APPARATUSES, METHODS AND SYSTEMS FOR A MARGINAL CONTRIBUTION TO PERFORMANCE PLATFORM

RELATED APPLICATIONS

[0001] Applicant hereby claims priority under 35 USC §119 for United States provisional patent application serial no. 61/152,588 filed February 13, 2009, entitled "APPARATUSES, METHODS AND SYSTEMS FOR A MARGINAL CONTRIBUTION TO PERFORMANCE PLATFORM," attorney docket no. 17209-093PV.

[0002] The entire contents of the aforementioned application is herein expressly incorporated by reference.

FIELD

[0003] The present invention is directed generally to an apparatuses, methods, and systems of financial trading, and more particularly, to APPARATUSES, METHODS AND SYSTEMS FOR A MARGINAL CONTRIBUTION TO PERFORMANCE PLATFORM.

BACKGROUND

[0004] Financial trading involves the buying and/or selling of financial instruments, such as stocks, bonds, derivatives, commodities, and/or the like, often within the context of a market exchange. Trades may be executed on behalf of customers by one or more traders, who may employ various trading strategies depending on the particular goals and/or requirements of the customer, trader, or trade in question.

Among such strategies are market orders, whereby the trader executes a trade immediately at the current market price for the traded instrument, and limit orders, whereby the trader waits to execute the trade until a counterparty is found willing to trade at a pre-set price. Electronic trading systems have also come about, employing information technology systems to facilitate trades.

SUMMARY

[0005] The APPARATUSES, METHODS AND SYSTEMS FOR A MARGINAL CONTRIBUTION TO PERFORMANCE PLATFORM (hereinafter "MCP PLATFORM") achieves significant gains in increasing the utility and efficacy of algorithmic trading. In one embodiment, the MCP Platform facilitates optimized trading of financial instruments by facilitating an optimization framework that optimizes order-placement. The MCP Platform may minimize a total, generalized cost associated with a trade by separating the overall optimization process into current order placement decisions and future order placement decisions. Future order placement decisions may then be evaluated analytically and the current order placement decisions evaluated numerically to effectively reduce the dimensionality of the optimization problem and facilitate optimization to be performed quickly and efficiently.

[0006] In one embodiment, the MCP PLATFORM may receive a request from a trader indicating that a client has placed an order for a security with the trader. The MCP PLATFORM may retrieve order parameters and security parameters from the request. The MCP PLATFORM may also retrieve market parameters from one or more exchanges. If all required inputs for executing the client's order have not been received,

the MCP PLATFORM may attempt to retrieve and/or construct such inputs based on the supplied data and/or based on the data in the MCP PLATFORM database and/or other data sources accessible by the MCP PLATFORM.

[0007] Stochastic processes for market variables may be specified based on the retrieved data. The implementation shortfall measuring the additional trading cost incurred for trading the security above the benchmark cost and calculated based on the retrieved security parameters may be specified. An implementation cost may be specified as a function of expected implementation shortfall and/or related values, such as risk aversion, and optimized to determine placement policies and/or trading schedules yielding minimal implementation cost values. The MCP PLATFORM may define a total implementation cost as the sum of the instantaneous implementation cost for a given order placement decision and the corresponding go-forward optimal implementation cost, as specified by a selected paradigm. Once the total implementation cost has been determined, the MCP PLATFORM may determine an optimal current order placement decision by, in one implementation, minimizing the total implementation cost as a function of order placement decisions, time horizon, and/or the like. The determined optimal current order placement may be passed to a trading system to execute a trade of the specified instruments and/or number of shares. If additional shares remain to be traded, the MCP PLATFORM may update the total implementation cost model with current values including current market data values, and the updated total implementation cost may be minimized again to yield the next optimal current order placement.

[0007a] According to an aspect, there is provided a processor-implemented method for trading a financial instrument, comprising: receiving an order request that includes a total quantity of shares and a time horizon within which to execute a transaction; retrieving order parameters, security parameters, and market parameters associated with the transaction; evaluating analytically a future implementation cost as a function of future number of shares to trade after a current time interval based on at least some of the retrieved parameters; evaluating numerically a current implementation cost as a function of current number of shares to trade during the current time interval based on at least some of the retrieved parameters; optimizing via a processor a total implementation cost to determine an optimal current number of shares from the total quantity to trade during the current time interval, wherein the total implementation cost is a sum of the future implementation cost and the current implementation cost; and executing an order for the optimal current number of shares of the financial instrument.

[0007b] According to another aspect, there is provided a financial instrument trading apparatus, comprising: a memory; a processor disposed in communication with said memory, and configured to issue a plurality of processing instructions stored in the memory, wherein the processor issues instructions to: receive an order request that includes a total quantity of shares and a time horizon within which to execute a transaction; retrieve order parameters, security parameters, and market parameters associated with the transaction; evaluate analytically a future implementation cost as a function of future number of shares to trade after a current time interval based on at least some of the retrieved parameters; evaluate numerically a current implementation cost as a function of current number of shares to trade during the current time interval based on at least some of the retrieved parameters; optimize via a processor a total implementation cost to determine an optimal current number of shares from the total quantity to trade during the current time interval, wherein the total implementation cost is a sum of the future implementation cost and the current implementation cost; and execute an order for the optimal number of shares of the financial instrument.

[0008] This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key or essential features of the claimed subject matter, nor is it intended to be used as an aid in determining the scope of the claimed subject matter.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] The accompanying appendices and/or drawings illustrate various non-limiting, example, inventive aspects in accordance with the present disclosure:

[0010] FIGURE 1 provides an overview of the entities interactions with and among MCP system components in one embodiment of the MCP PLATFORM;

[0011] FIGURE 2 illustrates data flow in one embodiment of the MCP PLATFORM operation;

[0012] FIGURE 3 shows an implementation of overall logic flow in one embodiment of the MCP PLATFORM operation;

[0013] FIGURE 4 is of a logic flow diagram illustrating selection of stochastic processes in one embodiment of the MCP PLATFORM;

[0014] FIGURE 5 is of a logic flow diagram illustrating specifying implementation shortfall as a stochastic integral in one embodiment of the MCP PLATFORM;

[0015] FIGURE 6 is of a logic flow diagram illustrating specifying implementation cost function in one embodiment of the MCP PLATFORM;

[0016] FIGURE 7 is of a logic flow diagram illustrating constructing a paradigm for evaluating the implementation cost in one embodiment of the MCP PLATFORM;

[0017] FIGURE 8 is of a logic flow diagram illustrating the determined optimal current order placement in one embodiment of the MCP PLATFORM;

[0018] FIGURE 9 is of a drawing illustrating a non-limiting example of an Order Placement Decision question that may be solved by the MCP PLATFORM;

[0019] FIGURE 10 is of a block diagram illustrating embodiments of the MCP PLATFORM controller;

[0020] FIGURE 11 is of screen shot diagram illustrating one embodiment of the MCP PLATFORM user interface.

[0021] The leading number of each reference number within the drawings indicates the figure in which that reference number is introduced and/or detailed. As such, a detailed discussion of reference number 101 would be found and/or introduced in Figure 1. Reference number 201 is introduced in Figure 2, etc.

DETAILED DESCRIPTION

MCP PLATFORM

[0022] This disclosure details aspects of the MCP PLATFORM. It is to be understood that, depending on the particular needs and/or characteristics of a trader, buyer or seller of a financial instrument, financial instrument modeler and/or other MCP PLATFORM user, financial product, market and/or exchange, computer implementation, and/or the like, various embodiments of the MCP PLATFORM may be implemented that enable a great deal of flexibility and customization. The instant disclosure discusses embodiments and/or applications of the MCP PLATFORM directed to trading strategies for financial instruments, such as those that may be traded on a market exchange. However, it is to be understood that the system described herein may be readily configured and/or customized for a wide range of other applications and/or implementations. For example, aspects of the MCP PLATFORM may be adapted for non-exchange traded financial instruments, trading of other types of traded goods, services, commodities, and/or the like, predictive modeling, and/or any of a wide variety of other applications.

[0023] Furthermore, aspects of the MCP PLATFORM may be configured to generate, administer, and/or manage trading strategies for a wide variety of different financial instruments, securities, and/or the like beyond specific embodiments and/or implementations described in detail herein. For example, MCP PLATFORM generated trading strategies may be employed with and/or linked to equities, debts, derivatives, notes (e.g., structured notes), stocks, preferred shares, bonds, treasuries, debentures,

options, futures, swaps, rights, warrants, commodities, currencies, funds, long and/or short positions, ETFs, insurance and/or risk transfer agreements, annuities, and/or other assets or investment interests. It should be noted that the MCP PLATFORM and/or MCP PLATFORM analysis may be configured to provide particular utility to investors, traders, brokers, specialists, and/or the like seeking to fulfill specific goals and/or purposes, such as to minimize risks, avoid impact costs, and/or the like. It is to be understood that the MCP PLATFORM may be further adapted to other implementations or investment, finance, and/or risk management applications.

[0024] In order to optimize various Order Placement Decisions during order execution, the MCP PLATFORM, in one embodiment, may seek to minimize implementation cost of trading, wherein implementation cost may include elements of impact cost, cost of risk, the literal cost of the stock, and/or the like. In one embodiment, the MCP PLATFORM may separate the overall Order Placement Decision optimization into two parts: (1) the current Order Placement Decisions, and (2) all future Order Placement Decisions. For example, a client may wish to buy 10,000 shares of S&P 500 ETF over the next hour. The MCP PLATFORM may seek to estimate and minimize the cost of trading by placing market orders at the ask price for shares of this ETF in such a way that all 10,000 shares will be purchased over the next hour. In order to decide the sizes of these market orders, the MCP PLATFORM may separate estimating the cost of trading into two parts to reduce the complexity of the estimation. In the first part, the MCP PLATFORM may use a paradigm (i.e. a mathematical model) for estimating the size of the optimal market order that minimizes the cost of trading in the current time period, such as in the next 1 minute. In the second part, the MCP PLATFORM may use a paradigm for estimating the size of the optimal market order

that minimizes the cost of trading in all the later time periods, such as over the next 59 minutes. By comparing the costs of placing market orders with different sizes, the MCP PLATFORM may determine the optimal market order size in the current time period that minimizes the total estimated cost of trading. The MCP PLATFORM attempts to minimize the cost of trading by repeating these calculations in each of the subsequent time periods.

[0025] Optimization of Order Placement Decisions during order execution is a very tough problem – often referred to as "adaptive" or "dynamic" optimization. The problem is that one has so many variables to optimize, literally tens of thousands of individual order placements, that even the most advanced numerical optimization methods take several minutes to converge on a solution. Unfortunately, these decisions need to be re-optimized several times a second. In one embodiment to facilitate this optimization, the MCP PLATFORM may use derivative pricing theory and stochastic control theory to analytically evaluate part (2) related to future placement assuming a continuum limit, and may use numerical techniques to evaluate part (1) related to current placement to reduce the dimensionality of the optimization problem. In one implementation, using this approach may reduce the dimensionality of the optimization problem by three or more orders of magnitude, and may facilitate solving the optimization problem by the MCP PLATFORM in milliseconds or faster, as opposed to previous processing/time intensive analysis that involve calculations done over the course of minutes. These processing time savings are significant within the realm of financial transactions – where time literally is money. The MCP PLATFORM may also use less computational power to solve the optimization problem, reducing hardware requirements and increasing scalability.

[0026] FIGURE 1 provides an overview of the entities interactions with and among MCP system components in one embodiment of the MCP PLATFORM. In Figure 1, a client 105a or 105b places an order to trade securities with a trader 110. In one embodiment, the client may place the order using a communication network 115. In one implementation, the communication network 115 may be the Internet. In another implementation, the communication network 115 may be a private communication network between a client and the trader. It is to be understood, that while only one communication network 115 is shown for clarity, multiple distinct communication networks may exist connecting each of the system components and affiliated entities with each other. Furthermore, although illustrated as separate entities, it is to be understood that the MCP Platform facilitates significant flexibility and certain features and functionality may be combined or separated depending on the particular working implementation and the needs of the various users. For example, aspects of the MCP Platform 120 and the MCP PLATFORM database 140 may be part of a trading platform 121 having a variety of other components 122. In another example, aspects of the MCP Platform 120, the MCP PLATFORM database 140, and Trader 110 may be combined into a single financial transaction facilitation system. In another example, a client may place an order over a telephone network, while the trader may communicate with the MCP PLATFORM 120 over the Internet.

[0027] Upon receiving the order from the client, the trader, also referred to herein as a user, may use the MCP PLATFORM 120 to execute the order in an optimal or near optimal way. In one embodiment, the MCP PLATFORM may use data from various sources, such as from exchanges 125a and/or 125b, supplied by the trader from the trader database 130, from the MCP PLATFORM database 140, and/or the like, to

determine an optimized trading solution that includes schedule facilitation for execution on one or more exchanges 125a and/or 125b. By way of non-limiting example only, exchanges 125a, 125b may be any financial exchanges such as the NYSE, NASDAQ, and/or the like.

[0028] Below is a table illustrating some of the notation used in the following description of the MCP PLATFORM. It is to be understood that this is not an exhaustive list of notation used in the description, and that other notation may appear throughout the specification.

X : *remaining position*

y : *market order trade rate*

N : *initial position*

T : *horizon*

μ : *drift*

σ : *volatility*

Σ : *covariance matrix*

f : *fill rate*

δ : *half spread*

$\bar{y}$: *market volume flow rate*

γ : *temporary market impact function exponent*

α : *temporary market impact function coefficient*

$\phi : \frac{\alpha}{\bar{y}^{\gamma}}$

Φ : *diagonal matrix with diagonal elements* ϕ_i

η : *permanent market impact function coefficient*

IS : *implementation shortfall*

U or IC : *implementation cost*

S : *price*

C : *diff between realized IS and expected IS*

λ or κ : *risk aversion coefficient*

β : *beta value with respect to some benchmark*

V : *cumulative volume profile*

v : *instantaneous volume profile*

[0029]

[0030] In one embodiment, the MCP PLATFORM also makes certain assumptions including: stock price follows an arithmetic random walk with drift μ and volatility σ; an order of N shares should be executed within an interval of length T; passive spread-capture strategy is estimated to be filled at a rate *f shares/sec*; and *y(t)* is the trading schedule of market orders, whose unit is *shares/sec*.

[0031] FIGURE 2 illustrates data flow in one embodiment of the MCP PLATFORM operation. At 205, the MCP PLATFORM may receive an indication from a trader that a client has placed an order for a security with the trader. In one embodiment, such an indication may be in the form of a request to determine an optimized trading schedule for the order.

[0032] In one embodiment, the request may include one or more order parameters, such as stock symbol, quantity of shares to trade, time within which to execute the order, and/or the like, specified by the client. At 210, the MCP PLATFORM may retrieve such order parameters from the request and place them in corresponding variables used by the MCP PLATFORM program code to store order parameters. For example, a C++ function that parses the request and copies the order parameters into corresponding variables may be used to facilitate the retrieval.

[0033] In another embodiment, the request may also include one or more security parameters, such as beta, drift, volatility, temporary market impact coefficient, temporary market impact exponent, and/or the like, specified by the trader. In one implementation, beta may indicate risk of the security compared to the market, drift may indicate how the price of the security varies with time, volatility may indicate the volatility of the security price, and temporary market impact coefficient and temporary market impact exponent may indicate how the price of a security may be temporarily affected by the temporary liquidity imbalance due to placement of an order. In one implementation, the trader may specify one or more security parameters based on values stored in the trader database 130. For example, the trader may calculate the value of beta based on historical data regarding the security stored in the trader database. At

215, the MCP PLATFORM may retrieve such security parameters from the request and place them in corresponding variables used by the MCP PLATFORM program code to store security parameters.

