PaddlePaddle Fluid

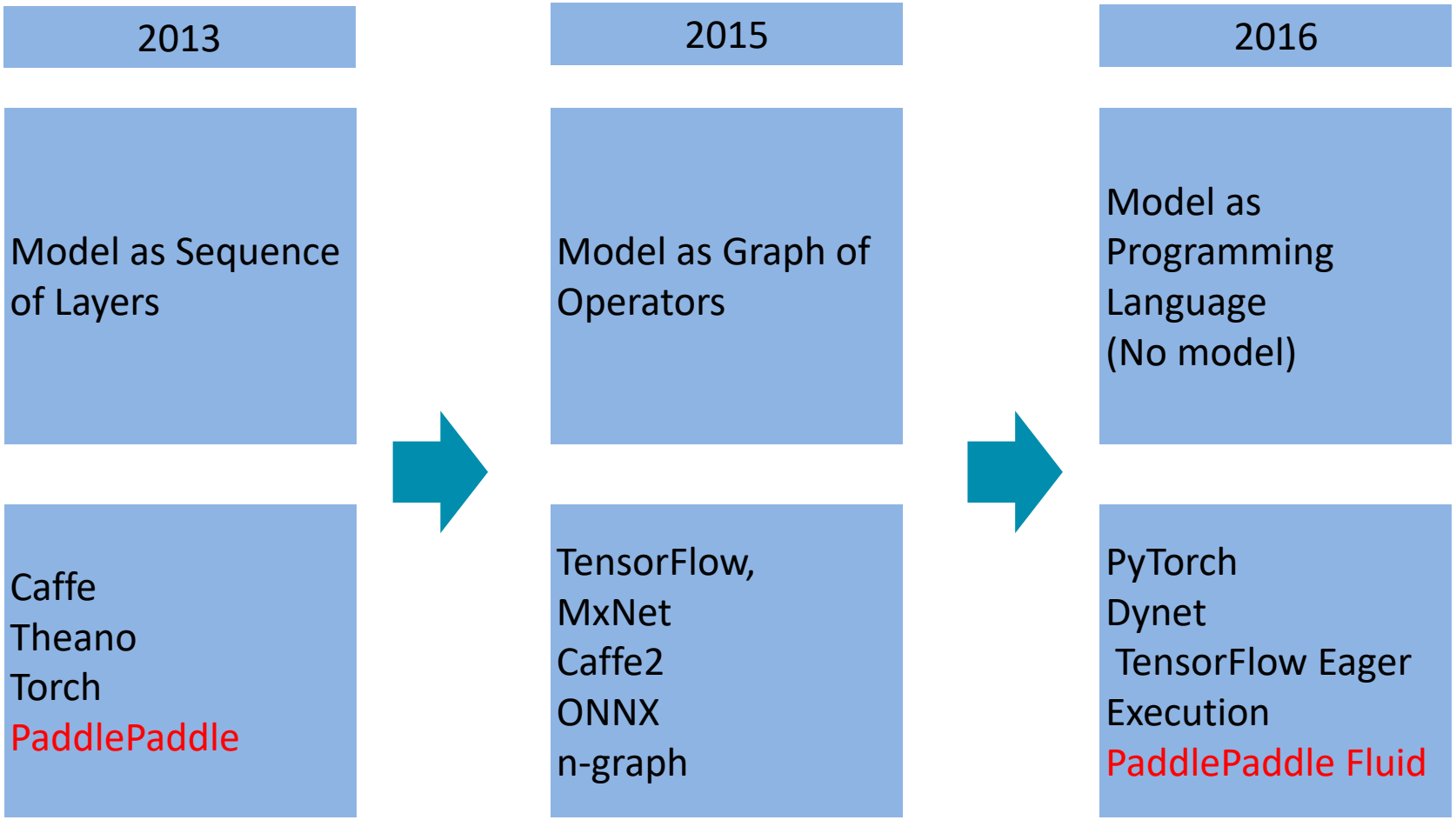
Towards a Deep Learning Programming Language

Qiaolongfei@Baidu

Contents

- The Evolution of Deep Learning Systems
- Why Fluid
- Design and Implement
- Towards a Compiled Programming Language

