Hbase 接口性能测试

1. **HBase API性能测试**
2. **写入测试**
3. 测试环境：

|  |  |
| --- | --- |
| 表文件大小 | 1.2G |
| 表字段数 | 3 |
| 操作线程数 | 1 |
| 字段平均长度 | 60B |

测试结果：

|  |  |  |  |
| --- | --- | --- | --- |
| 线程ID | 数据量 | 操作时间(秒） | 平均每秒处理量（行） |
| 1 | 1000万 | 235 | 42553 |

1. 测试环境：

|  |  |
| --- | --- |
| 表文件大小 | 1.5G |
| 表字段数 | 124 |
| 操作线程数 | 1 |
| Region个数 | 6 |

测试结果：

|  |  |  |  |
| --- | --- | --- | --- |
| 线程ID | 数据量 | 操作时间(秒） | 平均每秒处理量（行） |
| 1 | 100万 | 966 | 1035 |

备注：

1. 操作时间包括数据对象的解包时间43s，去除该时间，平均处理效率为1083.
2. 数据总共分6个region，测试记录看到在region分裂的时候hbase会停止服务，停止服务为10s-40s时长不等。
3. 和上一环境对比如下：

124个字段/3个字段=41.3

42553/1083=39.3

数据差不多，猜测HBase的性能主要还是和Put的数据长度有关。

1. 对于操作的对象大，进程的GC情况还未做优化。
2. 测试环境：

|  |  |
| --- | --- |
| 压缩 | Snappy |
| 操作线程数 | 1 |
| Region个数 | 2 |
| 数据量 | 100万 |

测试结果：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试次数 | 表文件大小 | 表字段数 | col value平均长度 | 操作时间(秒） | 平均每秒处理量（行） |
| 1 | 355.7 M | 30 | 72 | 215 | 4651 |
| 2 | 296.5 M | 15 | 144 | 88 | 11364 |
| 3 | 271.2 M | 10 | 216 | 77 | 12987 |
| 4 | 262.3 M | 5 | 432 | 61 | 16393 |

1. **Scan读取测试**
2. 测试环境：

|  |  |
| --- | --- |
| 表文件大小 | 1.5G |
| 表字段数 | 124 |
| Region个数 | 4 |

测试结果：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试线程数 | 线程ID | 数据量 | 查询数据 | 操作时间(秒） | 平均每秒处理量（行） |
| 1个线程 | 1 | 100万 | 200000 | 23 | 8696 |
| 2个线程 | 1 | 100万 | 200000 | 28 | 7143 |
| 2 | 100万 | 200000 | 28 | 7143 |
| 3个线程 | 1 | 100万 | 200000 | 35 | 5714 |
| 2 | 100万 | 200000 | 34 | 5882 |
| 3 | 100万 | 200000 | 35 | 5714 |
| 10个线程 | 1 | 100万 | 200000 | 45 | 4444 |
| 2 | 100万 | 200000 | 43 | 4651 |
| 3 | 100万 | 200000 | 38 | 5263 |
| 4 | 100万 | 200000 | 43 | 4651 |
| 5 | 100万 | 200000 | 39 | 5128 |
| 6 | 100万 | 200000 | 45 | 4444 |
| 7 | 100万 | 200000 | 43 | 4651 |
| 8 | 100万 | 200000 | 43 | 4651 |
| 9 | 100万 | 200000 | 46 | 4348 |
| 10 | 100万 | 200000 | 44 | 4545 |

1. **Get读取测试**
2. 测试环境：

|  |  |
| --- | --- |
| 表文件大小 | 1.5G |
| 表字段数 | 124 |
| Region个数 | 4 |

测试结果：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试线程数 | 线程ID | 数据量 | 查询数据 | 操作时间(秒） | 平均每秒处理量（行） |
| 1个线程 | 1 | 100万 | 50000 | 91 | 549 |
| 2个线程 | 1 | 100万 | 50000 | 96 | 521 |
| 2 | 100万 | 50000 | 96 | 521 |
| 3个线程 | 1 | 100万 | 50000 | 98 | 510 |
| 2 | 100万 | 50000 | 98 | 510 |
| 3 | 100万 | 50000 | 98 | 510 |
| 10个线程 | 1 | 100万 | 50000 | 100 | 500 |
| 2 | 100万 | 50000 | 100 | 500 |
| 3 | 100万 | 50000 | 100 | 500 |
| 4 | 100万 | 50000 | 100 | 500 |
| 5 | 100万 | 50000 | 100 | 500 |
| 6 | 100万 | 50000 | 100 | 500 |
| 7 | 100万 | 50000 | 100 | 500 |
| 8 | 100万 | 50000 | 100 | 500 |
| 9 | 100万 | 50000 | 100 | 500 |
| 10 | 100万 | 50000 | 100 | 500 |