[0034] The MCP PLATFORM may retrieve market parameters from one or more exchanges 220. The market parameters retrieved by the MCP PLATFORM may include security bid price, security ask price, volume, and/or the like parameters that facilitate the operation of the MCP PLATFORM. In one embodiment, the MCP PLATFORM may be part of a financial transaction facilitation system and the market parameters may be retrieved by making a C++ function call to a module of the financial transaction facilitation system that provides market data. In another embodiment, the market parameters may be retrieved from an exchange through a market data feed via a communications network. In one implementation, the data feed may be in an XML format and may take on the following form:

```
<XML>
        <Asset>
            <asset_symbol>XYZ</asset_symbol>
            <bid_price>100.01</bid_price>
            <ask_price>100.02</ask_price>
            <volume>500,000</volume>
        </Asset>
</XML>
```

[0035] In another implementation, the data feed may be a text file containing data fields separated by a separator character, such as "|" and may take on the following form:

```
asset_symbol | bid_price | ask_price | volume
```

[0036] In one implementation, data from a market data feed may be extracted by parsing a data file, such as by using the Perl m// operator (e.g. m/.*|/). In another implementation, data from multiple data feeds may be received and combined in a variety of ways. For example, if two or more feeds supply data regarding the same market parameter, the MCP PLATFORM may pick the value from a preferred feed, take the average value, and/or pick the value supplied by the greatest number of feeds.

[0037] The MCP PLATFORM may make a determination at 225 as to whether all required inputs for executing the client's order have been received. In one embodiment, the list of the required inputs used by the MCP PLATFORM may be specified by a MCP PLATFORM administrator and may be stored in a configuration file used by the MCP PLATFORM. In another embodiment, the list of these parameters may be hard coded into a software program that facilitates the operation of the MCP PLATFORM, such as a software program written in C++.

[0038] If not, at 230 the MCP PLATFORM may attempt to retrieve and/or construct such inputs based on the supplied data and/or based on the data in the MCP PLATFORM database and/or other data sources accessible by the MCP PLATFORM. For example, if the MCP PLATFORM did not receive a security parameter specifying beta, the MCP PLATFORM may attempt to construct a measure of beta based on historical market data, if such data is available in MCP PLATFORM database and/or other data sources accessible by the MCP PLATFORM.

[0039] In one embodiment, security parameters may be stored in the MCP PLATFORM database, such as a SQL database, using Microsoft SQL Server, MySQL,

and/or the like. In one implementation, the security parameters may be inserted into the database using an INSERT command that may take on the following form:

```
INSERT INTO SecurityParametersTable (drift, volatility, beta)
VALUES (drift_value, volatility_value, beta_value)
```

[0040] In one implementation, security parameters may be retrieved from the MCP PLATFORM database using a SELECT command that may take on the following form:

```
SELECT drift, volatility, beta
FROM SecurityParametersTable
```

[0041] If the MCP PLATFORM is not able to retrieve and/or construct the missing input values, the MCP PLATFORM may produce an error message alerting the trader 235. Otherwise, the MCP PLATFORM may analytically evaluate a future implementation cost as a function of future number of shares to trade at 240. In one embodiment, the MCP PLATFORM may split the time horizon into a current time period $(t_{now}, t_{now}+\Delta t)$ in which to trade ΔN shares and a future time period $(t_{now}+\Delta t, T)$ in which to trade $N_{\text{remaining}}$ shares that remain to be traded. In order to analytically evaluate the future implementation cost, the MCP PLATFORM may retrieve an implementation cost function evaluated at functional extrema based on an implementation shortfall, as described in more detail in Figure 3. In one implementation, such implementation cost function may be selected by the user and specified in the request.

[0042] In order to numerically evaluate an instantaneous implementation cost as a function of current number of shares to trade at 245, the MCP PLATFORM may

retrieve an instantaneous implementation cost function expressed as the sum of the direct cost of trading ΔN shares immediately, and the impact cost of trading ΔN shares immediately. In one implementation, such implementation cost function may be selected by the user and specified in the request.

[0043] The total implementation cost is the sum of the future implementation cost and the instantaneous implementation cost. At 250, the MCP PLATFORM may determine an optimal current number of shares ΔN to trade at 255 from the remaining quantity of shares $N_{remaining}$ by finding the optimal current number of shares to trade that minimizes the total implementation cost based on current parameters (e.g. Δt, T, expected spread, volatility, impact coefficients, and/or the like).

[0044] If the MCP PLATFORM determines at 260 that all N shares have not yet been traded, the MCP PLATFORM may update the total implementation cost model with current parameters. The MCP PLATFORM may continue to re-optimize and re-execute orders for an optimal current number of shares until the MCP PLATFORM determines that the total quantity of shares has been traded 270.

[0045] FIGURE 3 shows an implementation of overall logic flow in one embodiment of the MCP PLATFORM operation. At 301, upon receiving a request to determine an optimized trading schedule for a client order, the MCP PLATFORM may retrieve one or more market parameters (e.g. security bid price, security ask price, volume), security parameters (e.g. beta, drift, volatility) and/or order parameters (e.g. security symbol, time within which to execute the order) from one or more data sources.

[0046] Stochastic processes for market variables may be specified based on the retrieved data at 305. In one implementation, market variables may comprise the price

and/or volume trading of a financial instrument. In one implementation, the stochastic process specified for market variables comprises an arithmetic random walk. For example, the stochastic process specified for the price of a financial instrument may comprise an arithmetic random walk and may take on the following form:

$$d\hat{S}(t) = \mu dt + \sigma dB(t)$$

[0047] In this stochastic process, the way in which a security's intrinsic price $d\hat{S}(t)$ changes over time is expressed based on drift μ, which may indicate how the price of the security varies based on time t, and volatility σ, which may indicate the volatility of the security price based on a random walk variable $dB(t)$. In one embodiment, the values of drift and volatility may be based on the retrieved security parameters. For further detail regarding selection of stochastic processes see Figure 4.

[0048] The implementation shortfall may be defined as the difference between the realized trading cost and some target benchmark. For example, a target benchmark, such as an arrive price benchmark, may specify an optimal trading cost for trading a security. The implementation shortfall measures the additional trading cost incurred for trading the security above the benchmark cost. In one embodiment, a client and/or a trader may choose a benchmark based on market expectations, risk tolerance, and/or the like. For example, the arrival price benchmark may be harder to game, but may be noisier than the VWAP benchmark. In one embodiment, the benchmark cost may be calculated based on the retrieved security parameters. The implementation shortfall may be specified as a stochastic integral 310. In one implementation, implementation shortfall (IS) may be defined as follows:

$$IS = \int_0^T \left(y \cdot S_m(t, y) + f \cdot S_p(t) - \tilde{y}(t) \cdot \hat{S}(t) \right) dt,$$

where $\hat{S}(t)$ is the fundamental stock price (i.e., mid-quote), $\tilde{y}(t)$ is the benchmark trading schedule, y is the market order schedule, $S_m(t, y)$ is the observed price of executing market order y, f is the fill rate of passive limit orders, and $S_p(t)$ is the observed price of executing passive limit orders f.

[0049] In this implementation, the fundamental stock price may be based on the stochastic process describing how a security price changes over time and make take on the following form:

$$\hat{S}(t) = S_0 + \mu t + \sigma B(t)$$

[0050] The realized price of executing a market order may also be based on the stochastic process describing how a security price changes over time and make take on the following form:

$$S_m(t, y) = \hat{S}(t) + \delta_a + \alpha \left(\frac{y}{\bar{y}} \right)^{\gamma}$$

where $\bar{y}(t)$ is the expected market volume flow rate, α is the temporary impact coefficient, and market orders pay spread δ_a as well as temporary market impact $\alpha \left(\frac{y}{\bar{y}} \right)^{\gamma}$.

[0051] In one embodiment, the implementation shortfall may comprise a stochastic integral of the difference between a realized cost for a traded instrument and an arrive price benchmark cost, and may take on the following form:

$$
\begin{aligned}
IS &= \int_0^T \left(y \cdot S_m(t,y) + f \cdot S_p(t) - \tilde{y}(t) \cdot \hat{S}(t) \right) dt \\
&= \int_0^T \left(y \cdot S_m(t,y) + f \cdot S_p(t) \right) dt - S_0 \cdot N \\
&= \int_0^T \left(y \cdot (S_m(t,y) - S_0) + f \cdot (S_p(t) - S_0) \right) dt \\
&= \int_0^T \left(y \cdot \delta - f \cdot \delta + \alpha \left(\frac{y}{\bar{y}} \right)^{\gamma} \cdot y \right) dt + \int_0^T (\mu t + \sigma B)(y + f) dt \\
&= \int_0^T \left(y \cdot \delta - f \cdot \delta + \alpha \left(\frac{y}{\bar{y}} \right)^{\gamma} \cdot y \right) dt + \int_0^T \mu X dt + \int_0^T \sigma X dB
\end{aligned}
$$

where the last step follows from integration by parts.

[0052] In another embodiment, the implementation shortfall may comprise a stochastic integral of the difference between a realized cost for a traded instrument and a volume weighted average price (VWAP) benchmark cost. Let $V(t) = \int_0^t v(t)dt$ denote the normalized cumulative volume profile. *V(t)* may be a monotonically increasing function, where *V(0)*=0 and *V(T)*=1. In one implementation, the implementation shortfall corresponding to the VWAP price benchmark may be given by:

$$
\begin{aligned}
IS &= \int_0^T \left(y \cdot \delta - f \cdot \delta + \alpha \left(\frac{y}{\bar{y}} \right)^{\gamma} \cdot y \right) dt + \int_0^T (f + y - \tilde{y}) \hat{S}(t) dt \\
&= \int_0^T \left(y \cdot \delta - f \cdot \delta + \alpha \left(\frac{y}{\bar{y}} \right)^{\gamma} \cdot y \right) dt + \int_0^T \mu \left(X - (1 - V) N \right) dt \\
&\quad + \int_0^T \sigma \left(X - (1 - V) N \right) dB
\end{aligned}
$$

where the last step follows from integration by parts. Since the volume profile can be highly uncertain, VWAP analysis tends to be more difficult than its arrival price

counterpart. See Figure 5 for further details regarding specifying the implementation shortfall as a stochastic integral.

[0053] An implementation cost, also referred to herein as utility, may be specified as a function of expected implementation shortfall and/or related values 315. The function specifying implementation cost may be selected by a trader, by the MCP PLATFORM administrator, or may be hard coded into the MCP PLATFORM based on beliefs regarding the tradeoffs between using a less accurate proxy of traders' impatience when trading securities, such as a mean-variance-measure - associated with less computational complexity, and using a better proxy, such as a mean-risk-measure, having increased computational complexity.

[0054] In one embodiment, implementation cost weighs both the expected transaction cost and execution risk. For example, the implementation cost may comprise a sum of an expectation value for the implementation shortfall and an expectation risk, such as a variance and/or standard deviation of the implementation shortfall multiplied by a risk tolerance and/or aversion factor. For example, the risk aversion factor κ may be provided by the client, may be selected by the trader, or may be a default value in the MCP PLATFORM. In one implementation, the risk aversion factor may be specified quantitatively. In another implementation, the risk aversion factor may be specified as a qualitative value (e.g.: aggressive, moderate, conservative) and converted by the MCP PLATFORM into a predetermined corresponding quantitative value.

[0055] In one implementation, implementation cost may be measured using a mean-variance-measure and may take on the following form:

$$IC = E[IS] + K\mathrm{Var}[IS]$$

[0056] In another implementation, implementation cost may be measured using a quadratic-utility-measure and may take on the following form:

$$IC = E[IS] + KE[IS^2]$$

[0057] In yet another implementation, implementation cost may be measured using a mean-risk-measure and may take on the following form:

$$IC = E[IS] + K\sqrt{(\mathrm{Var}[IS])}$$

[0058] In one embodiment, the preferred implementation cost measure may be selected based on security parameters (e.g. beta, drift, volatility), order parameters (e.g. security symbol, time within which to execute the order), and/or the like. For example, if a client desires to execute a trade involving a large portfolio of stocks within one hour, the mean-variance-measure may be selected as the preferred implementation cost measure. However, if a client desires to execute a trade involving a single security over the course of a day, the mean-risk-measure may be selected as the preferred implementation cost measure. In another embodiment, the preferred implementation cost measure may be selected based on easy of use associated with the implementation cost measure as perceived by the client and/or the user. See Figure 6 for further details regarding specifying the implementation cost.

[0059] In various embodiments, the implementation cost may be optimized to determine placement policies and/or trading schedules yielding minimal implementation cost values. These extrema may then be determined by evaluating the implementation cost at those placement policies and/or trading schedules to yield

optimized implementation cost at 320. A preferred trading strategy used in this evaluation may be selected by a MCP PLATFORM administrator, selected by a MCP PLATFORM user, hard coded into a software program, and/or the like. For example, the preferred trading strategy may be selected based on computational complexity, market conditions, set of securities, desired speed of execution, and/or the like. For example, if the order parameters indicate that a client wants to trade a single stock within the course of a day, an adaptive mean-variance strategy for a single stock may be used. If the order parameters indicate that a client wants to trade a portfolio of stocks within the course of a few minutes, a non-adaptive mean-variance strategy for a portfolio of stocks may be used instead. A variety of paradigms may be used to evaluate implementation cost in various embodiments as illustrated by non-limiting paradigm examples described in the section entitled Implementation Cost Paradigms. See Figure 7 for further details regarding constructing a paradigm for evaluating the implementation cost.

[0060] An immediate contribution to the implementation cost, e.g., an instantaneous implementation cost, may be expressed as a function of current and/or immediate order placement and/or trading decisions 325. In one implementation, the immediate contribution to implementation cost is given by:

$$U_I(\Delta N) = \delta_{a,o} \cdot \min(\Delta N, N_{touch}) + (\delta_{a,o} + \delta_p) \cdot \max(\Delta N - N_{touch}, 0) + \alpha \cdot \Delta N / (\bar{\gamma} \cdot \Delta t) * (1/2 + 1/(1 + e^{-(N_{touch}/\bar{N}_{touch})}))$$

where the first two terms represent direct cost of trading ΔN shares immediately, and the third term represents the impact cost.

[0061] The MCP PLATFORM may define a total implementation cost as the sum of the instantaneous implementation cost for a given order placement decision and the corresponding go-forward optimal implementation cost, as specified by the selected paradigm, at 330. Once the total implementation cost has been determined, the MCP PLATFORM may determine an optimal current order placement decision by, in one implementation, minimizing the total implementation cost as a function of order placement decisions, time horizon, and/or the like 335. See Figures 8 and 9 for further details regarding the determined optimal current order placement.

[0062] The determined optimal current order placement may, in one implementation, be passed to a trading system to execute a trade of the specified instruments and/or number of shares 340. In another implementation, the optimal current order placement may be provided for display to a broker, trader, investor, and/or the like to aid and/or instruct a trade made thereby. A determination may be made as to whether there are more trades to be executed 345, such as for one or more financial instruments for which the total implementation cost has been established at 335. If so, then the implementation cost model may be updated with the current book of tradable instruments 350, and the updated total implementation cost may be minimized again to yield the next optimal current order placement 335. Otherwise, the current round of trading is done 355.

[0063] FIGURE 4 is of a logic flow diagram illustrating selection of stochastic processes in one embodiment of the MCP PLATFORM. In Figure 4, information regarding which market variables should be specified as stochastic processes may be retrieved 405. In one implementation, such information may be stored in a MCP

PLATFORM configuration file. In another embodiment, such information may be hard coded into a C++ software program.

