主办：

Tencent 腾讯 PURECPP

特别支持：

腾讯大学 Tencent University

NEWER IS BETTER _

2020 Pure C++大会

蚂蚁集团
ANT GROUP

Ray: 通用的分布式计算框架

陈昊@蚂蚁集团

2020-12-26

About me

- 蚂蚁集团
  - 2018 – Present。
  - 负责分布式计算引擎Ray。
  - Ray开源社区Committer。

- Facebook
  - 2015 – 2018.
  - Worked on Instagram infra efficiency.

主要内容

蚂蚁集团
ANT GROUP

- Ray是什么?
- Ray的架构。为什么用C++?
- Ray开源社区和应用场景。

分布式应用变得越来越重要

- 流计算/批计算系统。
- 机器学习。
- 推荐系统。
- 搜索引擎。
- …

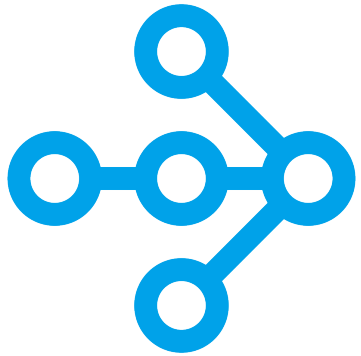
从海量数据中挖掘价值

开发分布式系统为什么这么难?

分布式系统开发的常见难点:

- 分布式组件通信。
- 任务调度。
- 故障恢复。
- 数据存储和传输。
- 元数据管理。
- 部署运维。
- …

如何让分布式系统开发更简单?



RAY

- 简单、通用的分布式编程API，适用于任意计算模式的分布式系统。
- 高性能和可扩展性。
- 解决分布式系统的通用问题，屏蔽分布式系统开发的难点。

像编写单机程序一样编写分布式程序

蚂蚁集团 ANT GROUP

通用分布式编程API

Key idea：不绑定计算模式，不引入新的概念，
而是把单机编程的概念分布式化。

两个核心概念：Task & Actor。

Ray task API

- Function ➡️ Remote function
- 无状态计算单元

```python
def heavy_compute(value):
    # Heavy computation here.
    # return ...

res = [heavy_compute(i)
       for i in range(10000)]
print(res)
```

把普通function变成remote function

```python
@ray.remote
def heavy_compute(value):
    # Heavy computation here.
    # return ...

res = [heavy_compute.remote(i)
       for i in range(10000)]
print(ray.get(res))
```

异步调用，远程执行

Ray Task：把Function分布式化

Ray actor API

- Class/Object ➡️ Actor
- 有状态计算单元

```python
class Counter:

    def __init__(self):
        self.value = 0

    def increment(self):
        self.value += 1
        return self.value
```

把普通Class变成Actor Class

```python
@ray.remote
class Counter:
    # ...

counter = Counter.remote()

res = [counter.increment.remote()
       for _ in range(3)]

assert ray.get(res) == [1, 2, 3]
```

创建远程Actor对象，返回Actor Handle

方法

Ray Actor：把Class/Object分布式化

多语言API

- 支持Python、Java（Ray 1.0）、C++

```
int Plus(int x, int y) {
  return x + y;
}

auto res = Ray::Task(
  Plus, 1, 2).Remote();

std::out << *(res.Get());
```

