我对后端优化的一点想法

DTCC2012

About Me

- Jametong@童家旺
- work@alipay (2010.8-)
- work@alibaba(2005.5-2010.8)
- work@浙江移动台州公司 (2003.12-2005.5)
- Blog @ http://www.dbthink.com/
- mail@ jametong@gmail.com
- Weibo @ Jametong

内容简介

- 什么是优化?
- 响应时间 Vs 吞吐量
- 性能与可伸缩性 (Performance Vs Scalability)
- Instrument & metrics
- 需要了解的一点硬件知识
- 常见案例分析
- 引用资料

什么是优化(1)

- The fastest way to do something is don‘t do it
  - Anonymous
- Two ways to improve performance, do it less or do it faster
  - Anonymous
- Performance is all about code path
  - From Cary Millsap
  - http://carymillsap.blogspot.com/2010/09/my-otn-interview-at-oow2010-which-hasnt.html

什么是优化(2)

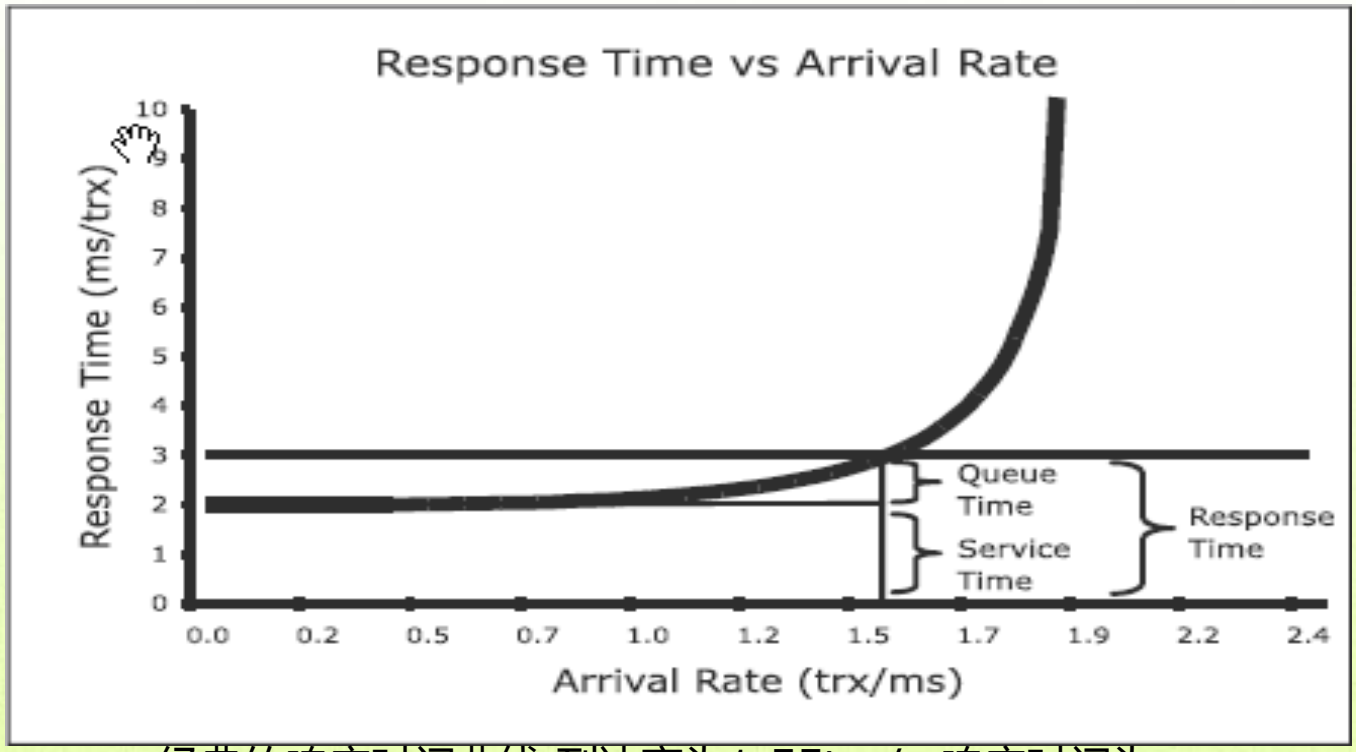
- 不访问不必要的数据
  - 使用B*Tree/hash等方法定位必要的数据
  - 使用column Store或分表的方式将数据分开存储
- 合理的利用硬件来提升访问效率
  - 使用缓存消除对数据的重复访问
  - 使用批量处理来减少交互次数（磁盘、网络）
  - 使用新硬件来降低后端的延时，提高效率
- 提高系统的吞吐量
  - 对工作单元进行细化，减少串行操作
  - 优化硬件配置，提高整体的TCO与硬件利用率
  - 合理的拆分（水平、垂直拆分）以提高系统的整体吞吐能力