[0064] In one embodiment, the MCP PLATFORM may determine whether a stochastic process for a market variable was selected by a MCP PLATFORM user at 410. For example, the MCP PLATFORM user may select a stochastic process for the market variable deemed to be best suited for given market conditions, set of securities, desired speed of execution, and/or the like. If so, the MCP PLATFORM may check whether other market variables need to be specified as stochastic processes at 425 and return to 410 until all the market variables on the list have been determined to be specified as stochastic processes.

[0065] If not, in one embodiment, the MCP PLATFORM may check at 415 whether a stochastic process for a market variable has been selected as a default value, such as by a MCP PLATFORM administrator. For example the MCP PLATFORM administrator may determine how the stochastic process for the market variable should be specified, and may update a configuration file storing default values to indicate that a selected stochastic process for the market variable should be used. In one implementation, the MCP PLATFORM administrator may update the default settings using a graphical user interface by typing in, selecting a radio button, selecting an option in a select box, and/or the like associated with the defined stochastic process for the market variable in an administration control panel of the MCP PLATFORM. In another implementation, the stochastic process for a market variable may be a default value by being hard coded into a software program, such as a software program written in C++. If there is a default value, the MCP PLATFORM may check whether other market variables

need to be specified as stochastic processes at 425 and return to 410 until all the market variables on the list have been determined to be specified as stochastic processes.

[0066] If not, in one embodiment, the MCP PLATFORM may prompt the MCP PLATFORM user to select the stochastic process for the market variable at 420, such as by using a user interface of the MCP PLATFORM (e.g. an input box, a radio button, a select box, and/or the like), by selecting and/or entering it into an excel spreadsheet, and/or the like. Once all market variables have been specified as stochastic processes, the MCP PLATFORM proceeds to 310 of Figure 3.

[0067] FIGURE 5 is of a logic flow diagram illustrating specifying implementation shortfall as a stochastic integral in one embodiment of the MCP PLATFORM. In Figure 5, information regarding available benchmarks may be retrieved at 505. In one implementation, the list of available benchmarks may be specified in a configuration file of the MCP PLATFORM. In another implementation, the list of available benchmarks may be retrieved from the MCP PLATFORM database.

[0068] At 510, information regarding the available benchmark related parameters may be retrieved. In one embodiment, such information may be specified by the user of the MCP PLATFORM (e.g. risk tolerance). In another embodiment, the MCP PLATFORM may generate such parameters based on available data (e.g. historical data stored in the MCP PLATFORM database).

[0069] At 515, a determination may be made regarding whether the user selected a preferred benchmark. In one implementation, this determination may be made based on data contained in the user request. In another implementation, this determination may be based on the available benchmark related parameters (e.g. a user profile

specifying the client's and/or the user's risk tolerance, stored in the MCP PLATFORM database).

[0070] If the user selected a preferred benchmark, the MCP PLATFORM may retrieve and/or apply such benchmark at 520. For example, such user selected benchmark may be the Volume Weighted Average Price (VWAP) benchmark. If the user did not select a preferred benchmark, the MCP PLATFORM may select a system default benchmark at 525. In one implementation, such system default benchmark may be the Arrival Price benchmark.

[0071] In one embodiment, implementation shortfall may be specified as a stochastic integral at 530 based on the selected benchmark and the specified stochastic processes for market variables.

[0072] FIGURE 6 is of a logic flow diagram illustrating specifying implementation cost function in one embodiment of the MCP PLATFORM. In Figure 6, information regarding available implementation cost measures may be retrieved at 605. In one implementation, the list of available implementation cost measures may be specified in a configuration file of the MCP PLATFORM. In another implementation, the list of available implementation cost measures may be retrieved from the MCP PLATFORM database.

[0073] At 610, information regarding the available implementation cost measure related parameters may be retrieved. In one embodiment, such information may be specified by the user of the MCP PLATFORM (e.g. ease of use associated with an implementation cost measure as perceived by the client and/or the user). In another

embodiment, the MCP PLATFORM may generate such parameters based on available data (e.g. historical data stored in the MCP PLATFORM database).

[0074] At 615, a determination may be made regarding whether the user selected a preferred implementation cost measure. In one implementation, this determination may be made based on data contained in the user request. In another implementation, this determination may be based on the available implementation cost measure related parameters (e.g. a user profile specifying the client's and/or the user's perception of ease of use associated with an implementation cost measure, stored in the MCP PLATFORM database).

[0075] If the user selected a preferred implementation cost measure, the MCP PLATFORM may retrieve and/or apply such implementation cost measure at 620. For example, such user selected implementation cost measure may be the mean-risk-measure. If the user did not select a preferred implementation cost measure, the MCP PLATFORM may select a system default implementation cost measure at 625. In one implementation, such system default implementation cost measure may be the mean-variance-measure.

[0076] In one embodiment, implementation cost function may be specified based on the selected implementation cost measure at 630.

[0077] FIGURE 7 is of a logic flow diagram illustrating constructing a paradigm for evaluating the implementation cost in one embodiment of the MCP PLATFORM. In Figure 7, information regarding selected implementation shortfall and implementation cost function may be retrieved at 705. For example, in one implementation, such information may be retrieved from a cache storing information output at 530 and 630.

In one embodiment, information regarding market parameters may also be retrieved at 710.

[0078] A paradigm of the go-forward optimal implementation cost may be constructed at 720. In one embodiment, such paradigm may be constructed based on a evaluation of the implementation cost function at the functional extrema with respect to order placement policies, using variational calculus and stochastic control theory, associated with the selected implementation shortfall, implementation cost function, and market parameters. In one implementation, the evaluation of the implementation cost function at the functional extrema with respect to order placement policies may be stored in a database. In another implementation, the evaluation of the implementation cost function at the functional extrema with respect to order placement policies may be hard coded into a C++ software program.

[0079] FIGURE 8 is of a logic flow diagram illustrating the determined optimal current order placement in one embodiment of the MCP PLATFORM. At 805, information regarding market parameters used in the evaluation of the total implementation cost may be retrieved. In one embodiment, numerical techniques may be used to minimize the total implementation cost based on the market parameters. At 815, a market order quantity associated with the minimized total implementation cost may be obtained. In one implementation, this may be a value of a C++ variable corresponding to the market order quantity that is associated with the minimized total implementation cost.

[0080] FIGURE 9 is of a drawing illustrating a non-limiting example of an Order Placement Decision question that may be solved by the MCP PLATFORM: do we submit

a market order now paying a half spread and market impact or wait on a limit order saving a half spread with the risk of not filling the order immediately and with the risk of the price deviating away from the arrival price benchmark further? As described above, in one embodiment, the MCP PLATFORM may solve this question by determining the optimal strategy or near optimal strategy for trading N shares in time interval T. In one embodiment, the MCP PLATFORM may output the market order size that should be placed during each sub-interval of the time interval T. It is to be understood that the MCP PLATFORM may solve a variety of Order Placement Decision questions. Other non-limiting examples of such questions may include: How many shares should I show now as a limit order with limit equal to the current best-bid? How many shares should I execute now as market orders? At what limit price should I post my shares on ECN-Z? How many shares should I post to DarkPoolX as Peg-Mid orders?

[0081] FIGURE 11 is of screen shot diagram illustrating one embodiment of the MCP PLATFORM user interface. In one embodiment, a trader may place an order with the MCP PLATFORM by specifying various parameters. As illustrated in figure 11, the trader may input order parameters such as security symbol, order quantity, time horizon, and/or the like. The trader may also choose a benchmark to use such as arrival price benchmark, VWAP benchmark, and/or the like. The trader may also specify an implementation cost measure to use such as mean-variance-measure, quadratic-utility-measure, mean-risk-measure, and/or the like. The trader may also specify risk tolerance that the MCP PLATFORM should use to determine a risk-aversion coefficient. The trader may submit the order by clicking on the "Submit Order" button.

Implementation Cost Paradigms

[0082] In one embodiment, a non-adaptive mean-variance strategy for trading a single stock with arrival price benchmark may be used. A non-adaptive trading strategy is a trading schedule *y(t)* that only depends on time, and the stochastic component comes from the random walk *dB(t)* of the mid-quote price. Using the non-adaptive mean-variance strategy may be less computationally intensive than using an adaptive mean-variance strategy. In one implementation, the expected implementation shortfall and the variance of the implementation shortfall for such a strategy may be expressed as:

$$E[IS] = \int_0^T \left(\delta \cdot y(t) - f \cdot \delta + \frac{\alpha}{\overline{y}^{\gamma}} \cdot y(t)^{1+\gamma} + (\mu - f\eta) X(t) \right) dt + \frac{\eta}{2} N^2$$

$$Var[IS] = \int_0^T X^2(t) \sigma^2 dt$$

[0083] Depending on a user's expectation of market performance, the temporary market impact function exponent γ may take on a variety of values. In one implementation, the user may set the temporary market impact function exponent γ to between 0 (inclusive) and 1. In this implementation, the paradigm for evaluating the implementation cost may be as follows:

Model

$$IS = \int_0^T \left[(S - \delta) f + \left(S + \delta + \alpha \left(\frac{y}{\bar{y}} \right)^{\gamma} \right) y \right] dt - N \cdot S_0$$

$$\int_0^T (y + f)\, dt = N$$

$$dS = \sigma dz$$

$$y = y(t),$$

Utility

$$U = E[IS] + \kappa Var[IS]$$

Results

We need to solve the following second-order ODE

$$\frac{\alpha\gamma(1+\gamma)}{\bar{y}^{\gamma}} \frac{\ddot{X}}{(-\dot{X} - f)^{1-\gamma}} = (\mu - f\eta) + 2\kappa\sigma^2 X$$

with the boundary condition

$$\begin{cases} X(0) = N \\ X(T) = 0 \end{cases}$$

The above ODE can be solved efficiently by any standard ODE solver such as the one provided by IMSL.

[0084]

[0085] In another implementation, the user may set the temporary market impact function exponent γ to 1. In this implementation, the paradigm for evaluating the implementation cost may be as follows (for notational brevity, zero drift and zero permanent impact is assumed):

Model

$$IS = \int_0^T \left[(S - \delta) f + \left(S + \delta + \alpha \frac{y}{\bar{y}} \right) y \right] dt - N \cdot S_0$$

$$\int_0^T (y + f)\, dt = N \,, \qquad dS = \sigma dz \,, \qquad y = y(t),$$

Utility

$$U = E[IS] + \kappa Var[IS]$$

Definitions

$$\tau = \sqrt{\frac{\alpha}{\kappa \sigma^2 \bar{y}}} \,, \qquad T^* = \min \left[T, \tau \sinh^{-1} \left(\frac{N}{f \tau} \right) \right]$$

$$\beta = \frac{T^*}{\tau} \,, \quad \tilde{t} = t/T^*, \quad \tilde{X} = X/N, \quad \tilde{f} = f/(N/T^*), \quad \tilde{y} = y/(N/T^*)$$

Results

$$\tilde{y}(\tilde{t}) = \begin{cases} \beta \frac{\cosh(\beta(1-\tilde{t}))}{\sinh(\beta)} - \tilde{f} & when \quad \tilde{t} \le 1 \\ 0 & \tilde{t} > 1 \end{cases}$$

$$\tilde{X}(\tilde{t}) = \begin{cases} \frac{\sinh(\beta(1-\tilde{t}))}{\sinh(\beta)} & when \quad \tilde{t} \le 1 \\ 0 & \tilde{t} > 1 \end{cases}$$

$$\frac{E[IS]}{N} = \delta(1 - 2\tilde{f}) + \underbrace{\alpha \frac{N}{\bar{y} T}}_{VWAP\ Market\ Impact} \cdot \xi_1$$

$$\frac{Std[IS]}{N} = \underbrace{\sigma \sqrt{\frac{T}{3}}}_{VWAP\ Risk} \cdot \xi_2$$

$$U = E[IS] + \kappa \left(Std[IS] \right)^2$$

$$\xi_1 = \left(\tilde{f}(\tilde{f} - 2) + \beta \frac{0.5 \cdot \sinh(2\beta) + \beta}{\cosh(2\beta) - 1} \right) \frac{T}{T^*}$$

$$\xi_2 = \sqrt{\frac{1}{\beta} \cdot \frac{0.5 \cdot \sinh(2\beta) - \beta}{\cosh(2\beta) - 1}} \cdot \sqrt{\frac{3T^*}{T}}$$

[0086] In another embodiment, an adaptive mean-variance strategy for trading a single stock with arrival price benchmark may be used. An adaptive trading strategy is a trading policy that depends on time as well as the state space. The state space includes the remaining position and some representation of the price history. In the absence of

persistent major parameter mis-estimates on such things as drift, an adaptive strategy leads to lower implementation cost on average than its non-adaptive counterpart. Assume market price follows a driftless arithmetic random walk $dS = \sigma dB(t)$. Consider an order of size N with trading horizon T. For brevity, assume zero limit-order fill rate (i.e., $f = 0$). Let $X(t)$ denote the remaining position of the order, and the position change is due to a trading policy $y(\cdot)$, i.e., $dX(t) = -y(\cdot)dt$. The implementation shortfall IS may have the following dynamics: $d(IS) = (\phi y^2 + \mu X)dt + \sigma X dB(t)$, where a linear temporary impact is assumed and $\phi = \frac{a}{y}$ is the impact coefficient. Let C represent the difference between IS and its expected value, namely, $C = IS - E[IS]$. Then E[C] = 0. Moreover, $d(C) = \left[\phi(y^2 - Ey^2) + \mu(X - EX)\right] dt + \sigma X dB(t)$, where Ey^2 indicates the expected value of the random variable y^2. The objective is to find the optimal trading policy y^* that minimizes the objective function $F = E[IS(T)] + \kappa \cdot Var[IS(T)]$ where κ is the risk aversion coefficient:

$$\begin{aligned} y^* &= \arg\min_{y(\cdot)} F \\ &= \arg\min_{y(\cdot)} \{E[IS(T)] + \kappa \cdot Var[IS(T)]\} \end{aligned}$$

[0087] In one implementation, the optimal adaptive strategy may be formulated in a form amenable to efficient numerical procedures. Let the state space be $\{t, X, C\}$. the whole objective function becomes:

$$F = \int_0^T \left(\phi E[y^2] + \kappa\sigma^2 E[X^2] + 2\kappa\phi E[Cy^2]\right) dt$$

[0088] To find an efficient way to calculate *E[IS(T)]* and *Var[IS(T)]*, the dynamics of the first and second moments of X and C may be as follows:

$E[IS]$

$$\begin{aligned}\frac{dE[IS]}{dt} &= \phi E[y^2] \\ &= \phi\tilde{y}^2 \text{ since } E[C] \text{ is equal to zero}\end{aligned}$$

$E[X]$

$$\begin{aligned}\frac{dE[X]}{dt} &= -E[y] \\ &\approx -\tilde{y}(1+\frac{a}{2}EC-\frac{a^2}{8}E[C^2]) \text{ by second-order Taylor expansion} \\ &= -\tilde{y}\left(1-\frac{a^2}{8}E[C^2]\right) \text{ since } E[C] \text{ is equal to zero}\end{aligned}$$

$E[X^2]$

$$\begin{aligned}\frac{dE[X^2]}{dt} &= -2E[yX] \\ &\approx -2\tilde{y}\left(E[X]+\frac{a}{2}E[XC]-\frac{a^2}{8}E[XC^2]\right) \text{ by second-order Taylor expansion}\end{aligned}$$

$Var(IS)$

$$\begin{aligned}\frac{dVar[IS]}{dt} &= \frac{dE[C^2]}{dt} \\ &= E[2\phi C(y^2-Ey^2)+\sigma^2X^2] \\ &= 2a\phi\tilde{y}^2E[C^2]+\sigma^2E[X^2]\end{aligned}$$

$$E[XC]$$

$$\begin{aligned}\frac{dE[XC]}{dt} &= E[\phi X(y^2 - Ey^2) - yC] \\ &= a\phi\tilde{y}^2 E[XC] - \tilde{y}E[C\sqrt{1+aC}] \\ &\approx a\phi\tilde{y}^2 E[XC] - \frac{a}{2}\tilde{y}E[C^2] + \frac{a^2}{8}\tilde{y}E[C^3] \text{ by second-order Taylor expansion}\end{aligned}$$

[0089] In the case of a linear market impact function, the four ordinary differential equations (ODEs) and boundary conditions may be as follows:

$$\begin{aligned}\frac{dE[X]}{dt} &= -\tilde{y}\left(1 - \frac{a^2}{8}E[C^2]\right) \\ \frac{dE[X^2]}{dt} &= -2\tilde{y}\left(E[X] + \frac{a}{2}E[XC]\right) \\ \frac{dE[C^2]}{dt} &= 2a\phi\tilde{y}^2 E[C^2] + \sigma^2 E[X^2] \\ \frac{dE[XC]}{dt} &= a\phi\tilde{y}^2 E[XC] - \frac{a}{2}\tilde{y}E[C^2]\end{aligned}$$

$$\begin{aligned}E[X]|_{t=0} &= N \\ E[C^2]|_{t=0} &= 0 \\ E[X^2]|_{t=0} &= N^2 \\ E[XC]|_{t=0} &= 0\end{aligned}$$

[0090] An efficient one-dimensional search over a to find the optimal a^* that minimizes the mean-variance objective function may be done. For each potential value of a, the above system of ODEs may be solved, which can be done on the order of one hundredth of a second.