Evolution of Deep Learning Systems



Evolution of Deep Learning Systems

- The expression ability is more and more flexible
  - 1. Sequence of Layer
    - Symbolic Programming paradigm(define and run)
    - Suitable for CNN
  - 2. Graph of Operators
    - Symbolic Programming paradigm(define and run)
    - Suitable for RNN
  - 3. Imperative programming
    - Define by run
    - Can describe arbitrary models

Symbolic Programming

```
x = layer.data("image")
l = layer.data("label")
f = layer.fc(x, W)
s = layer.softmax(f)
c = layer.mse(l, s)
```

describes the model

```
for i in xrange(1000): # train for 1000 iterations
    m = read_minibatch()
    forward({input=x, data=m}, minimize=c)
    backward(...)

print W # print the trained model parameters.
```

describes the training process

- Problem :
  - hard to debug
  - Hard to re-iterate fast on a program

Imperative programming

```
W = tensor(...)

for i in xrange(1000): # train for 1000 iterations
    m = read_minibatch()
    x = m["image"]
    l = m["label"]
    f = layer.fc(x, W)
    s = layer.softmax(f)
    c = layer.mse(l, s)
    backward()

print W # print the trained model parameters.
```

Define by run:

model configuration is in the training loop.

- Advantage:
  - Easy to debug
  - Can use the control-flow operators of host language
  - More flexible to express training process

