High performance computing

Arthur Whitney
kx.com
Chairman and founder
domain

trading (million events per second)
analysis (trillion orders, quotes, trades, ..)
realtime risk management
surveillance
monte carlo simulation
customers

banks
hedge funds
exchanges
data providers
..
kdb+ avg db: 350 billion records
kdb+ max: one trillion records

in the last 12 months ..
buy/sell orders: add, modify, delete

400 billion buy records
400 billion sell records
130 billion quotes
10 billion trades
realtime trading

3 billion complex transactions per day
peak 300,000 transactions per second
memory ops (not flops)

<table>
<thead>
<tr>
<th>Mode</th>
<th>MOPS</th>
<th>cache</th>
<th>mem</th>
<th>flash</th>
<th>disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>seq</td>
<td>1000M</td>
<td>200M</td>
<td>?</td>
<td>50M</td>
<td></td>
</tr>
<tr>
<td>rnd</td>
<td>1000M</td>
<td>10M</td>
<td>?</td>
<td>0.0001M</td>
<td></td>
</tr>
</tbody>
</table>

ROPS (records per second)

- select: 1M–100M
- insert: 1M–10M
- update: 100K+
new language

general purpose programming
relational database, timeseries analysis, messaging, webserver, ..

always try to take over the entire stack.
observation

good

people are willing to learn new languages for benefits in expression and performance, e.g. our parallel language and rdbms(kdb+)

bad

still hard to use even 10’s of cores well except for monte carlo and trivial scans
q (aka kdb+)

parallel programming language
parallel primitives, e.g. x+y
parallel operators, e.g. x[..]’y

parallel rdbms + timeseries
select insert update delete
select from trade where 0<deltas price
leftjoin, asofjoin, windowjoin, ..
regnms

/ 1.7 seconds (1.4 with 2core)
select from aj[`sym`time;trade;quote]
  where not price within(bid;ask)

/ 2.7 seconds (1.7 with 2core)
select from wj[-3000 1000;`sym`time;
    trade;(quote;(max;`ask);(min;`bid))]
  where not price within(bid;ask)
price mbs (dec 2007)

$10,000,000,000,000
  100,000,000 loans
  10,000,000 pools
  10,000 deals
  100,000 bonds

1000 paths (over 360 months each)
1000 cpu grid. 20 hours to 20 minutes.
tpcd example

l - lineitem
o - order
c - customer
p - part
s - supply
n - nation
r - region
sql92 (query 8)

```
select year, sum(case when name='BRAZIL' then rev
else 0 end)/sum(rev) from(
    select extract(year from o.d) as year, l.x*(1-l.xd) as rev, n2.name
    from p, s, l, o, c, n n1, n n2, r
    where p.p=l.p and s.s=l.s and l.o=o.o and o.c=c.c
    and c.n=n1.n and n1.r=r.r and r.name='AMERICA'
    and s.n=n2.n and o.d between date'1995-01-01' and
    date'1996-12-31' and
    p.t='ECONOMY ANODIZED STEEL') t
group by year order by year
```
q (query 8)

```
select rev wavg s.n=`BRAZIL
by o.d.year from l
where
  o.c.n.r=`AMERICA,
o.d.year in 1995 1996,
p.t=`"ECONOMY ANODIZED STEEL"
```
language

functional
atom, list, dict
short programs
byte code interpreter
code goes to data
reference count (no cycles)

100K c code. 1000 lines.