[0091] In the case of a power-law market impact function, the four ordinary differential equations (ODEs) and boundary conditions may be as follows:

$$\frac{dE[X]}{dt} = -\tilde{y}\left(1 - \frac{\gamma a^2}{2(1+\gamma)^2}E[C^2]\right)$$
$$\frac{dE[X^2]}{dt} = -2\tilde{y}\left(E[X] + \frac{a}{1+\gamma}E[XC]\right)$$
$$\frac{dE[C^2]}{dt} = 2a\phi\tilde{y}^{1+\gamma}E[C^2] + \sigma^2 E[X^2]$$
$$\frac{dE[XC]}{dt} = a\phi\tilde{y}^{1+\gamma}E[XC] - \frac{1}{1+\gamma}a\tilde{y}E[C^2]$$

$$E[X]\big|_{t=0} = N$$
$$E[C^2]\big|_{t=0} = 0$$
$$E[X^2]\big|_{t=0} = N^2$$
$$E[XC]\big|_{t=0} = 0$$

[0092] As in the linear-market-impact case, an efficient one-dimensional search over a to find the optimal a^* that minimizes the mean-variance objective function may be done.

[0093] In another embodiment, a non-adaptive mean-variance strategy for trading a portfolio of stocks with arrival price benchmark may be used. In this embodiment, the price of the *i*th stock $\hat{S}_i(t)$ is assumed to have an arithmetic random walk as follows:

$$d\tilde{S}_i(t) = \mu_i dt + \sum_j \Gamma_{ij} dB_j(t) = \mu_i dt + \Gamma_i dB(t)$$

[0094] The random walks of the *i*th stock and the *j*th stock are correlated, and the correlation can be described using the covariance matrix Σ, where its element is:

[0095] $$\Sigma_{ij} = cov(d\tilde{S}_i(t), d\tilde{S}_j(t)) \Rightarrow \Sigma = \Gamma\Gamma^T$$

[0096] In one implementation, the paradigm for evaluating the implementation cost may be as follows:

The expected cost can be written as

$$E[IS] = \int_0^T \left(2\tilde{\mu}^T \cdot \Lambda Z(t) + \dot{Z}^T(t) \cdot \dot{Z}(t)\right) dt + \delta^T \cdot N,$$

and the execution risk is

$$Var[IS] = \int_0^T \frac{1}{\kappa} Z^T \cdot \Lambda Z(t) dt.$$

[0097] The close-form solutions may be obtained as follows:

$$\begin{aligned}
E[IS] =& \tilde{Z}^T \cdot M_1 \tilde{Z} + \tilde{\mu}^T \cdot M_2 \tilde{Z} + \tilde{\mu}^T \cdot M_3 \tilde{\mu} + \delta^T \cdot N \\
\kappa Var[IS] =& \tilde{Z}^T \cdot M_4 \tilde{Z} + \tilde{\mu}^T \cdot M_5 \tilde{Z} + \tilde{\mu}^T \cdot M_6 \tilde{\mu} \\
M_1 =& \frac{1}{2}\lambda \frac{\sinh(2\lambda T) + 2\lambda T}{\cosh(2\lambda T) - 1} \\
M_2 =& 2\lambda \frac{\cosh(\lambda T) - 1}{\sinh(\lambda T)} + 2\lambda \frac{(\cosh(\lambda T) - 1)(\sinh(\lambda T) - \lambda T)}{\cosh(2\lambda T) - 1} \\
M_3 =& 4\lambda \frac{\cosh(\lambda T) - 1}{\sinh(\lambda T)} - 2\Lambda T + 2\lambda \frac{(\cosh(\lambda T) - 1)(\sinh(\lambda T) - \lambda T)}{\cosh(2\lambda T) - 1} \\
M_4 =& \frac{1}{2}\lambda \frac{\sinh(2\lambda T) - 2\lambda T}{\cosh(2\lambda T) - 1} \\
M_5 =& 2\lambda \frac{(\cosh(\lambda T) - 1)(\sinh(\lambda T) + \lambda T)}{\cosh(2\lambda T) - 1} - 2\lambda \frac{\cosh(\lambda T) - 1}{\sinh(\lambda T)} \\
M_6 =& 2\lambda \frac{(\cosh(\lambda T) - 1)(\sinh(\lambda T) + \lambda T)}{\cosh(2\lambda T) - 1} - 4\lambda \frac{\cosh(\lambda T) - 1}{\sinh(\lambda T)} + \Lambda T \\
\lambda =& \Lambda^{1/2}
\end{aligned}$$

[0098] The implementation cost may be written as follows:

$$
\begin{aligned}
IC =& E[IS] + \kappa Var[IS] \\
=& \tilde{Z}^T \cdot \lambda \frac{\cosh(\lambda T)}{\sinh(\lambda T)} \tilde{Z} + \tilde{\mu}^T \cdot 2\lambda \frac{\cosh(\lambda T) - 1}{\sinh(\lambda T)} \tilde{Z} \\
& + \tilde{\mu}^T \cdot \left(2\lambda \frac{\cosh(\lambda T) - 1}{\sinh(\lambda T)} - \Lambda T \right) \tilde{\mu} + \delta^T \cdot N
\end{aligned}
$$

[0099] In the special case with zero drift, the terms become simplified as follows:

$$
\begin{aligned}
E[IS] =& \frac{1}{2} \tilde{Z}^T \cdot \lambda \frac{\sinh(2\lambda T) + 2\lambda T}{\cosh(2\lambda T) - 1} \tilde{Z} \\
\kappa Var[IS] =& \frac{1}{2} \tilde{Z}^T \cdot \lambda \frac{\sinh(2\lambda T) - 2\lambda T}{\cosh(2\lambda T) - 1} \tilde{Z} \\
IC =& \tilde{Z}^T \cdot \lambda \frac{\cosh(\lambda T)}{\sinh(\lambda T)} \tilde{Z} + \delta^T \cdot N
\end{aligned}
$$

[00100] In another implementation, a commercial optimizer may be used to obtain the trajectories of the constrained problem, and the trajectories obtained in the optimizer and the closed-form objective function obtained analytically may be combined as follows. Based on the optimal trajectories obtained in the optimizer, the entire trading horizon is divided into sub-intervals [$t_0 = 0$, t_1, t_2, ..., t_{n-1}, $t_n = T$], where a stock is either trading or not trading during each sub-interval. The trajectories will also provide the remaining shares of the portfolio at the end of each sub-period, [$\boldsymbol{N}_1, \boldsymbol{N}_2, ..., \boldsymbol{N}_n = 0$]. When trading during the sub interval ending at t_i, use the analytical framework to optimize with the boundary condition $\boldsymbol{X}(t_i) = \boldsymbol{N}_i$. Periodically the optimizer is called to update the sub-interval information.

[00101] The following notation is used. The elements of the stock position vector $\boldsymbol{X}$ are re-arranged such that the actively traded stocks are listed first, followed by the stocks not trading during the sub-interval:

$$X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$$

[00102] The diagonal matrix of temporary impact coefficient, the covariance matrix, and the drift term may also be re-arranged accordingly:

$$\Phi = \begin{bmatrix} \Phi_1 & 0 \\ 0 & \Phi_2 \end{bmatrix}, \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ {\Sigma_{12}}^T & \Sigma_{22} \end{bmatrix}, \mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$$

[00103] Denote the terminal stock positions as:

$$X(t_i) = \begin{bmatrix} N_1 \\ N_2 \end{bmatrix}$$

[00104] Denote the quantities traded during the sub-intervals as:

$$X(t_{i-1}) - X(t_i) = \begin{bmatrix} \Delta N_1 \\ \Delta N_2 \end{bmatrix} = \begin{bmatrix} \Delta N_1 \\ 0 \end{bmatrix}$$

[00105] The solution is given as follows:

$$\begin{aligned} X_1 =& {\Phi_1}^{-1/2} P^T Z \\ \dot{X}_1 =& {\Phi_1}^{-1/2} P^T \dot{Z} \\ Z =& \tilde{Z}_0 + \frac{\sinh(\Lambda^{1/2}(t_i - t))}{\sinh(\Lambda^{1/2}(t_i - t_{i-1}))} \tilde{Z} \\ &+ \left(\frac{\sinh(\Lambda^{1/2}(t - t_{i-1}))}{\sinh(\Lambda^{1/2}(t_i - t_{i-1}))} + \frac{\sinh(\Lambda^{1/2}(t_i - t))}{\sinh(\Lambda^{1/2}(t_i - t_{i-1}))} - 1 \right) \tilde{\mu} \end{aligned}$$

$$\dot{\tilde{Z}} = -\frac{\cosh(\Lambda^{1/2}(t_i - t))}{\sinh(\Lambda^{1/2}(t_i - t_{i-1}))}\Lambda^{1/2}\tilde{Z} + \left(\frac{\cosh(\Lambda^{1/2}(t - t_{i-1}))}{\sinh(\Lambda^{1/2}(t_i - t_{i-1}))} - \frac{\cosh(\Lambda^{1/2}(t_i - t))}{\sinh(\Lambda^{1/2}(t_i - t_{i-1}))}\right)\Lambda^{1/2}\tilde{\mu}$$

$$\tilde{Z} = P\Phi_1{}^{1/2}\Delta N_1$$

$$\tilde{Z}_0 = P\Phi_1{}^{1/2}N_1$$

$$\tilde{\mu} = \frac{1}{2}\Lambda^{-1}P\Phi_1{}^{-1/2}(\mu_1 + 2\Sigma_{12}X_2)$$

$$\Lambda = \kappa\Phi_1{}^{-1/2}\Sigma_{11}\Phi_1{}^{-1/2}$$

$$\lambda = \Lambda^{1/2}$$

[00106] Let $T = t_i - t_{i-1}$, and the cost function can be computed as follows:

$$\begin{aligned} IC = &\tilde{Z}^T \cdot \lambda\frac{\cosh(\lambda T)}{\sinh(\lambda T)}\tilde{Z} + \tilde{\mu}^T \cdot 2\lambda\frac{\cosh(\lambda T) - 1}{\sinh(\lambda T)}\tilde{Z} \\ &+ \tilde{Z}_0{}^T \cdot 2\lambda\frac{\cosh(\lambda T) - 1}{\sinh(\lambda T)}\tilde{Z} + \tilde{Z}_0{}^T \cdot 4\lambda\frac{\cosh(\lambda T) - 1}{\sinh(\lambda T)}\tilde{\mu} \\ &+ \tilde{\mu}^T \cdot \left(2\lambda\frac{\cosh(\lambda T) - 1}{\sinh(\lambda T)} - \Lambda T\right)\tilde{\mu} + \delta^T \cdot \Delta N \\ &+ T\left(\mu_2{}^T \cdot X_2 + \kappa X_2{}^T \cdot \Sigma_{22}X_2 + \tilde{Z}_0{}^T \cdot \Lambda\tilde{Z}_0\right) \end{aligned}$$

[00107] Using the linear impact model, both the future and current risk-adjusted costs are quadratic functions, so the optimal trading quantities can be determined analytically, and the optimal trading quantities can be obtained by minimizing the sum of the costs as follows:

$$\begin{aligned} C &= C_0(\Delta N) + C_1(N - \Delta N) \\ &= \Delta N^T \cdot (A_0 + A_1)\Delta N + (b_0 - b_1 - 2A_1N)^T \cdot \Delta N \\ &\quad + N^T \cdot A_1N + b_1{}^T \cdot N \end{aligned}$$

[00108] In another implementation, a dollar-neutral penalty term may also be added to the objective function to take into account dollar-neutrality as follows:

$$IC = \int_0^T \left[\delta^T y - \delta^T f + (\Phi y)^T y + \mu^T X + \kappa X^T \Sigma X + \lambda \sigma_B^2 \left(1^T X\right)^2\right] dt$$
$$= \int_0^T \left[\delta^T y - \delta^T f + (\Phi y)^T y + \mu^T X + (\kappa + \lambda) X^T \left(\frac{\kappa}{\kappa + \lambda}\Sigma + \frac{\lambda}{\kappa + \lambda} 1\sigma_B^2 1^T\right) X\right] dt$$

[00109] In another implementation, time-dependent parameters may be used. Let's break up the horizon [0; T] into two parts, namely, [0, T_1] and [T_1, T]. The parameters used in [0, T_1] are based on current market conditions, while the parameters used in [T_1, T] are estimated using some smoothing mechanism that takes into account both current observations and historical patterns. In this implementation, the total implementation cost may be expressed as follows:

$$IC = \int_0^{T_1} \left(\delta_1^T y + y^T \Phi_1 y + \kappa X^T \Sigma_1 X\right) dt + \int_{T_1}^T \left(\delta_2^T y + y^T \Phi_2 y + \kappa X^T \Sigma_2 X\right) dt$$
$$= \int_0^{T_1} \left(\delta_1^T y + \dot{Z}_1^T \dot{Z}_1 + Z_1^T \Lambda_1 Z_1\right) dt + \int_{T_1}^T \left(\delta_2^T y + \dot{Z}_2^T \dot{Z}_2 + Z_2^T \Lambda_2 Z_2\right) dt$$

where the subscripts 1 and 2 indicate the association with the first segment [0, T_1] and the second segment [T_1, T], respectively. $\boldsymbol{Z}_1$ and $\boldsymbol{Z}_2$ are some orthonormal transformations of $\boldsymbol{X}$. The boundary conditions are $\boldsymbol{X}(0) = \boldsymbol{N}$ and $\boldsymbol{X}(T) = 0$.

[00110] The optimal cost-to-go value for [T_1, T] is equal to:

$$C(T_1, T) = \delta_2^T X(T_1) + [Z_2(T_1)]^T \left[\Lambda_2^{1/2} coth\left(\Lambda_2^{1/2}(T - T_1)\right)\right] Z_2(T_1)$$

[00111] The optimal cost-to-go value for [0, T_1] is equal to:

$$C(0,\ T_1) = \delta_1^T [N - X(T_1)] + [Z_1(0) - Z_1(T_1)]^T \left[\Lambda_1^{1/2} coth(\Lambda_1^{1/2} T_1)\right] [Z_1(0) - Z_1(T_1)] + 2[Z_1(T_1)]^T \left[\Lambda_1^{1/2} \left(coth(\Lambda_1^{1/2} T_1) - csch(\Lambda_1^{1/2} T_1)\right)\right] [Z_1(0) - Z_1(T_1)]$$

[00112] The total cost-to-go value for [0, T] is equal to:

$$C(0,\ T) = C(0,\ T_1) + C(T_1,\ T)$$

[00113] In theory, the MCP PLATFORM may use more segments than just two, albeit at the cost of heavier computational burden. The MCP PLATFORM may apply this approach in a rolling horizon fashion so that it may distinguish between current market conditions and expected market conditions throughout the execution.

