Master Thesis

A forward oriented reinforcement learning approach to the problem of optimized trade execution

Axel Perschmann

August 7, 2017

Albert-Ludwigs-University Freiburg im Breisgau
Faculty of Engineering
Department of Computer Science
Machine Learning Lab

Motivation

Motivation

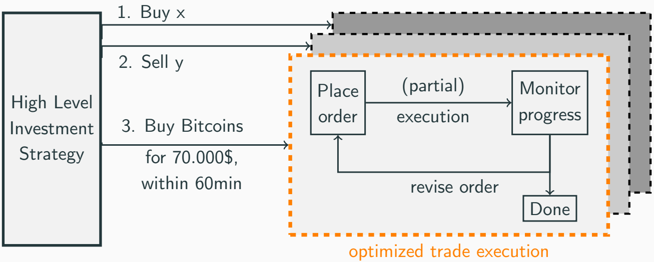
Motivation: (Large) profits at minimal risk

- *Profit* is made through beneficial trades
- *Risk* is reduced by diversification

High level trading strategies use *technical* and *fundamental analysis* to

- *estimate* whether individual assets are currently under- or overvalued
- *optimize* which assets shall be bought or sold at what share count

Investment decisions are then entrusted to a *trader*

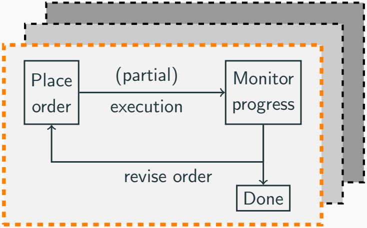


Optimized Trade Execution

Trading algorithm aims to **minimize inevitably occurring** costs:

- *Fees* charged by the market place organizer
- *Hidden costs* in form of worsening prices

In its simplest form the problem is defined by a particular financial instrument, which must be bought or sold within a fixed time horizon, while minimizing the expenditure for doing so.



Buy Bitcoins for 70.000$ within 60min

Objectives & Contributions

The following contributions to the problem of optimized trade execution have been made:

- A full-featured reinforcement learning environment for the simulation of trade execution is presented: The *Orderbook Trading Simulator*.
- An existing, *backward learning* reinforcement learning approach has been examined and was improved in detail.
- A novel, *forward oriented* reinforcement learning approach is presented. It applies growing batch learning, samples from a continuous state space and outperforms the foremost algorithm.

Outline of Contents

Trading Basics

Trading

Assets are typically traded through one of two basic market types:

Trading

Assets are typically traded through one of two basic market types:

Over-The-Counter (OTC) Markets

- Transactions take place directly between two parties.
- Prices are negotiated directly between buyers and sellers.

Trading

Assets are typically traded through one of two basic market types:

Over-The-Counter (OTC) Markets

- Transactions take place directly between two parties.
- Prices are negotiated directly between buyers and sellers.

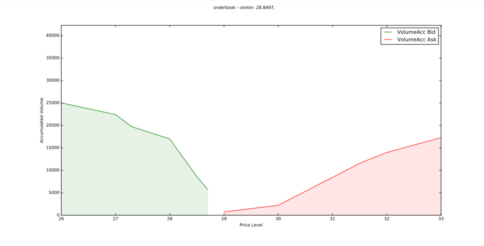
Exchange Markets

- Transactions are executed by a **broker**.
- Prices are determined by **Supply & Demand**.
  - Sellers can **ask** a price, they are aiming to make.
  - Buyers can **bid** a price, they are willing to pay.
  - *Asks* and *Bids* are maintained in a Limit Order Book.

Limit Order Book

The **Limit Order Book** (LOB) reflects supply (asks) and demand (bids).

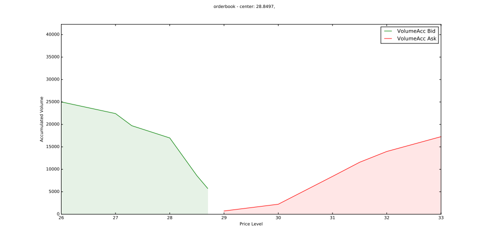
	Amount	Type	Volume	VolumeAcc	norm_Price
31.00	200.0	ask	6200.0	8425.0	1.074533
30.00	50.0	ask	1500.0	2225.0	1.039871
29.00	**25.0**	**ask**	725.0	725.0	1.005208
28.85	NaN	center	NaN	NaN	NaN
28.70	**200.0**	**bid**	5740.0	5740.0	0.994810
28.50	100.0	bid	2850.0	8590.0	0.987877
28.00	300.0	bid	8400.0	16990.0	0.970546



Limit Order Book

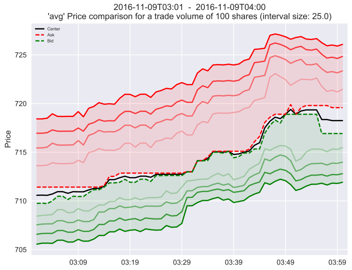
The **Limit Order Book** (LOB) reflects supply (asks) and demand (bids).

	Amount	Type	Volume	VolumeAcc	norm_Price
31.00	200.0	ask	6200.0	8425.0	1.074533
30.00	50.0	ask	1500.0	2225.0	1.039871
29.00	**25.0**	**ask**	725.0	725.0	1.005208
28.85	NaN	center	NaN	NaN	NaN
28.70	**200.0**	**bid**	5740.0	5740.0	0.994810
28.50	100.0	bid	2850.0	8590.0	0.987877
28.00	300.0	bid	8400.0	16990.0	0.970546

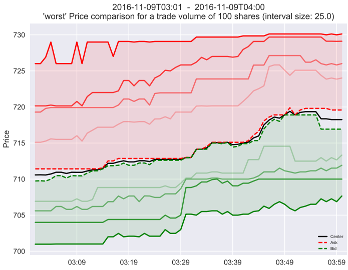


bid-ask spread: Difference between *best bid price* and *best ask price*

Limit Order Book: 60 Minute interval



(a) Average market price.



(b) Worst market price.

Prices and Orderbooks permanently change:

- Order fulfillments (Trade execution)
- Order cancelations
- Order arrivals
- Order updates ($+/-$ volume)

Trading costs

Investors must expect certain costs:

- **Fees** and **commissions** are contractually regulated.
  $\Rightarrow$ Fix costs, predictable to a large extend.
- **Slippage** is the deviation from the expected price.
  E.g. large orders can have a major impact on Supply & Demand.
  - Diminishing availability, worsening prices, ...

	Amount	Type	Volume	VolumeAcc	norm_Price
31.00	200.0	ask	6200.0	8425.0	1.074533
30.00	50.0	ask	1500.0	2225.0	1.039871
29.00	25.0	ask	725.0	725.0	1.005208
28.85	NaN	center	NaN	NaN	NaN
28.70	200.0	bid	5740.0	5740.0	0.994810
28.50	100.0	bid	2850.0	8590.0	0.987877
28.00	300.0	bid	8400.0	16990.0	0.970546

Hidden Costs: Slippage

Placing small orders, chances are high to get a good price. Larger orders *eat* into the book and are fulfilled at successively worse price levels.

