
ArchData
技术峰会成都站
主办方：
中生代技术
FRESHMAN TECHNOLOGY
快CTO
互联网创业技术服务平台
2018年1月27日成都菁蓉国际广场3W COFFICE 蓉漂茶馆
ArchData技术峰会成都站

润乾
RAQSOFT

中生代技术
FRESHMAN TECHNOLOGY

IT大咖说
知识共享平台

轻量级大数据计算引擎

创新技术推动应用进步

主讲人:蒋步星

RAQSOFT

RAQSOFT

目录 contents

沉重的大数据计算

1

集群

单机性能不被关注，

依靠集群规模

2

内存

避开外存计算困难，

指望巨大内存

3

框架

框架体系庞大复杂，

试图包罗万象

轻量级计算需求

大数据的技术本质是高性能

提高性能的需求无处不在

不总是有那么大的数据量

低延迟即时响应业务数据量并不大

不总是适合部署大数据平台

即时查询常常有被集成需求

临时性数据处理来不及建设大数据平台

不总是可以扩容硬件（内存）

需求

大数据计算开发难度大

大数据平台对SQL查询关注过多

性能比拼的主要阵地

优化SQL性能几乎无助于降低开发难度

大量过程计算的开发难度还很大

用SQL很难描述，复杂SQL优化效果不好

仍需大量底层的编码，经常编写UDF

提高性能本质上是降低开发难度

复杂运算的自动优化靠不住，需要快速编写高性能算法

难度大

举例：漏斗转换计算

设备ID	时间戳	事件名称	事件属性	应用标识	日期
1111	1490970560539	搜索商品	{类别=A……}	3zprjdg7yyp5	20170227
2222	1490965747548	浏览商品	{价格=50……}	3zprjdg7yyp5	20170227
3333	1490972189107	提交订单	{……}	3zprjdg7yyp5	20170227
……	……	……	……	……	……

更多问题

- 股票连续3天上涨后再涨1天的概率和平均涨幅，按所属板块和时间段分类对比
- 与去年同期的收入销售额对比分析，要考虑到节假日的影响
- 一周内累计登录时长超过一小时的用户占比，但要除去登录时长小于1分钟的误操作情况
- 本月内下午5-6点断档情况超过90%的商品
- 信用卡在最近三个月内最长连续消费的天数分布情况，考虑实施连续消费10天后积分三倍的促销活动
- ……

集群透明化

大数据平台努力实现集群透明化

单机与集群一致

网络存储系统+自动任务分配

透明化提高代码兼容性，降低开发难度

透明化难以获得最优性能

高性能计算方案因场景而异，可能是矛盾的

透明化只能选择最保险的方法，一般是性能较差的那个

透明化框架对资源消耗严重

透明化与高性能的权衡

数据分布

节点文件系统：可控冗余，内存利用

网络文件系统

任务分配

程序员分配：根据节点能力安排任务，无框架资源消耗

系统自动分配

轻量级计算的技术特征

目标：过程计算

可集成性

数据源开放性

- 直接文件计算

注重单机优化

- 多线程并行

权衡集群透明与高性能

- 节点文件存储，不用网络文件系统
- 多个单机运算，不用统一集群框架

目录 contents

集算器 – 技术特征

面向过程计算

无缝应用集成

多样性数据源接口 直接文件计算

单机优化技术 多线程并行

无中心集群结构 自由数据分布和任务分配

集算器 – 敏捷语法体系

某支股票最长连续涨了多少交易日

```
1  select max(连续日数)
2  from (select count(*) 连续日数
3          from (select sum(涨跌标志) over(order by 交易日) 不涨日数
4                  from (select 交易日,
5                          case when 收盘价>lag(收盘价) over(order by 交易日)
6                          then 0 else 1 end 涨跌标志
7                  from 股价表) )
8          group by 不涨日数)
```

SQL

	A
1	=股价表.sort(交易日)
2	=0
3	=A1.max(A2=if(收盘价>收盘价[-1],A2+1,0))