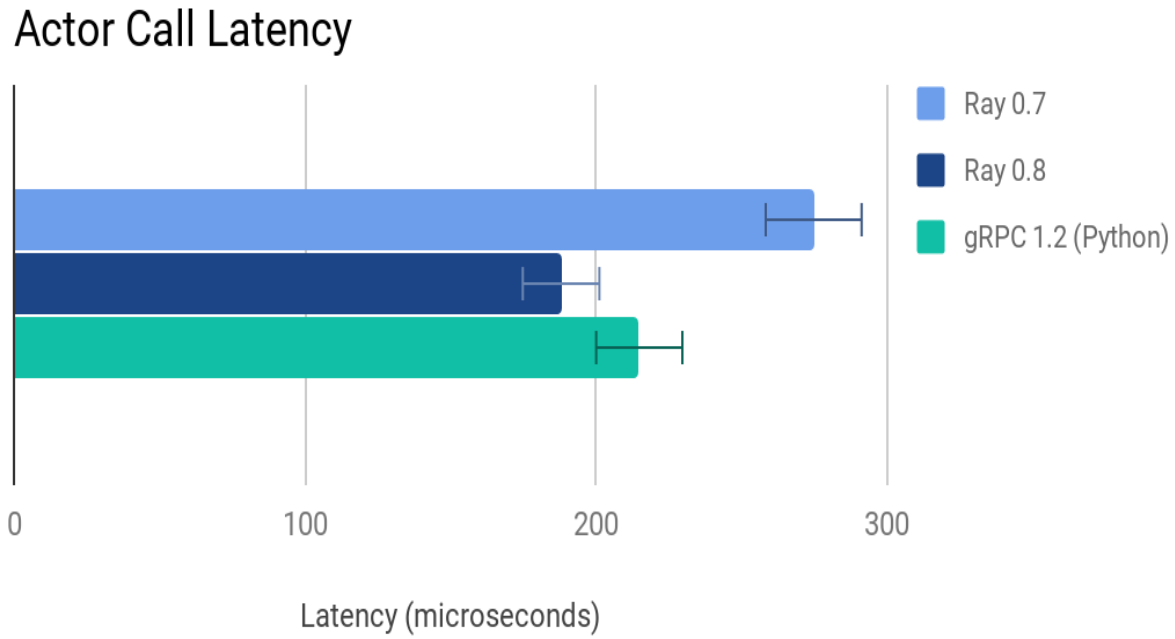
C++ Task

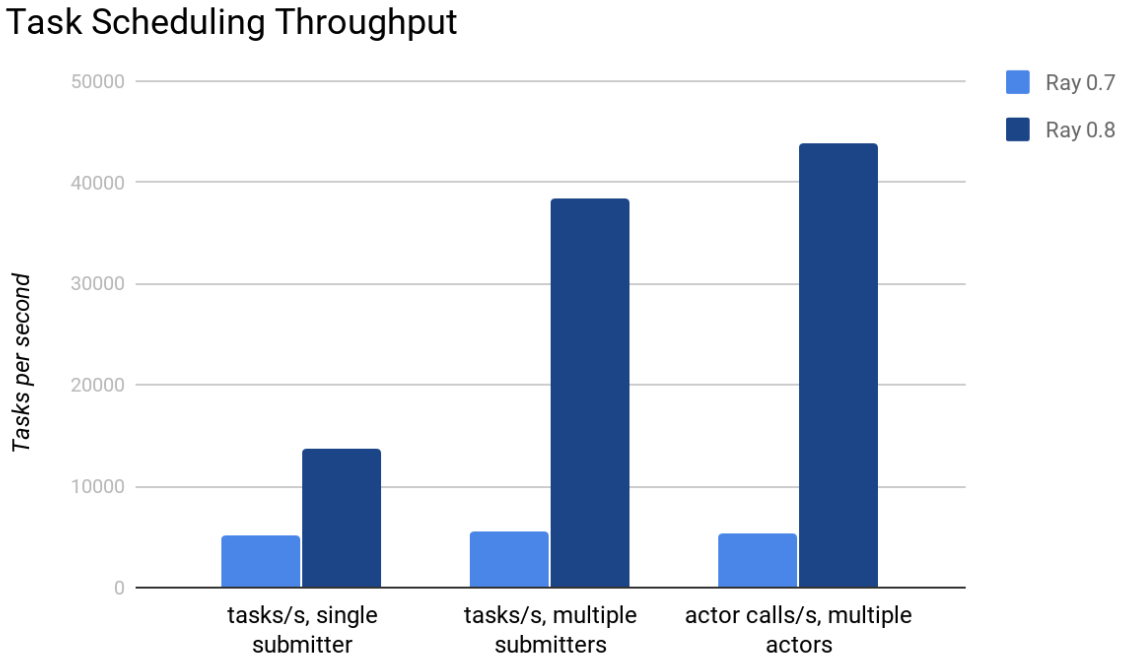
```
class Counter {
 public:
  int count;

  Counter() { count = 0; }

  static Counter *Create() {
    return new Counter();
  }

  int Add(int x) {
    count += x;
    return count;
  }
};

ActorHandle<Counter> actor =
  Ray::Actor(Counter::Create).Remote();

auto res = actor.Task(
  &Counter::Add, 1).Remote();

std::out << *(res.Get());
```

