**接口文档说明：**

**Class Openks\_fleet:**

Openks\_Fleet是一个基类，transpiler和pslib都是Fleet的实现。

**参数：**

mode(Mode)：实现Openks\_Fleet模式。

**返回：**None

**主要函数方法：**

* **init(role\_maker=None)**

fleet初始化，需在使用其他接口前先调用，用于定义多机环境配置

* **is\_worker()**

Parameter Server训练中使用，判断当前节点是否为Worker节点，是则返回True，否则返回False

* **is\_server(model\_dir=None)**

Parameter Server训练中使用，判断当前节点是否是Server节点，是则返回True，否则返回False

* **init\_server()**

Parameter Server训练中，fleet加载model\_dir中保存的模型相关参数进行parameter server的初始化

* **run\_server()**

Parameter Server训练中使用，用来启动server端服务

* **init\_worker()**

Parameter Server训练中使用，用来启动worker端服务

* **stop\_worker()**

训练结束后，停止worker

* **distributed\_optimizer(optimizer, strategy=None)**

分布式优化算法装饰器，用户可带入单机optimizer，并配置分布式训练策略，返回一个分布式的optimizer

在继承并重写了父类相关函数外还定义了\_transpile()函数

* **init\_server()**

Parameter Server训练中，fleet需要执行executor以运行初始化startup program，当然，如果model\_dir中已经有保存好的模型相关参数，就加载之并用于parameter server的初始化

* **init\_worker()**

首先，worker需要等待所有的参数服务器pserver启动；第二，运行executor的初始化startup program；第三，等待所有的worker全部初始化

检查所有的服务器是否正常

创建一个Communicator()类，Communicator用于分布式训练的信息传递，是CPP类Communicator()的封装

* **run\_server()**

运行executor来执行server的主程序main program

* **stop\_worker()**

训练结束后，关闭executor，停止worker

* **distributed\_optimizer(optimizer, strategy=None)**

分布式优化算法装饰器，用户可带入单机optimizer，并配置分布式训练策略，返回一个分布式的optimizer

* **\_transpile()**

传入之前设置的config（同步异步模式/是否使用GEO……）并根据config设置main\_program和transpiler；

设置startup\_program 、init\_server、init\_worker运行的program

**输入:** config、DistributedStrategy类的对象，包含训练信息

**输出：**无

**功能：**

根据config的信息和参数，将输入的program转化为分布式的program

**Class Openks\_DistributeTranspilerConfig(object):**

**说明：**

存储配置的class文件，为transpiler分布式任务提供配置

**重要的参数：**

slice\_var\_up (bool)

为多个Pserver（parameter server）将tensor切片, 默认为True。

split\_method (PSDispatcher)

可使用 RoundRobin 或者 HashName。

注意: 尝试选择最佳方法来达到Pserver间负载均衡。

min\_block\_size (int)

block中分割(split)出的元素个数的最小值。

**返回：**

初始化后的DistributeTranspilerConfig实例

**返回类型：**

实例（DistributeTranspilerConfig）

**代码：**

from paddle.fluid.transpiler.ps\_dispatcher import RoundRobin

import paddle.fluid as fluid

config = fluid.DistributeTranspilerConfig()

config.slice\_var\_up = True

config.split\_method = RoundRobin

config.min\_block\_size = 81920

**Class Openks\_RoleMakerBase(object):**

**类说明：**

RoleMakerBase作为一个基类，用于给当前分布式训练中的进程安排一个role

**函数：**

**def \_\_init\_\_(self)**

创建RoleMakerBase对象，对实例的worker节点、server节点列表进行初始化。

参数：无

返回：无

返回类型：无

**def is\_worker(self)**

Parameter Server训练中使用，判断当前节点是否是Worker节点，是则返回True，否则返回False

参数：无

返回：是否为worker节点

返回类型：bool

**def is\_server(self)**

Parameter Server训练中使用，判断当前节点是否是Server节点，是则返回True，否则返回False

参数：无

返回：是否为Server节点

返回类型：bool

**def is\_first\_worker(self):**

判断当前节点是否是第一个worker节点，是则返回True，否则返回False

参数：无

返回：是否为第一个worker节点

返回类型：Bool

**def worker\_num(self):**

返回当前总共的节点数

参数：无

返回：总共的节点数

返回类型：Int

**def worker\_index(self):**

返回当前worker节点id

参数：无

返回：当前节点id

返回类型：int

**def server\_index(self):**

返回当前server节点id

参数：无

返回：当前server节点id

返回类型：int

**def get\_trainer\_endpoints (self):**

返回trainer endpoints

参数：无

返回：trainer endpoints

返回类型：self.\_server\_endpoints 实例

**def to\_string (self):**

按照特定格式返回RoleMaker实例信息

参数：无

返回：实例信息

返回类型：string

**def all\_gather(self, input):**

根据input值，返回trainers和pservers之间的所有gather

参数：

input: int 或者 float

返回：trainers和pservers之间的所有gather

返回类型：values组成的list

**def all\_reduce\_worker(self, input, output, mode="sum"):**

trainers间所有的reduce

参数：

input: list

output:list

mode:string, 有sum, min, max三种模式

返回：无

返回类型：无

**def barrier\_worker(self):**

当前trainer之间的barrier

参数: 无

返回： 无

返回类型：无

**def barrier\_all (self):**

所有trainer之间的barrier

参数：无

返回：无

返回类型：无