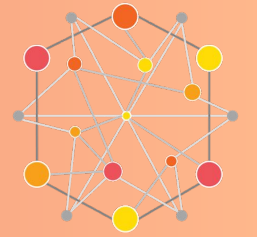
ExaWorks
https://exaworks.org

Tutorial: Developing Robust and Scalable Next Generation Workflows Applications and Systems

ISC-HPC 2022



Lawrence Livermore National Laboratory
Argonne NATIONAL LABORATORY
Brookhaven National Laboratory
OAK RIDGE National Laboratory
U.S. DEPARTMENT OF ENERGY | Office of Science

Swift/T

http://swift-lang.org/Swift-T

Swift/T: Enabling high-performance scripted workflows

Write site-independent scripts, translates to MPI

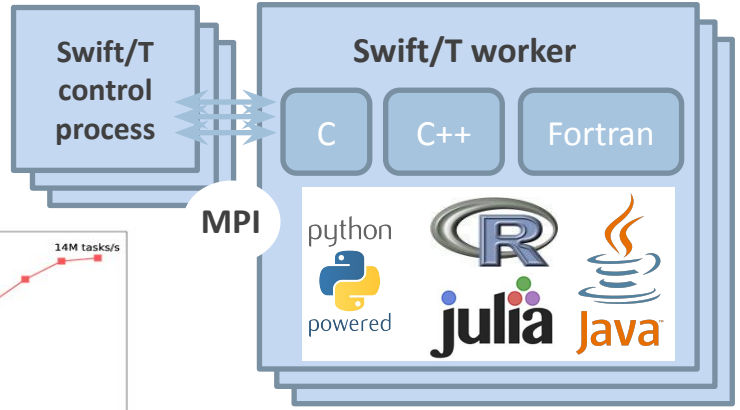
Automatic task parallelization and data movement

Invoke native code, script fragments in Python and R

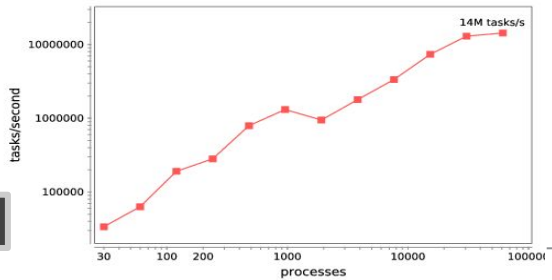
Rapidly subdivide large partitions for MPI jobs in multiple ways (MPI 3.0)

SWIFT/T

2018 R&D 100 WINNER



64K cores of Blue Waters
2 billion Python tasks
14 million Pythons/s




```
$ spack install stc
```

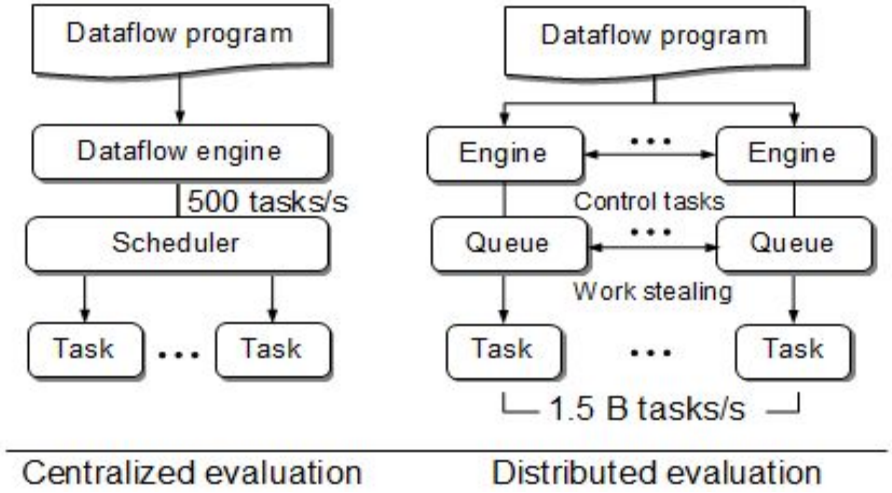
```
$ conda install -c lightsource2-tag swift-t
```

Swift/T: Scalable data flow programming for distributed-memory task-parallel applications
Proc. CCGrid 2013.

