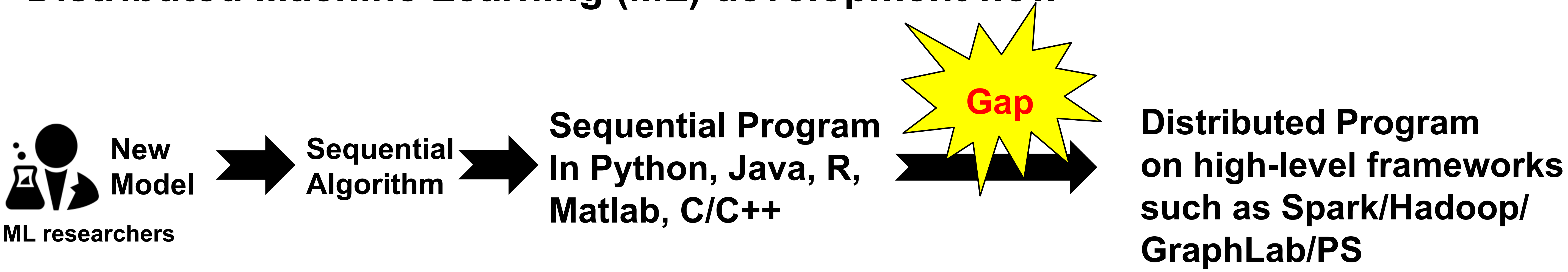
STRADS-AP: Simplifying Distributed Machine Learning Programming without Introducing a New Programming Model

Jin Kyu Kim[1], Abutalib Aghayev[1], Garth A. Gibson[1,2,3], Eric P. Xing[1,4]

[1]Carnegie Mellon University, [2]Vector Institute,
[3]University of Toronto ,[4]Petuum Inc.

Distributed ML Programming is Difficult

Distributed Machine Learning (ML) development flow



ML researchers

New Model

Sequential Algorithm

Sequential Program In Python, Java, R, Matlab, C/C++

Gap

Distributed Program on high-level frameworks such as Spark/Hadoop/GraphLab/PS

The cost of using high-level frameworks

- Mold a sequential ML program to a framework-specific programming model
- Change data structure design and computation routine
- Often deliver suboptimal performance

STRADS-AP aims to simplify conversion of sequential ML program into a distributed ML program almost mechanically

Easy Conversion of Seq. ML into Dist. ML

```
// part1: pretraining
declare data structure D for input
declare data structure P for parameters
```

Sequential data structures (i.e. map, vector) for input data and model parameters
→ Challenge1: scalability limit

```
// part2: training
for(iter=0; iter<MAX; iter++)
  for(i=0; i<N; i++){
    read a part of input D and parameter P
    write to a part of parameters P
  }
```

Source of parallelism: repetitive, static control flow, reorderable
→ Challenge2: data dependencies among loop bodies

Structure pattern of targeting ML programs

Easy Conversion of Seq. ML into Dist. ML

```
// part1: pretraining
declare data structure D for input
declare data structure P for parameters
```

Sequential data structures (i.e. map, vector) for input data and model parameters
→ Challenge1: scalability limit

```
// part2: training
for(iter=0; iter<MAX; iter++)
  for(i=0; i<N; i++){
    read a part of input D and parameter P
    write to a part of parameters P
  }
```

Source of parallelism: repetitive, static control flow, reorderable
→ Challenge2: data dependencies among loop bodies

Structure pattern of targeting ML programs

⇩

```
declare distributed data structures D
declare distributed data structures P
for(iter=0; iter<MAX; iter++)
  ParallelFor(N,[D,P](int i){
    read a part of input D and parameter P
    write to a part of parameters P
  })
```

STRADS-AP Distributed Program

Solution1: distributed data structures (i.e. dmap, dvector)
→ allows index based random R/W access from any node

Solution2: parallel loops (Sync/AsyncFor)
→ parallelize loop bodies with different consistency level