[00114] In another embodiment, a dynamic mean-variance strategy for trading a portfolio of stocks with arrival price benchmark may be used. Along the same lines as in the single-stock case, the dynamic portfolio execution problem is to minimize the following objective function:

$$F = E\left[\int_0^T \left(\delta^T y + y^T \Phi y + \kappa X^T \Sigma X\right) dt\right] + 2\kappa \int_0^T Cov\left(IS,\ y^T \Phi y\right) dt$$

[00115] In this embodiment, the following ODEs and corresponding boundary conditions may be used:

$$\frac{dE[X]}{dt} = -\left(1 - \frac{a^2}{8}E[C^2]\right)\tilde{y}$$
$$\frac{dE[X^T\Sigma X]}{dt} = -2\tilde{y}^T\Sigma\left(E[X] + \frac{a}{2}E[CX]\right)$$
$$\frac{dE[C^2]}{dt} = 2a\tilde{y}^T\Phi\tilde{y}E[C^2] + E[X^T\Sigma X]$$
$$\frac{dE[CX]}{dt} = a\tilde{y}^T\Phi\tilde{y}E[CX] - \frac{a}{2}E[C^2]\tilde{y}$$

1

$$E[X]\big|_{t=0} = N$$
$$E[C^2]\big|_{t=0} = 0$$
$$E[X^T\Sigma X]\big|_{t=0} = N^T\Sigma N$$
$$E[CX]\big|_{t=0} = 0$$

2

3 **[00116]** In another embodiment, a dynamic mean-variance strategy for trading a
4 single stock with VWAP benchmark may be used. Let v_t be the normalized
5 instantaneous volume profile corresponding to [0, T]. Denote the cumulative volume
6 profile $\int_0^t v_t dt$ by V_t. Basically, V_t is equal to the market volume in [0, t] as a fraction of
7 the market volume in [0, T]. Then $V_0 = 0$ and $V_T = 1$. Also, by definition, $dV_t = v_t dt$. If
8 the volume profile is flat (i.e., volume time instead of calendar time is used), then V_t is
9 equal to t/T. Since the total market volume in [0, T] is not observable until time T and
10 the final market volume is countable, V_t is a semimartingale.

11 **[00117]** Let S_t be the mid-quote price that follows some arithmetic random walk.
12 Then the VWAP price in the interval [0, T] is:

$$VWAP_{[0,\,T]} = \int_0^T S_t dV = \int_0^T v \cdot S_t dt$$

13

14 **[00118]** The VWAP price in the interval [0, t] is:

$$VWAP_{[0,\ t)} = \frac{\int_0^t v_t \cdot S_t dt}{V_t}$$

[00119] In the absence of uncertainty in V_t, using calculus gives:

$$\begin{aligned} d(VWAP_t) &= \frac{v_t}{V_t}(P_t - VWAP_t)\,dt \\ &= (P_t - VWAP_t))\,d(logV) \end{aligned}$$

[00120] For uncertain volume profile, one option is to model V_t as a Brownian bridge. Let v_t represent the expected instantaneous volume profile. Let σ be the volatility corresponding to the Brownian bridge. In this case $dV_t = v_t dt + \sigma d(BB)$ where BB is a standard Brownian bridge and can be assumed to be independent of the price Brownian motion.

[00121] Let $Cov(P_t,\ V_t)$ be the instantaneous covariance between S_t and V_t. The implementation shortfall corresponding to the VWAP benchmark is as follows:

$$\begin{aligned} IS &= \left[\int_0^T \alpha\left(\frac{y}{\bar{y}}\right)^{\gamma} \cdot y dt + \int_0^T (f+y)\,S_t dt\right] - N \cdot \int_0^T v \cdot S dt \\ &= \left[\int_0^T \alpha\left(\frac{y}{\bar{y}}\right)^{\gamma} \cdot y dt + \int_0^T X dS + NS_0\right] - N \cdot \int_0^T S dV \\ &= \left[\int_0^T \alpha\left(\frac{y}{\bar{y}}\right)^{\gamma} \cdot y dt + \int_0^T X dS + NS_0\right] - N \cdot S_T \\ &\quad + N \cdot \int_0^T V dS - N \int_0^T Cov(S_t,\ V_t) dt \\ &= \int_0^T \alpha\left(\frac{y}{\bar{y}}\right)^{\gamma} \cdot y dt + \int_0^T [X - N \cdot (1-V)]\,dS - N \int_0^T Cov(S_t,\ V_t) dt \end{aligned}$$

[00122] Since the last term is exogenous (i.e., it is constant from the mean-variance perspective), it may be dropped it from consideration. Assuming $dS = \sigma dB$, then:

$$dIS = \alpha \left(\frac{y}{\bar{y}}\right)^{\gamma} \cdot y dt + \sigma \left[X - N \cdot (1 - V)\right] dB$$

[00123] Denote the cumulative trade quantity by Y_t.In this case, $Y_t = N - X_t$. Define P&L at time t by $\int_0^t dIS$, namely:

$$\begin{aligned} IS_t &= \int_0^t \alpha \left(\frac{y}{\bar{y}}\right)^{\gamma} \cdot y dt + \int_0^t \left[X - N \cdot (1 - V)\right] dS \\ &= \int_0^t \alpha \left(\frac{y}{\bar{y}}\right)^{\gamma} \cdot y dt + \int_0^t (f + y) S dt + (NV_t - Y_t) S_t - NV_t \cdot VWAP_t \end{aligned}$$

where NV_t is the target cumulative trade quantity at time t. The sum of the first two terms in the last equation corresponds to the actual cost paid for buying Y_t shares in [0,t). The sum of this actual cost and the third term represents the "realized" cost of trading NV_t shares by time t (assuming trading at time t is frictionless). In other words, if Y_t is greater (less) than NV_t, the MCP PLATFORM sells (buys) the difference at midquote at time t. The last term in the above equation is NV_t times the realized VWAP price between 0 and t. Therefore, P&L at time t is the difference between the "realized" cost and the benchmark cost.

[00124] If the objective function is set to be of quadratic utility type, then the dynamic objective function is as follows:

$$F = E\left[\int_0^T \phi y^2 + \kappa \sigma^2 \left[X - N(1 - V)\right]^2 + 2\kappa\phi C y^2 dt\right]$$

and fits into the adaptive stochastic control framework.

[00125] Define VWAP slippage, denoted by *VS*, to be $\frac{\bar{S}-VWAP}{S_0}$. Let $\bar{S}$ indicate the average execution price seen by VWAP in [0, *T*]. Let $\overline{MI}$ represent the average (relative) market impact incurred in [0, *T*], namely, $\overline{MI} = \frac{1}{T}\int_0^T \frac{MI_t}{S_0} dt$. Let *PR* be the participation rate, namely, $PR = N/V_T$. In this case:

$$VS = \frac{\bar{S} - VWAP}{S_0} = (1 - PR) \cdot \overline{MI}$$

[00126] The above relationship also sheds some light on dynamic VWAP. If, in the presence of volume profile uncertainty, the performance of a dynamic VWAP strategy is close to $(1 - PR) \cdot \overline{MI}$, then little may be done to reduce the implementation cost.

[00127] In another embodiment, a non-adaptive mean-risk strategy for trading a single stock with arrival price benchmark may be used. In one implementation, the user may set the temporary market impact function exponent γ to between 0 (inclusive) and 1. In this implementation, the paradigm for evaluating the implementation cost may be as follows:

Model

$$IS = \int_0^T \left[(S - \delta) f + \left(S + \delta + \alpha \sqrt{\frac{y}{\bar{y}}} \right) y \right] dt - NS_0$$

$$\int_0^T (y + f)\, dt = N$$

$$dS = \sigma dz$$

$$y = y(t).$$

Utility

$$U = E[IS] + \kappa \int_0^T \sigma\sqrt{t}(y+f)dt$$

Definitions

$$\beta = \frac{\kappa\sigma\sqrt{\bar{y}}}{\alpha}$$

$$T_{max} = \sqrt{\frac{27N}{2\beta^2} + \left(\frac{27f}{4\beta^2}\right)^2} - \frac{27f}{4\beta^2}$$

$$T^* = \min(T, T_{max})$$

$$b = \frac{\kappa\sigma\sqrt{T^*}}{\alpha\sqrt{\frac{N}{\bar{y}T^*}}} = \frac{\beta T^*}{\sqrt{N}}$$

$$a = \frac{4}{9}b + \sqrt{1 - \tilde{f} - \frac{2}{81}b^2}$$

$$\tilde{t} = t/T^*, \quad \tilde{X} = X/N, \quad \tilde{f} = f/(N/T^*), \quad \tilde{y} = y/(N/T^*)$$

Results

$$\tilde{y}(\tilde{t}) = \begin{cases} \left(a - \frac{2}{3}b\sqrt{\tilde{t}}\right)^2 & when \quad \tilde{t} \le 1 \\ 0 & \tilde{t} > 1 \end{cases}$$

$$\tilde{X}(\tilde{t}) = \begin{cases} \left(1-\tilde{t}\right)\left(1 + \frac{2}{9}b^2\tilde{t}\right) - \frac{8}{9}ab\tilde{t}\left(1 - \sqrt{\tilde{t}}\right) & when \quad \tilde{t} \le 1 \\ 0 & \tilde{t} > 1 \end{cases}$$

$$\frac{E[IS]}{N} = \delta(1-2\tilde{f}) + \underbrace{\alpha\sqrt{\frac{N}{\bar{y}T}}}_{VWAP\ Market\ Impact} \cdot\xi_1$$

$$\frac{Std[IS]}{N} = \underbrace{\sigma\sqrt{\frac{T}{3}}}_{VWAP\ Risk} \cdot\xi_2$$

$$\frac{U}{N} = \delta(1-2\tilde{f}) + \alpha\sqrt{\frac{N}{\bar{y}T}}\cdot\xi_3$$

$$\xi_1 = \left(a^3 - \frac{4}{3}a^2b + \frac{2}{3}ab^2 - \frac{16}{135}b^3\right)\cdot\sqrt{\frac{T}{T^*}}$$

$$\xi_2 = \sqrt{1 - \frac{88}{315}ab + \frac{1}{9}b^2 + \frac{16}{567}a^2b^2 + \frac{2}{405}b^4}\cdot\sqrt{\frac{T^*}{T}}$$

$$\xi_3 = \left(a^3 - \frac{2}{3}a^2b + \frac{8}{135}b^3\right)\cdot\sqrt{\frac{T}{T^*}}$$

[00128] In another implementation, the user may set the temporary market impact function exponent γ to 1. In this implementation, the paradigm for evaluating the implementation cost may be as follows (for notational brevity, zero drift and zero permanent impact is assumed):

Model

$$IS = \int_0^T \left[(S-\delta)f + \left(S + \delta + \alpha\frac{y}{\bar{y}}\right)y\right]dt - NS_0$$

$$\int_0^T (y+f)\,dt = N$$

$$dS = \sigma dz$$

$$y = y(t),$$

Utility

$$U = E[IS] + \kappa\int_0^T \sigma\sqrt{t}(y+f)dt$$

Definitions

$$\beta = \frac{\kappa\sigma\bar{y}}{\alpha}$$

$$T_{max} = T\left(2\frac{N}{\beta T^{1.5}} - 2\frac{fT}{N}\cdot\frac{N}{\beta T^{1.5}} + \frac{2}{3}\right)^2$$

$$T^* = \min(T, T_{max})$$

$$b = \frac{\kappa\sigma\sqrt{T^*}}{\alpha\frac{N}{\bar{y}T^*}} = \frac{\beta(T^*)^{1.5}}{N}$$

$$\tilde{t} = t/T^*, \quad \tilde{X} = X/N, \quad \tilde{f} = f/(N/T^*), \quad \tilde{y} = y/(N/T^*)$$

Results

$$\tilde{y}(\tilde{t}) = \begin{cases} 1 + \frac{b}{3} - \tilde{f} - \frac{b}{2}\sqrt{\tilde{t}} & when \quad \tilde{t} \le 1 \\ 0 & \tilde{t} > 1 \end{cases}$$

$$\tilde{X}(\tilde{t}) = \begin{cases} 1 - \left(1 + \frac{b}{3}\right)\tilde{t} + \frac{b}{3}(\tilde{t})^{1.5} & when \quad \tilde{t} \le 1 \\ 0 & \tilde{t} > 1 \end{cases}$$

$$\frac{E[IS]}{N} = \delta(1 - 2\tilde{f}) + \underbrace{\alpha\frac{N}{\bar{y}T}}_{VWAP\ Market\ Impact}\cdot\xi_1$$

$$\frac{Std[IS]}{N} = \underbrace{\sigma\sqrt{\frac{T}{3}}}_{VWAP\ Risk}\cdot\xi_2$$

$$\frac{U}{N} = \delta(1 - 2\tilde{f}) + \alpha\frac{N}{\bar{y}T}\cdot\xi_3$$

$$\xi_1 = \left(1 + \tilde{f}^2 - 2\tilde{f} + \frac{b^2}{72}\right)\cdot\frac{T}{T^*}$$

$$\xi_2 = \sqrt{1 - \frac{11}{105}b + \frac{b^2}{252}}\cdot\sqrt{\frac{T^*}{T}}$$

$$\xi_3 = \left(1 + \tilde{f}^2 - 2\tilde{f} + \frac{2}{3}b - \frac{b^2}{72}\right)\cdot\frac{T}{T^*}$$

[00129] In one implementation, the MCP PLATFORM may re-optimize every time the MCP logic wakes up. Suppose the MCP PLATFORM is at t_0. The goal is to find the

optimal ΔN shares that the MCP PLATFORM should trade in the next Δt seconds using market orders, which is part of the optimal policy that minimizes the overall cost integrated from t_0 to T.

[00130] The integral from t_0 to T is broken up into two parts, namely, the integral from t_0 to $t_0 + \Delta t$ (*immediate cost*) and the integral from $t_0 + \Delta t$ to T (*future cost*). The MCP PLATFORM may approximate the integral from t_0 to $t_0+\Delta T$ by a rectangle. The first integral may be based on real-time parameter observations and accounts for current market conditions. The second integral may be based on smoothed parameter estimates.

[00131] Even though the MCP PLATFORM has a closed-form solution when the cost is integrated from 0 to T, the MCP PLATFORM does not have a closed-form solution when the lower limit of the cost integral is positive. To evaluate the optimal cost-to-go from $t_0 + \Delta t$ to T, the MCP PLATFORM may use efficient numerical techniques built around the closed-form solution corresponding to the cost integral from 0 to T.

[00132] For example, the MCP PLATFORM may use Bellman's principle of optimality. Suppose there is an optimal trade-out trajectory from 0 to T *and* a tail of the trajectory from t to T for some $0 < t < T$. If the actual $X(t)$ is equal to the number of remaining shares projected by the initial trade-out trajectory, the tail is the optimal solution that minimizes the cost integral from t to T. If other parameters constant are held constant, $X(t)$ is monotonically increasing in the order size. Therefore, to minimize the cost integral from t to T, the MCP PLATFORM may first perform an efficient search (e.g., bisection search or golden search) that finds the fictitious order size for which $X(t)$

is equal to the observed number of remaining shares at t. For this fictitious order size, there is a closed-form solution of the trade-out trajectory from 0 to T. The tail of this trade-out trajectory from t to T is the optimal solution that minimizes the cost integral from t to T. Therefore, the future cost can be calculated in closed-form.