Centralized evaluation is a bottleneck at extreme scales

Had this (Swift/K):

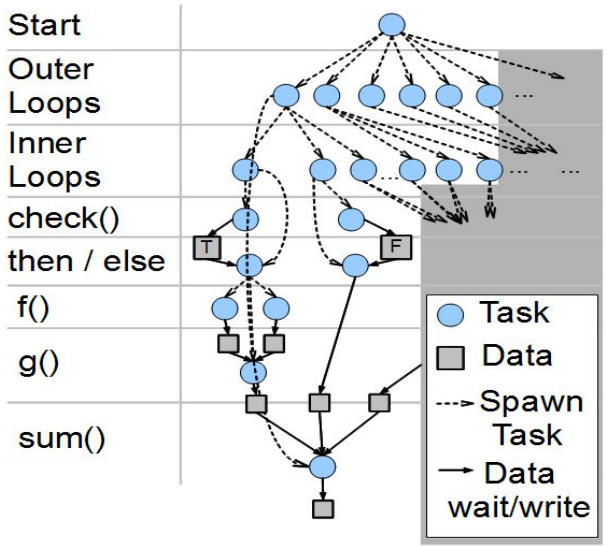
Now have this (Swift/T):



Turbine: A distributed-memory dataflow engine for high performance many-task applications. Fundamenta Informaticae 28(3), 2013

Swift/T: Fully parallel evaluation of complex scripts

```
int X = 100, Y = 100;
int A[][];
int B[];
foreach x in [0:X-1] {
  foreach y in [0:Y-1] {
    if (check(x, y)) {
      A[x][y] = g(f(x), f(y));
    } else {
      A[x][y] = 0;
    }
  }
  B[x] = sum(A[x]);
}
```

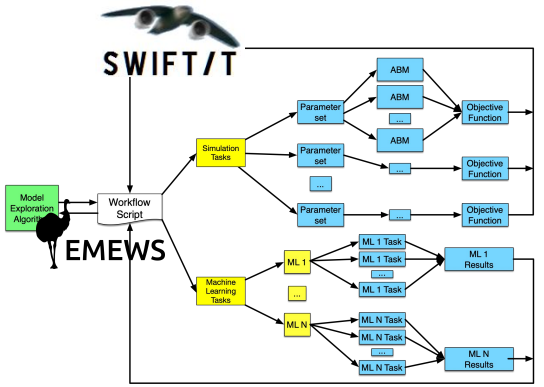


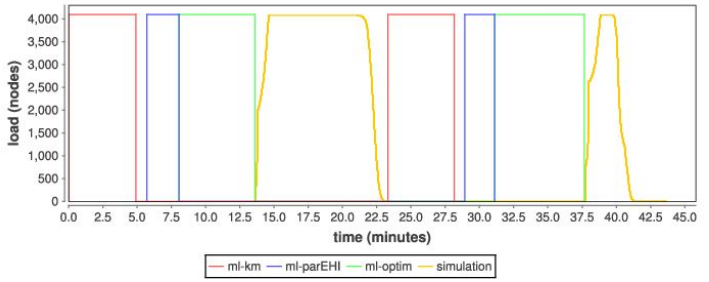
Compiler techniques for massively scalable implicit task parallelism.
SC 2014.

City-scale COVID-19 epidemic modeling

ACM Gordon Bell Special Prize
for High Performance Computing-Based COVID-19 Research
Finalist
2020

- **Observed city data**
  - Hospitalizations, cumulative deaths from Chicago Department of Public Health
  - Detailed line list data from Illinois Department of Public Health
- **ML Phases:**
  - Distribution of R code snippets via R `foreach %dopar%` syntax; work distributed by EMEWS
- **Simulation:**
  - Distribution of MPI tasks via Swift/T `@par` syntax.
    MPI communicators are dynamically allocated over in-order cores by Swift/T using `MPI_Comm_create_group()`
- **Key performance metric:** Scalability and Time to Solution
  - Keep the cores busy in presence of changing task types and workflow dynamics
- **Simulation phase** starts 1024 MPI tasks, each on a 256-rank MPI communicator
  - Assigns tasks at rate of 4,695 core-tasks per second, 73 communicators per second
- **The ML phases** are single-node, vendor-optimized R calls



