



Sandia
National
Laboratories

The ECP Proxy App Project: Highlights and Lessons Learned

Omar Aaziz

Jeanine Cook

PES GM Panel



Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Project Primary Goals



Compose and release a proxy application suite to represent the ECP application complex

- Release on a yearly cadence to track changes in application space
- Suite must be easy to use to ensure community adoption

Assess the fidelity of these proxy applications compared to their respective parents

- Since proxies are used in co-design and acquisition, we must know if and how they are/are not representative of parent behavior

Proxies used in Co-Design



Software development, used to evaluate

- Algorithmic options
- Code optimizations
- Programming models
- Performance portability

Proxies must represent behavior of applications we care about

Proxies must be easy to obtain, build, and execute across platforms

Hardware/system development, used to evaluate

- New architectural and system components
- Algorithms that drive behavior of hardware components

System Acquisition

- Performance/benchmarking

Project Highlights

ECP Proxy App Suite Release 4.0



Currently 64 proxies in ECP Proxy App Catalog

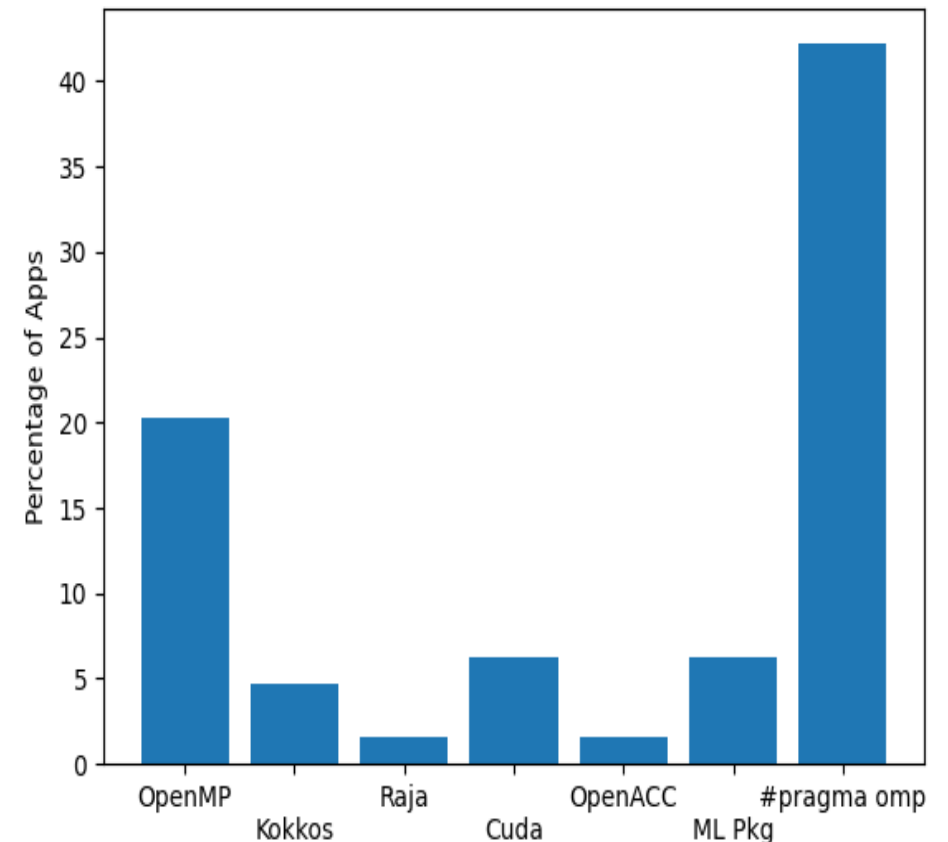
- <https://proxyapps.exascaleproject.org/app/>
- Organized by programming model
 - All programming models of interest to DOE

Implemented as spack packages for easier build and portability

- `spack info ecp-proxy-apps`
- `spack install ecp=proxy-apps@4.0`
- `spack list -tag proxy-app`