响应时间 Vs 吞吐量(1)

- 性能
  - 衡量完成特定任务的速度或效率
- 响应时间
  - 衡量系统与用户交互式多久能够收到响应
- 吞吐量
  - 衡量系统在单位时间里可以完成的任务量

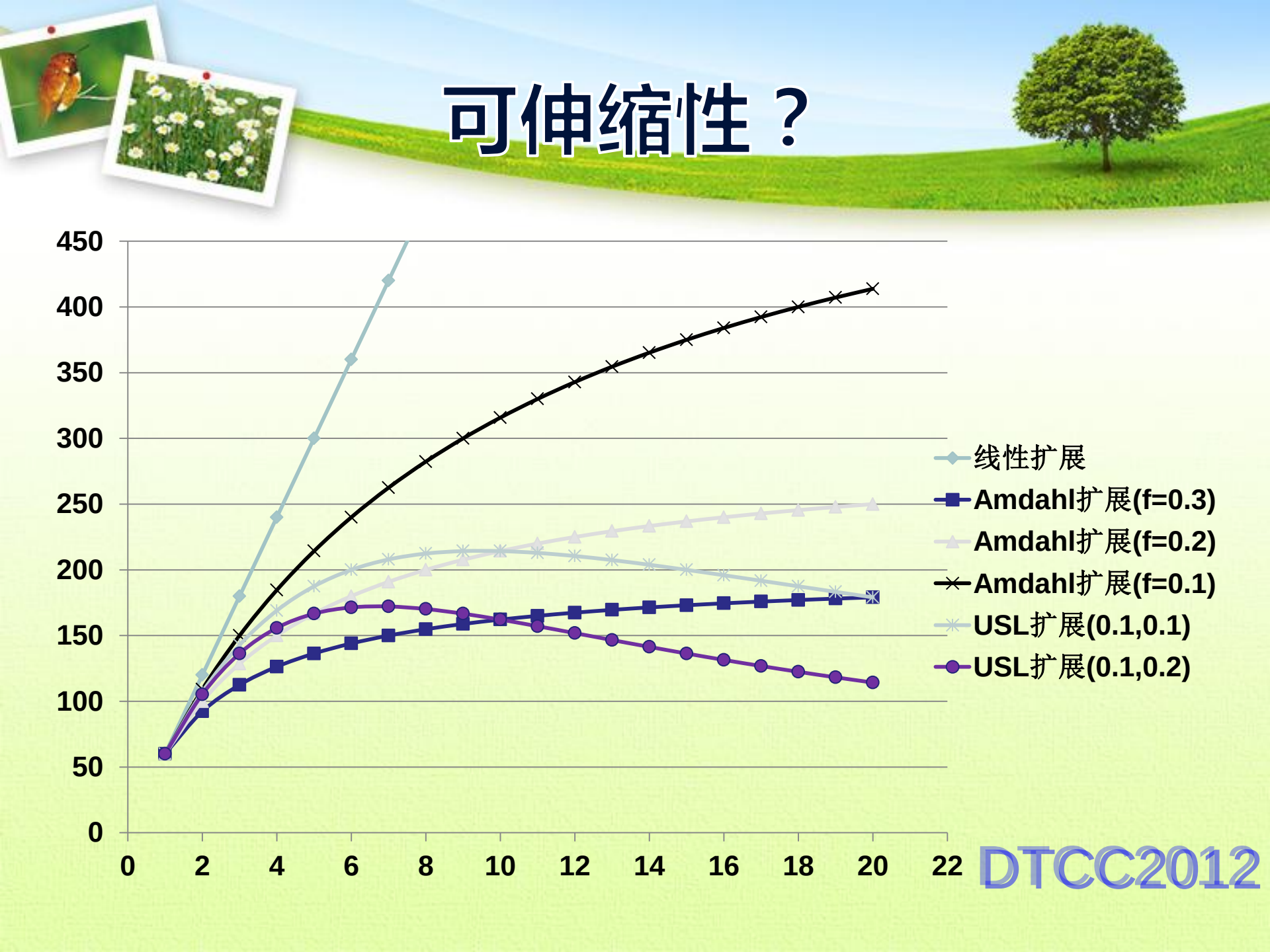
响应时间 Vs 吞吐量

Response Time = Service Time + Queue Time



经典的响应时间曲线.到达率为1.55trx/s,响应时间为3ms/Trx,服务时间为2ms/Trx,排队时间为1ms/trx

可伸缩性？


450
400
350
300
250
200
150
100
50
0
0
2
4
6
8
10
12
14
16
18
20
22
线性扩展
Amdahl扩展(f=0.3)
Amdahl扩展(f=0.2)
Amdahl扩展(f=0.1)
USL扩展(0.1,0.1)
USL扩展(0.1,0.2)

可伸缩性

- Amdahl's law
  - 使用多处理器进行 并行处理能够提升的性能的比例受限于程序中需要串行处理的比例
- USL Scalability
  - 使用多处理器进行 并行处理能够提升的性能的比例不仅受限于程序中需要串行处理的比例，还受限于进程之间的并发系数.

Instrument & Metrics

- What gets measured gets managed.
  - *Peter Drucker (彼得. 德魯克)*
- Don't guess, get the data
  - Anonymous

Instrument & Metrics@life

- 从杭州→北京（花费了6个半小时）

Instrument & Metrics@life

- 从杭州→北京（花费了6个半小时）
  - 13:00 – 13:15 从公司下楼到淘宝(15分钟)
  - 13:30 – 13:50 从淘宝出发到上出租车(20分钟)
  - 14:00 - 15:00 在出租车上,从淘宝<->机场(60分钟)
  - 15:10 - 15:20 拿机票(10分钟)
  - 15:25 - 15:50 安检(25分钟)
  - 16:00 – 17:00 在机场候机(60分钟)
  - 17:00 - 18:20 飞机上,杭州→北京(80分钟)
  - 18:20 - 18:40 到出租车上车点(20分钟)
  - 18:40 - 18:55 等待出租车(15分钟)
  - 18:55 - 19:50 机场到酒店(55分钟)

Instrument & Metrics@Oracle

- Metrics
  - v$sys_time_model & v$sess_time_model
  - v$sysstat & v$sesstat
  - v$system_event & v$session_event
  - v$session_wait & v$event_histogram
- Instrument
  - Extended 10046 trace

```
Elapsed times include waiting on following events:
  Event waited on                          Times   Max. Wait  Total Waited
  ---------------------------------------  Waited  ----------  ------------
  SQL*Net message to client                100001        0.00          0.19
  db file scattered read                      347        0.03          0.39
  SQL*Net message from client              100001        0.01          6.91
  db file sequential read                       2        0.00          0.00
********************************************************************************
```

DTCC2012

Instrument & Metrics@Linux

- vmstat & iostat & netstat & tcprstat & sar
- strace & oprofile & systemtap
- aspersa & latencytop & top