C++ Actor

性能和扩展性

- RPC通信：
  - 底层使用C++
  - Ray 0.8: worker-to-worker direct call





性能和扩展性

- Actor创建：8000/min。
- Actor故障恢复：2s。
- 扩展性：单集群支持千级节点、万级Actor。

分布式系统开发的常见难点

- ~~分布式组件通信。~~
- 任务调度。
- 故障恢复。
- ~~数据存储和传输。~~
- ~~元数据管理。~~
- 部署运维。
- …

任务调度

- 用户只需要关心资源需求，不需要关心物理信息

```
@ray.remote(num_cpu=1, num_gpu=2)
class MyActor:
  # ...
```

蚂蚁集团 ANT GROUP

自定义调度策略

蚂蚁集团 ANT GROUP

- Ray 1.0: Placement Group
- 支持collocation，anti-collocation等调度策略

```
pg = placement_group([{"GPU": 2}],
                     strategy="STRICT_PACK")

@ray.remote(num_gpus=1)
class GPUActor:
  # ...

[
  GPUActor.options(placement_group=pg).remote()
  for _ in range(2)
]
```

自动故障恢复

- 自动监控Actor健康状态，重启故障Actor。
- 实现HA能力。

```
@ray.remote(num_restarts=100)
class FaultTolerantActor:
  # ...

FaultTolerantActor.remote()
```