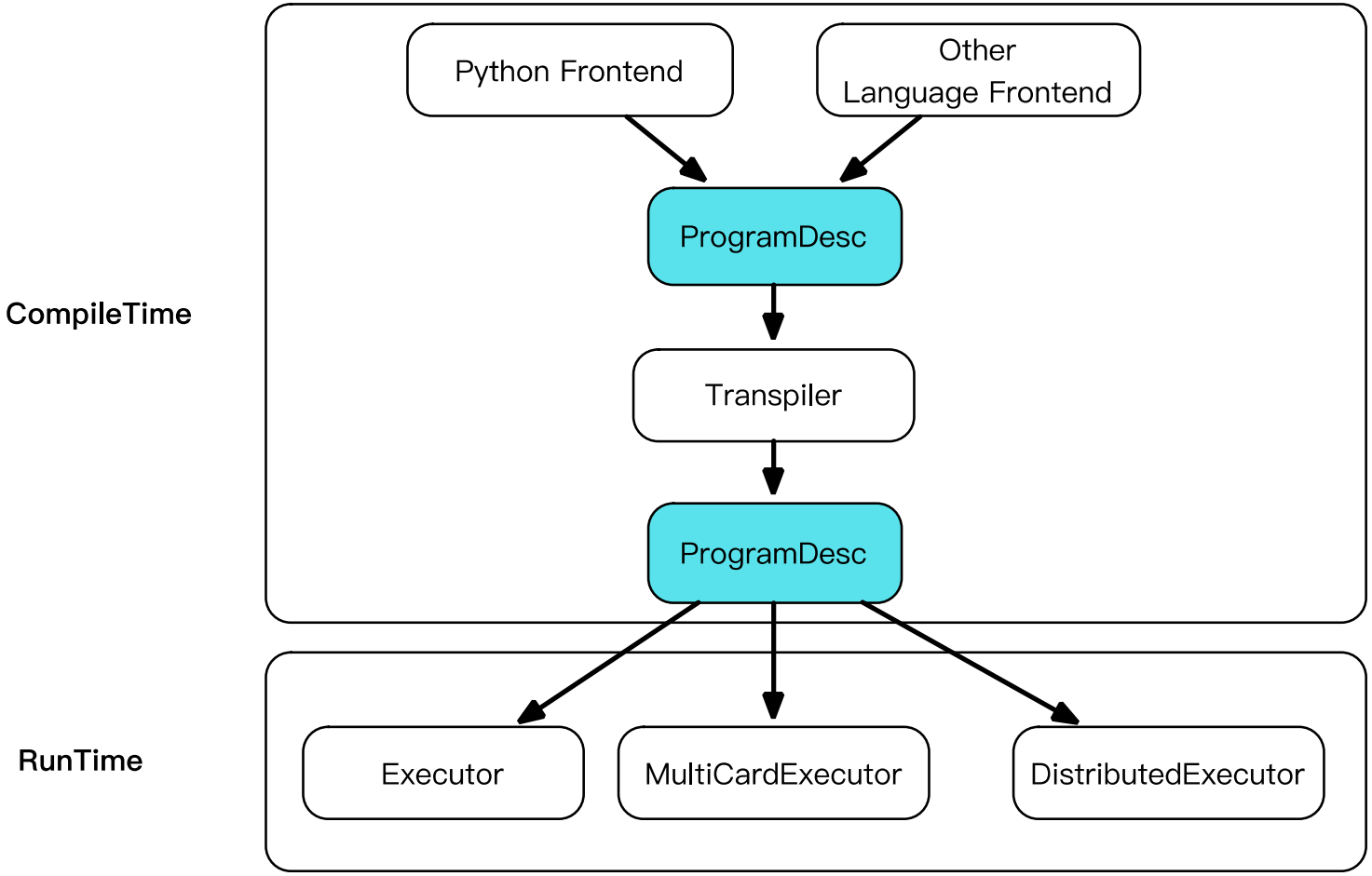
Why Fluid

- Fluid is similar to PyTorch, which describes the "process" of training or inference.
  - This brings Fluid the flexibility to define different non-standard models that haven't been invented yet.

- We are trying to push Fluid towards the directions of a compiler and a new programming language for deep learning.

General Compiler

The Design of Fluid



Seperated CompileTime and RunTime

- CompileTime
  - Describe Process of Trainer
  - The compile result is ProgramDesc
  - ProgramDesc is like AST in programming language
  - Use trasnpilers to optimize ProgramDesc
- RunTime
  - Executor will optimize and execute ProgramDesc during runtime

Turing Completeness

- In computability theory, a system of data-manipulation rules, such as a programming language, is said to be Turing complete if it can be used to simulate any Turing machine.

- For a programming language, if it provides ***if-then-else and loop***, it is Turing complete.

Nested Block

- PaddlePaddle-Fluid describe a deep learning application with nested blocks, not Graph of Operators

Programming Language	PaddlePaddle Fluid
For/while	WhileOp
If-Else, Switch	IfElseOp
Execute Instructions in order	Run Operators in order

RNN / Loop

```
x = sequence([10, 20, 30]) # shape=[None, 1]
m = var(0) # shape=[1]
W = var(0.314, param=true) # shape=[1]
U = var(0.375, param=true) # shape=[1]

rnn = pd.rnn()
with rnn.step():
  x_ = rnn.step_input(x)
  h = rnn.memory(init = m)
  hh = rnn.previous_memory(h)
  a = layer.fc(W, x_)
  b = layer.fc(U, hh)
  s = pd.add(a, b)
  act = pd.sigmoid(s)
  rnn.update_memory(h, act)
  rnn.output(a, b)
o1, o2 = rnn()
```

