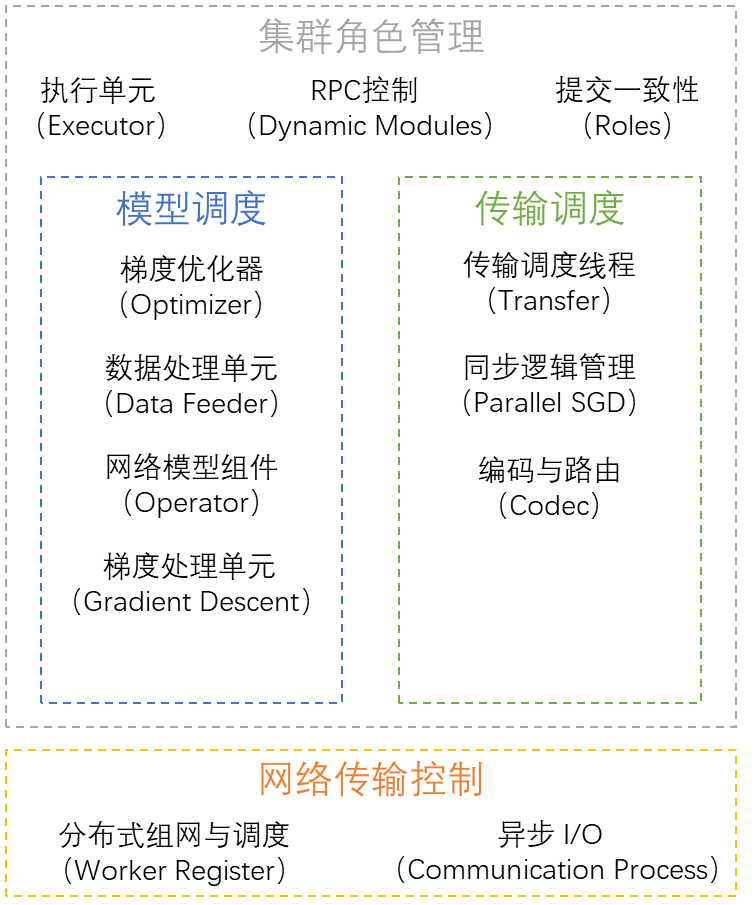
并行梯度下降实验框架

现有的分布式机器学习框架通常仅提供了神经网络模型的定制和一些简单的并行参数控制。在本文所涉及的应用领域下，这些简单的并行参数不能够充分适应本文对梯度编码和延迟控制的需求。本文设计一种并行的梯度下降算法（Parallel Stochastic Gradient Descent，P-SGD）框架，可以满足对模型的自定义控制需求（例如模型算法的修改，模型结构的修改等），同时又可以满足对网络数据的编码和路由控制需求（例如自定义的梯度转发逻辑，梯度编码和压缩等）。

