2 DPU and 2 x A100 GPU

Monte Cimone: 8-node **RISC-V** 4-core@1.2GHz (U740 Sifive SoC) HPC compute cluster (processors, main memory, non-volatile storage, and interconnect)



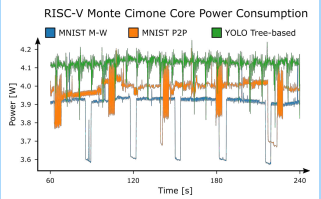
results YOLOv5 inference



	Δ energy/FLOP (CPU only)	energy /FLOP	avg CPU power (idle)	TDP /socket
x86-64 (Intel)	6.3 nJ	12.8 nJ	44 W	125 W
ARM-v8 (Ampere)	0.9 nJ	3.2 nJ	15 W	250 W
RISC-V (SiFive)	1.7 nJ	15.9 nJ	3.4 W	5 W

G. Mittone, N. Tonci, R. Birke, I. Colonnelli, D. Medic, A. Bartolini, R. Esposito, E. Parisi, F. Beneventi, M. Polato, M. Torquati, L. Benini, M. Aldinucci: Experimenting with Emerging ARM and RISC-V Systems for Decentralised Machine Learning. CoRR abs/2302.07946 (2023)

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101034126. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Spain, Germany, France, Italy, Poland, Switzerland, Norway.



YOLOv5 tree-based inference results on a decentralised inference tree (148 frames per leaf)	root + 2 leaves time (s)	root + 2 leaves energy/leaf (J): Δ (tot)	root + 4 leaves time (s)	root + 4 leaves energy/leaf (J): Δ (tot)	root + 7 leaves time (s)	root + 7 leaves energy/leaf (J): Δ (tot)
x86-64 (Intel)	**19.76**	1520 (2389)	**19.38**	1491 (2343)	**19.01**	1462 (2298)
ARM-v8 (Ampere)	37.16	**291 (848)**	39.88	**312 (910)**	43.15	**338 (985)**
RISC-V (SiFive)	1201.51	841 (4926)	1205.77	844 (4943)	1212.77	848 (4972)
Intel-Ampere	35.65	—	35.65	—	36.10	—