COMP226

Assignment 1, slides 2

Rahul Savani

rahul.savani@liverpool.ac.uk

Today

- Example with **more than one order at the same price**
- A (new) **sorting function** for the books in `skeleton.R`
- No `Inf`/`-Inf`, but `NA` instead in the output
- **Message by message logging** with `book.summarise`

Please

Download `comp226_a1_v3.zip`

Recall initial book

```
$ Rscript solution.R input/book_1.csv input/empty.txt
$ask
  oid price size
1   a   105  100

$bid
  oid price size
1   b    95  100

Total volume: 100 100
Best prices: 95 105
Mid-price: 100
Spread: 10
```

Ex: message_same_price.txt

```
$ Rscript solution.R input/book_1.csv input/message_ex_same_price.txt
$ask
  oid price size
2   j   105  132
1   a   105  100

$bid
  oid price size
1   b    95  100
2   k    95   71

Total volume: 171 232
Best prices: 95 105
Mid-price: 100
Spread: 10
```

Earlier messages closer to top of the book

Price-time precedence

- Orders are executed according to price time precedence
- Best price first, but **when two orders have the same price, the earlier one is executed first**
- We provide `book.sort` that respects price-time precedence
- It relies on the fact that the order ids increase as follows:

  ```
  a < k < ab < ba
  ```

  where < is indicating **"comes before"** in the message files

book.sort (in skeleton.R)

```r
book.sort <- function(book, sort_bid=T, sort_ask=T) {
    if (sort_ask && nrow(book$ask) >= 1) {
        book$ask <- book$ask[order(book$ask$price,
                                   nchar(book$ask$oid),
                                   book$ask$oid,
                                   decreasing=F),]
        row.names(book$ask) <- 1:nrow(book$ask)
    }

    if (sort_bid && nrow(book$bid) >= 1) {
        book$bid <- book$bid[order(-book$bid$price,
                                   nchar(book$bid$oid),
                                   book$bid$oid,
                                   decreasing=F),]
        row.names(book$bid) <- 1:nrow(book$bid)
    }
    book
}
```

You are welcome (and encouraged) to use book.sort

Example output

```
$ask
  oid price size
8   a   105  100
7   o   104  292
6   r   102  194
5   k    99   71
4   q    98  166
3   m    98   88
2   j    97  132
1   n    96  375

$bid
  oid price size
1   b    95  100
2   l    95   29
3   p    94   87
4   s    91  102

Total volume: 318 1418
Best prices: 95 96
Mid-price: 95.5
Spread: 1
```

The rownames are now: 1,2,... starting from the best prices

Empty book: `Inf`/`-Inf` -> `NA`

The old version of the code produced `Inf` or `-Inf` for prices when the book was empty; now we use `NA`

```
$ Rscript solution.R input/book_1.csv input/message_ex_cross.txt
$ask
[1] oid   price size
<0 rows> (or 0-length row.names)

$bid
  oid price size
1   c   106    1
2   b    95  100

Total volume: 101 0
Best prices: 106 NA
Mid-price: NA
Spread: NA
```

book.summarise

```r
book.summarise <- function(book, with_stats=T) {
    if (nrow(book$ask) > 0)
        book$ask <- book$ask[nrow(book$ask):1,]

    print(book)

    if (with_stats) {
        clean <- function(x) { ifelse(is.infinite(x), NA, x) }

        total_volumes <- book.total_volumes(book)
        best_prices <- lapply(book.best_prices(book), clean)
        midprice <- clean(book.midprice(book))
        spread <- clean(book.spread(book))

        cat("Total volume:", total_volumes$bid, total_volumes$ask, "\n")
        cat("Best prices:", best_prices$bid, best_prices$ask, "\n")
        cat("Mid-price:", midprice, "\n")
        cat("Spread:", spread, "\n")
    }
}
```

Logging in every step

```
book.reconstruct <- function(data, init=NULL, log=F) {

    if (nrow(data) == 0) return(book)
    if (is.null(init)) init <- book.init()

    book <- Reduce(
         function(b, i) {
             new_book <- book.handle(b, data[i,])
             if (log) {
                 cat("Step", i, "\n\n")
                 book.summarise(new_book, with_stats=F)
                 cat("====================\n\n")
             }
             new_book
         },
         1:nrow(data), init,
    )
    book.sort(book)
}
```

Turn on logging by changing the default argument: `log=T`