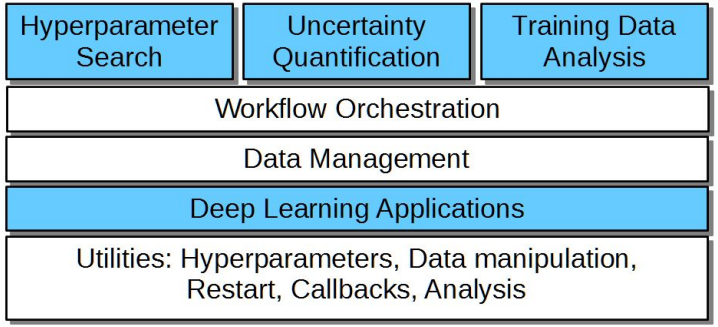


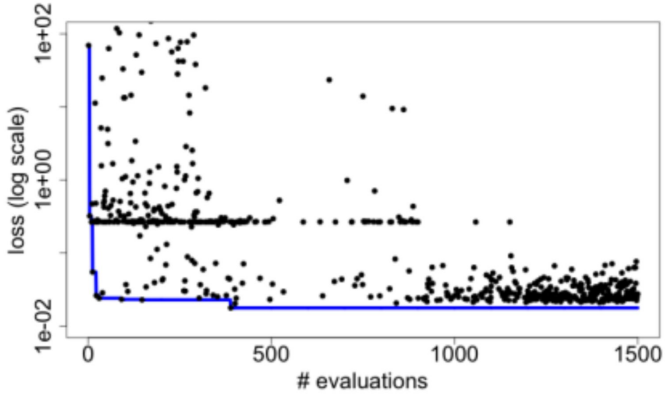
A population data-driven workflow for COVID-19 modeling and learning. IJHPCA 2021.

ECP CANDLE Hyperparameter optimization

- Goal: Develop an exascale deep learning environment for cancer, enabling the most challenging deep learning problems in cancer research to run on the most capable supercomputers
- Neural networks have a large number of possible configuration parameters, called hyperparameters
- CANDLE/Supervisor consists of several high-level workflows
- Capable of modifying/controlling application parameters dynamically as the workflow progresses and training runs complete
- Distribute work across large computing infrastructure, manage progress
- Underlying applications are Python programs that use Keras/TensorFlow

  - Hyperparameter search plot:
  - Search trajectory of mlrMBO (R model-based optimization) algorithm
  - Each iteration does 300 evaluations (batch size)
  - Minimum and average performance on validation data set decreases as the ME algorithm learns



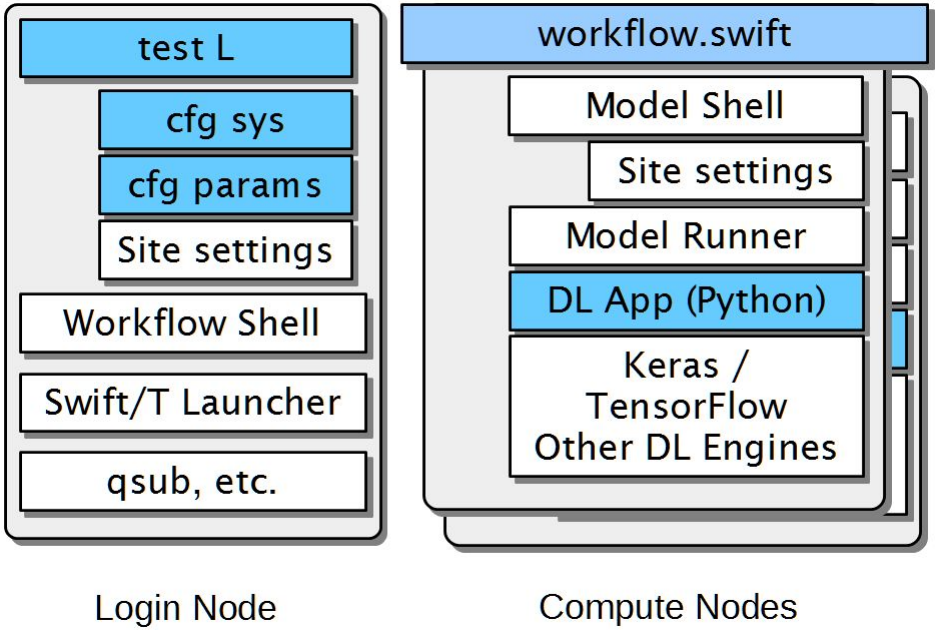


CANDLE/Supervisor: A workflow framework for machine learning applied to cancer research. BMC Bioinform. 2018.

CANDLE/Supervisor Implementation

Script schematic

- Runs start with a test script
- CFG scripts contain settings for a system or parameters for a given study (e.g., search space)
- Reusable site settings
- The workflow shell script sets up the run
- Swift/T launches and manages the workflow
- Reusable Model scripts set up each app run
- The DL app uses Keras/TF plus CANDLE Python utilities



Exercises

https://github.com/ExaWorks/Tutorial