Promoting Accurate Calculation of Insider Trading Liability

COMP 523 Project Proposal (Spring 2016)

Andrew Chin
Associate Professor, University of North Carolina School of Law

Abstract

Full enforcement of the insider trading laws benefits the public interest by improving confidence in the fairness and efficiency of the stock market. Millions of dollars are recovered annually for the investing public under a federal law that prohibits corporate insiders from profiting from short-term trading in the securities of the companies they control. Unfortunately, courts and attorneys have found it difficult to apply accurate methods for calculating liability under this law, which utilize modern algorithms for solving linear programming problems. Instead, they have relied on a simpler but sometimes inaccurate greedy algorithm devised by a judge in the 1940s. As a result, one plaintiff recently demanded and recovered $29,889.05 under this law, when an accurate calculation of the insider’s liability would have yielded $35,366.40.

A basic online tool for calculating accurate insider trading liability is available, but the toy examples it supports are not well suited to demonstrating the practical need for an accurate alternative to the greedy algorithm. To encourage everyday use by the interested public (including investors, corporate officers and directors, legal and financial advisers, regulators, and judges), considerable additional functionality will be needed, including data portability and tracking of regulatory filings.

Legal Background

Section 16(b) of the Securities Exchange Act of 1934 provides that specified corporate insiders must surrender “any profit realized … from any purchase and sale, or any sale and purchase, of any equity security” of the company within a six-month period. This is a strict liability provision, meaning that profits must be surrendered even if they were not attained through the misuse of inside information, and even if the paired purchase and sale events are part of a larger pattern of trades that also incurs losses. As a consequence, full enforcement of Section 16(b) requires liability to be calculated as the maximum profit that can be derived from the pairing of an insider’s purchases and sales, with each transaction pair taking place within a six-month period.

To aid enforcement of Section 16(b), corporate insiders are required to report transactions in their company’s stock on SEC Form 4. An example is shown below. These filings are available to the public on the SEC’s Web site at <http://www.sec.gov/search/search.htm> (including in XML format), and any shareholder can bring suit based on the information disclosed.
Mathematical Background

Section 16(b) liability calculation presents a straightforward optimization problem amenable to solution by standard linear programming techniques.

For example, consider the following pattern of trades:

<table>
<thead>
<tr>
<th>Date</th>
<th>Shares Bought (Sold)</th>
<th>Price/Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan. 1</td>
<td>1,000</td>
<td>$ 9</td>
</tr>
<tr>
<td>Feb. 15</td>
<td>(400)</td>
<td>$ 8</td>
</tr>
<tr>
<td>Mar. 1</td>
<td>2,000</td>
<td>$ 8</td>
</tr>
<tr>
<td>May 1</td>
<td>800</td>
<td>$ 7</td>
</tr>
<tr>
<td>June 15</td>
<td>(1,200)</td>
<td>$ 10</td>
</tr>
<tr>
<td>Sept. 1</td>
<td>1,000</td>
<td>$ 6</td>
</tr>
<tr>
<td>Oct. 15</td>
<td>(2,400)</td>
<td>$ 9</td>
</tr>
</tbody>
</table>
For $i = 1,2,3,4$ and $j = 1,2,3$, let $p_{ij}$ denote the per-share profit recoverable under Section 16(b) from pairing the $i$-th purchase and $j$-th sale in this table (counting chronologically). For example, by pairing the shares purchased on May 1 for $7$/share (i.e., the third purchase) with the shares sold on Feb. 15 for $8$/share (i.e., the first sale), the trader may be held liable for a profit of $1$/share; this fact may be expressed as $p_{31} = 1$. On the other hand, the first purchase on Jan. 1 and third sale on Oct. 15 are more than six months apart, so $p_{13} = 0$. Thus we can compute the vector $P = (p_{11}, p_{12}, p_{13}, p_{21}, p_{22}, p_{23}, p_{31}, p_{32}, p_{33}, p_{41}, p_{42}, p_{43}) = (0,1,0,2,0,1,3,2,0,4,3)$.