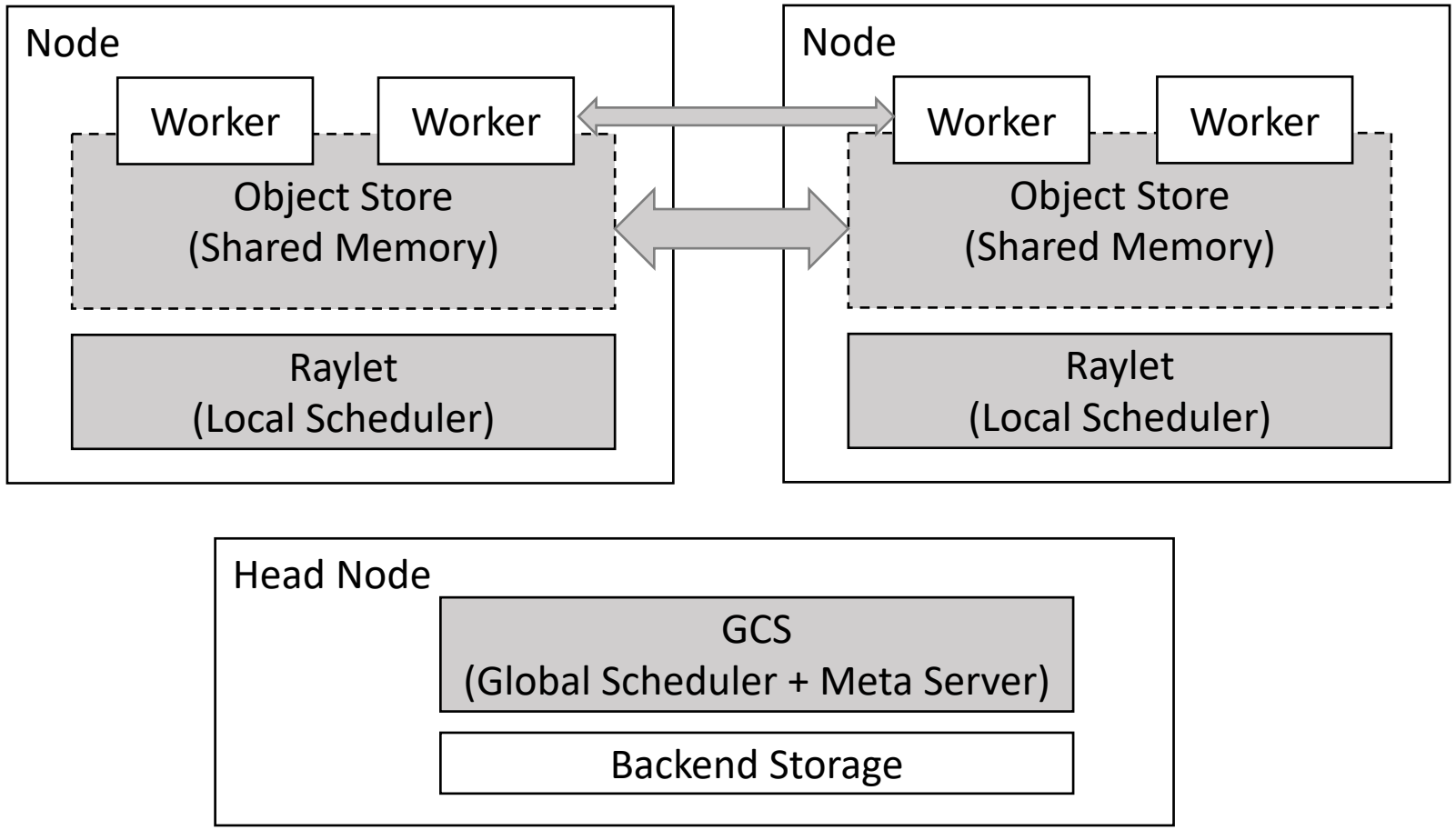
蚂蚁集团 ANT GROUP

部署

- Ray自带的部署工具支持：
  - 私有集群
  - K8S
  - 主流云服务商

- Auto-scaler： 弹性伸缩

Ray的架构

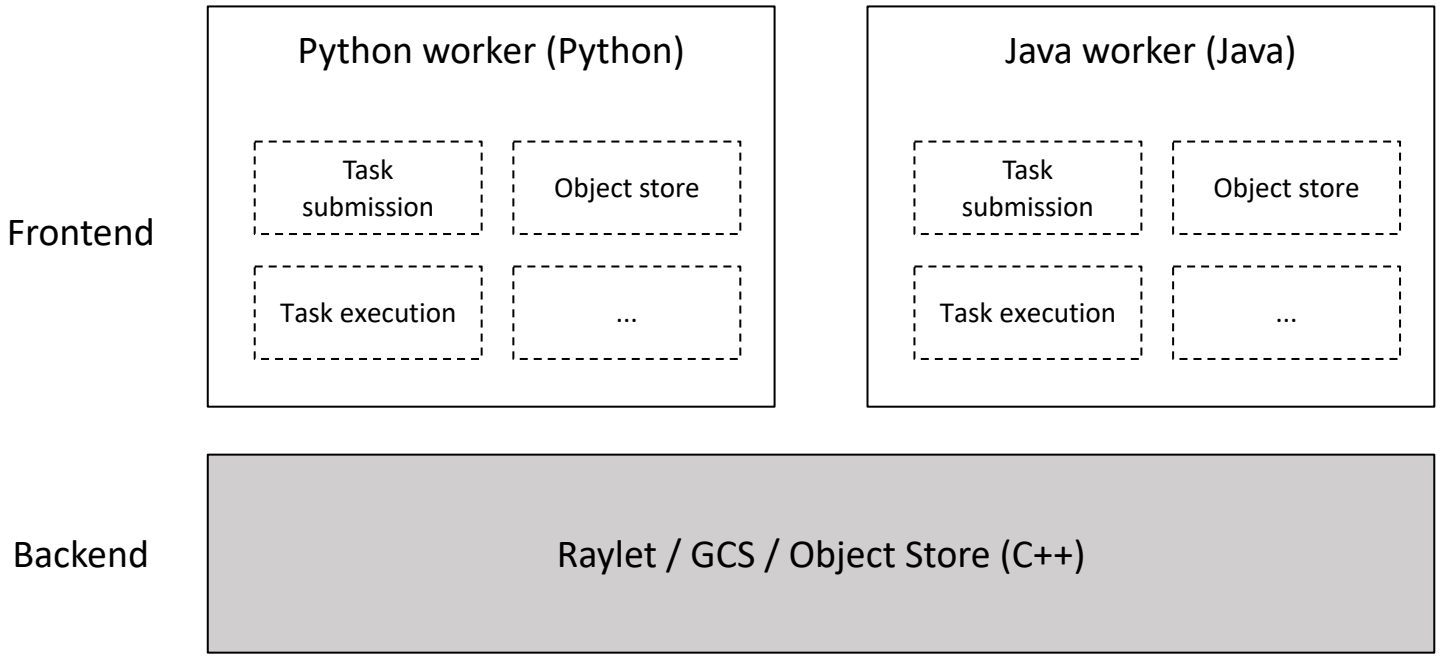


Written in C++

为什么使用C++?