集算脚本

思考：按照自然思维怎么做？

语法体系更容易描述人的思路

数据模型不限制高效算法实现

集算器 – 面向过程计算

完整的循环分支控制

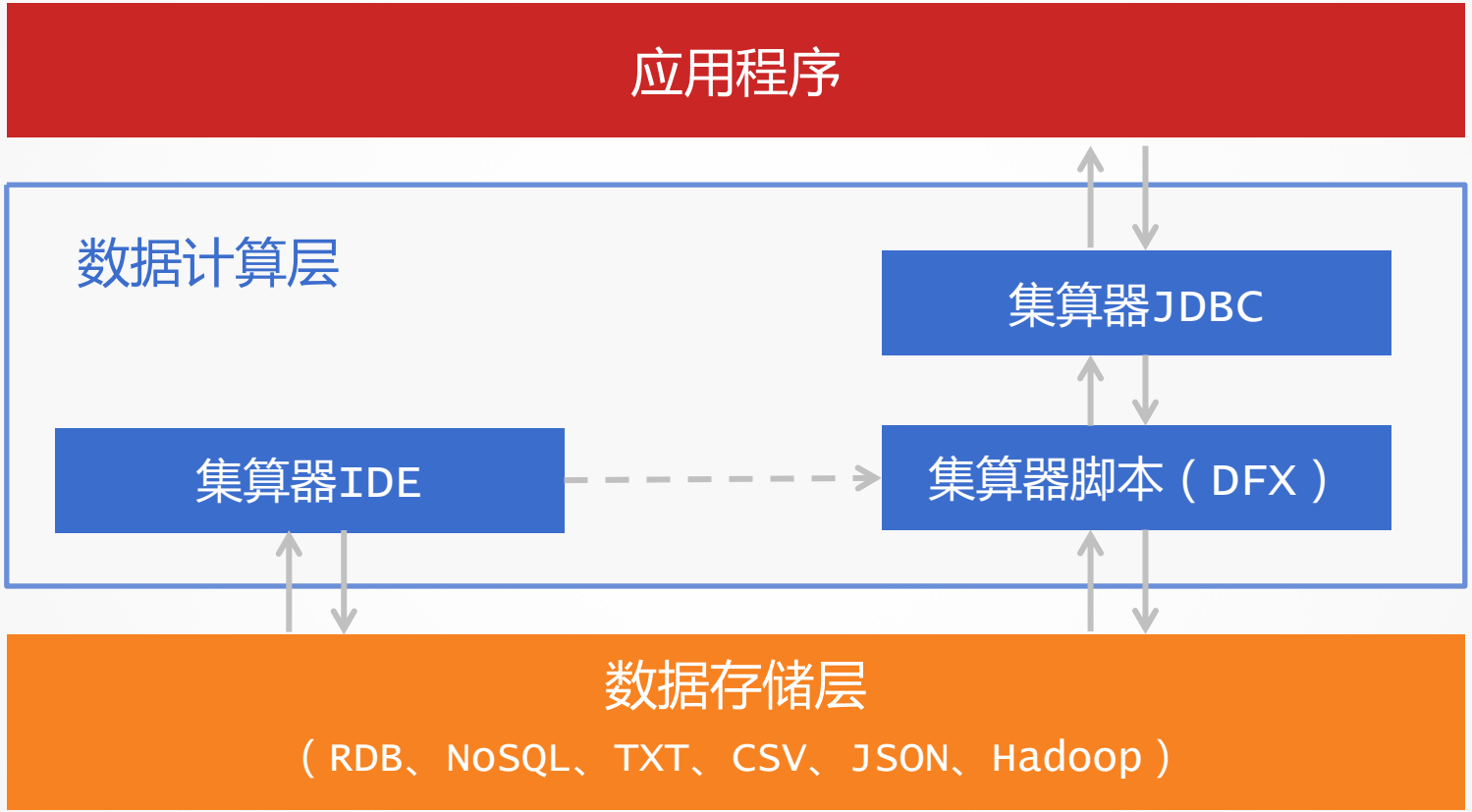
	A	B	C	D	E	F
1	=esProc.query("SELECT 订单ID AS 合同,订购日期 AS 日期,客户 ,订单金额 AS 金额,员工ID AS 销售 FROM 销售记录表 WHERE year(订购日期)=? OR year(订购日					
2	=esProc.query("select * from 员工表")					
3	>A1.run(销售=A2.select@1(编号:A1.销售))		/字段值 是记录			
4	=A1.group(销售)					
5	=create(销售,今年销售额,去年销售额,增长率,客户数,大客户数,大客户占比)					
6	for A4	=A6(1).销售.姓名				
7		=A6.select(year(日期)==年份).sum(金额)		/今年销售额		
8		=A6.select(year(日期)==年份-1).sum(金额)		/去年销售额		
9		=B8/B7-1	/增长率			
10		=A6.group(客户).(~.sum(金额))				
11		=B10.count()	/客户数			
12		=B10.count(~>=[B7,B8])	/大客户数			
13		=B12/B11	/大客户占比			
14		>A5.insert(0,B6,B7,B8,B9,B11,B12,B13)				
15	result A5					