P-SGD共由四个组件部分构成，分别是集群角色管理、模型调度、传输调度和网络传输控制，这些组件之间的关系如图C.1所示。下面逐一介绍这些组件的作用及其设计原理。



图C.1 并行梯度下降实验框架基本组件结构图

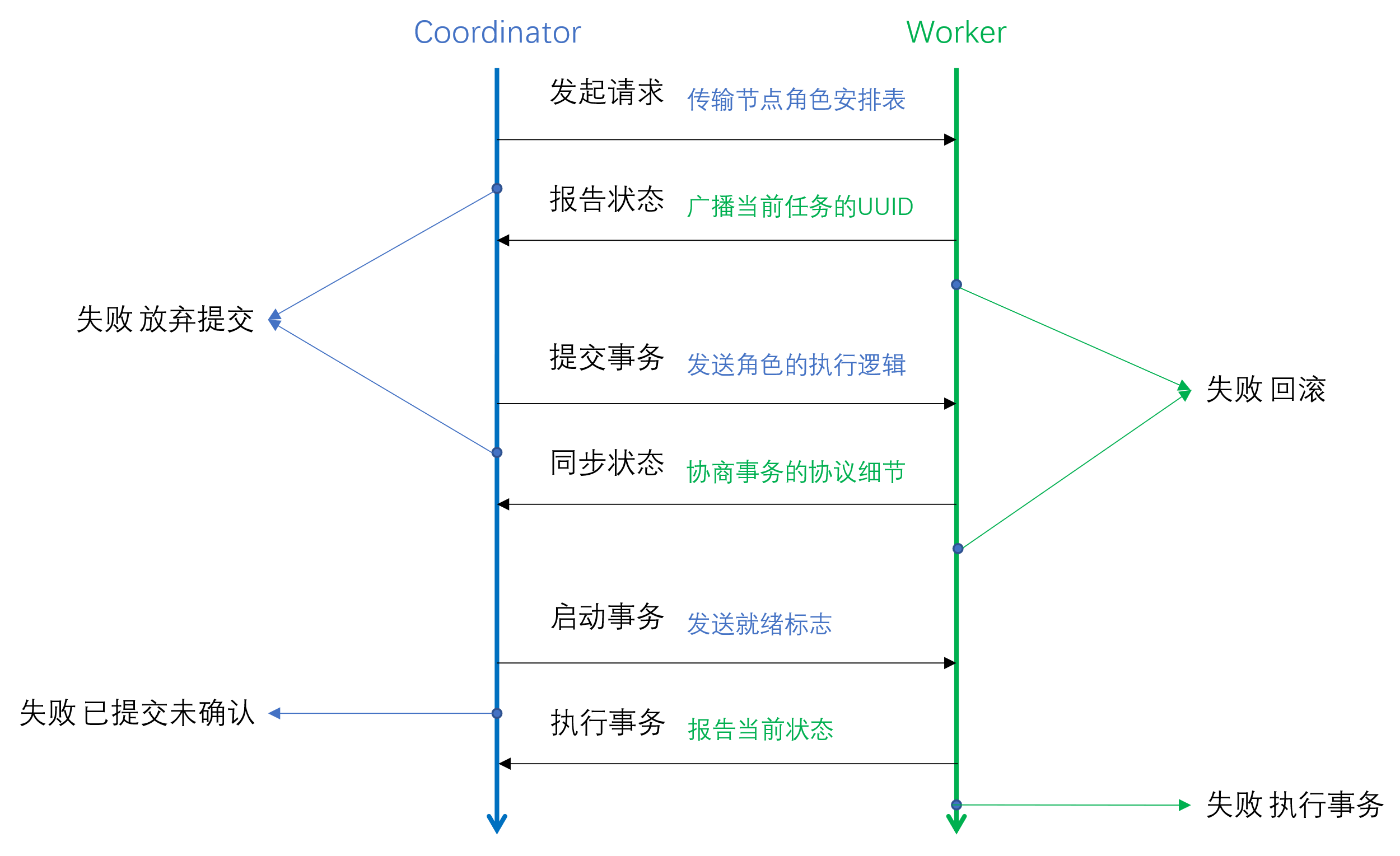
## 集群角色管理

集群角色管理主要实现了对集群中节点的角色分配，集群角色管理由代码中的*roles*、*executor*、*models*实现。

### 基本功能

集群角色管理提供了两种角色，一种是客户端（Client）、另一种是分布式服务节点（Worker）。客户端角色用于提交事务以及获取当前的提交结果：角色Coordinator负责提交事务，并保证提交结果的一致性；角色Reclaimer负责获取上次提交的事务的执行结果，并不会影响集群的状态。分布式服务节点是在机器上以服务的形式启动的，一旦一个计算机启动了Worker进程，那么这个计算机就可以作为分布式集群的一部分使用，等待事务的提交。分布式服务节点支持无人值守管理，当运行P-SGD任务时，不需要向Worker拷贝数据集或代码，Worker会自行获取要执行的代码、要训练的模型并从提交方下载所需的数据集。

Coordinator负责控制提交以及提交的一致性，集群在使用Coordinator提交一个任务后，仅可能存在两种状态：1.被提交的集群同时开始执行相同的事务；2.被提交的集群回滚到上个事务执行后的状态。Coordinator事务提交的状态转换如图C.2所示。



图C.2 协调者（Coordinator）的事务提交流程图

### 实现原理

集群角色管理实现了事务的提交控制、RPC逻辑控制两个主要功能，其中模块*roles*实现了事务的提交控制部分，*executor*和*models*实现RPC的相关支持逻辑。

1. 事务提交控制

事务的提交控制依照提交逻辑分为三个部分：1.事务的提交；2.事务的确认；3.事务的启动。三部分分别对应Coordinator模块的三组方法：和 控制了事务的发起和提交过程，其中可以为一组节点分配角色和执行逻辑，可以为一个节点分配角色和执行逻辑；控制了事务的确认，通过状态标志集合（）来记录节点的状态；控制了事务的最终执行和事务执行结果的回收，同样是通过标志集合（）实现。

1. RPC支持逻辑

RPC支持逻辑主要依赖Python的反射机制实现，实现的关键组件为。在调用时，通过模块信息查找模块对应的脚本位置，并将对应的模块名称、类名称、脚本内容发送至执行方。执行方先将脚本内容编译并执行，然后通过类名称获取类对象的构造方法，最后构造对象返回。只要调用双方约定了调用接口，具体的调用逻辑代码可以直接由Coordinator传输给Worker，不需要额外的人工操作。