[00133] Let M denote the upper bound on ΔN. For each round-lot ΔN between 0 and M, the MCP PLATFORM may calculate the sum of the immediate cost and future cost and choose the one that minimizes the overall cost.

[00134] The immediate cost is a trivially convex function in ΔN. The future cost is also convex in ΔN since it is convex in $X(t) - \Delta N$. Why is the future cost convex in $X(t) - \Delta N$ (i.e., the number of remaining shares at $t + \Delta t$)? The incremental cost of trading one more share is increasing in the number of remaining shares, which is a plain English way of defining convexity. Since the sum of two convex functions is convex, the overall cost function is convex in ΔN. As such, when the MCP PLATFORM conducts a search for the optimal ΔN, the MCP PLATFORM can stop as soon as it finds a local optimal solution, which is guaranteed to be globally optimal due to the convexity property of the overall cost function.

[00135] In another implementation, the MCP PLATFORM may receive an order having time T beyond market close for the current day. Without loss of generality, let's assume the MCP PLATFORM will be done with execution by the end of the next trading day. Let T_1, T_2 and T_3 indicate current day's market close, next day's market open and next day's market close, respectively. The boundary conditions are $X(0) = N$, $X(T_3) = 0$ and $X(t) = M \; \forall \; T_1 \leq t \leq T_2$, where M is the remaining quantity at the end of the current trading day. The optimal M^* can be obtained using an efficient one-dimensional

search technique. Let σ_1 and σ_2 indicate intra-day volatility and overnight-holding volatility, respectively. Then the MCP PLATFORM can break up a penalty function for risk into three parts as follows:

$$\int_0^{T_1} \frac{\sigma_1 X}{2\sqrt{t}} dt + \int_{T_1}^{T_2} \frac{\sigma_2 M}{2\sqrt{t}} dt + \int_{T_2}^{T_3} \frac{\sigma_1 X}{2\sqrt{t}} dt$$

which is equivalent to:

$$\int_0^{T_1} \sigma_1 \sqrt{t} y dt + M(\sigma_2 - \sigma_1)\left(\sqrt{T_2} - \sqrt{T_1}\right) + \int_{T_2}^{T_3} \sigma_1 \sqrt{t} y dt$$

[00136] The MCP PLATFORM may calculate the cost of optimally trading N - M shares in day 1, for which the MCP PLATFORM has a closed-form solution via Euler-Lagrange equation with the appropriate boundary condition. The overnight holding risk can be calculated in closed-form as well. The cost of optimally trading M shares in day 2 can be computed semi-analytically through a combination of closed-form solution, Bellman's principle of optimality, warm-start-based bisection search and the convexity property.

[00137]

[00138]

MCP PLATFORM Controller

[00139] FIGURE 10 illustrates inventive aspects of a MCP PLATFORM controller 1001 in a block diagram. In this embodiment, the MCP PLATFORM controller 1001 may serve to aggregate, process, store, search, serve, identify, instruct, generate, match,

and/or facilitate interactions with a computer through financial trading technologies, and/or other related data.

[00140] Typically, users, which may be people and/or other systems, may engage information technology systems (e.g., computers) to facilitate information processing. In turn, computers employ processors to process information; such processors 1003 may be referred to as central processing units (CPU). One form of processor is referred to as a microprocessor. CPUs use communicative circuits to pass binary encoded signals acting as instructions to enable various operations. These instructions may be operational and/or data instructions containing and/or referencing other instructions and data in various processor accessible and operable areas of memory 1029 (e.g., registers, cache memory, random access memory, etc.). Such communicative instructions may be stored and/or transmitted in batches (e.g., batches of instructions) as programs and/or data components to facilitate desired operations. These stored instruction codes, e.g., programs, may engage the CPU circuit components and other motherboard and/or system components to perform desired operations. One type of program is a computer operating system, which, may be executed by CPU on a computer; the operating system enables and facilitates users to access and operate computer information technology and resources. Some resources that may be employed in information technology systems include: input and output mechanisms through which data may pass into and out of a computer; memory storage into which data may be saved; and processors by which information may be processed. These information technology systems may be used to collect data for later retrieval, analysis, and manipulation, which may be facilitated through a database program. These information

technology systems provide interfaces that allow users to access and operate various system components.

[00141] In one embodiment, the MCP PLATFORM controller 1001 may be connected to and/or communicate with entities such as, but not limited to: one or more users from user input devices 1011; peripheral devices 1012; an optional cryptographic processor device 1028; and/or a communications network 1013.

[00142] Networks are commonly thought to comprise the interconnection and interoperation of clients, servers, and intermediary nodes in a graph topology. It should be noted that the term "server" as used throughout this application refers generally to a computer, other device, program, or combination thereof that processes and responds to the requests of remote users across a communications network. Servers serve their information to requesting "clients." The term "client" as used herein refers generally to a computer, program, other device, user and/or combination thereof that is capable of processing and making requests and obtaining and processing any responses from servers across a communications network. A computer, other device, program, or combination thereof that facilitates, processes information and requests, and/or furthers the passage of information from a source user to a destination user is commonly referred to as a "node." Networks are generally thought to facilitate the transfer of information from source points to destinations. A node specifically tasked with furthering the passage of information from a source to a destination is commonly called a "router." There are many forms of networks such as Local Area Networks (LANs), Pico networks, Wide Area Networks (WANs), Wireless Networks (WLANs), etc. For example, the Internet is generally accepted as being an interconnection of a

multitude of networks whereby remote clients and servers may access and interoperate with one another.

[00143] The MCP PLATFORM controller 1001 may be based on computer systems that may comprise, but are not limited to, components such as: a computer systemization 1002 connected to memory 1029.

Computer Systemization

[00144] A computer systemization 1002 may comprise a clock 1030, central processing unit ("CPU(s)" and/or "processor(s)" (these terms are used interchangeable throughout the disclosure unless noted to the contrary)) 1003, a memory 1029 (e.g., a read only memory (ROM) 1006, a random access memory (RAM) 1005, etc.), and/or an interface bus 1007, and most frequently, although not necessarily, are all interconnected and/or communicating through a system bus 1004 on one or more (mother)board(s) 1002 having conductive and/or otherwise transportive circuit pathways through which instructions (e.g., binary encoded signals) may travel to effect communications, operations, storage, etc. Optionally, the computer systemization may be connected to an internal power source 1086. Optionally, a cryptographic processor 1026 may be connected to the system bus. The system clock typically has a crystal oscillator and generates a base signal through the computer systemization's circuit pathways. The clock is typically coupled to the system bus and various clock multipliers that will increase or decrease the base operating frequency for other components interconnected in the computer systemization. The clock and various components in a computer systemization drive signals embodying information throughout the system. Such transmission and reception of instructions embodying information throughout a

computer systemization may be commonly referred to as communications. These communicative instructions may further be transmitted, received, and the cause of return and/or reply communications beyond the instant computer systemization to: communications networks, input devices, other computer systemizations, peripheral devices, and/or the like. Of course, any of the above components may be connected directly to one another, connected to the CPU, and/or organized in numerous variations employed as exemplified by various computer systems.

[00145] The CPU comprises at least one high-speed data processor adequate to execute program components for executing user and/or system-generated requests. Often, the processors themselves will incorporate various specialized processing units, such as, but not limited to: integrated system (bus) controllers, memory management control units, floating point units, and even specialized processing sub-units like graphics processing units, digital signal processing units, and/or the like. Additionally, processors may include internal fast access addressable memory, and be capable of mapping and addressing memory 529 beyond the processor itself; internal memory may include, but is not limited to: fast registers, various levels of cache memory (e.g., level 1, 2, 3, etc.), RAM, etc. The processor may access this memory through the use of a memory address space that is accessible via instruction address, which the processor can construct and decode allowing it to access a circuit path to a specific memory address space having a memory state. The CPU may be a microprocessor such as: AMD's Athlon, Duron and/or Opteron; ARM's application, embedded and secure processors; IBM and/or Motorola's DragonBall and PowerPC; IBM's and Sony's Cell processor; Intel's Celeron, Core (2) Duo, Itanium, Pentium, Xeon, and/or XScale; and/or the like processor(s). The CPU interacts with memory through instruction

passing through conductive and/or transportive conduits (e.g., (printed) electronic and/or optic circuits) to execute stored instructions (i.e., program code) according to conventional data processing techniques. Such instruction passing facilitates communication within the MCP PLATFORM controller and beyond through various interfaces. Should processing requirements dictate a greater amount speed and/or capacity, distributed processors (e.g., Distributed MCP PLATFORM), mainframe, multi-core, parallel, and/or super-computer architectures may similarly be employed.Alternatively, should deployment requirements dictate greater portability, smaller Personal Digital Assistants (PDAs) may be employed.

[00146] Depending on the particular implementation, features of the MCP PLATFORM may be achieved by implementing a microcontroller such as CAST's R8051XC2 microcontroller; Intel's MCS 51 (i.e., 8051 microcontroller); and/or the like. Also, to implement certain features of the MCP PLATFORM, some feature implementations may rely on embedded components, such as: Application-Specific Integrated Circuit ("ASIC"), Digital Signal Processing ("DSP"), Field Programmable Gate Array ("FPGA"), and/or the like embedded technology. For example, any of the MCP PLATFORM component collection (distributed or otherwise) and/or features may be implemented via the microprocessor and/or via embedded components; e.g., via ASIC, coprocessor, DSP, FPGA, and/or the like. Alternately, some implementations of the MCP PLATFORM may be implemented with embedded components that are configured and used to achieve a variety of features or signal processing.

[00147] Depending on the particular implementation, the embedded components may include software solutions, hardware solutions, and/or some combination of both

hardware/software solutions. For example, MCP PLATFORM features discussed herein may be achieved through implementing FPGAs, which are a semiconductor devices containing programmable logic components called "logic blocks", and programmable interconnects, such as the high performance FPGA Virtex series and/or the low cost Spartan series manufactured by Xilinx. Logic blocks and interconnects can be programmed by the customer or designer, after the FPGA is manufactured, to implement any of the MCP PLATFORM features. A hierarchy of programmable interconnects allow logic blocks to be interconnected as needed by the MCP PLATFORM system designer/administrator, somewhat like a one-chip programmable breadboard. An FPGA's logic blocks can be programmed to perform the function of basic logic gates such as AND, and XOR, or more complex combinational functions such as decoders or simple mathematical functions. In most FPGAs, the logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory. In some circumstances, the MCP PLATFORM may be developed on regular FPGAs and then migrated into a fixed version that more resembles ASIC implementations. Alternate or coordinating implementations may migrate MCP PLATFORM controller features to a final ASIC instead of or in addition to FPGAs. Depending on the implementation all of the aforementioned embedded components and microprocessors may be considered the "CPU" and/or "processor" for the MCP PLATFORM.

Power Source

[00148] The power source 1086 may be of any standard form for powering small electronic circuit board devices such as the following power cells: alkaline, lithium hydride, lithium ion, lithium polymer, nickel cadmium, solar cells, and/or the like.

Other types of AC or DC power sources may be used as well. In the case of solar cells, in one embodiment, the case provides an aperture through which the solar cell may capture photonic energy. The power cell 1086 is connected to at least one of the interconnected subsequent components of the MCP PLATFORM thereby providing an electric current to all subsequent components. In one example, the power source 1086 is connected to the system bus component 1004. In an alternative embodiment, an outside power source 1086 is provided through a connection across the I/O 1008 interface. For example, a USB and/or IEEE 1394 connection carries both data and power across the connection and is therefore a suitable source of power.

Interface Adapters

[00149] Interface bus(ses) 1007 may accept, connect, and/or communicate to a number of interface adapters, conventionally although not necessarily in the form of adapter cards, such as but not limited to: input output interfaces (I/O) 1008, storage interfaces 1009, network interfaces 1010, and/or the like. Optionally, cryptographic processor interfaces 1027 similarly may be connected to the interface bus. The interface bus provides for the communications of interface adapters with one another as well as with other components of the computer systemization. Interface adapters are adapted for a compatible interface bus. Interface adapters conventionally connect to the interface bus via a slot architecture. Conventional slot architectures may be employed, such as, but not limited to: Accelerated Graphics Port (AGP), Card Bus, (Extended) Industry Standard Architecture ((E)ISA), Micro Channel Architecture (MCA), NuBus, Peripheral Component Interconnect (Extended) (PCI(X)), PCI Express, Personal Computer Memory Card International Association (PCMCIA), and/or the like.

[00150] Storage interfaces 1009 may accept, communicate, and/or connect to a number of storage devices such as, but not limited to: storage devices 1014, removable disc devices, and/or the like. Storage interfaces may employ connection protocols such as, but not limited to: (Ultra) (Serial) Advanced Technology Attachment (Packet Interface) ((Ultra) (Serial) ATA(PI)), (Enhanced) Integrated Drive Electronics ((E)IDE), Institute of Electrical and Electronics Engineers (IEEE) 1394, fiber channel, Small Computer Systems Interface (SCSI), Universal Serial Bus (USB), and/or the like.

[00151] Network interfaces 1010 may accept, communicate, and/or connect to a communications network 1013. Through a communications network 1013, the MCP PLATFORM controller is accessible through remote clients 1033b (e.g., computers with web browsers) by users 1033a. Network interfaces may employ connection protocols such as, but not limited to: direct connect, Ethernet (thick, thin, twisted pair 10/100/1000 Base T, and/or the like), Token Ring, wireless connection such as IEEE 802.11a-x, and/or the like. Should processing requirements dictate a greater amount speed and/or capacity, distributed network controllers (e.g., Distributed MCP PLATFORM), architectures may similarly be employed to pool, load balance, and/or otherwise increase the communicative bandwidth required by the MCP PLATFORM controller. A communications network may be any one and/or the combination of the following: a direct interconnection; the Internet; a Local Area Network (LAN); a Metropolitan Area Network (MAN); an Operating Missions as Nodes on the Internet (OMNI); a secured custom connection; a Wide Area Network (WAN); a wireless network (e.g., employing protocols such as, but not limited to a Wireless Application Protocol (WAP), I-mode, and/or the like); and/or the like. A network interface may be regarded as a specialized form of an input output interface. Further, multiple network

interfaces 1010 may be used to engage with various communications network types 1013. For example, multiple network interfaces may be employed to allow for the communication over broadcast, multicast, and/or unicast networks.

[00152] Input Output interfaces (I/O) 1008 may accept, communicate, and/or connect to user input devices 1011, peripheral devices 1012, cryptographic processor devices 1028, and/or the like. I/O may employ connection protocols such as, but not limited to: audio: analog, digital, monaural, RCA, stereo, and/or the like; data: Apple Desktop Bus (ADB), IEEE 1394a-b, serial, universal serial bus (USB); infrared; joystick; keyboard; midi; optical; PC AT; PS/2; parallel; radio; video interface: Apple Desktop Connector (ADC), BNC, coaxial, component, composite, digital, Digital Visual Interface (DVI), high-definition multimedia interface (HDMI), RCA, RF antennae, S-Video, VGA, and/or the like; wireless: 802.11a/b/g/n/x, Bluetooth, code division multiple access (CDMA), global system for mobile communications (GSM), WiMax, etc.; and/or the like. One typical output device may include a video display, which typically comprises a Cathode Ray Tube (CRT) or Liquid Crystal Display (LCD) based monitor with an interface (e.g., DVI circuitry and cable) that accepts signals from a video interface, may be used. The video interface composites information generated by a computer systemization and generates video signals based on the composited information in a video memory frame. Another output device is a television set, which accepts signals from a video interface. Typically, the video interface provides the composited video information through a video connection interface that accepts a video display interface (e.g., an RCA composite video connector accepting an RCA composite video cable; a DVI connector accepting a DVI display cable, etc.).