天然分步、层次清晰、直接引用单元格名**无需定义变量**

集算器 – 开发环境


执行、调试执行、单步执行
设置断点
esProc 2012 - Trial Edition
文件(F) 编辑(E) 程序(P) 工具(T) 窗口(W) 帮助(H)
A1
=file("E:\\esProc\\file\\股票记录.txt").import@t()
E:\esProc\file\4连续上涨的股票.dfx
A
B
C
1 =file("E:\\esProc\\file\\股票记录.txt").import@t()
2 =A1.derive(:上涨天数)
3 =A2.sort(股票编码,交易日期)
4 =A3.group@o(股票编码)
5 =A4.run(~.run(上涨天数=if(收盘价<=收盘价[-1],1,上涨天数[-1]+1)))
6 =A5.select(~.max(上涨天数)>arg1).(股票编码)
7 result A6
8
9 /特点:
10 1、支持相对定位访问，比上期、移动平均、累计等常
11
12
A1
股票编码 交易日期 收盘价 上涨天数
120089 2009-01-01 50.24 1
120123 2009-01-01 10.35 1
120136 2009-01-01 43.37 1
120141 2009-01-01 41.86 1
120170 2009-01-01 194.63 1
120243 2009-01-01 15.75 1
120319 2009-01-01 1.36 1
网格变量
序号 名称 值
网格结果所见即所得，易于调试；方便引用中间结果
语法简单，符合自然思维，比其他高级开发语言更简单

集算器 – 应用结构


应用程序
数据计算层
集算器JDBC
集算器IDE
集算器脚本（DFX）
数据存储层
（RDB、NoSQL、TXT、CSV、JSON、Hadoop）

数据源接口

- 高效二进制压缩文件、列式存储
- RDB：Oracle,DB2,MS SQL,MySQL,PG,….
- TXT/CSV，JSON/XML，EXCEL
- Hadoop：HDFS，HIVE，HBASE
- MongoDB，REDIS，…
- HTTP、ALI-OTS

  … …
- 内置接口，即装即用

大数据计算技术

1 遍历技术

2 连接解决

3 存储格式

4 分段并行

5 集群分布

遍历技术

遍历是大数据计算的基础

2 连接解决

连接计算是结构化数据的最大难点

3 存储格式

有效的存储格式是高性能的保证

4 分段并行

数据分段是并行计算的基础

集群分布

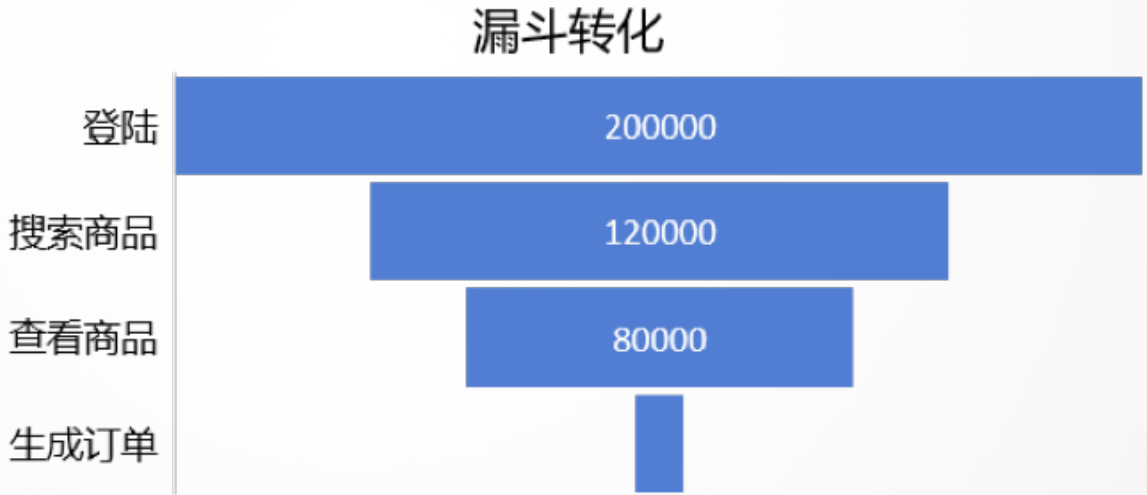
数据与运算的自由分布
获得更高性能

目录 contents

问题解释

电商漏斗分析帮助运营人员分析多步骤过程中每一步的转化与流失情况

如：依次有序触发 “登陆”→“搜索商品”→“查看商品”→“生成订单“ 事件的人群的转化流失情况



关键点

日期过滤　如：仅针对2017-02-01至2017-02-28之间的事件

时间窗口　如：目标事件得在5天之内发生的

目标事件的有序性　如：先浏览商品后放入购物车，反之则不算

事件属性过滤　如：brand=="Apple"，其他品牌的不予统计

难点

时间窗口跨天

如：2017-01-01 12:00 至2017-01-06 12:00之间也是5天

目标事件的顺序不确定

由输入参数决定目标事件的顺序，可能是"浏览商品->放入购物车"，也可能是"放入购物车->浏览商品"