Proxy App Quality Standards and Best Practices

- <https://proxyapps.exascaleproject.org/standards/>

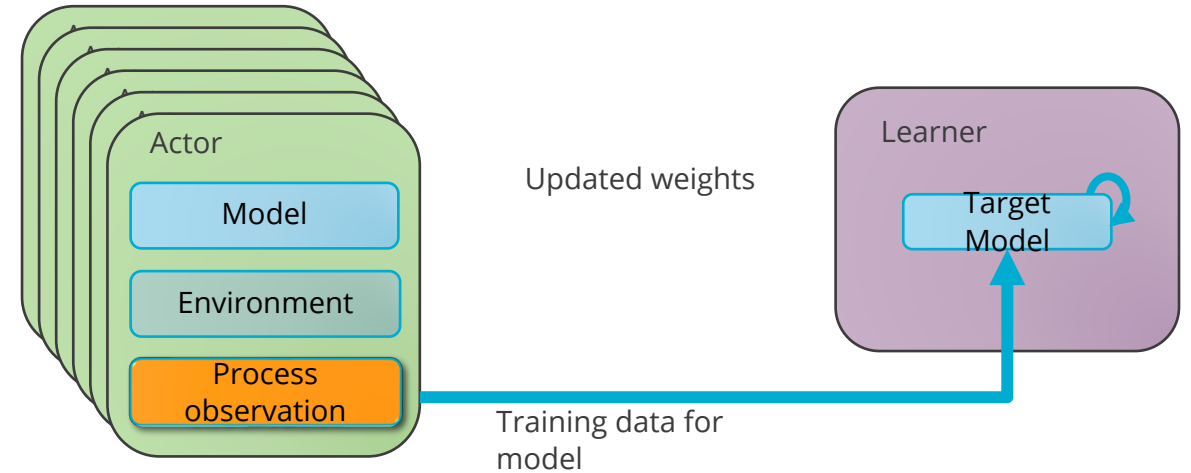


New ML Proxies



miniRL

- Is a reinforcement learning proxy app derived from EXARL
- Captures important communication and computational aspects
- Demonstrated using Deep Q-Network and a synthetic environment called ExaCartpole
- Provides modular agents, environments, and workflows
- minitally
 - Implements tallying system from OpenMC
- Qtensor-mini
 - Tensor contraction simulator



- HyPar
 - Finite-difference framework to solve any general hyperbolic-parabolic PDEs on structured grids
- IMEXLBM
 - Lattice Boltzmann Method proxy app suite for heterogeneous systems

ECP Proxy Apps

POC: ECP Proxy Apps Team (ecp-proxy-apps@llnl.gov)

URL: <https://proxyapps.exascaleproject.org/>

Publications and reports: url

Ways to contribute?

- Submit proxy apps: <https://proxyapps.exascaleproject.org/submit-app/>
- Develop or improve spackage for proxy apps: <https://github.com/spack/spack/>

Proxy Application Fidelity Assessment



Develop method to assess fidelity of proxies that are used in hardware co-design

- Only use proxies that are designed to represent **computational/memory**/communication behavior of parent applications

Key insights

- An application generates a “fingerprint” during execution on a particular system
 - Reflects how the application behaves in response to system constraints
- Similar application fingerprints mean application responds similarly to particular design constraint and to changes in that particular constraint
 - Expect codes with similar dependence/bottleneck on memory bandwidth to derive similar benefit from memory bandwidth improvement
- Fingerprints must be easy and fast to collect

Method

- Evaluated several ML Similarity techniques
- Fingerprint is 500-700 (based on platform) performance counter events
- Used an average of (take from SC paper)

Experimental Platform



Application fingerprint from two different platforms (IBM & Intel)

- ~700 IBM events; ~500 Intel events
 - Can compare proxy/parent on single platform
 - Have to be careful across platforms
 - Cross-platform events can be very different → group so that only similar events are compared

Used 8 proxy/parent pairs and 5 others that were either proxies or applications (not paired)

Run configuration fixed

- 4 nodes, 128 ranks

Normalize input/problem to consume ~50% of memory

~4500 total runs

- 5 for each application x # groups
- Average over 5 runs, then over ranks → single value for each event

Use LDMS and PAPI sampler

- Sample rate: 1 sec

Apply similarity

- LIST the techniques we used in SC paper

Entire Fingerprint



IBM P9

ExaMiniMD	0	19	26	24	19	19	47	22	21	22	18	21	50	18	19	20	20	16	15	17	19
LAMMPS	19	0	14	10	19	11	55	24	16	13	24	26	61	17	24	23	15	14	12	19	10
sw4lite	26	14	0	7	25	17	52	21	17	11	24	26	59	24	23	22	25	23	17	17	18
sw4	24	10	7	0	22	16	54	22	16	13	24	26	61	21	23	23	21	19	15	18	14
SWFFT	19	19	25	22	0	20	45	19	21	24	16	17	52	8	17	16	14	14	8	18	13
HACC	19	11	17	16	20	0	51	23	16	15	23	26	57	18	23	23	17	16	14	18	15
MiniQMC	47	55	52	54	45	51	0	37	49	54	37	36	19	46	39	39	54	49	46	41	51
QMCPack	22	24	21	22	19	23	37	0	21	24	8	10	45	22	13	12	28	21	16	12	20
miniVite	21	16	17	16	21	16	49	21	0	16	20	24	54	19	20	19	23	17	14	14	17
vite	22	13	11	13	24	15	54	24	16	0	24	27	59	24	22	22	23	21	17	17	19
Nekbone	18	24	24	24	16	23	37	9	20	24	0	0	42	18	10	9	25	19	14	11	19
Nek5000	21	26	26	26	17	26	36	10	24	27	8	0	43	20	12	12	27	21	16	15	21
XSbench	50	61	59	61	52	57	19	45	54	59	42	43	0	53	44	44	60	55	53	47	58
openmc	18	17	24	21	8	18	46	22	19	24	18	20	53	0	19	18	12	11	8	18	12
picsarlite	19	24	23	23	17	23	39	13	20	22	10	12	44	19	0	0	25	21	15	12	21
picsar	20	23	22	23	16	23	39	12	19	22	8	12	44	18	3	0	25	20	14	11	21
amg2013	20	15	25	21	14	17	54	28	23	23	25	27	60	12	25	25	0	15	14	25	13
Castro	16	14	23	19	14	16	49	21	17	21	19	21	55	11	21	20	15	0	10	17	11
Laghos	15	12	17	15	8	14	46	16	14	17	14	16	53	8	15	14	14	10	8	12	8
pennant	17	19	17	18	18	18	41	12	14	17	11	15	47	18	12	11	25	17	12	8	18
snap	19	10	18	14	13	15	51	20	17	19	19	21	58	12	21	21	13	11	8	18	8