## 模型调度

模型调度主要实现了对神经网络的支持，包括基础的神经网络组件（损失函数、激活函数、卷积等）和神经网络与分布式互操作的组件（节点数据采样、并行优化算法等）。同时提供了一个通用模型接口，供用户使用。模型调度部分由代码中的*nn*模块实现。

### 基本功能

1. 激活函数（Activation）

激活函数包含了神经网络常用的激活函数类型，有线性整流单元（ReLU）、Sigmoid型函数、双曲正切函数（Tanh）、Softmax激活函数、Leak ReLu、线性激活（Linear）。此外，用户可以通过实现抽象类来实现自定义的激活函数，该激活函数可以放置于模型层（Layer）中使用，也可以单独组合成算子（Operator）使用。

1. 数据集（Data）

数据集包含了两部分，一部分是用于基本类型的处理，即，另一种是用于和P-SGD互操作的。此外，用户可以通过实现接口来定义自己的数据集类型，数据集类型传入类中的方法即可用于模型的训练。

1. 梯度下降（Gradient Descent）

梯度下降包含了常用的梯度下降优化器，有AdaDelta、AdaGrad、ADAM、Gradient Decay、RMS Prop和minibatch SGD。此外，用户可以通过实现接口来定义自己的梯度下降优化器，梯度下降优化器通过传入类中的方法来接收指定的优化器用于训练。

1. 表示学习层（Layer）

层可以用于构建神经网络模型，包含了Batch Normalization、Convolutional Layer、Dense Layer等。此外，用户可以通过实现抽象类来实现自定义的层，自定义的层可以用于构建模型。

1. 损失函数（Loss）

损失函数用于配置神经网络模型的优化指标，模块中包含了常用的均方误差（MSE）、交叉熵（Cross Entropy）和双曲正切（Tanh）型损失函数。此外，用户可以通过实现接口定义自己的损失函数，损失函数可以传入类的方法，用于定义模型的优化目标。

1. 评判指标（Metric）

评判指标可以用于评估最终模型的性能，模块中包含了常用的评判指标，包括二值准确率（Binary Accuracy）、分类准确率（Categorial Accuracy）、相对误差（Relative Error）、均方误差（Mean Square Error）、相对均方误差（Relative Mean Square Error）、等效错误率（Equal Error Rate）、真阳性率（True Positive）、假阳性率（False Positive）、真阴性率（True Negative）、假阴性率（False Negative）、ROC线下面积（Area Under the Curve）。此外，用户可以通过实现接口来定义自己的评判指标，评判指标可以传入类的方法，用于在evaluate阶段评估模型的性能。

1. 操作（Operation）

操作是神经网络中的基本元素，模块中实现了加（Add）、减（Sub）、乘（Multiply）、幂次（Power）、卷积（Convolution）操作。使用操作可以将Value连接起来，操作的返回结果依然是操作，通过定义操作树的方式可以实现计算图的定义，通过实现抽象类并定义计算图，可以实现自定义的映射模型。

1. 优化方法（Optimizer）

优化方法主要用于与P-SGD的互操作，模块中包含了梯度下降（Gradient Descent）、梯度上升（Gradient Ascent）、双重缓冲（Double Buffering）、并行梯度下降（Parallel Stochastic Gradient Descent）、模型平均化（Parameter Averaging）。