	Amount	Type	Volume	VolumeAcc	norm_Price
31.00	200.0	ask	6200.0	8425.0	1.074533
30.00	50.0	ask	1500.0	2225.0	1.039871
29.00	25.0	ask	725.0	725.0	1.005208
28.85	NaN	center	NaN	NaN	NaN
28.70	200.0	bid	5740.0	5740.0	0.994810
28.50	100.0	bid	2850.0	8590.0	0.987877
28.00	300.0	bid	8400.0	16990.0	0.970546

20 shares:	150 shares:	500 shares
Buy 20 @ 29	Buy 25 @ 29$ Buy 50 @ 30$ Buy 75 @ 31$	Buy 25 @ 29$ Buy 50 @ 30$ Buy 200 @ 31$ Buy 100 @ 31.50$ Buy 75 @ 32$ Buy 50 @ 33$
580$	4,549.50$	15,625$
Avg: 29$ Slippage: 0.00$ Cost: 0.00$	Avg: 30.33$ Slippage 1.33 $ Cost 200 $	Avg: 31.25$ Slippage 2.25 $ Cost 1,125 $

Order types: Simple Market Order

Scenario: You want to buy 100 shares of AIWC.

	Amount	Type	Volume	VolumeAcc	norm_Price
31.00	200.0	ask	6200.0	8425.0	1.074533
30.00	50.0	ask	1500.0	2225.0	1.039871
29.00	25.0	ask	725.0	725.0	1.005208
28.85	NaN	center	NaN	NaN	NaN
28.70	200.0	bid	5740.0	5740.0	0.994810
28.50	100.0	bid	2850.0	8590.0	0.987877
28.00	300.0	bid	8400.0	16990.0	0.970546

Solution: Buy them right away for the current market price:

- from Alice: $\Rightarrow 25 * 29\$ = 725\$$
- from Bob and Cedar: $\Rightarrow (20 + 30) * 30\$ = 1500\$$
- from David: $\Rightarrow 25 * 31\$ = 775\$$

Total: 3000$ (avg price: 30$)

Fast, but costly (Average price differs from best price)
$\Rightarrow$ Slippage of 1$/share $\Rightarrow$ Loss of 100$ compared to best ask.

Order types: Limit Order

Scenario: You want to buy 100 shares of AIWC, at max 30$/share.

	Amount	Type	Volume	VolumeAcc	norm_Price
31.00	200.0	ask	6200.0	8425.0	1.074533
30.00	50.0	ask	1500.0	2225.0	1.039871
29.00	25.0	ask	725.0	725.0	1.005208
28.85	NaN	center	NaN	NaN	NaN
28.70	200.0	bid	5740.0	5740.0	0.994810
28.50	100.0	bid	2850.0	8590.0	0.987877
28.00	300.0	bid	8400.0	16990.0	0.970546

Solution: Place a limit order (i.e. a *bid*): BUY 100 @30$

75 shares are matched immediately.

Order types: Limit Order

Scenario: You want to buy 100 shares of AIWC, at max 30$/share.

	Amount	Type	Volume	VolumeAcc	norm_Price
32.00	75.0	ask	2400.0	11750.0	1.049274
31.50	100.0	ask	3150.0	9350.0	1.032879
31.00	200.0	ask	6200.0	6200.0	1.016485
30.50	NaN	center	NaN	NaN	NaN
30.00	25.0	bid	750.0	750.0	0.983695
28.70	200.0	bid	5740.0	6490.0	0.941068
28.50	100.0	bid	2850.0	9340.0	0.934510

Solution: Place a limit order (i.e. a *bid*): BUY 100 @30$

75 shares are matched immediately.

25 remaining orders are added to orderbook and wait for matching offers.

- from Alice: $\Rightarrow 25 * 29\$ = 725\$$
- from Bob and Cedar: $\Rightarrow (20 + 30) * 30\$ = 1500\$$

Total: 2225$ (avg price: 29.67$)

Reduced Slippage, but not guaranteed to execute fully

Order strategy: Submit and Revise

Combination of Market Order and Limit Order:

- Place limit order at t=[0, 15, 30, 45] and leave it for 15 minutes
- At t=59 replace Limit Order with Market Order
- Accepting more slippage towards end of period
  - Check trade progresse very 15 minutes
- Exploit Market/Orderbook Features
  - spread size
  - bid/ask imbalance (more people trying to sell or to buy?)
  - current market price

Order strategy: Submit and Revise

Combination of Market Order and Limit Order:

- Place limit order at t=[0, 15, 30, 45] and leave it for 15 minutes
- At t=59 replace Limit Order with Market Order
- Accepting more slippage towards end of period
    - Check trade progresse very 15 minutes
- Exploit Market/Orderbook Features
    - spread size
    - bid/ask imbalance (more people trying to sell or to buy?)
    - current market price

Idea: Use Reinforcement Learning to find optimal limits

Orderbook Trading Simulator

Historic Order Book Data

Most research is based on large data sets

- complete logs: order arrivals, cancelations, updates, matches
- years of 'stock exchange' data (resolution: milliseconds)

Historic Order Book Data

Most research is based on large data sets

- complete logs: order arrivals, cancelations, updates, matches
- years of 'stock exchange' data (resolution: milliseconds)

I work on

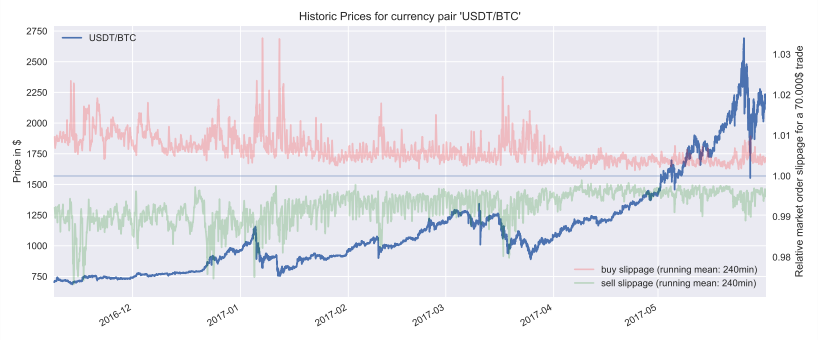
- Self recorded data from **Poloniex** (Digital Assets Market Place):
- Orderbook *snapshots* between Nov 10th 2016 and May 31st 2017

  9 Currencypairs: 'USDT_BTC', 'BTC_ETH', 'BTC_XMR', 'BTC_XRP', 'BTC_FCT', 'BTC_NAV', 'BTC_DASH', 'BTC_MAID', 'BTC_ZEC'
- Datasize: ~30KB per snapshot (resolution: 1 minute)

Call `https://poloniex.com/public?command=returnOrderBook&currencyPair=USDT_BTC&depth=5000`


```
{"asks":[["705.450000",2.772181],["706.170000",0.052838],
    ... ], "bids":[["705.000000",0.158232],["703.700000"
    ,0.001250], ... ], "isFrozen": 0, "seq": 63413296}
```

Historic Order Book Data (2): Bitcoins



Historic BTC prices between Nov, 10th 2016 and May, 31 2017.