事件属性不确定

浏览商品事件有brand属性，登录事件可能啥属性都没有。

属性以json串的形式存在

传统计算方案

SQL？存储过程？

显然极不适合写这类过程计算

有序计算是SQL的软肋

MapReduce？Spark?

相当于在写UDF

UDF！

理论上可以提高性能

开发工作量太大

缺乏通用性

聚合算法解说 针对单个用户

T =时间窗口 k=目标事件个数 C=long(date(2030,12,31))缺省的初始时间戳

初值有三个：M=最大事件序号，初值为0
A=k+1个最大事件序号，初值为0
S=k+1个初始时间戳，初值为C

假设k=3,T=3600000

M
3

A
3
1
0
0

S
1483600000000
1483602000000
1924876800000
1924876800000

事件ID	时间戳
1	1483200000000
事件ID	时间戳
2	1483201000000
事件ID	时间戳
1	1483600000000
事件ID	时间戳
2	1483601000000
事件ID	时间戳
1	1483602000000
事件ID	时间戳
3	1483602500000

实际代码 聚合算法

	A	B	C	D	E
7	=[0]*dNum	=to(dNum).([0]*I)	=to(dNum).([B1]*I)	=sdID-1	=now()
8	for A3,	for A8	if (p=B8.用户ID-D7,A7(p))==k	next	
9			>n=B8.事件ID, t=O+86400000*B8.日期+B8.时间戳, A=B7(p), S=C7(p)	for t-S(1)>T	>A.shift(),S.shift()
10			>i=A.pselect(n>~)	if n!=A(i)+1	next
11			>if((A(i)=n)==1,S(i)=t)	>if(i==1&&A7(p)<n, A7(p)=n)	>if(i>1&&A(i)==A((j=i-1)), (A.shift(j),S.shift(j)))

优化举例 用户码表和事件码表

- 大数据计算主要慢在遍历
- group函数很好用，用一次遍历一次
- for循环代码复杂，本身效率不如group
- 如何遍历一次同时造出用户和事件码表？

优化方案 数据管道

优化结果 管道的应用

不用管道

	A	
1	=file(文件名).cursor@b()	产生游标
2	=A1.groups(#1:用户ID)	算出用户ID码表
3	=file(文件名).cursor@b()	重新产生游标
4	=A3.groups(#3:事件ID)	算出事件ID码表

代码对比

用管道

	A	
1	=file(文件名).cursor@b()	产生游标
2	=channel()	创建管道
3	=A2.groups(#1:用户ID)	算出用户ID码表
4	>A1.push(A2)	把数据压入管道
5	=A1.groups(#3:事件ID)	算出事件ID码表

源数据6亿条，用户数400万，事件数10个

实际代码 用户码表和事件码表

利用管道, 实现一次遍历多次数据处理

	A	B
1	="c:/funnel_data/"	数据存放目录
2	=directory(A1+"2017*").sort().(file(A1+~:"UTF-8"))	列出目录下所有数据文件
3	=A2.(~.cursor()).conjx()	对所有数据文件循环产生游标，合并成一个对外游标
4	=channel()	产生一个管道（用于一次遍历多次数据处理）
5	=A4.groups(#1:用户id)	对管道数据遍历，造出用户码表
6	>A3.push(A4)	把游标推入管道
7	=A3.groups(#3:事件ID,#4:事件名称;iterate(~~&#5.import@j().fname()):属性名称)	对游标遍历数据，获得事件id、名称、属性名称组成的码表
8	=file(A1+"event.bin").export@b(A7)	把事件码表存入文件
9	=A4.result()	从管道取出运算结果--用户码表
10	=file(A1+"user"+string(A9.len())+".bin").export@b(A9)	把用户码表存入文件，并把用户数编入文件名，方便后续使用