To maximize the total profit recoverable under Section 16(b), we must find the number of shares $x_{ij}$ for each pair of purchases and sales for which the total recoverable profit $\sum_{i,j} p_{ij}x_{ij}$ is maximum, subject to the constraints

$$\sum_{j} x_{ij} \leq 1,000 \quad \sum_{j} x_{ji} \leq 400$$
$$\sum_{j} x_{2j} \leq 2,000 \quad \sum_{j} x_{i2} \leq 1,200$$
$$\sum_{j} x_{3j} \leq 800 \quad \sum_{j} x_{i3} \leq 2,400$$
$$\sum_{j} x_{4j} \leq 1,000 \quad \forall i, j : x_{ij} \geq 0$$

This linear programming problem may be solved by standard techniques; e.g., the simplex method. The solution vector is:

$$X = (x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}, x_{41}, x_{42}, x_{43}) = (0,0,0,1200,0,0,0,800,0,0,1000),$$

for a maximum recoverable profit $P \cdot X$ of $7,000$.

In contrast, Judge Charles Edward Clark in 1943 devised a greedy algorithm to calculate Section 16(b) liability, which he referred to as the “lowest-in, highest-out” method. The algorithm consists of iteratively pairing the purchase at the lowest per-share price with the sale at the highest per-share price within six months for which shares can be matched, and removing transactions when no more remaining shares remain to be paired with them. Judge Clark’s method results in the following pairings, for a total profit of $5,800:

<table>
<thead>
<tr>
<th>Shares</th>
<th>Purchase Date</th>
<th>Price/Share</th>
<th>Sale Date</th>
<th>Price/Share</th>
<th>Profit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,000</td>
<td>Sept. 1</td>
<td>$6$</td>
<td>June 15</td>
<td>$10$</td>
<td>$4,000</td>
</tr>
<tr>
<td>200</td>
<td>May 1</td>
<td>$7$</td>
<td>June 15</td>
<td>$10$</td>
<td>$600</td>
</tr>
<tr>
<td>600</td>
<td>May 1</td>
<td>$7$</td>
<td>Oct. 15</td>
<td>$9$</td>
<td>$1,200</td>
</tr>
</tbody>
</table>

Courts and attorneys continue to use Judge Clark’s method, most likely because of their unfamiliarity with linear programming. The plaintiff in *Chechele v. Vicis Capital, LLC* recently demanded and received $29,889.05 based on a “lowest-in, highest-out calculation,” when an accurate calculation of the insider’s liability would have yielded $35,366.40. The availability of off-the-shelf linear programming software (e.g., the Solver function in Excel, the linprog function in Matlab, and public domain/open source implementations of many well-known algorithms) is of no help, because courts and attorneys lack confidence in their ability to formulate particular Section 16(b) liability calculations as linear programming problems.
A specific, user-friendly software solution would therefore provide an immediate public benefit in aiding the full and vigorous enforcement of the nation’s insider trading laws.

**Deliverables**

- Despite the relatively simple hypothetical counterexample to Judge Clark’s algorithm above, thus far the only known actual cases of an undercalculated Section 16(b) award are *Chechele* and another case, both of which involve several hundred purchases and sales. A Web calculator capable of manipulating spreadsheet files to run Judge Clark’s algorithm and to calculate the vectors $P$, $X$, and the Section 16(b) liability $P \cdot X$ is needed to verify these counterexamples and to address similar cases that may arise in the future.

- The SEC Web site publishes the Form 4 filings it receives each day at http://www.sec.gov/edgar/searchedgar/currentevents.htm. For each insider identified in these filings, it would be desirable to query the SEC Form 4 database for other trading activity and calculate their potential liability exposure, producing a summary report that can be reviewed by regulators, compliance counsel and other interested attorneys.

- It would also be desirable to scrape the SEC Form 4 database (or use other data files published on the SEC Web site) to compile statistics on the prevalence of short-swing trading by insiders and the size and complexity of transaction sequences that might be the subject of a Section 16(b) challenge.

- (Stretch) Use advanced linear programming algorithms, sparse matrix methods, and other complexity-reducing techniques to improve the efficiency of the software solution for large trading patterns (i.e., millions or tens of millions of transactions), such as may arise under high-frequency trading scenarios.

**Further Information**