Orderbook Trading Simulator (OTS)

Task: Buy 100 bitcoins over a period of 4*15=60 minutes

```
ots = OrderbookTradingSimulator(orderbooks=window_60orderbooks,
    volume=100, tradingperiods=4, periodlength=15)

summary = ots.trade(limit=713.0)  # t=0
summary = ots.trade(limit=715.0)  # t=15
display(ots.history)  # Slippage: 340.85$
```

Orderbook Trading Simulator (OTS)

Task: Buy 100 bitcoins over a period of 4*15=60 minutes

```
ots = OrderbookTradingSimulator(orderbooks=window_60orderbooks,
    volume=100, tradingperiods=4, periodlength=15)

summary = ots.trade(limit=713.0)  # t=0
summary = ots.trade(limit=715.0)  # t=15
display(ots.history)  # Slippage: 340.85$
```

- maintains a *masterbook*.

(1) `master += diff(ob[t] - ob[t-1])`, drop negative share counts.

(2) perform trade: buy until given limit

(3) `master -= bought bitcoins`

(4) done, if `volume==0` or $t == T - 1$ (`forced=True`)

Orderbook Trading Simulator (OTS)

Task: Buy 100 bitcoins over a period of 4*15=60 minutes

```
ots = OrderbookTradingSimulator(orderbooks=window_60orderbooks,
    volume=100, tradingperiods=4, periodlength=15)

summary = ots.trade(limit=713.0)  # t=0
summary = ots.trade(limit=715.0)  # t=15
display(ots.history)  # Slippage: 340.85$
```

- maintains a *masterbook*.
  1. `master += diff(ob[t] - ob[t-1])`, drop negative share counts.
  2. perform trade: buy until given limit
  3. `master -= bought bitcoins`
  4. done, if `volume==0` or $t == T - 1$ (`forced=True`)
- returns detailed trading statistics

	ASK	BID	CENTER	SPREAD	LIMIT	VOLUME	vol_traded	CASH	cash_traded	...	avg	forced	initMAvg	low	high	cost
03:01	711.42	709.74	710.58	1.68	713.0	100.00	46.77	0.00	-33280.72	...	711.51	False	718.42	711.42	713.00	43.48
03:16	712.52	711.86	712.19	0.66	715.0	53.23	28.90	-33280.72	-20630.99	...	713.99	False	718.42	712.52	715.00	98.53
03:31	715.10	712.98	714.04	2.12	717.5	24.33	6.68	-53911.71	-4780.95	...	716.16	False	718.42	715.10	717.41	37.28
03:46	718.60	717.77	718.18	0.83	720.0	17.65	17.65	-58692.66	-12706.15	...	719.73	False	718.42	718.60	720.00	161.57

Orderbook Trading Simulator (OTS): Implementation

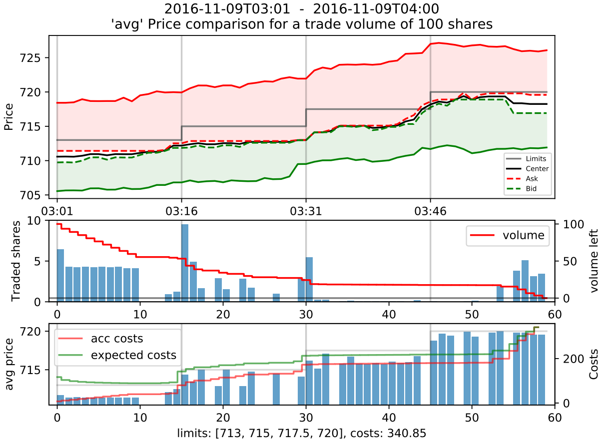


Figure 2: Visualization of an exemplary trading strategy.

Reinforcement Learning Approaches

Reinforcement Learning Framework

Use RL to find optimal limits for Submit & Revise Strategies.

Actions current_best $+ [-0.4, -0.3, ..., 0.9, 1.0]$ %

ask=705.45, $a = +0.1 \Rightarrow limit = 705.45 * 1.001 = 706.16$

Reinforcement Learning Framework

Use RL to find optimal limits for Submit & Revise Strategies.

Actions current_best + $[-0.4, -0.3, ..., 0.9, 1.0]$ %

ask=705.45, $a = +0.1 \Rightarrow \textit{limit} = 705.45 * 1.001 = 706.16$

Costs Difference between expected and obtained price (Slippage)

cost = vol_traded * (avg_paid - initial_center)

$cost = 17.65 * (719.73 - 710.58) = \mathbf{161.57}$

	ASK	BID	CENTER	SPREAD	LIMIT	VOLUME	vol_traded	CASH	cash_traded	...	avg	forced	initMAvg	low	high	cost
03:01	711.42	709.74	710.58	1.68	713.0	100.00	46.77	0.00	-33280.72	...	711.51	False	718.42	711.42	713.00	43.48
03:16	712.52	711.86	712.19	0.66	715.0	53.23	28.90	-33280.72	-20630.99	...	713.99	False	718.42	712.52	715.00	98.53
03:31	715.10	712.98	714.04	2.12	717.5	24.33	6.68	-53911.71	-4780.95	...	716.16	False	718.42	715.10	717.41	37.28
03:46	718.60	717.77	718.18	0.83	720.0	17.65	17.65	-58692.66	-12706.15	...	719.73	False	718.42	718.60	720.00	**161.57**

Reinforcement Learning Framework

Use RL to find optimal limits for Submit & Revise Strategies.

Actions current_best + $[-0.4, -0.3, ..., 0.9, 1.0]$ %

ask=705.45, $a = +0.1 \Rightarrow \mathit{limit} = 705.45 * 1.001 = 706.16$

Costs Difference between expected and obtained price (Slippage)

cost = vol_traded * (avg_paid - initial_center)

$$cost = 17.65 * (719.73 - 710.58) = \mathbf{161.57}$$

	ASK	BID	CENTER	SPREAD	LIMIT	VOLUME	vol_traded	CASH	cash_traded	...	avg	forced	initMAvg	low	high	cost
03:01	711.42	709.74	710.58	1.68	713.0	100.00	46.77	0.00	-33280.72	...	711.51	False	718.42	711.42	713.00	43.48
03:16	712.52	711.86	712.19	0.66	715.0	53.23	28.90	-33280.72	-20630.99	...	713.99	False	718.42	712.52	715.00	98.53
03:31	715.10	712.98	714.04	2.12	717.5	24.33	6.68	-53911.71	-4780.95	...	716.16	False	718.42	715.10	717.41	37.28
03:46	718.60	717.77	718.18	0.83	720.0	17.65	17.65	-58692.66	-12706.15	...	719.73	False	718.42	718.60	720.00	**161.57**

States *remaining time*, *remaining volume*, [orderbook features, ...]