优化举例 事件ID

- 目标事件判断[10007,10004,10010]
- 事件顺序判断
- 对着数组查找成员很慢

优化方案 事件ID序号化

事件ID	事件名称	属性名称
10001	启动	[]
10002	登陆	[]
10003	搜索商品	[content,page_num]
10004	浏览商品	[brand,price]
10005	收藏商品	[]
10006	加入购物车	[]
10007	生成订单	[price_all]
10008	订单付款	[]
10009	评价商品	[how]
10010	退订商品	[]

预处理阶段

id
1
2
3
4
5
6
7
8
9
10

目标事件:[4,7,9,10]

实时计算阶段

Member
0
0
0
1
0
0
2
0
3
4

=file(fPath+file).iselect@b(start:end,日期).select((事件ID=A1(事件ID),事件ID>0&&(${filter})))

优化结果 目标事件序号化

采用原事件ID

2	=[1004,1007, 1009,1010]
3	=T.select(A2.pos(事件ID)>0)

代码对比

目标事件序号化

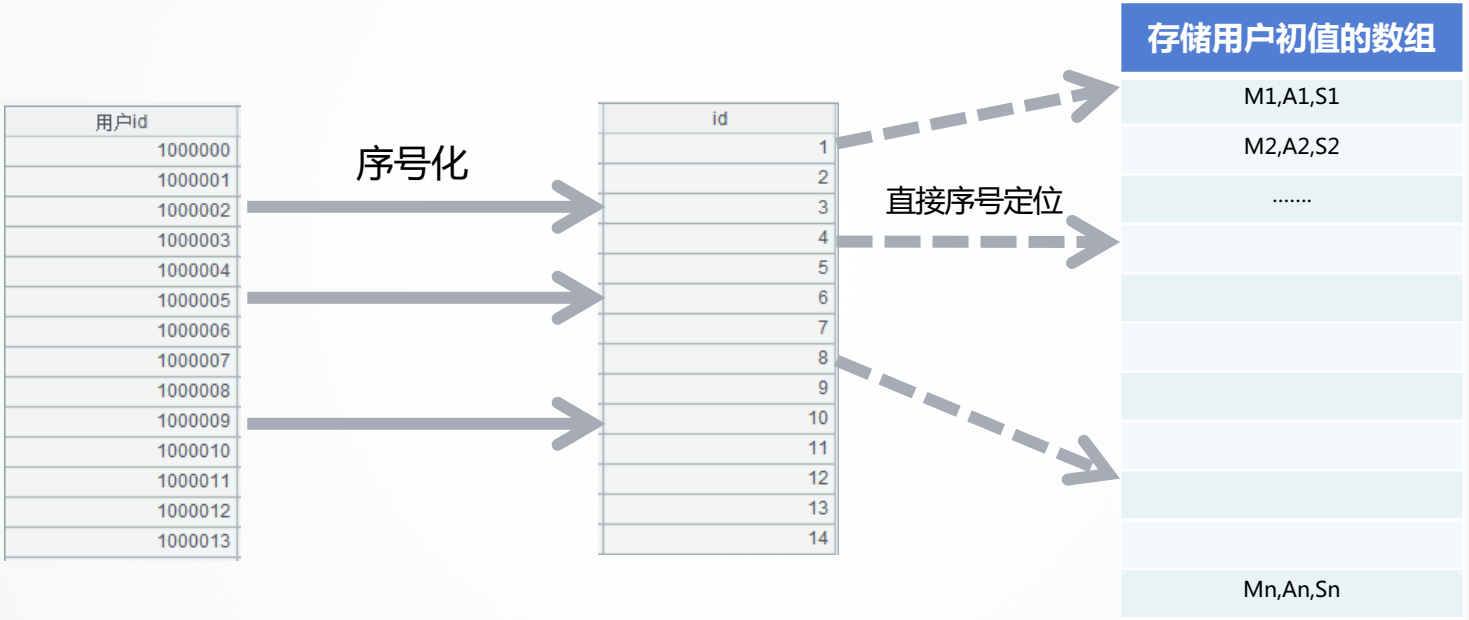
2	=[0,0,0,1,0,0,2,0,3,4]
3	=T.select((事 件 ID=A2(事 件 ID), 事 件 ID>0))

测试数据量：7500万条，选出1880万条

优化举例 用户ID序号化

- 每个用户需要存储三个初值M,A,S
- 400万个用户需要存400万组初值
- 从大数组里查找 成员很慢
- 序号直接定位快很多

优化方案 用户ID序号化



优化结果 用户序号化

采用原用户ID：

	A	
1	=long(date(2030,12,31))	起始时间戳的初值
2	=file(“user.bin”).import@b().derive(0:M,[0]*(k+1):A,[A1]*(k+1):S)	用户初值
3	=1000.(A2.select(用户ID==rand(4000000)+1000000)	需要遍历查找