[00153] User input devices 1011 may be card readers, dongles, finger print readers, gloves, graphics tablets, joysticks, keyboards, mouse (mice), remote controls, retina readers, trackballs, trackpads, and/or the like.

[00154] Peripheral devices 1012 may be connected and/or communicate to I/O and/or other facilities of the like such as network interfaces, storage interfaces, and/or the like. Peripheral devices may be audio devices, cameras, dongles (e.g., for copy protection, ensuring secure transactions with a digital signature, and/or the like), external processors (for added functionality), goggles, microphones, monitors, network interfaces, printers, scanners, storage devices, video devices, video sources, visors, and/or the like.

[00155] It should be noted that although user input devices and peripheral devices may be employed, the MCP PLATFORM controller may be embodied as an embedded, dedicated, and/or monitor-less (i.e., headless) device, wherein access would be provided over a network interface connection.

[00156] Cryptographic units such as, but not limited to, microcontrollers, processors 1026, interfaces 1027, and/or devices 1028 may be attached, and/or communicate with the MCP PLATFORM controller. A MC68HC16 microcontroller, manufactured by Motorola Inc., may be used for and/or within cryptographic units. The MC68HC16 microcontroller utilizes a 16-bit multiply-and-accumulate instruction in the 16 MHz configuration and requires less than one second to perform a 512-bit RSA private key operation. Cryptographic units support the authentication of communications from interacting agents, as well as allowing for anonymous transactions. Cryptographic units may also be configured as part of CPU. Equivalent

microcontrollers and/or processors may also be used. Other commercially available specialized cryptographic processors include: the Broadcom's CryptoNetX and other Security Processors; nCipher's nShield, SafeNet's Luna PCI (e.g., 7100) series; Semaphore Communications' 40 MHz Roadrunner 184; Sun's Cryptographic Accelerators (e.g., Accelerator 6000 PCIe Board, Accelerator 500 Daughtercard); Via Nano Processor (e.g., L2100, L2200, U2400) line, which is capable of performing 500+ MB/s of cryptographic instructions; VLSI Technology's 33 MHz 6868; and/or the like.

Memory

[00157] Generally, any mechanization and/or embodiment allowing a processor to affect the storage and/or retrieval of information is regarded as memory 1029. However, memory is a fungible technology and resource, thus, any number of memory embodiments may be employed in lieu of or in concert with one another. It is to be understood that the MCP PLATFORM controller and/or a computer systemization may employ various forms of memory 1029. For example, a computer systemization may be configured wherein the functionality of on-chip CPU memory (e.g., registers), RAM, ROM, and any other storage devices are provided by a paper punch tape or paper punch card mechanism; of course such an embodiment would result in an extremely slow rate of operation. In a typical configuration, memory 1029 will include ROM 1006, RAM 1005, and a storage device 1014. A storage device 1014 may be any conventional computer system storage. Storage devices may include a drum; a (fixed and/or removable) magnetic disk drive; a magneto-optical drive; an optical drive (i.e., Blueray, CD ROM/RAM/Recordable (R)/ReWritable (RW), DVD R/RW, HD DVD R/RW etc.); an array of devices (e.g., Redundant Array of Independent Disks (RAID)); solid state

memory devices (USB memory, solid state drives (SSD), etc.); other processor-readable storage mediums; and/or other devices of the like. Thus, a computer systemization generally requires and makes use of memory.

Component Collection

[00158] The memory 1029 may contain a collection of program and/or database components and/or data such as, but not limited to: operating system component(s) 1015 (operating system); information server component(s) 1016 (information server); user interface component(s) 1017 (user interface); Web browser component(s) 1018 (Web browser); database(s) 1019; mail server component(s) 1021; mail client component(s) 1022; cryptographic server component(s) 1020 (cryptographic server); the MCP PLATFORM component(s) 1035; and/or the like (i.e., collectively a component collection). These components may be stored and accessed from the storage devices and/or from storage devices accessible through an interface bus. Although non-conventional program components such as those in the component collection, typically, are stored in a local storage device 1014, they may also be loaded and/or stored in memory such as: peripheral devices, RAM, remote storage facilities through a communications network, ROM, various forms of memory, and/or the like.

Operating System

[00159] The operating system component 1015 is an executable program component facilitating the operation of the MCP PLATFORM controller. Typically, the operating system facilitates access of I/O, network interfaces, peripheral devices, storage devices, and/or the like. The operating system may be a highly fault tolerant, scalable, and secure system such as: Apple Macintosh OS X (Server); AT&T Plan 9; Be

OS; Unix and Unix-like system distributions (such as AT&T's UNIX; Berkley Software Distribution (BSD) variations such as FreeBSD, NetBSD, OpenBSD, and/or the like; Linux distributions such as Red Hat, Ubuntu, and/or the like); and/or the like operating systems. However, more limited and/or less secure operating systems also may be employed such as Apple Macintosh OS, IBM OS/2, Microsoft DOS, Microsoft Windows 2000/2003/3.1/95/98/CE/Millenium/NT/Vista/XP (Server), Palm OS, and/or the like. An operating system may communicate to and/or with other components in a component collection, including itself, and/or the like. Most frequently, the operating system communicates with other program components, user interfaces, and/or the like. For example, the operating system may contain, communicate, generate, obtain, and/or provide program component, system, user, and/or data communications, requests, and/or responses. The operating system, once executed by the CPU, may enable the interaction with communications networks, data, I/O, peripheral devices, program components, memory, user input devices, and/or the like. The operating system may provide communications protocols that allow the MCP PLATFORM controller to communicate with other entities through a communications network 1013. Various communication protocols may be used by the MCP PLATFORM controller as a subcarrier transport mechanism for interaction, such as, but not limited to: multicast, TCP/IP, UDP, unicast, and/or the like.

Information Server

[00160] An information server component 1016 is a stored program component that is executed by a CPU. The information server may be a conventional Internet information server such as, but not limited to Apache Software Foundation's Apache,

Microsoft's Internet Information Server, and/or the like. The information server may allow for the execution of program components through facilities such as Active Server Page (ASP), ActiveX, (ANSI) (Objective-) C (++), C# and/or .NET, Common Gateway Interface (CGI) scripts, dynamic (D) hypertext markup language (HTML), FLASH, Java, JavaScript, Practical Extraction Report Language (PERL), Hypertext Pre-Processor (PHP), pipes, Python, wireless application protocol (WAP), WebObjects, and/or the like. The information server may support secure communications protocols such as, but not limited to, File Transfer Protocol (FTP); HyperText Transfer Protocol (HTTP); Secure Hypertext Transfer Protocol (HTTPS), Secure Socket Layer (SSL), messaging protocols (e.g., America Online (AOL) Instant Messenger (AIM), Application Exchange (APEX), ICQ, Internet Relay Chat (IRC), Microsoft Network (MSN) Messenger Service, Presence and Instant Messaging Protocol (PRIM), Internet Engineering Task Force's (IETF's) Session Initiation Protocol (SIP), SIP for Instant Messaging and Presence Leveraging Extensions (SIMPLE), open XML-based Extensible Messaging and Presence Protocol (XMPP) (i.e., Jabber or Open Mobile Alliance's (OMA's) Instant Messaging and Presence Service (IMPS)), Yahoo! Instant Messenger Service, and/or the like. The information server provides results in the form of Web pages to Web browsers, and allows for the manipulated generation of the Web pages through interaction with other program components. After a Domain Name System (DNS) resolution portion of an HTTP request is resolved to a particular information server, the information server resolves requests for information at specified locations on the MCP PLATFORM controller based on the remainder of the HTTP request. For example, a request such as http://123.124.125.126/myInformation.html might have the IP portion of the request "123.124.125.126" resolved by a DNS server to an information server at that IP address;

that information server might in turn further parse the http request for the "/myInformation.html" portion of the request and resolve it to a location in memory containing the information "myInformation.html." Additionally, other information serving protocols may be employed across various ports, e.g., FTP communications across port 21, and/or the like. An information server may communicate to and/or with other components in a component collection, including itself, and/or facilities of the like. Most frequently, the information server communicates with the MCP PLATFORM database 1019, operating systems, other program components, user interfaces, Web browsers, and/or the like.

[00161] Access to the MCP PLATFORM database may be achieved through a number of database bridge mechanisms such as through scripting languages as enumerated below (e.g., CGI) and through inter-application communication channels as enumerated below (e.g., CORBA, WebObjects, etc.). Any data requests through a Web browser are parsed through the bridge mechanism into appropriate grammars as required by the MCP PLATFORM. In one embodiment, the information server would provide a Web form accessible by a Web browser. Entries made into supplied fields in the Web form are tagged as having been entered into the particular fields, and parsed as such. The entered terms are then passed along with the field tags, which act to instruct the parser to generate queries directed to appropriate tables and/or fields. In one embodiment, the parser may generate queries in standard SQL by instantiating a search string with the proper join/select commands based on the tagged text entries, wherein the resulting command is provided over the bridge mechanism to the MCP PLATFORM as a query. Upon generating query results from the query, the results are passed over the bridge mechanism, and may be parsed for formatting and generation of a new results

Web page by the bridge mechanism. Such a new results Web page is then provided to the information server, which may supply it to the requesting Web browser.

[00162] Also, an information server may contain, communicate, generate, obtain, and/or provide program component, system, user, and/or data communications, requests, and/or responses.

User Interface

[00163] The function of computer interfaces in some respects is similar to automobile operation interfaces. Automobile operation interface elements such as steering wheels, gearshifts, and speedometers facilitate the access, operation, and display of automobile resources, functionality, and status. Computer interaction interface elements such as check boxes, cursors, menus, scrollers, and windows (collectively and commonly referred to as widgets) similarly facilitate the access, operation, and display of data and computer hardware and operating system resources, functionality, and status. Operation interfaces are commonly called user interfaces. Graphical user interfaces (GUIs) such as the Apple Macintosh Operating System's Aqua, IBM's OS/2, Microsoft's Windows 2000/2003/3.1/95/98/CE/Millenium/NT/XP/Vista/7 (i.e., Aero), Unix's X-Windows (e.g., which may include additional Unix graphic interface libraries and layers such as K Desktop Environment (KDE), mythTV and GNU Network Object Model Environment (GNOME)), web interface libraries (e.g., ActiveX, AJAX, (D)HTML, FLASH, Java, JavaScript, etc. interface libraries such as, but not limited to, Dojo, jQuery(UI), MooTools, Prototype, script.aculo.us, SWFObject, Yahoo! User Interface, any of which

may be used and) provide a baseline and means of accessing and displaying information graphically to users.

[00164] A user interface component 1017 is a stored program component that is executed by a CPU. The user interface may be a conventional graphic user interface as provided by, with, and/or atop operating systems and/or operating environments such as already discussed. The user interface may allow for the display, execution, interaction, manipulation, and/or operation of program components and/or system facilities through textual and/or graphical facilities. The user interface provides a facility through which users may affect, interact, and/or operate a computer system. A user interface may communicate to and/or with other components in a component collection, including itself, and/or facilities of the like. Most frequently, the user interface communicates with operating systems, other program components, and/or the like. The user interface may contain, communicate, generate, obtain, and/or provide program component, system, user, and/or data communications, requests, and/or responses.

Web Browser

[00165] A Web browser component 1018 is a stored program component that is executed by a CPU. The Web browser may be a conventional hypertext viewing application such as Microsoft Internet Explorer or Netscape Navigator. Secure Web browsing may be supplied with 128bit (or greater) encryption by way of HTTPS, SSL, and/or the like. Web browsers allowing for the execution of program components through facilities such as ActiveX, AJAX, (D)HTML, FLASH, Java, JavaScript, web browser plug-in APIs (e.g., FireFox, Safari Plug-in, and/or the like APIs), and/or the

like. Web browsers and like information access tools may be integrated into PDAs, cellular telephones, and/or other mobile devices. A Web browser may communicate to and/or with other components in a component collection, including itself, and/or facilities of the like. Most frequently, the Web browser communicates with information servers, operating systems, integrated program components (e.g., plug-ins), and/or the like; e.g., it may contain, communicate, generate, obtain, and/or provide program component, system, user, and/or data communications, requests, and/or responses. Of course, in place of a Web browser and information server, a combined application may be developed to perform similar functions of both. The combined application would similarly affect the obtaining and the provision of information to users, user agents, and/or the like from the MCP PLATFORM enabled nodes. The combined application may be nugatory on systems employing standard Web browsers.

Mail Server

[00166] A mail server component 1021 is a stored program component that is executed by a CPU 1003. The mail server may be a conventional Internet mail server such as, but not limited to sendmail, Microsoft Exchange, and/or the like. The mail server may allow for the execution of program components through facilities such as ASP, ActiveX, (ANSI) (Objective-) C (++), C# and/or .NET, CGI scripts, Java, JavaScript, PERL, PHP, pipes, Python, WebObjects, and/or the like. The mail server may support communications protocols such as, but not limited to: Internet message access protocol (IMAP), Messaging Application Programming Interface (MAPI)/Microsoft Exchange, post office protocol (POP3), simple mail transfer protocol (SMTP), and/or the like. The mail server can route, forward, and process incoming and

outgoing mail messages that have been sent, relayed and/or otherwise traversing through and/or to the MCP PLATFORM.

[00167] Access to the MCP PLATFORM mail may be achieved through a number of APIs offered by the individual Web server components and/or the operating system.

[00168] Also, a mail server may contain, communicate, generate, obtain, and/or provide program component, system, user, and/or data communications, requests, information, and/or responses.

Mail Client

[00169] A mail client component 1022 is a stored program component that is executed by a CPU 1003. The mail client may be a conventional mail viewing application such as Apple Mail, Microsoft Entourage, Microsoft Outlook, Microsoft Outlook Express, Mozilla, Thunderbird, and/or the like. Mail clients may support a number of transfer protocols, such as: IMAP, Microsoft Exchange, POP3, SMTP, and/or the like. A mail client may communicate to and/or with other components in a component collection, including itself, and/or facilities of the like. Most frequently, the mail client communicates with mail servers, operating systems, other mail clients, and/or the like; e.g., it may contain, communicate, generate, obtain, and/or provide program component, system, user, and/or data communications, requests, information, and/or responses. Generally, the mail client provides a facility to compose and transmit electronic mail messages.

Cryptographic Server

[00170] A cryptographic server component 1020 is a stored program component that is executed by a CPU 1003, cryptographic processor 1026, cryptographic processor interface 1027, cryptographic processor device 1028, and/or the like. Cryptographic processor interfaces will allow for expedition of encryption and/or decryption requests by the cryptographic component; however, the cryptographic component, alternatively, may run on a conventional CPU. The cryptographic component allows for the encryption and/or decryption of provided data. The cryptographic component allows for both symmetric and asymmetric (e.g., Pretty Good Protection (PGP)) encryption and/or decryption. The cryptographic component may employ cryptographic techniques such as, but not limited to: digital certificates (e.g., X.509 authentication framework), digital signatures, dual signatures, enveloping, password access protection, public key management, and/or the like. The cryptographic component will facilitate numerous (encryption and/or decryption) security protocols such as, but not limited to: checksum, Data Encryption Standard (DES), Elliptical Curve Encryption (ECC), International Data Encryption Algorithm (IDEA), Message Digest 5 (MD5, which is a one way hash function), passwords, Rivest Cipher (RC5), Rijndael, RSA (which is an Internet encryption and authentication system that uses an algorithm developed in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman), Secure Hash Algorithm (SHA), Secure Socket Layer (SSL), Secure Hypertext Transfer Protocol (HTTPS), and/or the like. Employing such encryption security protocols, the MCP PLATFORM may encrypt all incoming and/or outgoing communications and may serve as node within a virtual private network (VPN) with a wider communications network. The cryptographic component facilitates the process of "security authorization" whereby access to a

resource is inhibited by a security protocol wherein the cryptographic component effects authorized access to the secured resource. In addition, the cryptographic component may provide unique identifiers of content, e.g., employing and MD5 hash to obtain a unique signature for an digital audio file. A cryptographic component may communicate to and/or with other components in a component collection, including itself, and/or facilities of the like. The cryptographic component supports encryption schemes allowing for the secure transmission of information across a communications network to enable the MCP PLATFORM component to engage in secure transactions if so desired. The cryptographic component facilitates the secure accessing of resources on the MCP PLATFORM and facilitates the access of secured resources on remote systems; i.e., it may act as a client and/or server of secured resources. Most frequently, the cryptographic component communicates with information servers, operating systems, other program components, and/or the like. The cryptographic component may contain, communicate, generate, obtain, and/or provide program component, system, user, and/or data communications, requests, and/or responses.