Reinforcement Learning Framework

Use RL to find optimal limits for Submit & Revise Strategies.

Actions current_best + $[-0.4, -0.3, ..., 0.9, 1.0]$ %

ask=705.45, $a = +0.1 \Rightarrow \mathit{limit} = 705.45 * 1.001 = 706.16$

Costs Difference between expected and obtained price (Slippage)

cost = vol_traded * (avg_paid - initial_center)

$cost = 17.65 * (719.73 - 710.58) = \mathbf{161.57}$

	ASK	BID	CENTER	SPREAD	LIMIT	VOLUME	vol_traded	CASH	cash_traded	...	avg	forced	initMAvg	low	high	cost
03:01	711.42	709.74	710.58	1.68	713.0	100.00	46.77	0.00	-33280.72	...	711.51	False	718.42	711.42	713.00	43.48
03:16	712.52	711.86	712.19	0.66	715.0	53.23	28.90	-33280.72	-20630.99	...	713.99	False	718.42	712.52	715.00	98.53
03:31	715.10	712.98	714.04	2.12	717.5	24.33	6.68	-53911.71	-4780.95	...	716.16	False	718.42	715.10	717.41	37.28
03:46	718.60	717.77	718.18	0.83	720.0	17.65	17.65	-58692.66	-12706.15	...	719.73	False	718.42	718.60	720.00	**161.57**

States *remaining time*, *remaining volume*, [orderbook features, ...]

Goal Minimize expected costs

Backward Approach

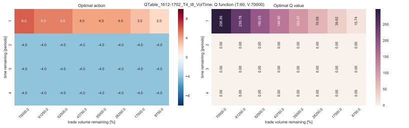
- Proposed by Nevmyvaka et al. in 2006 [2].
- Q-learning and dynamic programming
- Samples discretized state space in a backward, brute force manner.

```
Input: V=70.000$, H=60min, T=4, I=[12.5%..100%],
       L=[-4..10]
for t=1 to T do
    while not end of data do
        Transform (orderbook) → o_1..o_R;
        for i=0 to I do
            for a=0 to L do
                Set x = [t, i, o_1, ..., o_R];
                Simulate transition x → y;
                Calculate immediate cost_im(x, a);
                Look up argmax cost(y, p);
                Update cost([t, v, o_1, ..., o_R], a);
            end
        end
        Select the highest-payout action argmax cost(y, p)
          in every state y to output optimal policy
    end
end
```

Backward Approach

- Proposed by Nevmyvaka et al. in 2006 [2].
- Q-learning and dynamic programming
- Samples discretized state space in a backward, brute force manner.

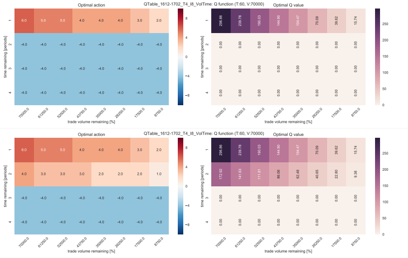
Input: V=70.000\$, H=60min, T=4, I=[12.5%..100%], L=[-4..10]
for *t=1 to T* **do**
    while *not end of data* **do**
        Transform (orderbook) $\rightarrow o_1..o_R$;
        for *i=0 to I* **do**
            for *a=0 to L* **do**
                Set $x = [t, i, o_1, ..., o_R]$;
                Simulate transition $x \rightarrow y$;
                Calculate immediate $\text{cost}_{im}(x, a)$;
                Look up argmax $\text{cost}(y, p)$;
                Update $\text{cost}([t, v, o_1, ..., o_R], a)$;
            end
        end
        Select the highest-payout action argmax $\text{cost}(y, p)$ in every state y to output optimal policy
    end
end



Backward Approach

- Proposed by Nevmyvaka et al. in 2006 [2].
- Q-learning and dynamic programming
- Samples discretized state space in a backward, brute force manner.

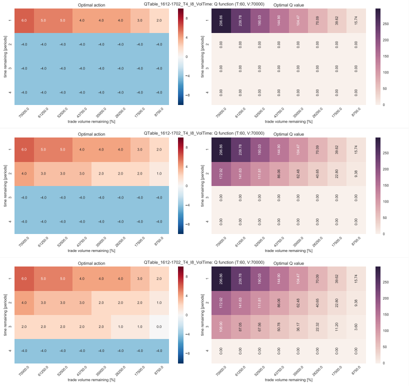
Input: V=70.000$, H=60min, T=4, I=[12.5%..100%], L=[-4..10]
for *t=1 to T* **do**
  while *not end of data* **do**
    Transform (orderbook) $\rightarrow o_1..o_R$;
    for *i=0 to I* **do**
      for *a=0 to L* **do**
        Set $x = [t, i, o_1, ..., o_R]$;
        Simulate transition $x \rightarrow y$;
        Calculate immediate $\text{cost}_{im}(x, a)$;
        Look up argmax $\text{cost}(y, p)$;
        Update $\text{cost}([t, v, o_1, ..., o_R], a)$;
      end
    end
    Select the highest-payout action argmax $\text{cost}(y, p)$ in every state y to output optimal policy
  end
end



Backward Approach

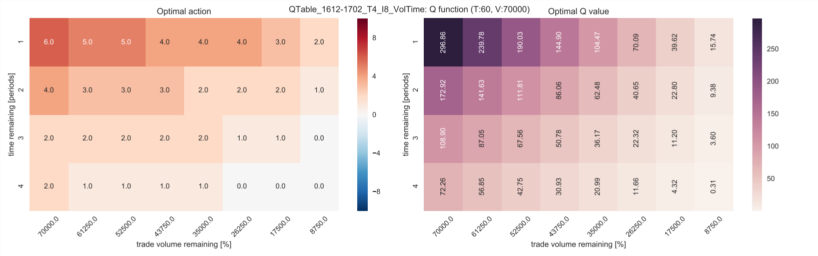
- Proposed by Nevmyvaka et al. in 2006 [2].
- Q-learning and dynamic programming
- Samples discretized state space in a backward, brute force manner.

Input: V=70.000$, H=60min, T=4, I=[12.5%..100%], L=[-4..10]
for *t=1 to T* **do**
  while *not end of data* **do**
    Transform (orderbook) $\rightarrow o_1..o_R$;
    for *i=0 to I* **do**
      for *a=0 to L* **do**
        Set $x = [t, i, o_1, ..., o_R]$;
        Simulate transition $x \rightarrow y$;
        Calculate immediate $\text{cost}_{im}(x, a)$;
        Look up argmax $\text{cost}(y, p)$;
        Update $\text{cost}([t, v, o_1, ..., o_R], a)$;
      end
    end
    Select the highest-payout action argmax $\text{cost}(y, p)$ in every state y to output optimal policy
  end
end



Backward Approach

- Proposed by Nevmyvaka et al. in 2006 [2].
- Q-learning and dynamic programming
- Samples discretized state space in a backward, brute force manner.