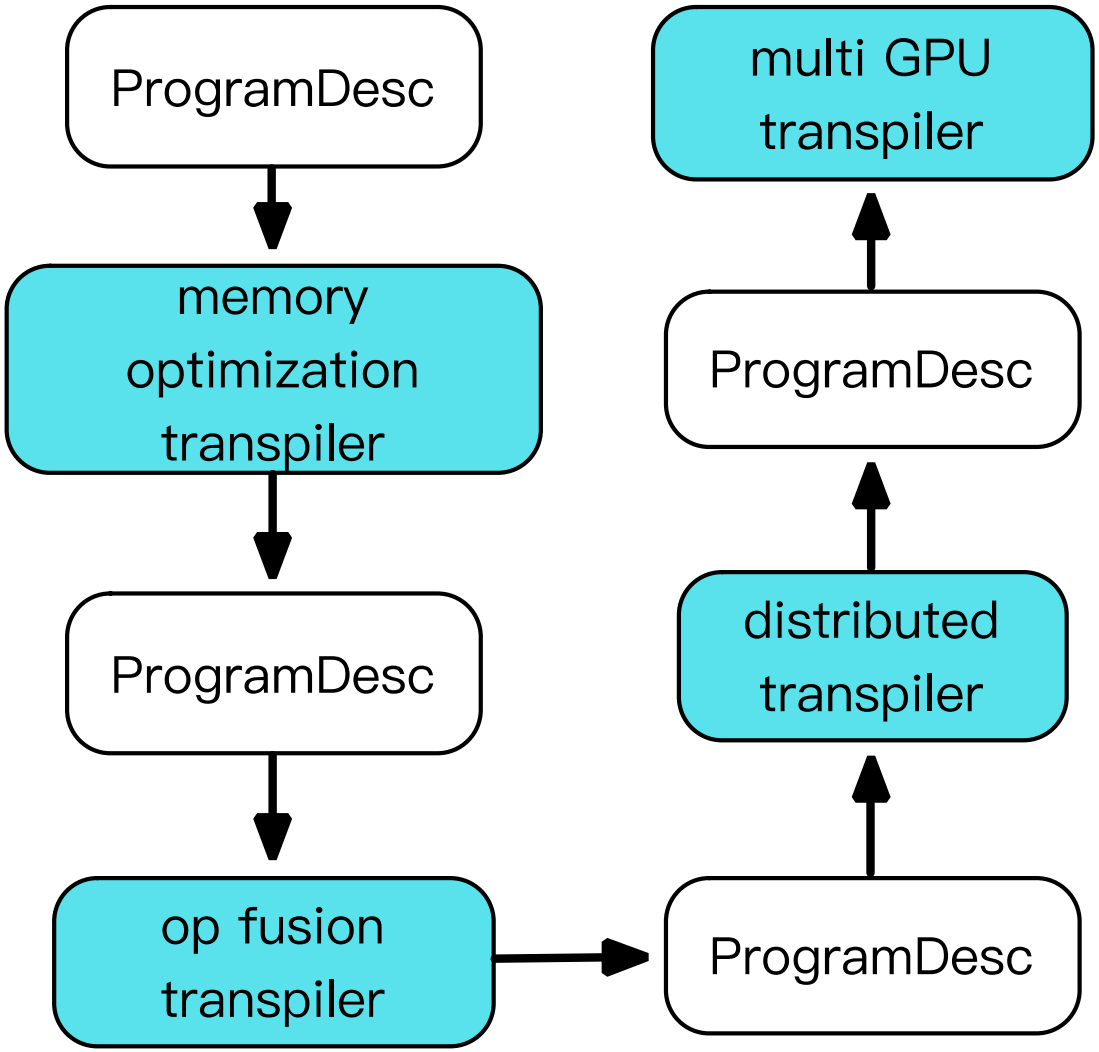
```
int* x = {10, 20, 30};
int* m = {0};
int* W = {0.314};
int* U = {0.375};

int mem[sizeof(x) / sizeof(x[0]) + 1];
int o1[sizeof(x) / sizeof(x[0]) + 1];
int o2[sizeof(x) / sizeof(x[0]) + 1];
for (int i = 1; i <= sizeof(x)/sizeof(x[0]); ++i) {
  int x = x[i-1];
  if (i == 1) mem[0] = m;
  int a = W * x;
  int b = Y * mem[i-1];
  int s = fc_out + hidden_out;
  int act = sigmoid(sum);
  mem[i] = act;
  o1[i] = act;
  o2[i] = hidden_out;
}
```