代码对比

用户ID序号化：

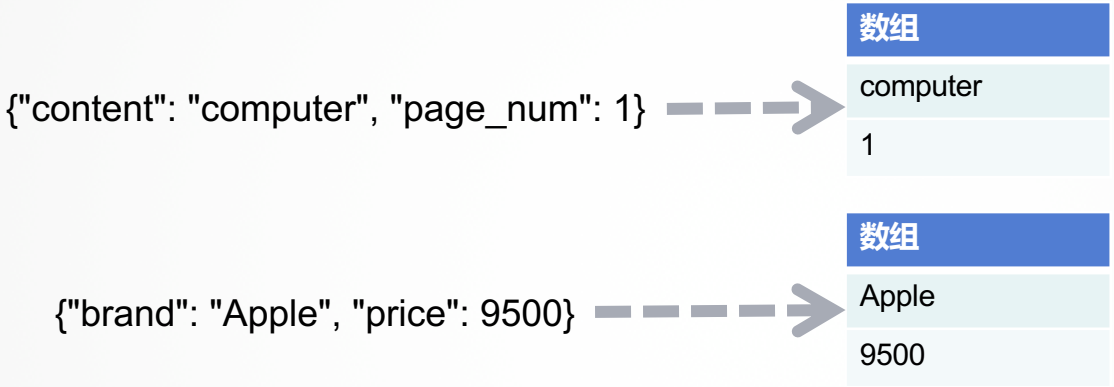
	A	
1	=long(date(2030,12,31))	起始时间戳的初值
2	=to(4000000).([0,[0]*(k+1),[A4]*(k+1)])	用户初值
3	=1000.(A2(rand(4000000)+1))	直接用序号定位

400万个用户，1000次的查找定位

优化举例 事件属性

- 属性以json串的形式保存
- 字符串比对很慢，难以实现复杂的过滤需求
- 不同事件的属性项不一样
- 转成json对象或者序表浪费存储空间

优化方案 事件属性数组化



造事件码表的时候把事件属性整理好，事先规定好属性顺序，连属性名也省了大量节约存储空间，也加快读数速度

优化结果 事件属性数组化

代码对比：

	A	B
1	=file(fPath+"20170101序表.bin").cursor@b().select(事件ID==10004 && 事件属性.brand=="Apple")	
2	for A1,100000	=B2+A2.len()

序表化

3	=file(fPath+"20170101数组化.bin").cursor@b().select(事件ID==4 && 事件属性(1)=="Apple")	
4	for A3,100000	=B4+A4.len()

数组化

性能解说：

事件属性以json串的形式存在，不管以字符串对比的方式，还是序表的方式过滤，都比较慢，如果变成数组的形式存储，直接用数组成员对比的方式过滤，则会快很多。

性能对比：

事件属性序表化	事件属性数组化
5.6秒	3.1秒

测试数据量：1000万，过滤出31万

实际代码 用户ID,事件ID序号化,事件属性数组化

序号化便于数组查找定位, 属性数组化便于存储和过滤

按用户ID拆分文件

	A	B	C	
11	=ceil(A9.len()/nFile)			nFile是传入的参数：文件个数 算出每个文件的用户数
12	=A2.(~.cursor()).conjx()			重新对文件产生游标
13	=A12.switch(#1,A9:用户id;#3,A7:事件ID)			把用户,事件变成对应的码表记录
14	for A13,10000	>A14.run(#1=#1.#,#3=#3.#,a=A7(#3). 属 性 名 称 ， b=#5.import@j(),#5=a.(b.field(~)))		把用户,事件变成码表序号,事件属性变成数组
15		=A14.group(int((#1-1)/A11))		根据每个文件能放的用户数分组
16		for B15	=file(A1+"tmp"+string(int((B15(1).#1-1)/A11)+1)+".bin").export@ba(B15,#1:用户ID,#2:时间戳,#3:事件ID,#4:事件名称,#5:事件属性,#6:日期)	把记录以追加的方式写入相应的文件里

实测问题：多线程与多进程

- 多线程并行数达到一定程度后性能不升反降
- 多线程之间内存共享，java需要分配内存
- 改用多进程后，并行数一直升到机器核数，性能始终呈上升状态

解决方案 多线程和多进程

代码对比：

	A	B	
17	=callx(fPath+"漏斗转化subnew.dfx",A15, wNum,[event]*p,filter,eNum, start,end,A12,A16, fPath+"funnel_data/")		调用子程序，并获得子程序返回的数据

