Equality Between Stock Trading and Min Cost Nonincreasing

Claire Yang

May 2021

1 Introduction

There are two problems:

- **Stock Trading:** Given a stock n-day price, you can only trade at most once a day, you can choose to buy a stock or sell a stock or not trade today, output the maximum profit can be achieved. [1]
- **Min Cost Nonincreasing:** Given an array A, what is the min cost to change A s.t. it is nonincreasing? cost = $sum(|x_i|)$ where x_i = the amount we changed A[i]

I noticed that the same algorithm can solve these two seemingly unrelated problems. So I will prove their equality using Primal-Dual theory.

2 Fractional Primal for Stock Trading

We try formulating the Dual for Stock Trading, then comparing it to the Primal for Min Cost Nonincreasing. They should in theory be the same.

A = array of prices, length n
M = n x n binary matrix
M[i][j] = 1 iff we bought on day i and sold on day j
max$\sum_{i<j}(A[j] - A[i]) * M[i][j]$
s.t.
$\sum_{j>i} M[i][j] <= 1 \quad \forall i \quad (*y_i)$ (buys on day i)
$\sum_{j<i} M[j][i] <= 1 \quad \forall i \quad (*z_i)$ (sells on day i)
$M[i][j] \geq 0$

3 Fractional Dual for Stock Trading

$min \sum_{i=1}^{n} y_i + \sum_{i=1}^{n} z_i$
s.t.

$y_i + z_j \geq A[j] - A[i] \quad \forall i < j$
$y_i, z_j \geq 0$

4 Interpret Fractional Dual for Stock Trading

y_i = how much we increased A[i]
z_j = how much we decreased A[j]

But the inequality does not express total change $(y_i - z_i)$

This is not helpful. It's hard to interpret the dual in such a way that it matches the Min Cost Nonincreasing problem. So let's try the other way around: formulating the Dual for Min Cost Nonincreasing then comparing it to the Primal for Stock Trading.

5 Fractional Primal for Min Cost Nonincreasing

min $\sum_{i=1}^{n} y_i + \sum_{j=1}^{n} z_j$
s.t.

$$A[i] + y_i - z_i >= A[j] + y_j - z_j \leftrightarrow y_i - y_j - z_i + z_j >= A[j] - A[i] \quad \forall i < j$$
$$y_i, z_j >= 0$$

6 Fractional Dual for Min Cost Nonincreasing

max $\sum_{i<j} (A[j] - A[i]) * M[i][j]$
s.t.
$\sum_{j>i} M[i][j] - \sum_{j<i} M[j][i] <= 1 \quad \forall i$ (buys on day i - sells on day i)
$\sum_{j<i} M[j][i] - \sum_{j>i} M[i][j] <= 1 \quad \forall i$ (sells on day i - buys on day i)
$M[i][j] \geq 0 \quad \forall (i, j)$

This is very similar to the primal for stock trading! Observe that the second difference is the first difference with the terms switched, so combine these two inequalities to obtain:
$-1 <= \sum_{j>i} M[i][j] - \sum_{j<i} M[j][i] <= 1 \quad \forall i$

This is just saying that on day i, the net trade (buys - sells) must reflect at most one transaction (either one buy in the end or one sell, or none).

7 Conclusions

7.1 Buy or Sell?

$\sum_{j>i} M[i][j] - \sum_{j<i} M[j][i]$ is an indication on how important it is to trade stock i. The closer it is to -1, the better it is to sell it. The closer to 1, the better it is to buy it.

7.2 Transform

These rules are different from those in the original problem, which allow only 1 transaction each day. However, we can transform the solution in this relaxed problem (M) to the solution in the constrained problem (M') by letting $M'[i][j] = \frac{1}{n}(\sum_{j>i} M[i][j] - \sum_{j<i} M[j][i])$. If this value is positive, its magnitude is an indication of how much we should make the trade pair (i,j). If the value is negative, its magnitude is an indication of how much we should make the trade pair (j,i).

Claim: $-1 \leq \sum_{j>i} M'[i][j] - \sum_{j<i} M'[j][i] \leq 1$ when $M'[i][j]$ is defined as above.

$$\sum_{j>i} M'[i][j] - \sum_{j<i} M'[j][i] =$$

$$\frac{1}{n}(\sum_{j>i}\sum_{k>i} M[i][k] - \sum_{j>i}\sum_{k<i} M[k][i] - \sum_{j>i}\sum_{k>j} M[j][k] + \sum_{j>i}\sum_{k<j} M[k][j] -$$

$$\sum_{j<i}\sum_{k>j} M[j][k] + \sum_{j<i}\sum_{k<j} M[k][j] + \sum_{j<i}\sum_{k>i} M[i][k] + \sum_{j<i}\sum_{k<i} M[k][i]) \quad (1)$$

$$\text{Note that} \sum_{j>i}\sum_{k>i} M[i][k] + \sum_{j<i}\sum_{k<i} M[k][i] = (n-1)\sum_{k>i} M[i][k]$$

$$\text{and} \sum_{j>i}\sum_{k<i} M[k][i] + \sum_{j<i}\sum_{k<i} M[k][i] = (n-1)\sum_{k<i} M[k][i]$$

$$\text{and} \sum_{j>i}\sum_{k<j} M[k][j] + \sum_{j<i}\sum_{k<j} M[k][j] - \sum_{j>i}\sum_{k>j} M[j][k] - \sum_{j<i}\sum_{k>j} M[j][k] =$$

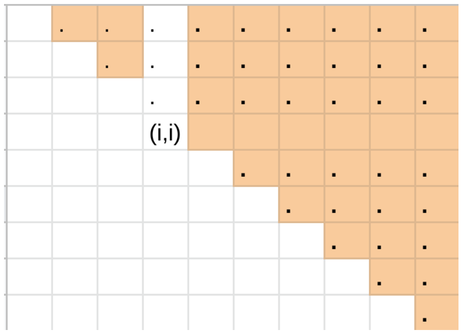
$$\sum_{k>i} M[i][k] - \sum_{k<i} M[k][i] \quad (2)$$

$$\text{so (1)} = \frac{1}{n}((n-1)\sum_{k>i} M[i][k] - (n-1)\sum_{k<i} M[k][i] + \sum_{k>i} M[i][k] - \sum_{k<i} M[k][i]) =$$

$$\sum_{k>i} M[i][k] - \sum_{k<i} M[k][i]$$

which is $\geq$ -1 and $\leq$ 1 by the formulation of the dual

Visual of Equation (2)



orange = and $\sum_{j>i}\sum_{k<j} M[k][j] + \sum_{j<i}\sum_{k<j} M[k][j]$
dots = $\sum_{j>i}\sum_{k>j} M[j][k] + \sum_{j<i}\sum_{k>j} M[j][k]$
so orange - dots = $\sum_{k>i} M[i][k] - \sum_{k<i} M[k][i]$

7.3 Integer Versions

In the original problem, you can't "0.4908 of the way" buy on day i. You must either buy, sell, or do nothing. This does not diminish the equality between the two problems, however. We proved equality between the fractional versions, so the integer versions must be equal as well.

8 Further Questions

1. The original stock trading problem forbids us from buying "0.4908" of stock on day i, but in real life, we can certainly control how much stock we buy and sell. Perhaps this could be useful in stock trading algorithms.
2. Minimum cost nonincreasing can be easily transformed into minimum cost increasing, decreasing, nondecreasing. I wonder if any of these have practical applications.

References

[1] https://www.lintcode.com/problem/best-time-to-buy-and-sell-stock-v/.