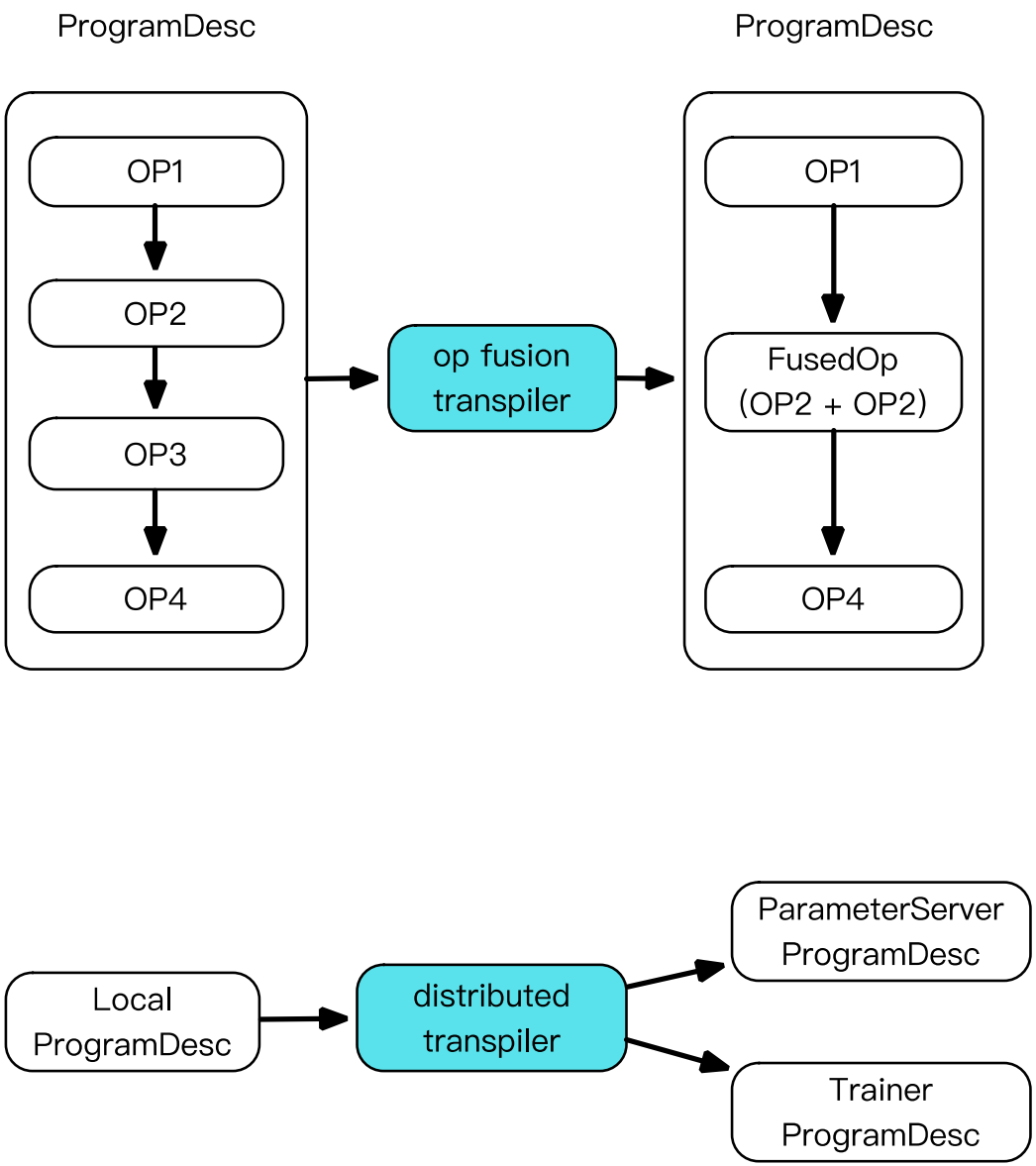
If-Else

```python
import paddle as pd

x = minibatch([10, 20, 30]) # shape=[None, 1]
y = var(1) # shape=[1], value=1
z = minibatch([10, 20, 30]) # shape=[None, 1]
cond = larger_than(x, 15) # [false, true, true]

ie = pd.ifelse()
with ie.true_block():
    d = pd.layer.add(x, y)
    ie.output(d, pd.layer.softmax(d))
with ie.false_block():
    d = pd.layer.fc(z)
    ie.output(d, d+1)
o1, o2 = ie(cond)
```