多线程并行

	A	B	
17	=callx(fPath+"漏斗转化subnew.dfx",A15, wNum,[event]*p,filter,eNum, start,end,A12,A16, fPath+"funnel_data/"; A11)		调用子程序，并获得子程序返回的数据

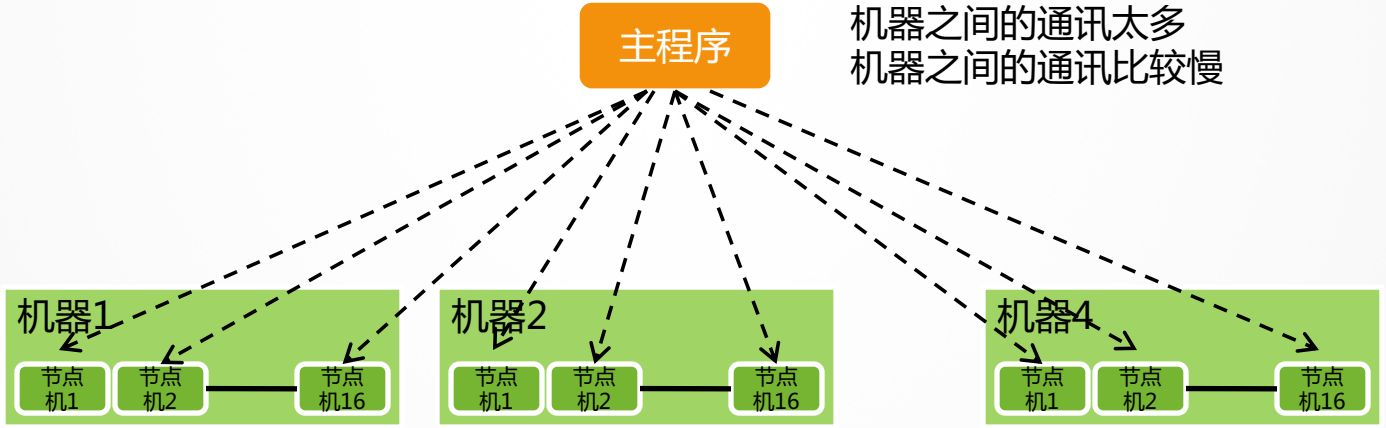
多进程并行

性能对比：

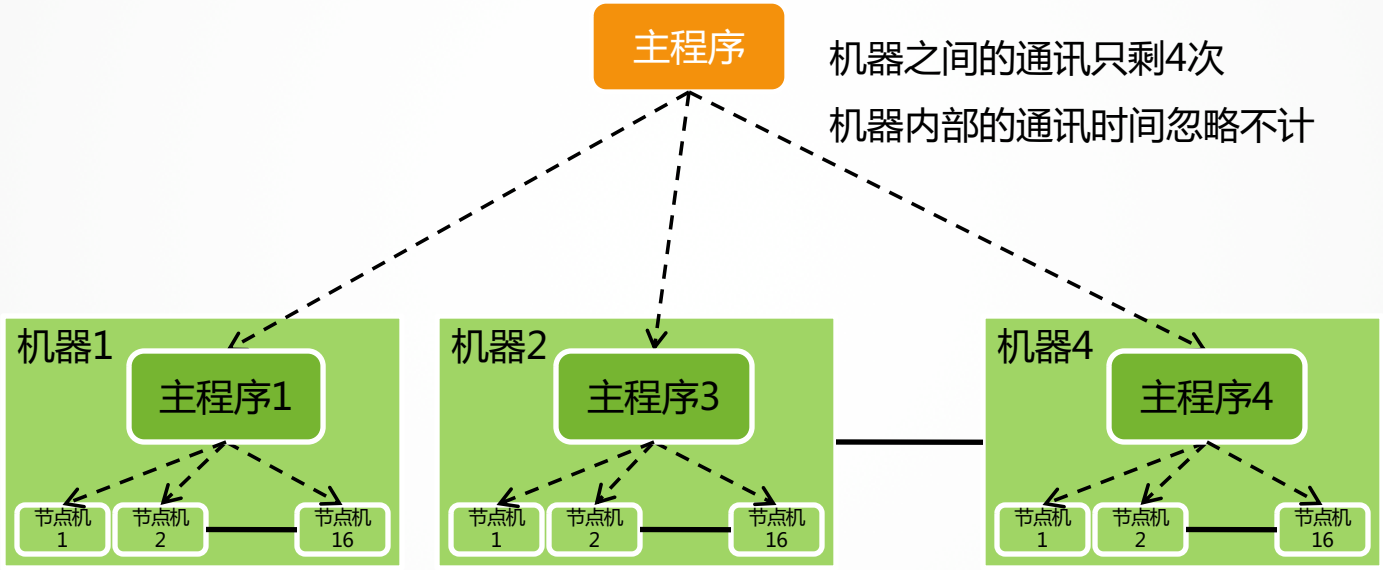
32核的服务器

实测问题：单机和多机

- 第一轮测试是单机，32核，没考虑多机通讯问题
- 第二轮实测机器是四台机器，每台16核
- 采用每台机器跑16个节点机的方式



解决方案 三层结构


主程序
机器之间的通讯只剩4次
机器内部的通讯时间忽略不计
机器1
主程序1
节点机1
节点机2
节点机16
机器2
主程序3
节点机1
节点机2
节点机16
机器4
主程序4
节点机1
节点机2
节点机16

优化结果 三层结构

代码对比：

两层结构

	A	B	
17	=callx(fPath+" 漏 斗 转 化 subnew.dfx",A15,wNum,[event]*p,filter,eNum,start,end,A12,A16,fPath+"funnel_data/"; A11)		调用子程序

三层结构

Main:

	A	B	
17	=callx("mainsub.dfx",wNum,[event]*4,filter,eNum,A3,start,end,A2,p,d,to(n),fPath+"day/",C7;B2)		调用二层主程序

Main_sub:

	A	B	
17	=callx("subday.dfx",dNum,wNum,[event]*d,filter,eNum,start,end,B2,A3,fPath;A2)		调用子程序

性能对比：

两层结构	三层结构
6.5秒	5.1-5.2秒

总结体会

过程计算是普遍问题

- 大数据平台大都在努力优化SQL，无助于此类问题的解决
- 过度依赖UDF，计算平台本身失去意义

性能优化是不断试错迭代的过程