Trained over: 4.154 periods á 60 minutes (Nov 2016 - Apr 2017)

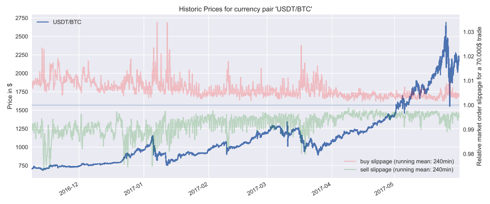
states = [time_left, volume_left]

32 states, 15 actions = 480 trials per period

Backward Approach - Discussion (1)

Subject of Trade

- Nevmyvaka: Buy 100 shares
- Here: Buy shares for 70.000$



Backward Approach - Discussion (2)

Action limit mapping

- Nevmyvaka: Forced crossing of the bid-ask spread.
- Here: no crossing, action=0 guarantees partial execution.

limit_buy = ask * (1 + (a/1000)) instead of limit_buy = bid + a

limit_sell = bid * (1 − (a/1000)) instead of limit_sell = ask − a,

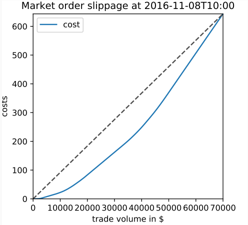
where $a \in [-4, -3, ..., +9, +10]$

	Amount	Type	Volume	VolumeAcc	norm_Price
31.00	200.0	ask	6200.0	8425.0	1.074533
30.00	50.0	ask	1500.0	2225.0	1.039871
29.00	25.0	ask	725.0	725.0	1.005208
28.85	NaN	center	NaN	NaN	NaN
28.70	200.0	bid	5740.0	5740.0	0.994810
28.50	100.0	bid	2850.0	8590.0	0.987877
28.00	300.0	bid	8400.0	16990.0	0.970546

Backward Approach - Discussion (3)

Discrete State Space

- No generalization to states never observed.
- Rounding required in learning process.
  - Loss of accuracy
  - Cost scaling problem:
    Non-linear slippage growth.

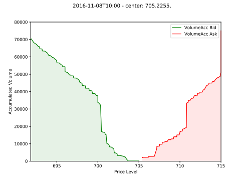


Markov Property

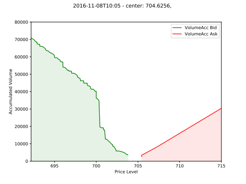
- States capture all relevant information from history.

Nevmyvaka: Backward approach does not incorporate preceding trades.

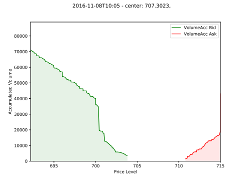
E.g., if starting at t=45 and a remaining trade volume of 17.500$:



(a) Original Orderbook.



(b) Assuming 52.500 shares being matched at t=0, then no further matches.



(c) Assuming 52.500 shares being matched evenly at 1.166 shares per minute.

Experiments

Data & Task

Data

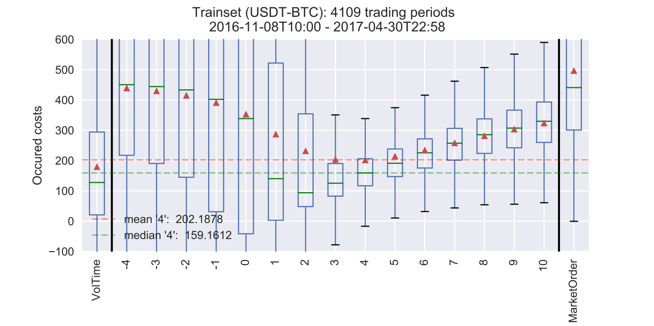
- Currency pair: USDT/BTC
- Training period: Nov 10th, 2016 - Apr, 30th 2017: 4154 orderbook windows á 60min.
- Test period: May 2017: 724 orderbook windows

Task:

- Buy Bitcoins worth of 70.000$
- Trading horizon: 60min
- 4 order revisions (i.e. period length=15min)
- 8 inventory levels (8.750$, ..., 70.000$)
  Sample transitions, collected in backward mode:
  $\Rightarrow 4.154 * 4 * 8 * 15 =$ **1.993.920 samples**

Baseline

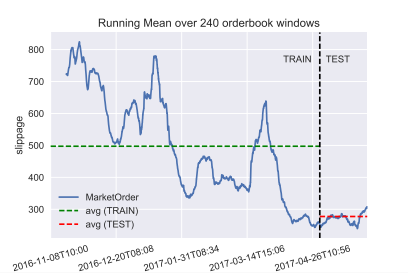
- Simple Market Order
- Optimal Submit & Leave strategie: a=4



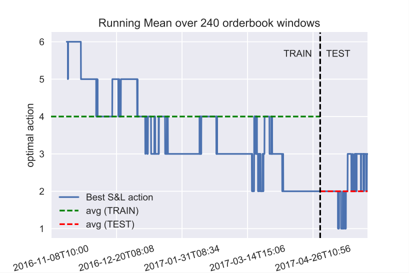
Average Submit & Leave costs over the full training period.

Baseline

- Simple Market Order
- Optimal Submit & Leave strategie: a=4 and a=2



(a) Observed Market Order Slippage.



(b) Best S&L action.

Backward Approach: Additional Market Variables

	slippage	med	std	perf_2	perf_4	perf_M	perf_VolTime
center_price_disc3	149.57	46.15	420.14	−3.44%	−15.19%	−46.10%	−0.70%
center_price_disc5	161.70	37.83	450.36	+4.39%	−8.32%	−41.73%	+7.35%
marketPrice_buy_worst_disc5	146.83	39.62	351.77	−5.21%	−16.75%	−47.08%	−2.52%
marketPrice_sell_worst_disc5	148.94	42.71	388.86	−3.85%	−15.55%	−46.33%	−1.12%
marketPrice_spread_disc5	150.46	44.15	371.32	−2.87%	−14.69%	−45.78%	−0.11%
marketPrice_imbalance_disc5	150.10	68.68	336.60	−3.10%	−14.90%	−45.91%	−0.35%
sharecount_buy_disc3	149.57	46.15	420.14	−3.44%	−15.19%	−46.10%	−0.70%
sharecount_buy_disc5	160.73	37.78	449.22	+3.76%	−8.87%	−42.08%	+6.70%
sharecount_imbalance_disc5	148.03	40.49	372.86	−4.44%	−16.07%	−46.65%	−1.73%
sharecount_sell_disc5	151.50	47.49	425.35	−2.20%	−14.10%	−45.40%	+0.58%
sharecount_spread_disc5	148.93	40.84	352.12	−3.86%	−15.56%	−46.33%	−1.13%
spread_disc3	147.41	35.48	370.96	−4.84%	−16.42%	−46.88%	−2.14%
spread_disc5	141.07	37.39	349.72	−8.93%	−20.02%	−49.16%	−6.34%
spread_disc9	142.80	36.69	364.76	−7.81%	−19.03%	−48.54%	−5.20%
VolTime	150.63	33.83	358.66	−2.76%	−14.60%	−45.72%	0.00%
2	154.90	68.62	389.15	0.00%	−12.17%	−44.18%	+2.84%
4	176.37	141.66	273.58	+13.86%	0.00%	−36.44%	+17.09%
MarketOrder	277.49	246.48	158.66	+79.14%	+57.33%	0.00%	+84.22%

Only *spread* is beneficial.