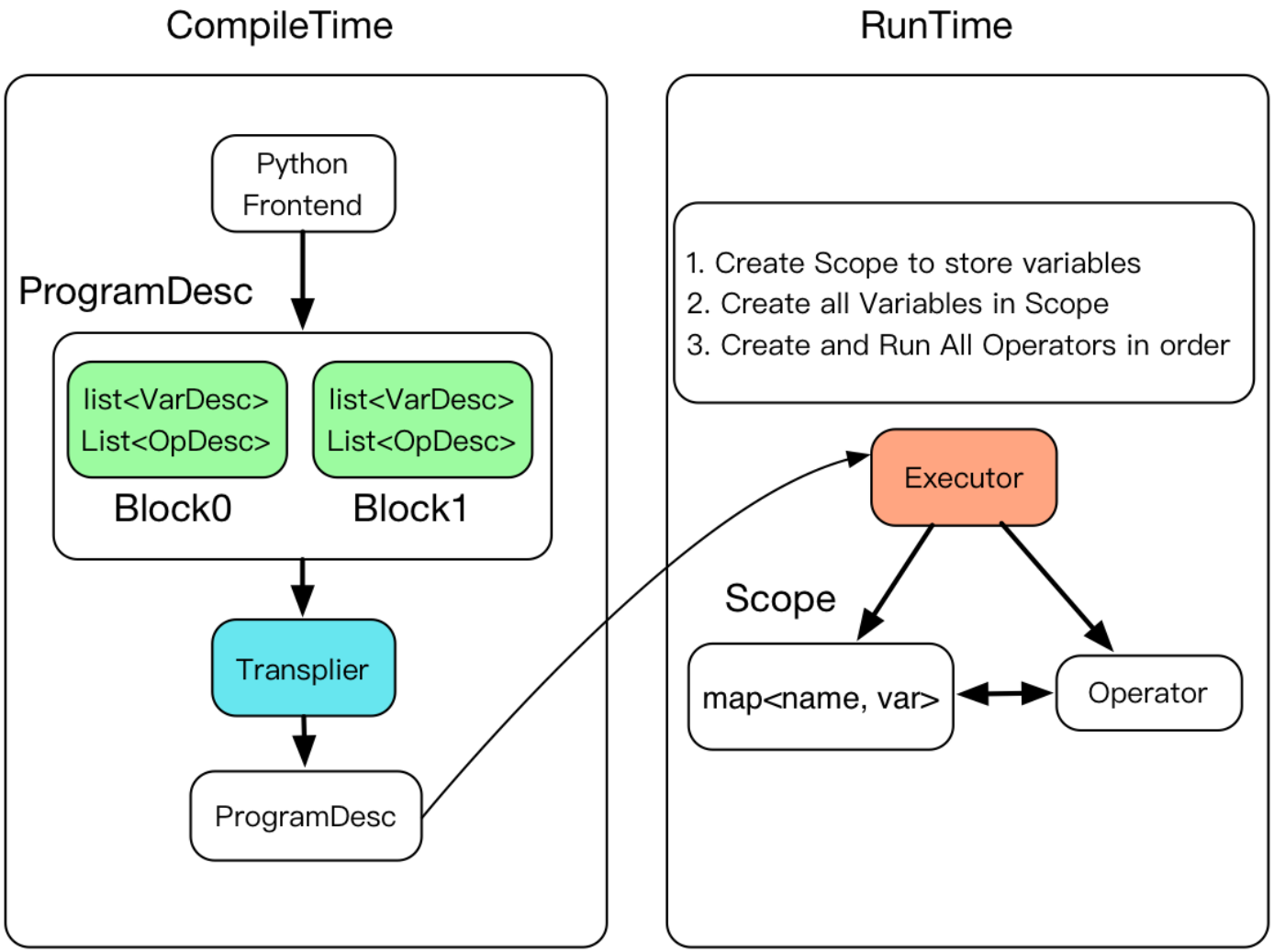
```cpp
namespace pd = paddle;

int x = 10;
int y = 1;
int z = 10;
bool cond = false;
int o1, o2;
if (cond) {
  int d = x + y;
  o1 = z;
  o2 = pd::layer::softmax(z);
} else {
  int d = pd::layer::fc(z);
  o1 = d;
  o2 = d+1;
}
```

Transpilers




ProgramDesc
OP1
OP2
OP3
OP4
op fusion
transpiler
ProgramDesc
OP1
FusedOp
(OP2 + OP2)
OP4
Local
ProgramDesc
distributed
transpiler
ParameterServer
ProgramDesc
Trainer
ProgramDesc

