**Historical Microstructure Computation**

**基本算法：**

1. 以十分钟为一个bucket,按下述公式计算每个bucket里的指标。最后用每个bucket的当天成交量计算加权平均

2. 最后取过去21天的中值。如果计算麻烦，则退化为以全天交易时间长度为一个bucket

**ats:Average Trade Size**

**tradePeriod: the average trade time between 2 trades**

**注:如果不用分bucket的方式来计算，则使用下述公式**

**quoteSize: the time weighted average quote size**

**turnoverPeriod: the average time spent for executing the whole trade**

**tickPeriod: the average time between price change**

**twSpread: time weighted spread**

skiped

**spreadPeriod**

或

其中，指第i笔trade发生时的spread =ask1 – bid1.

按照后一个公式，spread指这个bin内时间加权的max(ticksize, ask1-bid1)

两个quote的时间间隔

**adv20:average daily value 20**

**mdv21:median daily value 21**

**几个特殊处理规则**

**停牌处理**

注：对DailyFact，如果当日停牌，则不予计算

对Fact21，如果停牌，则往前找，直到凑够21个交易日，如果从当天往前60天的交易日里还凑不够21个交易日，那么当天的值不予计算。

这些缺失engine启动时按照default.sh或者default.sz替换并给出提示，方便查找交易问题。

**default值计算**

分为上海和深圳两个交易所：

对上海（深圳一样），当天上海交易所所有有意义的dailyFact，取中值

暂时不对SecurityType做区分，非股票的数量很少，取中值影响不大

**除数为0的处理**微观结构计算中除数为0的数据处理，对每个bucket来说  
1. 每天将除数不为0的进行计算，假设20130711有2000只股票，对每个微观结构值，每个bucket做个分位数分析，取出q1（1%分位数）和q99（99%分位数），  
   对于除数为0的，做如下运算。这样所有除数为0的bucket里面的值都替换为一个有效值。

2. 按照bucket流量的加权方式计算

|  |  |  |  |
| --- | --- | --- | --- |
|  | 除数为0处理方法 | 每日计算后处理方法，在每个组内 | 21日中值处理方法outlier处理方法 |
| ats | 替换为q1 | q1 – q99 | Floored at 1 lotsize |
| tradePeriod | q99 | q1 – q99 | [0.1, 10000] |
| quoteSize | q99 | q1 – q99 | Na |
| turnoverPeriod | 由前三个计算得出 | q1 – q99 | [0.1, 10000] |
| tickPeriod | q99 | q1 – q99 | [0.1, 10000] |
| spreadPeriod | 除数不会为0 | q1 – q99 | [0.1, 10000] |
| twspread | 除数不会为0 | q1 – q99 | For stock, floored at 0.0001  For future, floored at 0.00001 |

3. 对spreadPeriod，如果ask1或者bid1为0，则人工构造一个对手价  
a) ask=0, bid<>0, 则ask=bid + ticksize (0.01或0.001)  
b) bid=0, ask<>0, 则bid = ask - ticksize (0.01或0.001)

直接将spreadPeriod=0.001(ETF), spreadPeriod = 0.01(股票），spreadPeriod = 0.01(逆回购，这里的0.01）

4.  对twSpread,如果ask1或者bid1为0，则人工构造一个对手价。如：

a) ask=0, bid<>0, 则ask=bid + ticksize (0.01或0.001)  
b) bid=0, ask<>0, 则bid = ask - ticksize (0.01或0.001)

**特殊交易日的处理**

交易量分布计算时，需要取过去21天的中值

1. 对于股票，1月和7月的第一个交易日、当月的最后一个交易日剔除，然后再计算

2. 1月、7月的第1个交易日、当月的最后一个交易日应该单独计算过去21天的，因为这些日子的交易量分布比较特殊。

暂时的解决方法，剔除1中的特殊日期，然后用前一天代替。

**ccp, daccp的缺失处理**

ccp取最近一个交易日的收盘价

daccp取最近一个交易日的除权除利息日收盘价（即第二天交易所发布的preclose）

若ccp，daccp找不到相应的值，用-1处理

股票的closePrice和adjustedClose可取到；

基金和转债的可取到closePrice，设djustedClose=closePrice；

回购的closePrice和adjustedClose取不到，均设为-1

**volume distribution:**

**确定股票分组（待定）**

**1st step**

对每只股票，按照十分钟间隔，取每只股票当天所有bucket的成交量v(b1) ...v(b24)