Backward Approach: Look-ahead Variables

	slippage	med	std	perf_2	perf_4	perf_M	perf_VolTime
ob_direction_disc5	61.99	97.52	319.84	−59.98%	−64.85%	−77.66%	−58.84%
future_center5_disc5	141.61	41.79	389.58	−8.58%	−19.71%	−48.97%	−5.98%
future_center15_disc5	133.78	40.90	405.88	−13.63%	−24.15%	−51.79%	−11.18%
future_center60_disc5	131.85	47.63	391.46	−14.88%	−25.25%	−52.49%	−12.47%
VolTime	150.63	33.83	358.66	−2.76%	−14.60%	−45.72%	0.00%
2	154.90	68.62	389.15	0.00%	−12.17%	−44.18%	+2.84%
4	176.37	141.66	273.58	+13.86%	0.00%	−36.44%	+17.09%
MarketOrder	277.49	246.48	158.66	+79.14%	+57.33%	0.00%	+84.22%

Massive performance gain.

Backward Approach: Simulation of Proceeding trades

	slippage	med	std	perf_2	perf_4	perf_M	perf_VolTime	perf: Table 5
center_orig_disc3	158.97	48.93	431.16	+2.62%	−9.87%	−42.71%	+5.54%	−0.70%
center_orig_disc5	150.47	47.49	422.12	−2.86%	−14.69%	−45.78%	−0.11%	+7.35%
marketPrice_buy_worst_disc5	148.33	47.46	400.73	−4.24%	−15.90%	−46.55%	−1.53%	−2.52%
marketPrice_sell_worst_disc5	154.06	42.10	370.91	−0.54%	−12.65%	−44.48%	+2.28%	−1.12%
marketPrice_spread_disc5	151.12	53.07	395.68	−2.44%	−14.32%	−45.54%	+0.33%	−0.11%
marketPrice_imbalance_disc5	143.86	47.05	375.43	−7.13%	−18.44%	−48.16%	−4.49%	−0.35%
sharecount_buy_disc3	149.57	46.15	420.14	−3.44%	−15.19%	−46.10%	−0.70%	−0.70%
sharecount_buy_disc5	150.47	47.49	422.12	−2.86%	−14.69%	−45.78%	−0.11%	+6.70%
sharecount_imbalance_disc5	151.92	53.41	400.69	−1.92%	−13.86%	−45.25%	+0.86%	−1.73%
sharecount_sell_disc5	149.93	47.01	420.70	−3.21%	−14.99%	−45.97%	−0.47%	+0.58%
sharecount_spread_disc5	149.36	53.41	402.86	−3.58%	−15.31%	−46.17%	−0.84%	−1.13%
spread_disc3	145.63	41.89	346.00	−5.98%	−17.43%	−47.52%	−3.32%	−2.14%
spread_disc5	139.87	41.22	336.30	−9.70%	−20.69%	−49.59%	−7.14%	−6.34%
spread_disc9	141.30	42.80	336.88	−8.78%	−19.89%	−49.08%	−6.19%	−5.20%
ob_direction_disc5	62.03	97.52	320.76	−59.95%	−64.83%	−77.64%	−58.82%	−58.84%
VolTime_simulatedTrades	147.86	42.10	346.17	−4.55%	−16.17%	−46.72%	−1.84%	
VolTime	150.63	33.83	358.66	−2.76%	−14.60%	−45.72%	0.00%	
2	154.90	68.62	389.15	0.00%	−12.17%	−44.18%	+2.84%	
4	176.37	141.66	273.58	+13.86%	0.00%	−36.44%	+17.09%	
MarketOrder	277.49	246.48	158.66	+79.14%	+57.33%	0.00%	+84.22%	

$-2.76\% \Rightarrow -4.55\%$ [Volume, Time]

$-8.93\% \Rightarrow -9.70\%$ [Volume, Time, Spread]

Backward Approach: Action Consequences

	slippage	med	std	perf_2	perf_4	perf_M	perf_VolTime
_a_-4_disc5	149.84	40.54	368.50	−3.27%	−15.04%	−46.00%	−0.52%
_a_-3_disc5	150.70	37.95	372.15	−2.71%	−14.55%	−45.69%	+0.05%
_a_0_disc5	153.01	45.17	375.98	−1.22%	−13.25%	−44.86%	+1.58%
_a_1_disc5	148.56	41.60	385.79	−4.10%	−15.77%	−46.46%	−1.37%
_a_2_disc5	149.47	43.47	368.49	−3.50%	−15.25%	−46.13%	−0.77%
_a_3_disc5	145.04	42.42	365.27	−6.37%	−17.76%	−47.73%	−3.71%
_a_4_disc5	155.52	40.84	373.98	+0.40%	−11.82%	−43.95%	+3.25%
_a_5_disc5	152.05	38.53	372.03	−1.84%	−13.79%	−45.21%	+0.94%
_a_6_disc5	151.80	40.39	371.41	−2.00%	−13.93%	−45.29%	+0.78%
_a_7_disc5	148.61	40.39	363.67	−4.06%	−15.74%	−46.44%	−1.34%
_a_8_disc5	152.46	41.26	363.02	−1.57%	−13.56%	−45.06%	+1.22%
_a_9_disc5	151.59	40.84	363.68	−2.14%	−14.05%	−45.37%	+0.64%
_a_10_disc5	150.28	40.84	363.70	−2.99%	−14.80%	−45.84%	−0.23%
VolTime_simulatedTrades	147.86	42.10	346.17	−4.55%	−16.17%	−46.72%	−1.84%
VolTime	150.63	33.83	358.66	−2.76%	−14.60%	−45.72%	0.00%
2	154.90	68.62	389.15	0.00%	−12.17%	−44.18%	+2.84%
4	176.37	141.66	273.58	+13.86%	0.00%	−36.44%	+17.09%
MarketOrder	277.49	246.48	158.66	+79.14%	+57.33%	0.00%	+84.22%