The Execution of a Fluid Program

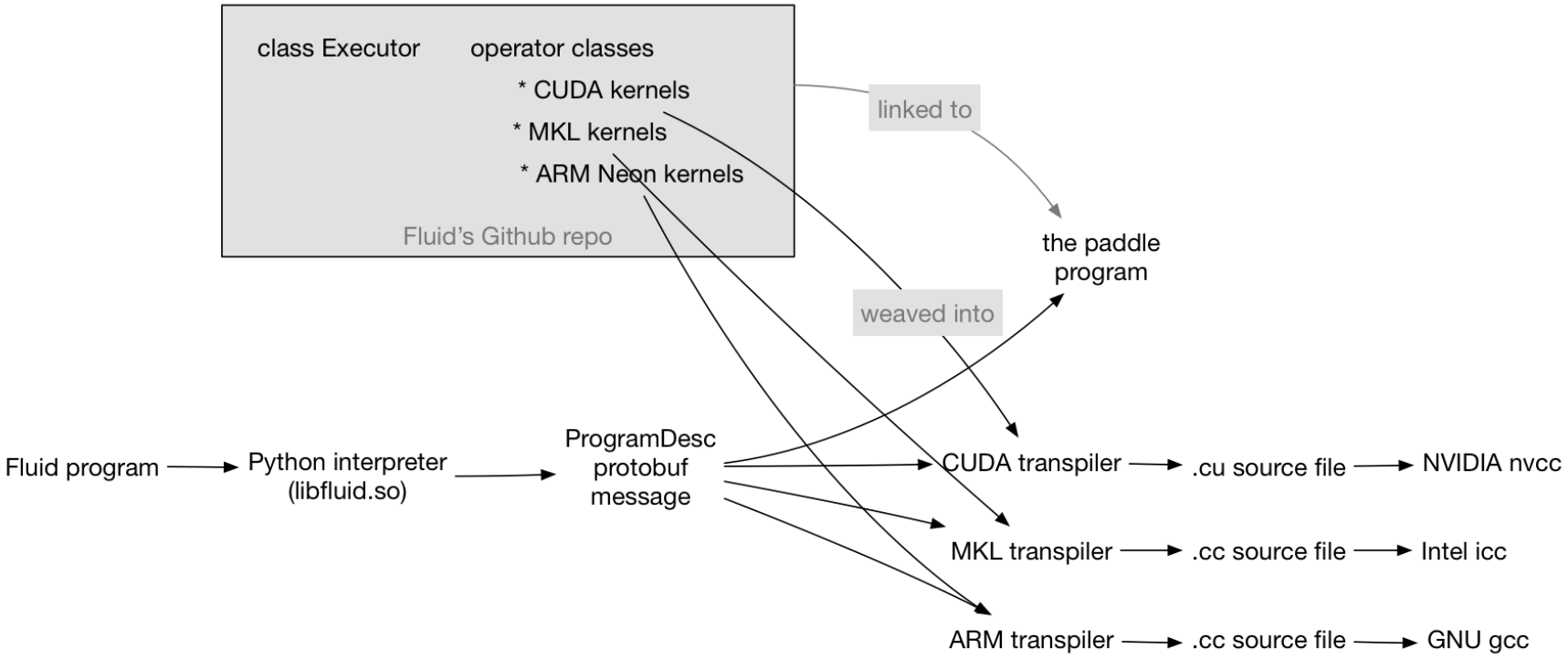


Compiled Language

- Fluid is moving towards the direction of a compiled programming language

Native Code Generator

Takes a ProgramDesc and generates a .cu (or .cc) file, which could be built by C++ compilers (gcc, nvcc, icc) into binaries



Application

- Video
- Image
- NLP
- GAN
- Reinforcement Learning

Repos

- Main Repo
  - https://github.com/PaddlePaddle/Paddle
- Model Bank
  - https://github.com/PaddlePaddle/models
- Book
  - https://github.com/PaddlePaddle/book
- Tape
  - https://github.com/PaddlePaddle/tape

Thank You!