bftrader开发计划

欢迎大家讨论！

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 任务 | 说明 | 预计完成时间 | 进度 | 里程碑 |
| Allinone版本 | | | | |
| Gateway接口定义 | Sdk/bftrader.proto  Sdk/bfgateway.proto  Sdk/bfproxy.proto | 2016.4.15 | 完成 | 0.1 |
| CtpGateway实现 | Ctp网关  Ctpgateway | 2016.4.25 | 完成 | 0.1 |
| Datafeed接口定义 | Sdk/bfdatafeed.proto | 2016.4.26 |  | 0.1 |
| Datafeed实现 | 历史数据服务  Datafeed | 2016.4.28 |  | 0.1 |
| Cta接口定义 | Sdk/bfcta.proto  Sdk/bfrobot.proto  Sdk/bfctaengine.proto | 2016.4.29 |  | 0.1 |
| Ctaengine实现 | Ctaengine | 2016.5.10 |  | 0.1 |
| Cta Python支持库 | 兼容vnpy策略接口  不一样的地方：  参数保存在cta  取参数调用函数 | 2016.5.15 |  | 0.1 |
| Python策略例子 | DataRecorder  DoubleEmaDemo  OrderManagementDemo | 2016.5.20 |  | 0.1 |
|  |  |  |  |  |
| 分布式五件套版本 | | | | |
| ctpGateway | 拆分独立运行 | 2016.6.1 |  | 0.2 |
| Ctaengine | 拆分独立运行 | 2016.6.1 |  | 0.2 |
| Datafeed | 拆分独立运行 | 2016.6.1 |  | 0.2 |
|  |  |  |  |  |
| 辅助工具 | | | | |
| BtGateway实现 | 回测网关，出报告和日志，图形界面 |  |  | 0.3 |
| Assist实现 | 远程管理工具 |  |  | 0.3 |
| ibGateway | 盈透网关 |  |  | 0.3 |
| Dataframe实现 | 常用时间序列函数库；C导出接口 |  |  | 0.3 |
|  |  |  |  |  |

数据服务设计-20160426

1. 目的

在云端运行的策略，需要补少量的数据才能运行，需要一个小型数据库来保存或者自己收集。

Cta数据库和btest数据库是不一样的，btest数据库是回测研发策略用的，可能有100年的数据，不可能搬到云端。

1. 需求

策略需要补当天（盘中中断或者中途连接上）或者前一天的tick（刚开盘，可能需要前一天的tick做一些事情）

策略需要查询1分-1周期最近的60根bar

1. 数据库设计

Cta数据库，由2个文件构成：

Tick：只保留收集最近两个交易日的tick，收盘后（以下午4点为准）自动清理过期的tick。

Bar：初始数据库手工拷贝到bar目录；之后每天收盘后，根据tick自动计算当天的bar补上。

1. API设计

getPreTick(to); //取to之前的那个tick，只能取两天内的。

getPreBar(period,to); //取to之前的那个bar，period为周期；只能取当天之前的bar。

1. 说明

A 设计如此细粒度的api，是考虑简单性。策略补tick和bar时候，每个ontick里面，可以只补100个，这样，不要多少个tick，就可以补齐了，而不会造成后续tick的堵塞。

B 当天的bar，由策略自己根据tick算！（需要讨论，或者当天的bar只能取1分钟的？）

1. 分歧

Gateway只负责推送tick和补tick，bar计算和保存+tick保存，都是策略的事情。

Gateway应该做一个最小型的数据中心支持策略跑起来，应该负责推送tick+bar，和保存tick+bar，比如各1024根。

数据服务设计-20160427

1. 由于大周期的k线，结束时间不好确定，如60分钟的，中场有休息，还有夜盘。每个合约都不一样，要数据中心统一生成，需要有交易时段表，才可以。交易时段，从开始到结束，分钟累加，就是k；对于策略来说，知道自己的交易时段，可以快速计算和生成，只需要提供历史1分钟的数据就可以了。
2. 这样，数据中心只需要保留tick m1 和 d1，其他周期，策略自己去算。Allinone的策略不带数据库。
3. Tick保留2-5天就行了；方案，切换actionday时候，记录actionday = curday，加上 pre\_curday =old\_actionday，类似这样就行。然后把x天之前的删除就是了。
4. M1保留得3600根，相当于半个月，用于计算60分钟的60均线。M1可以一直保留。
5. D1需要保留300天，相当于一年左右，用于计算周线的60均线。D1可以一直保留。
6. 提供当天的tick和m1的推送。(m1可以不推送，高人请忽略)。
7. 提供当天的tick和m1的查询。
8. 提供历史的tick m1 d1的查询。
9. 提供m1和d1的导入功能（3600根/300根）。

数据服务设计-20160428

1. 考虑到策略需要的数据千变万化比如需要加权指数，内置的数据中心很难算这个，既然做不到极致，那就不做。具体见：《关于数据收集和整理的一点说明.txt》
2. 只提供推送tick服务，提供内置数据库插入 查询 删除数据
3. 提供inserttick insertbar接口
4. 提供loadtick loadbar接口
5. 提供delteTick delteBar接口
6. 提供getContract addContract接口（保存加权指数），看是否要给出onContract的推送接口，和tick统一；这样数据服务完全是给策略使用的了，allinone本身自己不用了。