1. **Thrift接口测试**
2. **写入测试**
3. 测试环境：

* 类型一：

|  |  |
| --- | --- |
| 表文件大小 | 14.37M |
| 表字段数 | 3 |
| ThriftServer | 1 |

测试结果：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试线程数 | 线程ID | 数据量 | 操作时间(秒） | 平均每秒处理量（行） |
| 1个线程 | 1 | 100000 | 63 | 1587 |
| 2个线程 | 1 | 100000 | 89 | 1123 |
| 2 | 100000 | 89 | 1123 |
| 3个线程 | 1 | 100000 | 91 | 1098 |
| 2 | 100000 | 91 | 1098 |
| 3 | 100000 | 91 | 1098 |
| 10个线程 | 1 | 100000 | 104 | 961 |
| 2 | 100000 | 107 | 935 |
| 3 | 100000 | 107 | 935 |
| 4 | 100000 | 107 | 935 |
| 5 | 100000 | 103 | 971 |
| 6 | 100000 | 98 | 1020 |
| 7 | 100000 | 105 | 952 |
| 8 | 100000 | 103 | 971 |
| 9 | 100000 | 107 | 935 |
| 10 | 100000 | 107 | 935 |

* 类型二：

|  |  |
| --- | --- |
| 表文件大小 | 187.3M左右 |
| 表字段数 | 124 |
| ThriftServer | 1个，占用内存159M左右 |

测试结果：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试线程数 | 线程ID | 数据量 | 操作时间(秒） | 平均每秒处理量（行） |
| 1个线程 | 1 | 100000 | 190 | 526 |
| 2个线程 | 1 | 100000 | 208 | 481 |
| 2 | 100000 | 206 | 485 |
| 3个线程 | 1 | 100000 | 236 | 424 |
| 2 | 100000 | 242 | 413 |
| 3 | 100000 | 241 | 415 |
| 10个线程 | 1 | 100000 | 388 | 258 |
| 2 | 100000 | 391 | 256 |
| 3 | 100000 | 391 | 256 |
| 4 | 100000 | 387 | 259 |
| 5 | 100000 | 390 | 256 |
| 6 | 100000 | 392 | 255 |
| 7 | 100000 | 389 | 257 |
| 8 | 100000 | 387 | 258 |
| 9 | 100000 | 393 | 254 |
| 10 | 100000 | 390 | 256 |

1. **读取测试**
2. 测试环境：

|  |  |
| --- | --- |
| 表文件大小 | 1.5G |
| 表字段数 | 124 |
| Region个数 | 4 |

测试结果：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试线程数 | 线程ID | 数据量 | 查询数据 | 操作时间(秒） | 平均每秒处理量（行） |
| 1个线程 | 1 | 100万 | 50000 | 46 | 1087 |
| 2个线程 | 1 | 100万 | 50000 | 54 | 926 |
| 2 | 100万 | 50000 | 54 | 926 |
| 3个线程 | 1 | 100万 | 50000 | 59 | 847 |
| 2 | 100万 | 50000 | 59 | 847 |
| 3 | 100万 | 50000 | 59 | 847 |
| 10个线程 | 1 | 100万 | 50000 | 71 | 704 |
| 2 | 100万 | 50000 | 69 | 725 |
| 3 | 100万 | 50000 | 70 | 714 |
| 4 | 100万 | 50000 | 71 | 704 |
| 5 | 100万 | 50000 | 71 | 704 |
| 6 | 100万 | 50000 | 70 | 714 |
| 7 | 100万 | 50000 | 70 | 714 |
| 8 | 100万 | 50000 | 67 | 746 |
| 9 | 100万 | 50000 | 71 | 704 |
| 10 | 100万 | 50000 | 70 | 714 |

1. **Rest接口测试**
2. **写入测试**
3. 测试环境：

|  |  |
| --- | --- |
| 表文件大小 | 10M左右 |
| 表字段数 | 124 |
| 操作线程数 | 1 |
| RESTServer | 1个，占用内存 225M左右 |