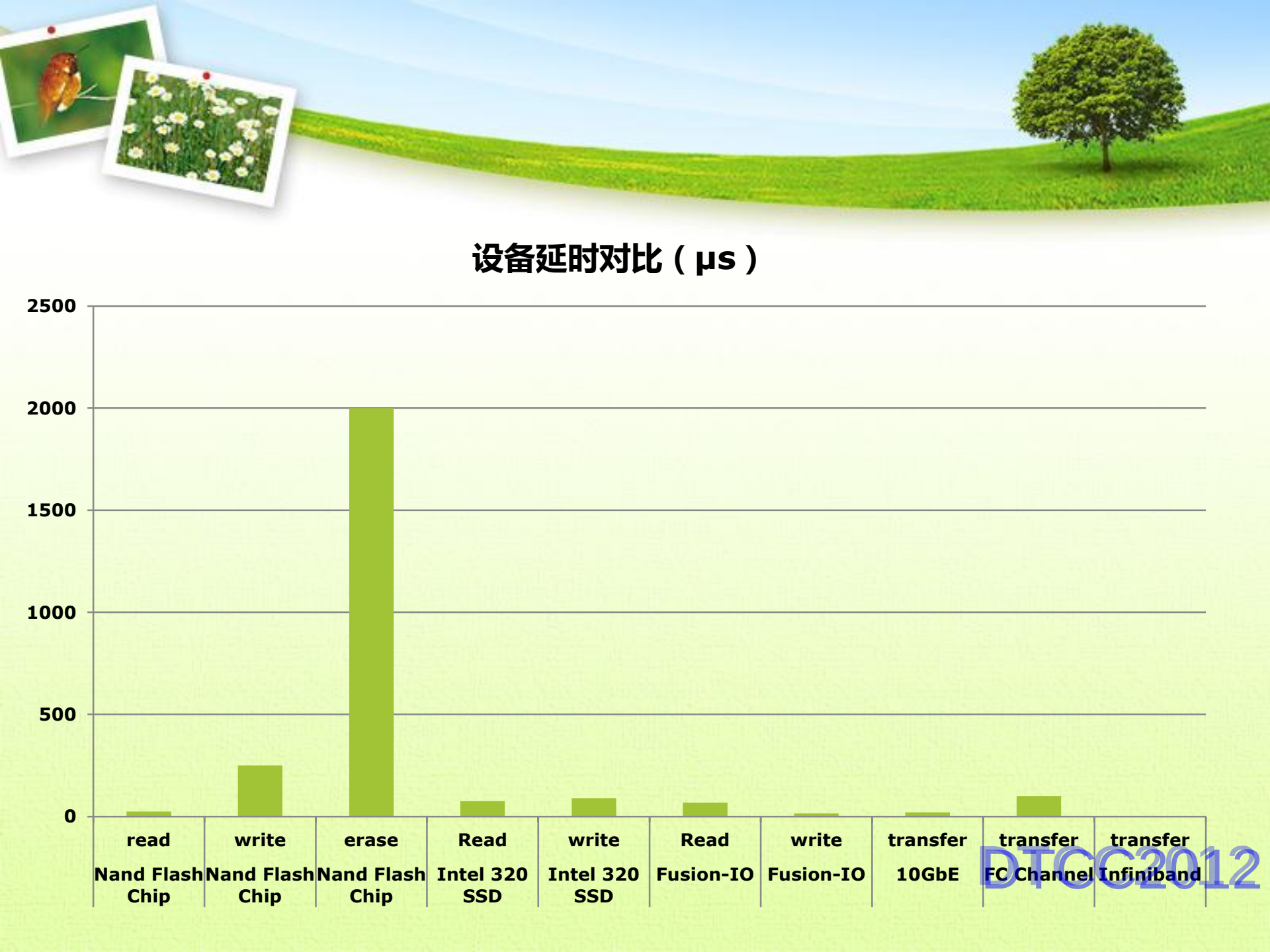
```
[oracle@mytest ~]$ ps -ef | grep dbw
oracle    8323     1  0  2010 ?        00:42:29 ora_dbw0_mytest
[oracle@mytest ~]$ strace -c -p 8323
Process 8323 attached - interrupt to quit
Process 8323 detached
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 98.39    0.007194          37       195           pwrite
  1.61    0.000118           0       644           times
  0.00    0.000000           0         1           read
  0.00    0.000000           0         1           open
  0.00    0.000000           0         1           close
  0.00    0.000000           0        10           getrusage
  0.00    0.000000           0        11        11 semtimedop
------ ----------- ----------- --------- --------- ----------------
100.00    0.007312                   863        11 total
```