The MCP PLATFORM Database

[00171] The MCP PLATFORM database component 1019 may be embodied in a database and its stored data. The database is a stored program component, which is executed by the CPU; the stored program component portion configuring the CPU to process the stored data. The database may be a conventional, fault tolerant, relational, scalable, secure database such as Oracle or Sybase. Relational databases are an extension of a flat file. Relational databases consist of a series of related tables. The tables are interconnected via a key field. Use of the key field allows the combination of

the tables by indexing against the key field; i.e., the key fields act as dimensional pivot points for combining information from various tables. Relationships generally identify links maintained between tables by matching primary keys. Primary keys represent fields that uniquely identify the rows of a table in a relational database. More precisely, they uniquely identify rows of a table on the "one" side of a one-to-many relationship.

[00172] Alternatively, the MCP PLATFORM database may be implemented using various standard data-structures, such as an array, hash, (linked) list, struct, structured text file (e.g., XML), table, and/or the like. Such data-structures may be stored in memory and/or in (structured) files. In another alternative, an object-oriented database may be used, such as Frontier, ObjectStore, Poet, Zope, and/or the like. Object databases can include a number of object collections that are grouped and/or linked together by common attributes; they may be related to other object collections by some common attributes. Object-oriented databases perform similarly to relational databases with the exception that objects are not just pieces of data but may have other types of functionality encapsulated within a given object. If the MCP PLATFORM database is implemented as a data-structure, the use of the MCP PLATFORM database 1019 may be integrated into another component such as the MCP PLATFORM component 1035. Also, the database may be implemented as a mix of data structures, objects, and relational structures. Databases may be consolidated and/or distributed in countless variations through standard data processing techniques. Portions of databases, e.g., tables, may be exported and/or imported and thus decentralized and/or integrated.

[00173] In one embodiment, the database component **1019** includes several tables **1019**a-e. A Users table **1019**a may include fields such as, but not limited to: model_ID,

model_name, parameter_ID(s), instrument_IDs, model_type, model, and/or the like. A Parameters table 1019b may include fields such as, but not limited to: parameter_ID, parameter_name, parameter_value(s), value_source(s), parameter_history, parameter_type, and/or the like. A Users table 1019c may include fields such as, but not limited to: user_ID, user_name, contact_info, login, password, instrument_ID, portfolio, trading_history, risk_aversion, payment_identifier(s), and/or the like. An Instruments table 1019d may include fields such as, but not limited to: instrument_ID, instrument_type, parameter_ID(s), current_market_price, historical_market_price, volume, trade_history, underlying_entity, exchange, trading_restrictions, volatility, and/or the like. A market data table 1019e may include fields such as, but not limited to: market_data_feed_ID, asset_ID, asset_symbol, asset_name, spot_price, bid_price, ask_price, and/or the like; in one embodiment, the market data table is populated through a market data feed (e.g., Bloomberg's PhatPipe, Dun & Bradstreet, Reuter's Tib, Triarch, etc.), for example, through Microsoft's Active Template Library and Dealing Object Technology's real-time toolkit Rtt.Multi.

[00174] In one embodiment, the MCP PLATFORM database may interact with other database systems. For example, employing a distributed database system, queries and data access by search MCP PLATFORM component may treat the combination of the MCP PLATFORM database, an integrated data security layer database as a single database entity.

[00175] In one embodiment, user programs may contain various user interface primitives, which may serve to update the MCP PLATFORM. Also, various accounts may require custom database tables depending upon the environments and the types of

clients the MCP PLATFORM may need to serve. It should be noted that any unique fields may be designated as a key field throughout. In an alternative embodiment, these tables have been decentralized into their own databases and their respective database controllers (i.e., individual database controllers for each of the above tables). Employing standard data processing techniques, one may further distribute the databases over several computer systemizations and/or storage devices. Similarly, configurations of the decentralized database controllers may be varied by consolidating and/or distributing the various database components 1019a-e. The MCP PLATFORM may be configured to keep track of various settings, inputs, and parameters via database controllers.

[00176] The MCP PLATFORM database may communicate to and/or with other components in a component collection, including itself, and/or facilities of the like. Most frequently, the MCP PLATFORM database communicates with the MCP PLATFORM component, other program components, and/or the like. The database may contain, retain, and provide information regarding other nodes and data.

The MCP PLATFORMs

[00177] The MCP PLATFORM component 1035 is a stored program component that is executed by a CPU. In one embodiment, the MCP PLATFORM component incorporates any and/or all combinations of the aspects of the MCP PLATFORM that was discussed in the previous figures. As such, the MCP PLATFORM affects accessing, obtaining and the provision of information, services, transactions, and/or the like across various communications networks.

[00178] The MCP PLATFORM component enables one to access, calculate, engage, exchange, generate, identify, instruct, match, process, search, serve, store, and/or

facilitate trading strategies, modeling of market factors and/or parameters associated with tradable financial instruments, optimization of said models, and/or the like and use of the MCP PLATFORM.

[00179] The MCP PLATFORM component enabling access of information between nodes may be developed by employing standard development tools and languages such as, but not limited to: Apache components, Assembly, ActiveX, binary executables, (ANSI) (Objective-) C (++), C# and/or .NET, database adapters, CGI scripts, Java, JavaScript, mapping tools, procedural and object oriented development tools, PERL, PHP, Python, shell scripts, SQL commands, web application server extensions, web development environments and libraries (e.g., Microsoft's ActiveX; Adobe AIR, FLEX & FLASH; AJAX; (D)HTML; Dojo, Java; JavaScript; jQuery(UI); MooTools; Prototype; script.aculo.us; Simple Object Access Protocol (SOAP); SWFObject; Yahoo! User Interface; and/or the like), WebObjects, and/or the like. In one embodiment, the MCP PLATFORM server employs a cryptographic server to encrypt and decrypt communications. The MCP PLATFORM component may communicate to and/or with other components in a component collection, including itself, and/or facilities of the like. Most frequently, the MCP PLATFORM component communicates with the MCP PLATFORM database, operating systems, other program components, and/or the like. The MCP PLATFORM may contain, communicate, generate, obtain, and/or provide program component, system, user, and/or data communications, requests, and/or responses.

Distributed MCP PLATFORMs

[00180] The structure and/or operation of any of the MCP PLATFORM node controller components may be combined, consolidated, and/or distributed in any number of ways to facilitate development and/or deployment. Similarly, the component collection may be combined in any number of ways to facilitate deployment and/or development. To accomplish this, one may integrate the components into a common code base or in a facility that can dynamically load the components on demand in an integrated fashion.

[00181] The component collection may be consolidated and/or distributed in countless variations through standard data processing and/or development techniques. Multiple instances of any one of the program components in the program component collection may be instantiated on a single node, and/or across numerous nodes to improve performance through load-balancing and/or data-processing techniques. Furthermore, single instances may also be distributed across multiple controllers and/or storage devices; e.g., databases. All program component instances and controllers working in concert may do so through standard data processing communication techniques.

[00182] The configuration of the MCP PLATFORM controller will depend on the context of system deployment. Factors such as, but not limited to, the budget, capacity, location, and/or use of the underlying hardware resources may affect deployment requirements and configuration. Regardless of if the configuration results in more consolidated and/or integrated program components, results in a more distributed series of program components, and/or results in some combination between a

consolidated and distributed configuration, data may be communicated, obtained, and/or provided. Instances of components consolidated into a common code base from the program component collection may communicate, obtain, and/or provide data. This may be accomplished through intra-application data processing communication techniques such as, but not limited to: data referencing (e.g., pointers), internal messaging, object instance variable communication, shared memory space, variable passing, and/or the like.

[00183] If component collection components are discrete, separate, and/or external to one another, then communicating, obtaining, and/or providing data with and/or to other component components may be accomplished through inter-application data processing communication techniques such as, but not limited to: Application Program Interfaces (API) information passage; (distributed) Component Object Model ((D)COM), (Distributed) Object Linking and Embedding ((D)OLE), and/or the like), Common Object Request Broker Architecture (CORBA), local and remote application program interfaces Jini, Remote Method Invocation (RMI), SOAP, process pipes, shared files, and/or the like. Messages sent between discrete component components for inter-application communication or within memory spaces of a singular component for intra-application communication may be facilitated through the creation and parsing of a grammar. A grammar may be developed by using standard development tools such as lex, yacc, XML, and/or the like, which allow for grammar generation and parsing functionality, which in turn may form the basis of communication messages within and between components. For example, a grammar may be arranged to recognize the tokens of an HTTP post command, e.g.:

```
w3c -post http://... Value1
```

[00184] where Value1 is discerned as being a parameter because "http://" is part of the grammar syntax, and what follows is considered part of the post value. Similarly, with such a grammar, a variable "Value1" may be inserted into an "http://" post command and then sent. The grammar syntax itself may be presented as structured data that is interpreted and/or other wise used to generate the parsing mechanism (e.g., a syntax description text file as processed by lex, yacc, etc.). Also, once the parsing mechanism is generated and/or instantiated, it itself may process and/or parse structured data such as, but not limited to: character (e.g., tab) delineated text, HTML, structured text streams, XML, and/or the like structured data. In another embodiment, inter-application data processing protocols themselves may have integrated and/or readily available parsers (e.g., the SOAP parser) that may be employed to parse communications data. Further, the parsing grammar may be used beyond message parsing, but may also be used to parse: databases, data collections, data stores, structured data, and/or the like. Again, the desired configuration will depend upon the context, environment, and requirements of system deployment.

[00185]

[00186]

[00187] The entirety of this application (including the Cover Page, Title, Headings, Field, Background, Summary, Brief Description of the Drawings, Detailed Description, Claims, Abstract, Figures, and otherwise) shows by way of illustration various embodiments in which the claimed inventions may be practiced. The advantages and features of the application are of a representative sample of embodiments only, and are

not exhaustive and/or exclusive. They are presented only to assist in understanding and teach the claimed principles. It should be understood that they are not representative of all claimed inventions. As such, certain aspects of the disclosure have not been discussed herein. That alternate embodiments may not have been presented for a specific portion of the invention or that further undescribed alternate embodiments may be available for a portion is not to be considered a disclaimer of those alternate embodiments. It will be appreciated that many of those undescribed embodiments incorporate the same principles of the invention and others are equivalent. Thus, it is to be understood that other embodiments may be utilized and functional, logical, organizational, structural and/or topological modifications may be made without departing from the scope and/or spirit of the disclosure. As such, all examples and/or embodiments are deemed to be non-limiting throughout this disclosure. Also, no inference should be drawn regarding those embodiments discussed herein relative to those not discussed herein other than it is as such for purposes of reducing space and repetition. For instance, it is to be understood that the logical and/or topological structure of any combination of any program components (a component collection), other components and/or any present feature sets as described in the figures and/or throughout are not limited to a fixed operating order and/or arrangement, but rather, any disclosed order is exemplary and all equivalents, regardless of order, are contemplated by the disclosure. Furthermore, it is to be understood that such features are not limited to serial execution, but rather, any number of threads, processes, services, servers, and/or the like that may execute asynchronously, concurrently, in parallel, simultaneously, synchronously, and/or the like are contemplated by the disclosure. As such, some of these features may be mutually contradictory, in that they cannot be simultaneously present in a single

embodiment. Similarly, some features are applicable to one aspect of the invention, and inapplicable to others. In addition, the disclosure includes other inventions not presently claimed. Applicant reserves all rights in those presently unclaimed inventions including the right to claim such inventions, file additional applications, continuations, continuations in part, divisions, and/or the like thereof. As such, it should be understood that advantages, embodiments, examples, functional, features, logical, organizational, structural, topological, and/or other aspects of the disclosure are not to be considered limitations on the disclosure as defined by the claims or limitations on equivalents to the claims.

CLAIMS

What is claimed is:

1. A processor-implemented method for trading a financial instrument, comprising:

receiving an order request that includes a total quantity of shares and a time horizon within which to execute a transaction;

retrieving order parameters, security parameters, and market parameters associated with the transaction;

evaluating analytically a future implementation cost as a function of future number of shares to trade after a current time interval based on at least some of the retrieved parameters;

evaluating numerically a current implementation cost as a function of current number of shares to trade during the current time interval based on at least some of the retrieved parameters;

optimizing via a processor a total implementation cost to determine an optimal current number of shares from the total quantity to trade during the current time interval, wherein the total implementation cost is a sum of the future implementation cost and the current implementation cost; and

executing an order for the optimal current number of shares of the financial instrument.

2. The method of claim 1, further comprising iteratively re-optimizing and re-executing until the total quantity of shares is traded.

3. The method of claim 1, wherein the evaluating analytically a future implementation cost is based on a stochastic process.

4. The method of claim 3, wherein the stochastic process comprises an arithmetic random walk with specified drift and volatility.

5. The method of claim 1, wherein the security parameters comprise a time-varying price and a time-varying volume for the tradable financial instrument.

6. The method of claim 1, wherein the evaluating analytically a future implementation cost further comprises calculating an implementation shortfall based on a benchmark cost.

7. The method of claim 6, wherein the benchmark cost is based on an arrive price.

8. The method of claim 6, wherein the benchmark cost is based on a volume weighted average price.

9. The method of claim 1, wherein the order is a market order.

10. A financial instrument trading apparatus, comprising:

a memory;

a processor disposed in communication with said memory, and configured to issue a plurality of processing instructions stored in the memory, wherein the processor issues instructions to:

receive an order request that includes a total quantity of shares and a time horizon within which to execute a transaction;

retrieve order parameters, security parameters, and market parameters associated with the transaction;

evaluate analytically a future implementation cost as a function of future number of shares to trade after a current time interval based on at least some of the retrieved parameters;

evaluate numerically a current implementation cost as a function of current number of shares to trade during the current time interval based on at least some of the retrieved parameters;

optimize via a processor a total implementation cost to determine an optimal current number of shares from the total quantity to trade during the current time interval, wherein the total implementation cost is a sum of the future implementation cost and the current implementation cost; and

execute an order for the optimal current number of shares of the financial instrument.

11. The apparatus of claim 10, further comprising instructions to iteratively re-optimize and re-execute until the total quantity of shares is traded.

12. The apparatus of claim 10, wherein the evaluating analytically a future implementation cost is based on a stochastic process.

13. The apparatus of claim 12, wherein the stochastic process comprises an arithmetic random walk with specified drift and volatility.

14. The apparatus of claim 10, wherein the security parameters comprise a time-varying price and a time-varying volume for the tradable financial instrument.

15. The apparatus of claim 10, wherein the evaluating analytically a future implementation cost further comprises calculating an implementation shortfall based on a benchmark cost.

16. The apparatus of claim 15, wherein the benchmark cost is based on an arrive price.

17. The apparatus of claim 15, wherein the benchmark cost is based on a volume weighted average price.

18. The apparatus of claim 10, wherein the order is a market order.