测试结果：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试线程数 | 线程ID | 数据量 | 操作时间(秒） | 平均每秒处理量（行） |
| 1个线程 | 1 | 10000 | 49 | 204 |
| 2个线程 | 1 | 10000 | 50 | 200 |
| 2 | 10000 | 50 | 200 |
| 3个线程 | 1 | 10000 | 58 | 172 |
| 2 | 10000 | 58 | 172 |
| 3 | 10000 | 58 | 172 |
| 10个线程 | 1 | 10000 | 117 | 85 |
| 2 | 10000 | 114 | 88 |
| 3 | 10000 | 117 | 85 |
| 4 | 10000 | 116 | 86 |
| 5 | 10000 | 114 | 88 |
| 6 | 10000 | 117 | 85 |
| 7 | 10000 | 117 | 85 |
| 8 | 10000 | 114 | 88 |
| 9 | 10000 | 114 | 88 |
| 10 | 10000 | 116 | 86 |

1. **Scan读取测试**
2. 测试环境：

|  |  |
| --- | --- |
| 表文件大小 | 1.5G |
| 表字段数 | 124 |
| Region个数 | 4 |

测试结果：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试线程数 | 线程ID | 数据量 | 查询数据 | 操作时间(秒） | 平均每秒处理量（行） |
| 1个线程 | 1 | 100万 | 20000 | 45 | 444 |
| 2个线程 | 1 | 100万 | 20000 | 71 | 282 |
| 2 | 100万 | 20000 | 72 | 278 |
| 3个线程 | 1 | 100万 | 20000 | 99 | 202 |
| 2 | 100万 | 20000 | 95 | 211 |
| 3 | 100万 | 20000 | 99 | 202 |
| 10个线程 | 1 | 100万 | 20000 | 175 | 114 |
| 2 | 100万 | 20000 | 175 | 114 |
| 3 | 100万 | 20000 | 175 | 114 |
| 4 | 100万 | 20000 | 175 | 114 |
| 5 | 100万 | 20000 | 175 | 114 |
| 6 | 100万 | 20000 | 169 | 118 |
| 7 | 100万 | 20000 | 169 | 118 |
| 8 | 100万 | 20000 | 175 | 114 |
| 9 | 100万 | 20000 | 169 | 118 |
| 10 | 100万 | 20000 | 175 | 114 |

1. **Get读取测试**
2. 测试环境：

|  |  |
| --- | --- |
| 表文件大小 | 1.5G |
| 表字段数 | 124 |
| Region个数 | 4 |

测试结果：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试线程数 | 线程ID | 数据量 | 查询数据 | 操作时间(秒） | 平均每秒处理量（行） |
| 1个线程 | 1 | 100万 | 10000 | 48 | 208 |
| 2个线程 | 1 | 100万 | 10000 | 47 | 213 |
| 2 | 100万 | 10000 | 47 | 213 |
| 3个线程 | 1 | 100万 | 10000 | 58 | 172 |
| 2 | 100万 | 10000 | 59 | 169 |
| 3 | 100万 | 10000 | 59 | 169 |
| 10个线程 | 1 | 100万 | 10000 | 141 | 71 |
| 2 | 100万 | 10000 | 141 | 71 |
| 3 | 100万 | 10000 | 141 | 71 |
| 4 | 100万 | 10000 | 141 | 71 |
| 5 | 100万 | 10000 | 140 | 71 |
| 6 | 100万 | 10000 | 141 | 71 |
| 7 | 100万 | 10000 | 141 | 71 |
| 8 | 100万 | 10000 | 141 | 71 |
| 9 | 100万 | 10000 | 141 | 71 |
| 10 | 100万 | 10000 | 141 | 71 |

1. **性能对比**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **HBase API接口** | **Thrift接口** | **Rest接口** | **说明** |
| 数据写入 | 高 | 中 | 低 |  |
| 数据Scan读取 | 高 | 中 | 低 | Thrift查询在客户端保持了一个Client长连接，查询性能比使用API方式Scan低，比Get略高 |
| 数据Get读取 | 高 |  | 低 |  |
| 编码难易程度 | 容易 | 中等 | 中等 | 使用Java语言，Thrift和Rest接口都很容易。 Thrift天生支持多语言，但是Rest接口就需要自己类库了。 |
| 多语言兼容性 | 不支持 | 支持 | 支持 |  |
| 系统稳定性 | 高 | 中等 | 中等 | 由于Thrift和Rest接口需要单独的开启一个服务端，虽然可以通过开启多个服务端来保证稳定性，但是还是增加了服务的一层不确定因素。 |

总结：

1. 如果是Java语言应用，建议直接使用HBase API访问HBase。如果非Java语言应用，还是优先考虑Thrift接口，其次考虑Rest接口。