- 需要敏捷的开发工具快速做出原型测试
- UDF开发太慢，不适应性能优化

集算器的代码敏捷与高效性

短小精干的代码

- 核心聚合算法10行，整个计算29行，加上数据整理的13行，总代码行数不超过50
- 完全没有用UDF

丰富的集合运算类库与高性能工具集

- 有序集合，位置引用，...
- 序号化、并行计算、数据自由分布、迭代式聚合、...

网格式代码直观易写

- 特别适合过程式运算

集算器的设计特点

面向过程式计算

- 原创的离散数据模型易于描述复杂过程
- 高性能算法可以迅速实现

注重单机性能优化

- 集群性能以单机为基础

注重小内存运算

- 不能事事指望大内存

集群透明性和高性能之间有效权衡

- 牺牲少量管理便利性换取高性能

新版本展望

集算器的数据仓库功能将进一步提高性能

- 排序后可采用列存，进一步减少存储量
- 组合表可用大文件代替碎文件，易于管理
- 序号键可高性能过滤出分段，适应更大数据量
- ForkReduce技术自动实现多节点运算，代码还能更短

敬请期待

横跨全年的品牌传播与实效营销盛宴

年度技术峰会+ArchData巡回
覆盖全国华北、华东、华南、西南、西北地理区域中心城市
200+媒体传播，受众超千万

年度技术峰会
北京、上海、成都
杭州技术沙龙
广州技术沙龙
深圳技术沙龙
西安技术沙龙
南京技术沙龙
厦门技术沙龙
北京技术沙龙
成都技术沙龙
上海技术沙龙

全域连横

同城合纵

以上海为例

年度技术峰会
成都技术沙龙
走进大数据案例企业
走进电商案例企业
走进金融案例企业
科技厂商探访
企业痛点诊断
旅游交流活动
成都技术沙龙
年终技术峰会

年度技术峰会+一杆到底，
围绕北京/上海/成都，经济标杆城市，步步为营，深耕用户
多重形式，复数传播，挖掘地域商务价值

合纵连横，在中国开发者群体中缔造品牌营销奇迹

中生代技术
FRESHMAN TECHNOLOGY

ArchData技术峰会
聚焦人工智能、大数据、基础架构、区块链等前沿课题

中生代技术提供咨询内训服务
技术架构，研发管理，敏捷开发，大数据
微服务，AI，机器学习等

中生代技术提供人才服务
对接研发主管，内推精准人才