Numbers you Should Know

- L1 cache reference 0.5 ns(1GHz CPU)
- Branch mispredict 5 ns
- L2 cache reference 7 ns
- Mutex lock/unlock 25 ns
- **Main memory reference 100 ns**
- Compress 1K bytes with Zippy 3,000 ns
- **Send 2K bytes over 1 Gbps network 20,000 ns**
- Read 1 MB sequentially from memory 250,000 ns
- **Round trip within same datacenter 500,000 ns**
- **Disk seek 10,000,000 ns (7200rpm SATA)**
- Read 1 MB sequentially from disk 20,000,000 ns
- Send packet CA->Netherlands->CA 150,000,000 ns

Storage层面的一些性能指标

- 传统磁盘的IO特性
- Nand Flash本身的IO特性
  - Read 25μs
  - Write (program) 250 μs
  - Erase 2000 μs
- Intel 320 SSD
  - Read 75 μs
  - Write 90 μs
- Fusion-IO
  - Read：68μs
  - Write: 15μs
- Storage通道的处理延时
  - 10GbE的Latency ： 10-50 μs
  - Fabric Channel 的latency : 20-50 μs
  - InfiniBand 的Latency ：1-3 μs


设备延时对比（μs）
2500
2000
1500
1000
500
0
read
Nand Flash Chip
write
Nand Flash Chip
erase
Nand Flash Chip
Read
Intel 320 SSD
write
Intel 320 SSD
Read
Fusion-IO
write
Fusion-IO
transfer
10GbE
transfer
FC Channel
transfer
Infiniband
DTCC2012

优化数据访问

- 优化传统B*Tree 优化数据访问
  - 如何更好的设计索引
- 使用Write Optimized B-Tree优化数据访问
  - Stratified B-trees （ Acunu)
  - LSM tree (BigTable,Cassandra,LevelDB)
  - Fractal Tree Indexes(TokuDB)
- 使用基于Hash的算法访问数据
  - Oracle Hash Cluster , BDB , Bitcask