No improvement

Backward Approach: Function Approximation

- RandomForestRegressor: BatchTree-agent
- 150 Trees

	slippage	med	std	perf_2	perf_4	perf_M	perf_VolTime
BT_VolTime_a4_	163.21	25.48	466.45	+5.36%	−7.46%	−41.18%	+8.35%
BT_VolTime_a*_	143.24	38.53	373.35	−7.53%	−18.78%	−48.38%	−4.90%
BT_VolTime	151.08	47.39	391.02	−2.47%	−14.34%	−45.55%	+0.30%
BT_VolTimeSpread	249.07	237.29	142.11	+60.79%	+41.22%	−10.24%	+65.36%
VolTime_simulatedTrades	147.86	42.10	346.17	−4.55%	−16.17%	−46.72%	−1.84%
VolTime	150.63	33.83	358.66	−2.76%	−14.60%	−45.72%	0.00%
2	154.90	68.62	389.15	0.00%	−12.17%	−44.18%	+2.84%
4	176.37	141.66	273.58	+13.86%	0.00%	−36.44%	+17.09%
MarketOrder	277.49	246.48	158.66	+79.14%	+57.33%	0.00%	+84.22%

VolTimeSpread: terrible performance

Single a*-feature: loss in performance

All a*-features simultaneously: improvement of −7.53%

Forward Approach

Idea: Collect more realistic transitions

- Simulate **full** trades
- Include trade history
- Sample from continuous state space

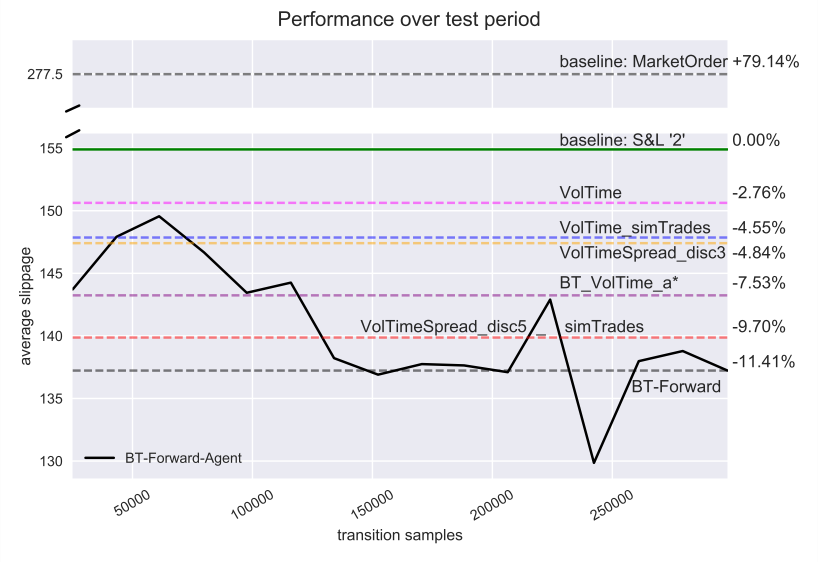
Forward Approach

Idea: Collect more realistic transitions

- Simulate **full** trades
- Include trade history
- Sample from continuous state space

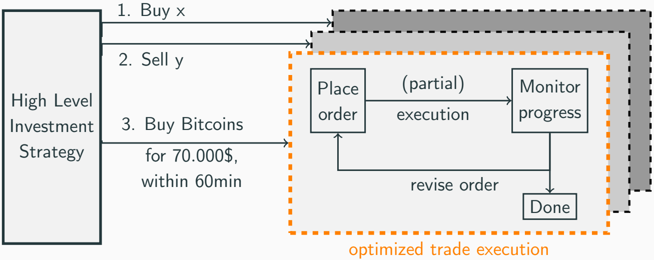
```
Input: data, V=70.000$, H=60min, T=4, L=[-4..10], E=60, retrain=256
Shuffle(data);
Split data into chunks of length retrain
while not end of chunks do
    for orderbook window in chunks do
        Init OTS(orderbook_window, V, H, T, L)
        for epoch=0 to E do
            Reset OTS to V = 100% and random time point (in H)
            repeat
                Set x_t = [time_left, volume_left, o_1, ..., o_R]
                Enquire ϵ-greedy action from model
                if action chain led to an end state previously then
                    choose other action
                end
                Apply action
                Remember transition {x_t, action, cost, x_{t+1}}
            until V = 0%;
        end
        Retrain model from collected transitions (growing batch)
    end
end
```

Forward Approach Performance



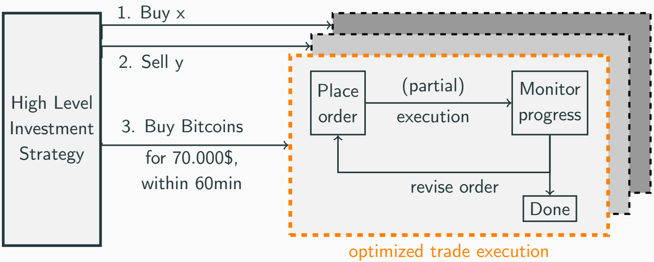
Conclusion

Conclusion



1. Buy x
2. Sell y
High Level Investment Strategy
3. Buy Bitcoins
for 70.000$,
within 60min
Place order
(partial)
execution
Monitor progress
revise order
Done
optimized trade execution

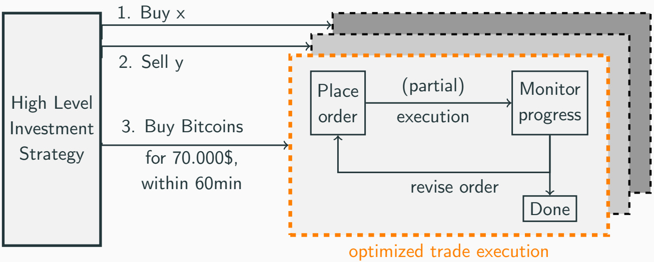
Conclusion



Goal: Minimize hidden costs (MarketOrder in may 2017: $\approx 277.49\$$)

- Submit & Leave Strategy: 0.00% (-44.18%)

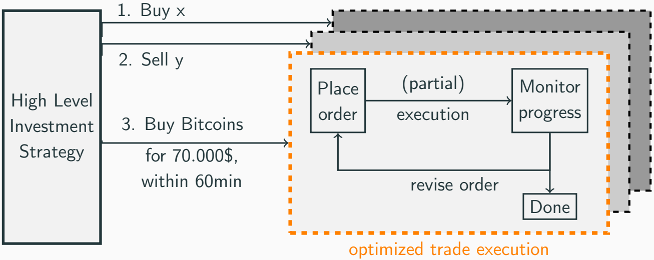
Conclusion



Goal: Minimize hidden costs (MarketOrder in may 2017: $\approx$ 277.49$)

- Submit & Leave Strategy: 0.00% ($-$44.18%)
- Backward approach by Nevmyvaka et al.: $-$4.84% ($-$46.88%)