**2nd step**

根据股票分组，对当天组的所有股票按照下列公式计算

以常用的10分钟间隔为例，则对每只股票每天要计算24个时间段，计算Bin1...Bin24

**3rd step, 转换成百分比**

**4th step**

注：开盘集合竞价和收盘集合竞价单算

**5th step - normalize**

指数和exchange两个维度进行计算，未来可能扩展为多个维度

最近21个交易日需要满足下列条件：(1)不是当月最后一个交易日；(2)不是1月或7月第一个交易日。

**Real Time Data Service**

1. **Level II market data**

op 今天开盘价

hp:从开盘到目前为止的最高价

lp:从开盘到目前为止的最低价

lu:limit up (当天涨停价）

ld:limit down (当天跌停价)

adv20-- **microstructure**

mdv21 -- **microstructure**

vs:当前的累计成交量

vs1s:上午成交量

vs2s:下午成交量

vwp: 从开盘到目前为止的vwap

vwpvs:从开盘到目前为止的符合VWAP条件的成交量

tp:最近一笔成交价

daccp:除权除息后的前一个交易日收盘价- **microstructure**

trdst: Trade Status -- 当前的交易时段 AUCT (集合竞价）、OBTRD(连续交易时段)、BREAK(午市休op盘)、CLOSE（收盘）、SUSP（停牌）、HALT（临时停牌）

auctype:集合竞价的类型 OPEN/CLOSE

indauc: 集合竞价中的拟成交价

indaucs：集合竞价中的拟成交量

vol：过去21天的年化volatility

Lotsize(100) - 从数据库取

ticksize(0.01)-从数据库取

1. **定时请求**

注册某股票从开盘时刻起每t(通常为1,5,60)分钟的一个snapshot，包括以下信息，但不仅限于下述数据:

vwp1: 在第1个t分钟内发生的交易的加权平均价

vwpvs1:在第1个t分钟内发生的交易量

ncnt1: 在第1个t分钟内发生的交易笔数

lgts1:在第一个t分钟内发生的最大的一笔交易量

每t分钟计算一次上述值，并通知

1. **区间请求**

注册某股票从t1到t2时刻每笔交易发生后的snapshot。需支持的请求接口可包括下述选项（部分选项不是必须项），但请考虑以后的可扩展性：

1. 开始时间 t1 (必须)
2. 结束时间 t2 (必须)
3. 时区(可选)
4. 价格(可选)
5. 是否包含集合竞价
6. Exclude成交量大于某个值的交易

该数据接口应包括以下信息，但不仅限于下述数据:

vwp:从t1到min(now, t2)间发生的交易的加权平均价

vwp\_below:从t1到min(now, t2)间发生的成交价低于价格参数的交易的加权平均价

vwp\_at:从t1到min(now, t2)间发生的成交价等于价格参数的交易的加权平均价

vwp\_above从t1到min(now, t2)间发生的成交价大于价格参数的交易的加权平均价

vwpvs:从t1到min(now, t2)间发生的总成交量

vwpvs\_below:从t1到min(now, t2)间发生的成交价低于价格参数的交易的总成交量

vwpvs\_at:从t1到min(now, t2)间发生的成交价等于价格参数的交易的总成交量

vwpvs\_above从t1到min(now, t2)间发生的成交价大于价格参数的交易的总成交量

hp:从t1到min(now, t2)间发生的交易的最高成交价

hp\_below

hp\_above

lp:从t1到min(now, t2)间发生的交易的最低成交价

lp\_below

lp\_above

tcnt:交易笔数

tcnt\_at

tcnt\_below

tcnt\_above

每当一笔交易发生时，需要更新以上值，并通知

**注：**

**1. 交易时段，engine可能重复发同一个key。**

对计算服务来说，第一次收到key，发送status=ACK和相应的field，默认值为“-”；接着立即计算，每一次有更新，按照前述算法完成计算后照常发送。后续如果收到重复的key，发送status=RESULT和最近一笔的计算值，后续照常计算。

**2. 中午休盘时，如果收到请求。**

1）如果重复的key，那么发送最近一个计算的值（不用再发ACK）

2）如果新请求，计算最近一个可用行情（或者成交）来时的值

对于实时计算，如果12:00发送[DATA\_CALC\_601600\_sh\_1\_rt](http://jg-yangfei1:8161/admin/send.jsp?JMSDestination=DATA_CALC_601600_sh_1_rt&JMSDestinationType=topic)，先发status=ACK和默认的key=value对，这里的value都是“-”，然后计算最后一笔行情来时的值（即11:30最后一笔行情来时的整个上午的统计值），开盘后正常计算和发送。

对于区间请求，如果12:00发送DATA\_CALC\_600435\_sh\_[excAMAuction=false:excLargeSize=5000:excSmallSize=100]\_20130905-09:15:00\_20130905-15:00:00\_Interval，先发status=ACK和默认的key=value对，这里的value都是“-”，然后按照前述算法计算整个早上的值并发布，下午开盘后照常计算和发布

对于rolling的，如果收到DATA\_CALC\_600000\_sh\_[interval=10]\_rolling，先发status=ACK和默认的key=value对，这里的value都是“-”，然后计算上午收盘前最后一个10分钟的值，并发布，开盘后照常计算。

**3.收盘后，如果收到请求，对于上述三种计算，按照下列计算方法**

1）实时数据：存储最后一笔行情时的计算值即全天的统计值。后续注册收到的都是这个值

2）区间数据：如果发送9:00-10:00的请求，那么计算这个区间内最后时间的相应值

3）定时数据：存储收盘前最后一个区间的值。例如下午四点请求DATA\_CALC\_600000\_sh\_[interval=10]\_rolling，那么发送的数据应当是14:50-15:00的相应值。