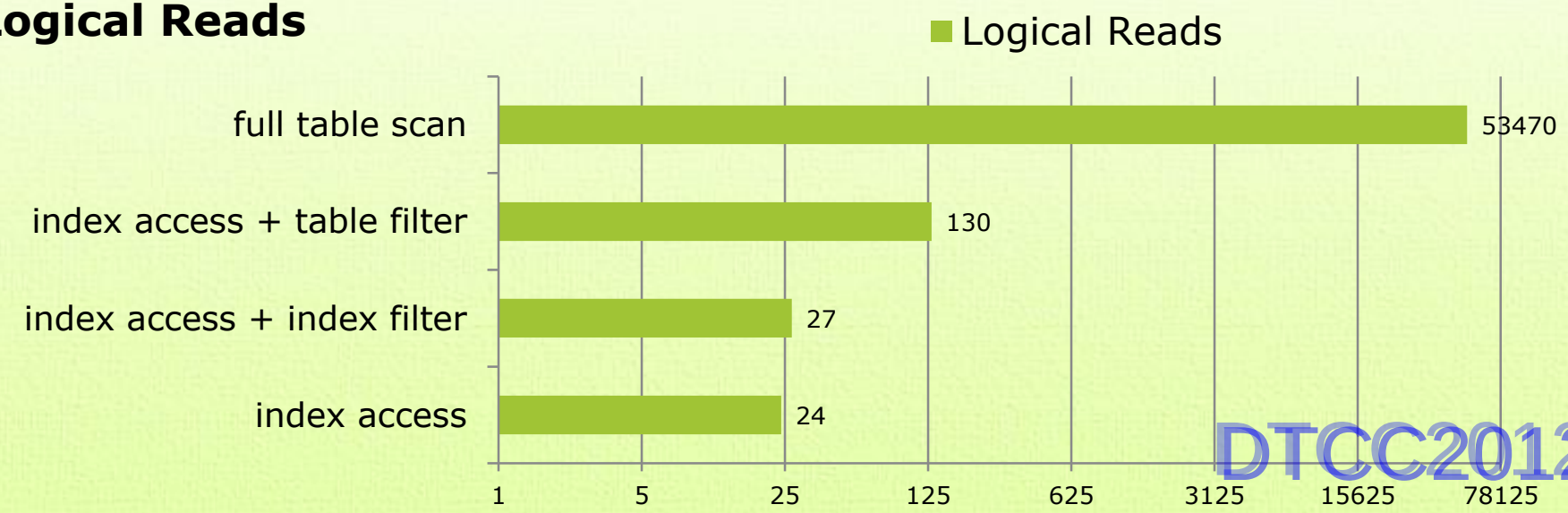
B*Tree优化数据访问

- 模拟场景
  - 表中有5个字段(id, status, type, gmt_create, content)
  - 主要基于三个字段的组合进行查询
    - 字段status, type总共有30个组合(status 5个值,type 6个值)
    - 每天有10000条记录,时间随机分布(共60天的数据)
  - 执行的查询,分页取第一个20条记录
  - select id,status,type,gmt_create,content from (select id,status,type,gmt_create,content,rownum rn
    from (select id,status,type,gmt_create,content from james_t
    where status = '00' and type = '05' and gmt_create >= trunc(sysdate) and gmt_create < trunc(sysdate+1)
    order by gmt_create desc) where rownum <= 20) where rn >= 0;
  - 分别创建三种不同的索引进行测试
    - create index james_t_idx on james_t(status, type, gmt_create);
    - create index james_t_idx on james_t (status, gmt_create);
    - create index james_t_idx on james_t (gmt_create, status, type);

B*Tree优化数据访问

index columns	description	Logical Reads
status, type, gmt_create	index access	24
gmt_create, status, type	index access + index filter	27
status, gmt_create	index access + table filter	130
no index	full table scan	53470

Logical Reads



行存储 Vs 列存储

- 场景介绍
  - 表VeryBigTable含有30个列
  - 表的记录数为50,000,000条
  - 平均每个用户为300条左右
  - 其中有2个列属于详细描述字段,平均长度为2k
  - 其它的列的总长度平均为250个字节
  - 此表上的查询有两种模式
    - 列出表中的主要信息(每次20条,不包含详细信息,90%的查询)
    - 查看记录的详细信息(10%的查询)
  - 保存在Oracle数据库中,默认block size(8k)
- 要求
  - 对此业务进行优化
  - 分析数据,说服开发部门实施此优化