Intel SKX

ExaMiniMD	0	3	15	22	31	14	84	79	35	35	56	54	72	17	46	46	15	23	27	29	17
LAMMPS	3	0	15	23	32	14	85	79	34	35	56	54	72	17	46	46	16	23	28	29	18
sw4lite	15	15	0	13	24	13	77	71	25	25	46	44	65	13	38	37	12	14	16	22	14
sw4	22	23	13	0	17	16	66	61	21	21	36	34	57	16	29	29	17	12	10	11	12
SWFFT	31	32	24	17	0	21	62	58	30	30	36	34	49	19	24	25	21	21	17	20	24
HACC	14	14	13	16	21	0	76	71	29	29	47	45	64	8	37	37	8	16	19	20	16
MiniQMC	84	85	77	66	62	76	0	11	63	63	36	37	29	74	42	42	77	68	62	61	71
QMCPack	79	79	71	61	58	71	11	0	59	58	32	33	33	70	38	38	72	63	57	57	66
miniVite	35	34	25	21	30	29	63	59	0	1	30	28	58	27	30	30	27	14	16	19	21
vite	35	35	25	21	30	29	63	58	1	0	30	28	58	28	30	29	27	14	17	19	22
Nekbone	56	56	46	36	36	47	36	32	30	30	0	0	36	45	20	19	46	35	30	31	41
Nek5000	54	54	44	34	34	45	37	33	28	28	3	0	38	43	19	19	45	34	29	29	39
XSbench	72	72	65	57	49	64	29	33	58	58	36	38	0	60	32	32	64	59	52	54	62
openmc	17	17	13	16	19	8	74	70	27	28	45	43	60	0	35	34	8	15	17	22	16
picsarlite	46	46	38	29	24	37	42	38	30	30	20	19	32	35	0	0	38	30	23	27	34
picsar	46	46	37	29	25	37	42	38	30	29	19	19	32	34	3	0	37	29	23	26	33
amg2013	15	16	12	17	21	8	77	72	27	27	46	45	64	8	38	37	0	15	18	23	16
Castro	23	23	14	12	21	16	68	63	14	14	35	34	59	15	30	29	15	0	8	14	13
Laghos	27	28	16	10	17	19	62	57	16	17	30	29	52	17	23	23	18	8	0	13	15
pennant	29	29	22	11	20	20	61	57	19	19	31	29	54	22	27	26	23	14	13	8	17
snap	17	18	14	12	24	16	71	66	21	22	41	39	62	16	34	33	16	13	15	17	8

Assessment Summary



ML methods can be used to understand proxy app fidelity

- Node and memory, and communication

COS on-going work

- Re-factoring event groupings to be more accurate within a single platform and to enable more accurate cross-platform comparisons
- Developing a method to determine ground truth
 - Unsupervised learning, ~recommendation system
- Examining ML techniques to reduce data collection
- Developing infrastructure to collect application fingerprints on ARM platforms

I/O assessment work

- Evaluation of MacSio (configurable to match various communication patterns)
- Explore the viability of creating a model to predict certain I/O behaviors of AMR-based applications based on combinations of input parameters and system variability
- Use this model to create a tool that will generate MACSio parameters from an AMReX input file (with placeholders for missing capabilities)

Big Lessons Learned

Assessing proxy fidelity is feasible using ML techniques and performance measurement

- Feedback must be given to proxy developers when proxy fidelity is low
 - Sometimes they haven't updated the proxy!

Curating a proxy app suite is more difficult than it sounds, but possible!

- In spite of 100's of proxies that exist, possible to
 - Organize, document, catalog for easy access
 - Spackify for ease of porting

Developers should understand motivation and use of proxy

- What is the proxy going to be used for?
 - Software optimization? Hardware co-design? Programming model exploration?
- What is it intended to represent?
 - Algorithm? Computational behavior? Programming model?

Proxy documentation must state intended use

Proxies must keep up with application space

Thanks!



The End