STRADS-AP Workflow

```
stradsap::dvector<T1> D;
stradsap::dmap<T2> P,Q;
float alpha(0.1);
for(i=0; i<N; i++){
  stradsap::parallel_for
   (N, [i, alpha, &D, &P, &Q](int j){
     - optimization routine
     - read i,j, alpha, elements of D
     - read/write elements of P,Q
  }, stradsap::ConsistencyModel);
  alpha *= 0.99;
}
```

STRADS-AP code

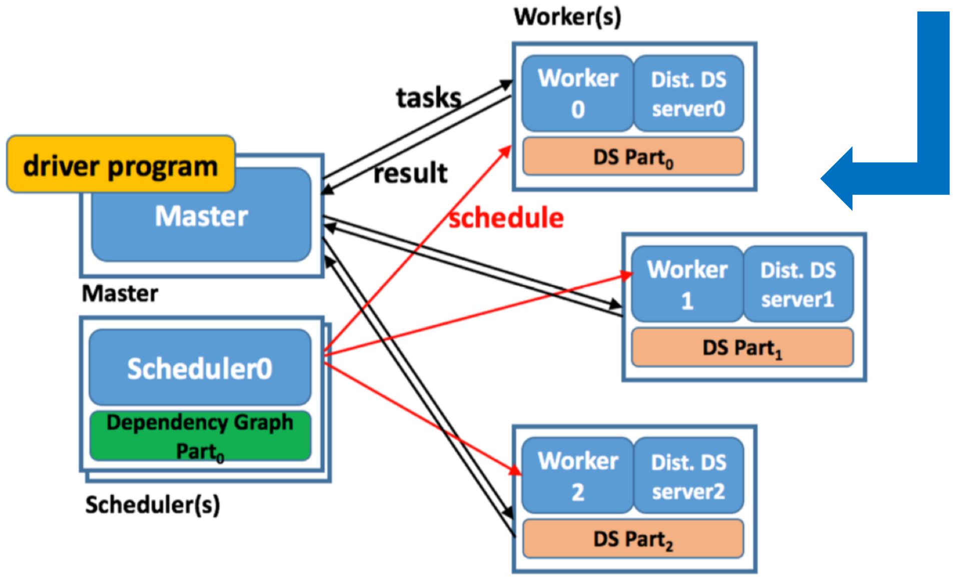
fill the lack of C++ language's reflection capability

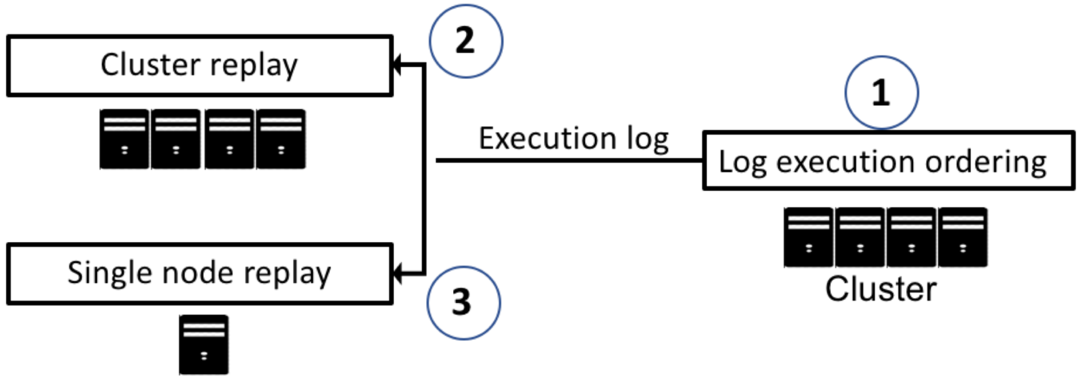
STRADS-AP preprocessor

Add Language specific augmentations

Native compiler

Binary code





STRADS-AP debugging

STRADS-AP runtime

Presentation schedule

STRADS-AP presentation at 3pm Wed July 10 in Track II