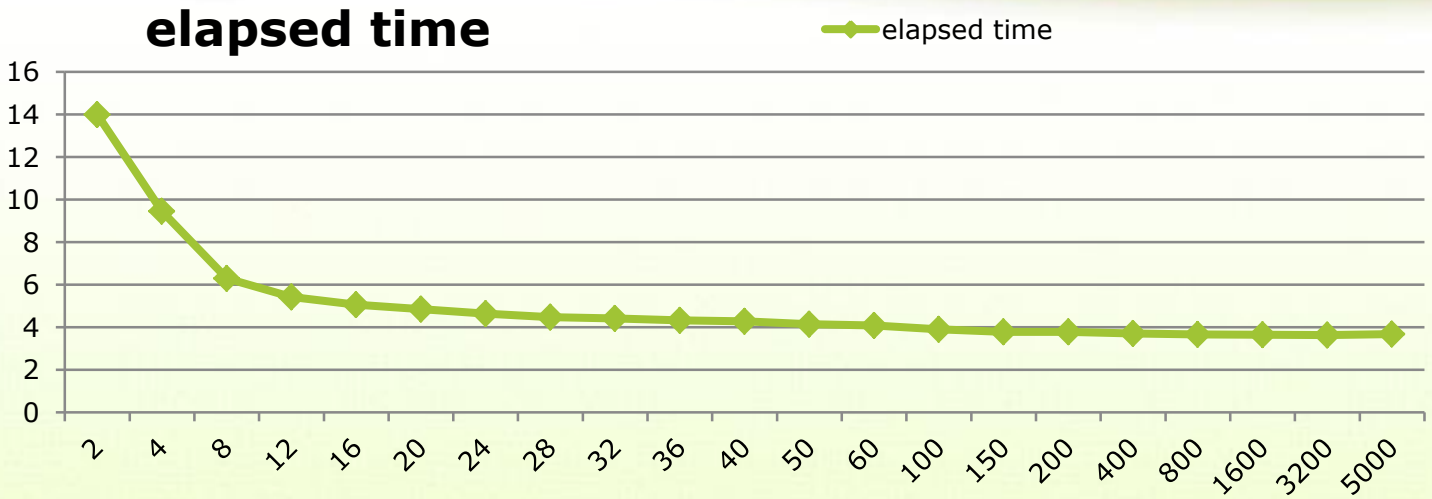
分表存储来提高性能(2)

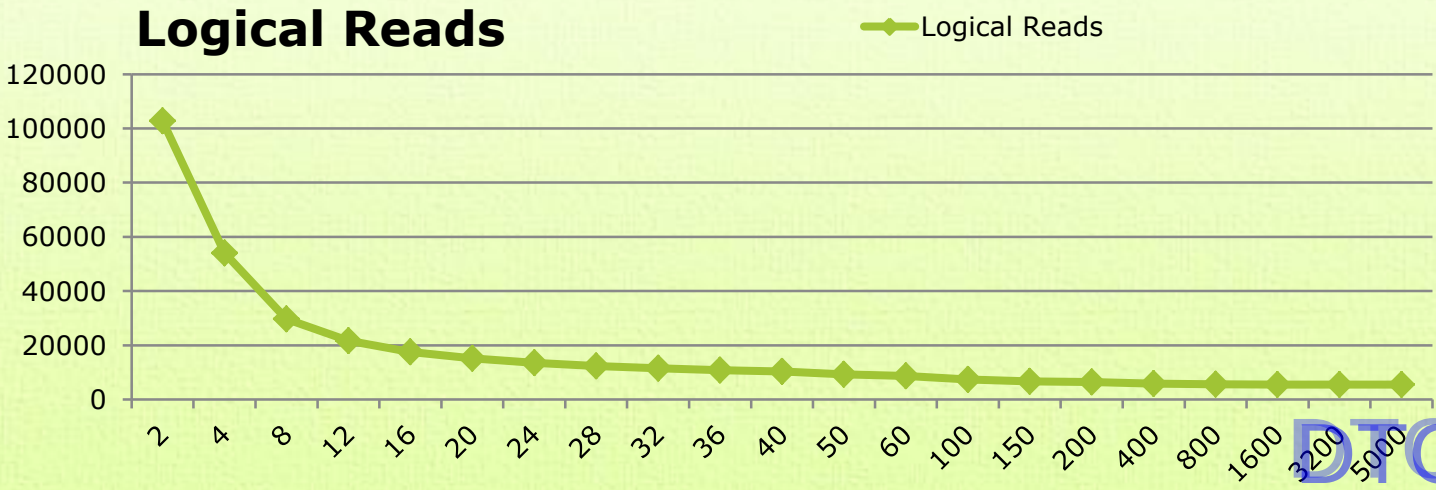
- 性能分析
  - 每块记录数
    - 8192 * 0.80(1) / 250 = 25.5 (主表)
    - 8192 * 0.80 / 2000 = 3.27(详情表)
    - 8192 * 0.80 / (2000 + 250) = 2.91
  - 访问的逻辑IO(内存块访问)
    - List的查询代价
    - 改进后=(300/25.5) * y + 4 + x = 4 + x + 11.8y = 4^2 + 7^3 + 11.8 * 1.5^4 = 28.7
    - 改进前=(300/2.91) * y + 4 + x = 4 + x + 103.y = 4 + 7 + 103 * 1.5 = 165.5
  - 访问涉及到的物理读(磁盘块访问)
    - List的查询代价(逻辑IO * (1 – 命中率))
    - 改进后=28.7 * (1 – 0.85^5) = 4.305
    - 改进前=165.5 * (1 – 0.85) = 24.825
  - 访问时间(ms)
    - 改进后=逻辑IO时间+物理IO时间= 28.7 * 0.01^6 + 4.305 * 7^7 = 30.422ms
    - 改进前=逻辑IO时间+物理IO时间= 165.5 * 0.01 + 24.825 * 7 = 175.43ms