1. 值（Value）

值是神经网络中参与操作的基本元素，模块提供了三种值类型：

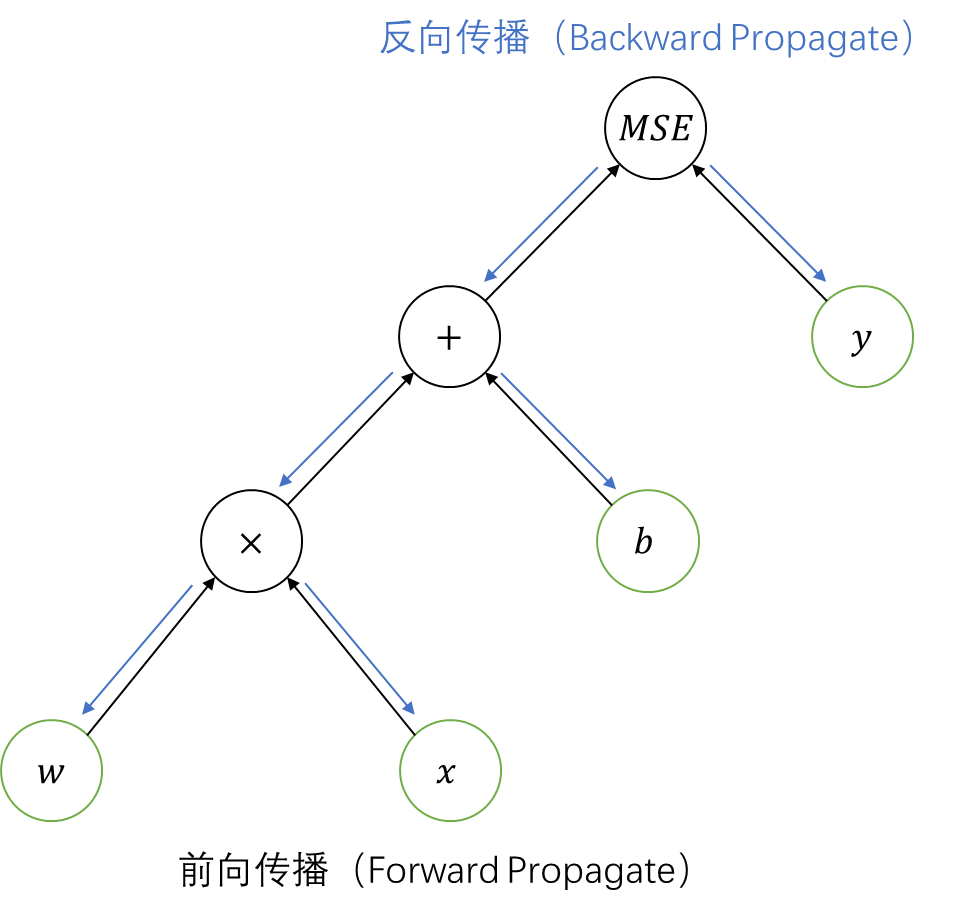
1.变量（Variable），变量可以参与计算，可以被计算梯度，能够参与到模型的前向与反向传播计算中。变量在使用前需要初始化，可以使用默认初始化也可以使用自定义值进行初始化，变量可以被梯度下降优化器注册并进行优化，是网络参数的一部分。

2.占位符（Place Holder），占位符可以参与计算，可以被计算梯度，能够参与到模型的前向传播计算中。占位符没有实值，需要在使用时注册具体的值参与计算，占位符也不会被修改，不会被梯度下降优化器优化。

3.权重（Weights），权重是模型层的一部分，用于提供给模型层使用，和梯度下降优化器进行互操作，没有单独使用的意义。权重属于变量的一种子类实现。

### 实现原理

模型调度模块依赖神经网络计算图实现，模块中每个定义的操作都是Operation的子类，且Operation也需要由Operation来构造。Operation作为计算图的基本节点，Value和Place Holder也是一种Operation，作为计算图的叶子节点。当创建模型时，我们使用加（Add）减（Sub）等操作把变量（Value）和占位符（Place Holder）连接起来组成计算图，并使用损失函数（Loss）定义计算图的根节点。当使用模型时，我们初始化叶子节点，为变量（Value）初始化，为占位符（Place Holder）定义实值，并使用图遍历算法逐个节点计算结果。而梯度下降优化器（Optimizer）就相当于一种图遍历算法，遍历计算图并计算对应的梯度信息，如图C.3所示。



图C.3 模型调度中计算图的前向传播与反向传播示意图（其中为模型输入，为模型参数，为模型样本标签，为均方误差损失）。

## 传输调度

传输调度主要实现了对P-SGD中梯度和参数传输的控制，主要处理计算节点之间的同步逻辑，数据通信的压缩编码，数据传输的路由过程。传输调度由代码中的*codec*和*psgd*模块实现。

### 基本功能

传输调度中的*psgd*模块控制了节点之间的同步状态，模块实现的同步状态只有两种，即完全同步（Synchronized SGD）和完全异步（Asynchronous SGD）：完全同步状态下，所有直接相连的计算节点总是需要互相等待任务进度，不存在某个节点比与其相连的其他节点多训练一个批次（对不直接相连的计算节点没有同步限制）；完全异步状态下，所有的计算节点依照自己的计算性能独立进行计算。

传输调度中的*codec*模块控制了梯度的发送方式与发送编码，通过和来完全控制从该节点流出和流入的数据的处理方式。传输调度控制接口*codec*也是本实验平台的核心内容，本实验平台的数据编码和路由控制就是基于*codec*来操纵的。

## 网络传输控制

网络传输控制主要实现了对P-SGD中的所有数据类型的传输和记录，为每个节点建立了一个异步的消息处理队列，简化了传输调度中的网络收发逻辑。网络传输控制组件由代码中的*network*模块实现。

### 基本功能

网络传输控制实现了基本的网络收发和节点标识的功能。网络传输控制组件通过32bit整形来唯一标识当前集群中的节点，并且支持向任意一个直接连接的节点发送信息。信息的类型是可序列化类型，可以发送任意实现了和方法的类。组件中维护了一个消息队列，由I/O线程负责处理队列中消息的收发，因此组件对消息的处理是异步的，也不需要上层逻辑维护消息的收发流程。I/O线程使用了多路复用技术，可以实现低开销多链接的维护。