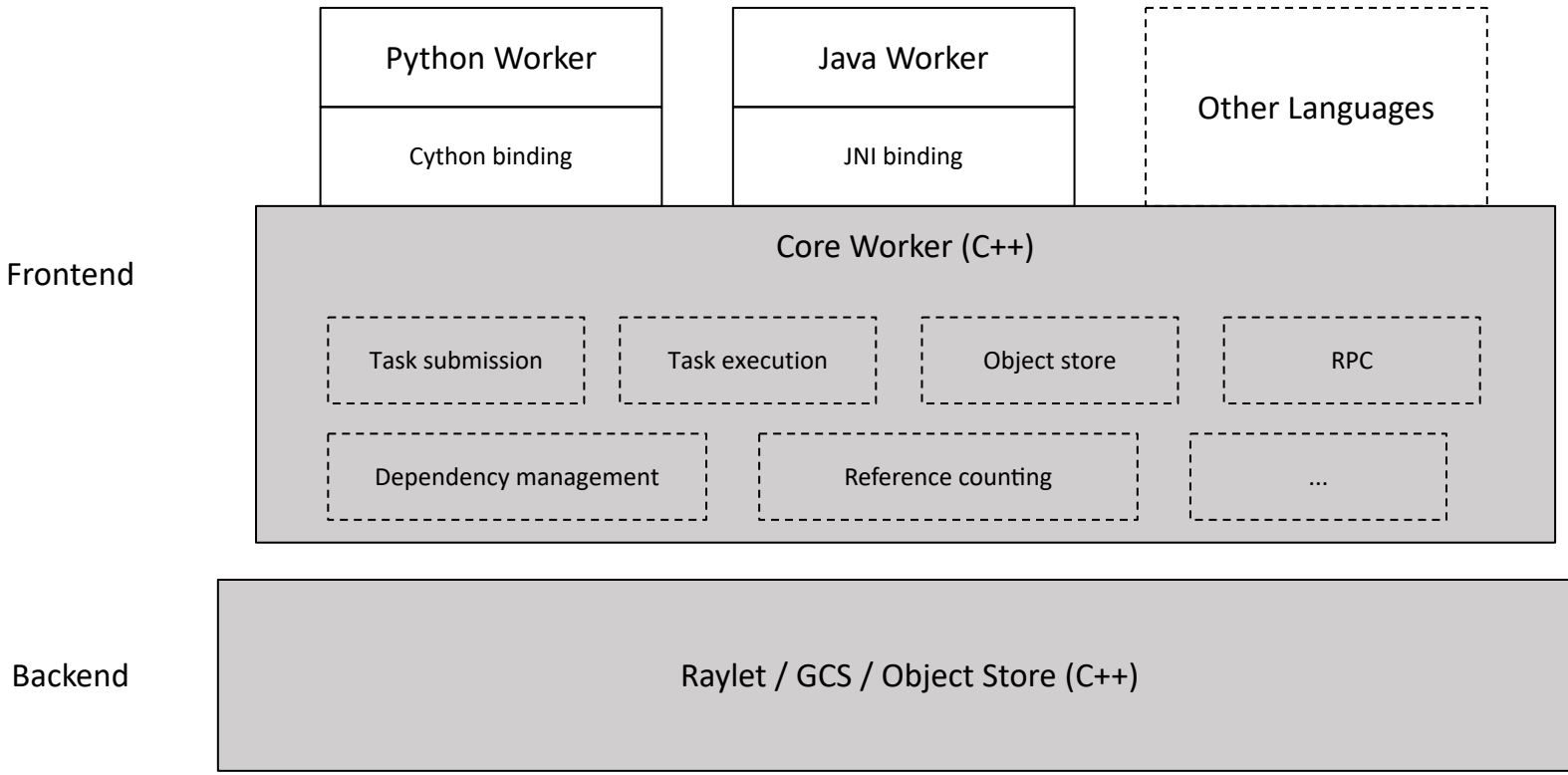
- 高性能、可扩展性。
- 多语言API。

多语言API：Worker老架构



- 代码重复
- 可维护性差

多语言API：Worker新架构



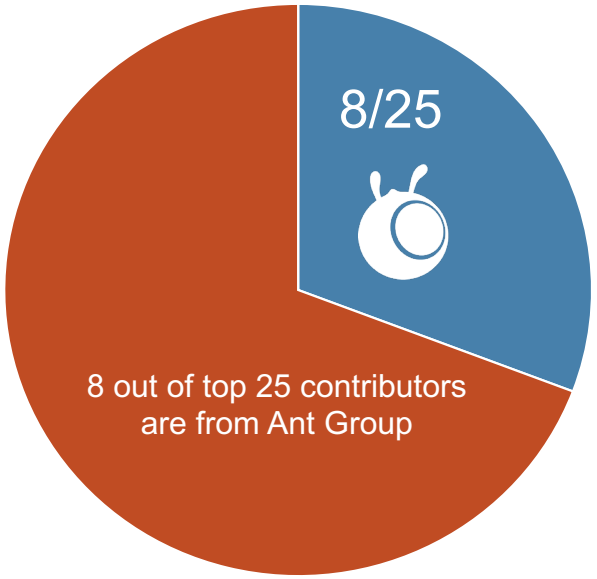
- 减少重复代码。
- 多语言API保持一致性。
- 更容易支持新语言API。
- 在Core Worker实现高级功能

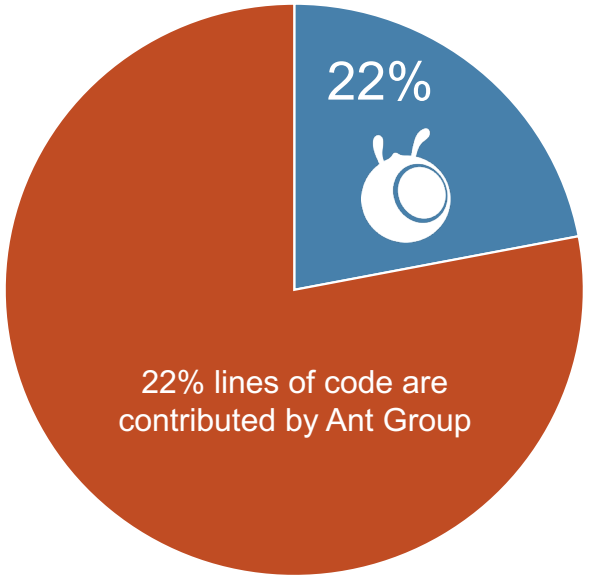
Ray开源社区

蚂蚁集团

Unwatch 407 | Unstar 14.2k | Fork 2.3k

蚂蚁的开源贡献





第二大contributor

Features:
- Java API
- Cross-language programming
- Actor fault tolerance
- Placement Group
- Multi-tenancy

Ray internals:
- Fault-tolerant GCS
- Pluggable GCS storage
- gRPC
- Worker-to-worker direct transport
- In-memory object store

DevOps:
- New dashboard
- Kubernetes deployment

Libraries:
- Streaming library

Ray在蚂蚁的应用

蚂蚁集团 ANT GROUP

- 已经大规模应用在生产环境（10万Core）。
- 应用于风控、营销、金融智能等蚂蚁核心业务。
- 连续三年平稳支持双十一。

新计算模式
（例如：图计算GeaFlow）

融合计算
（例如：在线机器学习）

谢谢！

微信搜一搜 Ray中文社区

蚂蚁集团
ANT GROUP

欢迎加入蚂蚁！

chenh@antfin.com