使用缓存优化重复查询

- 使用场景
  - 缓存的一致性维护问题
  - 数据的具体读写比
  - 变更频率
  - 业务对一致性的要求
  - 使用何种缓存方式.
- 注意事项
  - 考虑缓存的刷新策略
  - 考虑缓存的数据延迟对业务的影响
  - 考虑缓存失效时,系统的支撑能力
  - 参考缓存工具: MemCached, Tair, Redis

批量处理接口





批量处理的注意事项

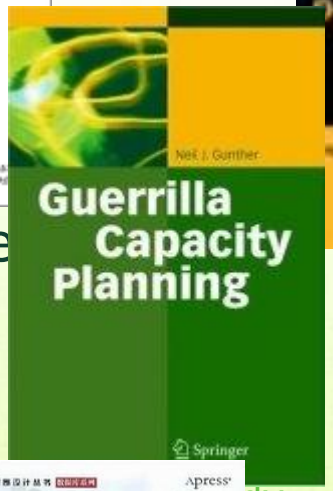
- 在有批量需求的情况下,尽可能提供批量处理接口,如memcached的multiget,或者cassandra的getslice,针对RDBMS,可以使用批量接口(如Oracle的Array Interface).
- 不要设置过大的批次大小
  - 这需要与对应的程序heap 空间
  - 设置过大可能会导致程序出现OOM错误
  - 网络send/receive_buffer大小(/proc/sys/net/core/wmem|rmem_max|default)

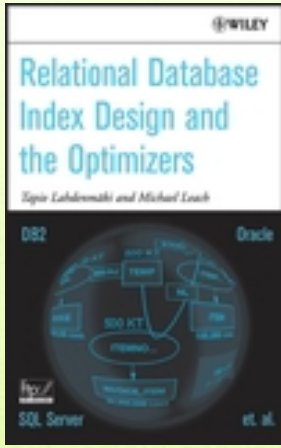
优化硬件配置

- Five minute rule?
  - Price / Performance Break Point?
- 一台机器配置的代价
  - 机架成本
  - 电力成本（机器电力？Cooling？）
  - CPU license成本（Oracle？Veritas ？）
  - 机器成本？
  - 使用PCI-E SSD？
  - 使用HBA？InfiniBand？交换机？
  - 多使用内存？多使用磁盘？
  - 单机故障对业务的影响代价？

Recommended Reading

- Optimizing Oracle Performanc
  - By Cary Millsap, Jeff Holt
- Forecasting Oracle Performanc
  - By Craig Shallahamer
- Guerrilla Capacity Planning
  - By Neil J. Gunther
- Relational Database Index Design and the
  - By Tapio Lahdenmaki
- Think Clearly About Performance
  - By Cary Millsap
  - http://www.method-r.com/dow ails/ rly-about-performance
- TroubleShooting Oracle P e
  - By Christian Antognini
- 性能诊断艺术
  - Translated By 冯大辉/胡怡文





谢谢

Q & A

DTCC2012