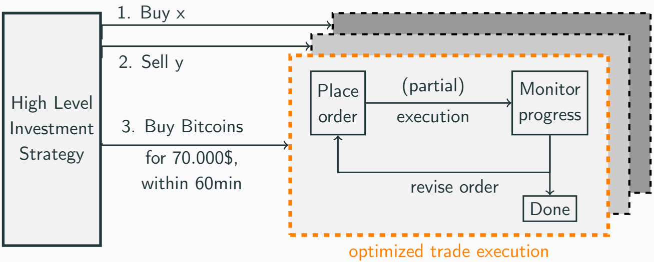
Conclusion



Goal: Minimize hidden costs (MarketOrder in may 2017: $\approx$ 277.49$)

- Submit & Leave Strategy: 0.00% ($-$44.18%)
- Backward approach by Nevmyvaka et al.: $-$4.84% ($-$46.88%)
- Backward approach improved: $-$9.70% ($-$49.59%)

Conclusion



Goal: Minimize hidden costs (MarketOrder in may 2017: $\approx$ 277.49$)

- Submit & Leave Strategy: 0.00% ($-$44.18%)
- Backward approach by Nevmyvaka et al.: $-$4.84% ($-$46.88%)
- Backward approach improved: $-$9.70% ($-$49.59%)
- Forward approach: $-$11.41% ($-$50.55%)

Future Work

Orderbook Trading Simulator

- Different data: high resolution log files

Future Work

Orderbook Trading Simulator

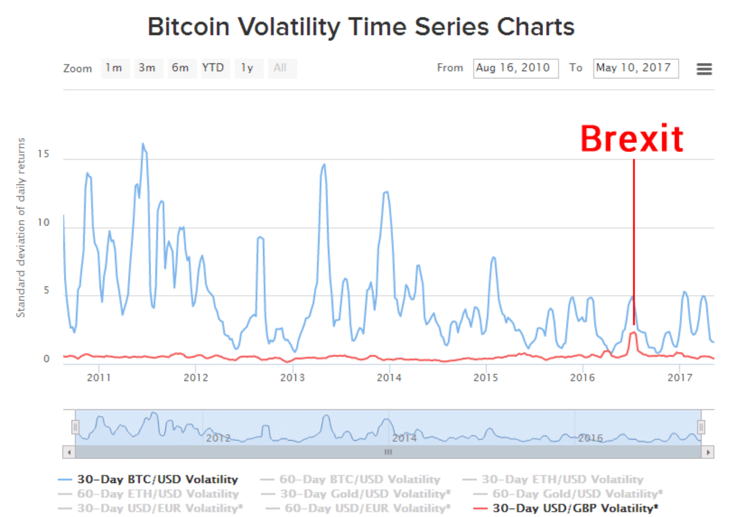
- Different data: high resolution log files

Forward Learning Approach

- (Deep) neural networks and replay memory
- Continuous action space
- Combine with specialized time series prediction methods
  e.g. Prevedex STaR [1]

Questions?

Volatiliy of USDT/BTC and (USDT/Pound



Source: `https://99bitcoins.com/bitcoin-volatility-explained`

Performance of forward agent

	slippage	med	std	perf_2	perf_4	perf_M	perf_VolTime
BT_Forward_samples_025.078	143.72	74.69	300.93	−7.22%	−18.52%	−48.21%	−4.59%
BT_Forward_samples_043.334	147.92	82.24	310.49	−4.51%	−16.13%	−46.69%	−1.80%
BT_Forward_samples_061.103	149.56	77.33	288.56	−3.45%	−15.20%	−46.10%	−0.71%
BT_Forward_samples_079.809	146.67	77.20	277.43	−5.31%	−16.84%	−47.14%	−2.62%
BT_Forward_samples_097.615	143.45	73.18	281.71	−7.40%	−18.67%	−48.31%	−4.77%
BT_Forward_samples_116.008	144.26	77.94	285.80	−6.87%	−18.21%	−48.01%	−4.23%
BT_Forward_samples_134.011	138.22	73.30	269.71	−10.77%	−21.63%	−50.19%	−8.24%
BT_Forward_samples_152.352	136.90	69.59	264.95	−11.62%	−22.38%	−50.66%	−9.11%
BT_Forward_samples_170.605	137.75	77.50	261.17	−11.07%	−21.90%	−50.36%	−8.55%
BT_Forward_samples_188.224	137.64	73.71	253.37	−11.14%	−21.96%	−50.40%	−8.62%
BT_Forward_samples_206.409	137.10	74.22	257.54	−11.49%	−22.27%	−50.59%	−8.98%
BT_Forward_samples_224.083	142.90	75.43	265.78	−7.75%	−18.98%	−48.50%	−5.13%
BT_Forward_samples_242.342	129.85	74.43	239.96	−16.17%	−26.37%	−53.20%	−13.79%
BT_Forward_samples_261.018	137.98	75.69	248.07	−10.92%	−21.77%	−50.27%	−8.39%
BT_Forward_samples_279.347	138.79	75.28	258.49	−10.40%	−21.31%	−49.98%	−7.86%
BT_Forward_samples_298.020	137.23	78.71	242.88	−11.41%	−22.19%	−50.55%	−8.89%
VolTime	150.63	33.83	358.66	−2.76%	−14.60%	−45.72%	0.00%
2	154.90	68.62	389.15	0.00%	−12.17%	−44.18%	+2.84%
4	176.37	141.66	273.58	+13.86%	0.00%	−36.44%	+17.09%
MarketOrder	277.49	246.48	158.66	+79.14%	+57.33%	0.00%	+84.22%

The circuit of masterbook adjustments

	t=0	t=1	t=2	t=3	t=4
ob[t]	705.45: 3.17 707.18: 7.99	705.45: 5.04 707.18: 1.53	705.45: 6.85 707.18: 1.53	705.45: 5.08 707.18: 7.77	705.45: 6.84 707.18: 7.77
diff = ob[t]-ob[t-1]	705.45: +3.17 707.18: +7.99	705.45: +1.87 707.18: -6.45	705.45: +1.81	705.45: -1.77 707.18: +6.24	705.45: +1.75
master = master + diff	705.45: 3.17 707.18: 7.99	705.45: 1.87 707.18: 1.53	705.45: 1.81 707.18: 1.53	705.45: -1.77 707.18: 7.77 drop negatives	705.45: 1.75 707.18: 7.77
trade now	705.45: 3.17	705.45: 1.87	705.45: 1.81	-	705.45: 1.75
traded total	705.45: 3.17	705.45: 5.04	705.45: 6.85	-	705.45: 8.6
master = master - trade	707.18: 7.99	707.18: 1.53	707.18: 1.53	707.18: 7.77	707.18: 7.77

Table 2: The circuit of masterbook adjustments

W. Groß.
Predicting time series with space-time convolutional and recurrent neural networks.
In M. Verleysen, editor, *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*. ESANN, Apr. 2017.

Y. Nevmyvaka, Y. Feng, and M. Kearns.
Reinforcement learning for optimized trade execution.
In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 673–680, New York, NY, USA, 2006. ACM.