



CENSORED EXPLORATION AND THE DARK POOL PROBLEM

Kuzman Ganchev, Michael Kearns, Yuriy Nevmyvaka,
Jennifer Wortman Vaughan

Computer & Information Science
University of Pennsylvania

June 19, 2009

STOCK EXCHANGES

- Exchange maintains sorted list of orders called “order books”
- submitting an order \Rightarrow trade or add to order books

price	volume	
\$20	2000	}sell
\$12.34	100	
\$11.56	300	
\$10.34	200	}buy
\$ 2.12	1000	

STOCK EXCHANGES

- Exchange maintains sorted list of orders called “order books”
- submitting an order \Rightarrow trade or add to order books
- **DEFINITION** liquidity = volumes available to trade

price	volume	
\$20	2000	}sell
\$12.34	100	
\$11.56	300	
\$10.34	200	}buy
\$ 2.12	1000	

STOCK EXCHANGES

- Exchange maintains sorted list of orders called “order books”
- submitting an order \Rightarrow trade or add to order books
- **DEFINITION** liquidity = volumes available to trade
- order books are visible to all participants

price	volume	
\$20	2000	}sell
\$12.34	100	
\$11.56	300	
\$10.34	200	}buy
\$ 2.12	1000	

STOCK EXCHANGES

- Exchange maintains sorted list of orders called “order books”
- submitting an order \Rightarrow trade or add to order books
- **DEFINITION** liquidity = volumes available to trade
- order books are visible to all participants
- visibility allows price discovery, easier regulation

price	volume	
\$20	2000	}sell
\$12.34	100	
\$11.56	300	
\$10.34	200	}buy
\$ 2.12	1000	

STOCK EXCHANGES

- Exchange maintains sorted list of orders called “order books”
- submitting an order \Rightarrow trade or add to order books
- **DEFINITION** liquidity = volumes available to trade
- order books are visible to all participants
- visibility allows price discovery, easier regulation
 - * hard to execute large orders *

price	volume	
\$20	2000	}sell
\$12.34	100	
\$11.56	300	
\$10.34	200	}buy
\$ 2.12	1000	

DARK POOLS

- order books are *hidden* (good for information hiding)
- priced pegged to light exchange
- ≈ 40 dark pools exist in the US
- compete intensely for orders
- **PROBLEM:** how do we split up a large order among dark pools?
 - Can't observe which dark pool has liquidity
 - Delays associated with trying many

DARK POOL MODEL

A mathematical model for dark pools.

- w.l.o.g. assume we are buying shares
- each pool has probability distribution over liquidity
- each time we submit an order, available liquidity is sampled
- we observe only the amount we traded

DARK POOL MODEL

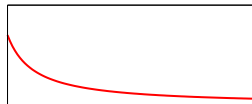
A mathematical model for dark pools.

- w.l.o.g. assume we are buying shares
- each pool has probability distribution over liquidity
- each time we submit an order, available liquidity is sampled
- we observe only the amount we traded

BUYER

DARKPOOL

Prob. of liquidity at DP 1



DARK POOL MODEL

A mathematical model for dark pools.

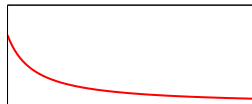
- w.l.o.g. assume we are buying shares
- each pool has probability distribution over liquidity
- each time we submit an order, available liquidity is sampled
- we observe only the amount we traded

BUYER

DARKPOOL

Submit 5,000

Prob. of liquidity at DP 1



DARK POOL MODEL

A mathematical model for dark pools.

- w.l.o.g. assume we are buying shares
- each pool has probability distribution over liquidity
- each time we submit an order, available liquidity is sampled
- we observe only the amount we traded

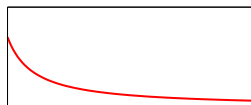
BUYER

DARKPOOL

Submit 5,000

3,000 shares available

Prob. of liquidity at DP 1



DARK POOL MODEL

A mathematical model for dark pools.

- w.l.o.g. assume we are buying shares
- each pool has probability distribution over liquidity
- each time we submit an order, available liquidity is sampled
- we observe only the amount we traded

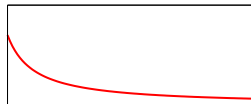
BUYER

DARKPOOL

Submit 5,000

3,000 shares available
← liquidity = 3,000

Prob. of liquidity at DP 1



DARK POOL MODEL

A mathematical model for dark pools.

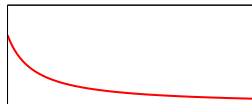
- w.l.o.g. assume we are buying shares
- each pool has probability distribution over liquidity
- each time we submit an order, available liquidity is sampled
- we observe only the amount we traded

BUYER

DARKPOOL

Submit 5,000

Prob. of liquidity at DP 1



DARK POOL MODEL

A mathematical model for dark pools.

- w.l.o.g. assume we are buying shares
- each pool has probability distribution over liquidity
- each time we submit an order, available liquidity is sampled
- we observe only the amount we traded

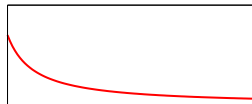
BUYER

DARKPOOL

Submit 5,000

6,000 shares available

Prob. of liquidity at DP 1



DARK POOL MODEL

A mathematical model for dark pools.

- w.l.o.g. assume we are buying shares
- each pool has probability distribution over liquidity
- each time we submit an order, available liquidity is sampled
- we observe only the amount we traded

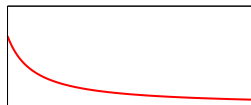
BUYER

DARKPOOL

Submit 5,000

6,000 shares available
← liquidity $\geq 5,000$

Prob. of liquidity at DP 1



DARK POOL PROBLEM

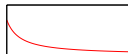
Every timestep, client gives us an order to split among dark pools

- 1 optimal policy when distributions are *known*
- 2 learn optimal policy when distributions are *unknown*

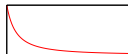
BROKER

DARK POOLS

Prob. of liquidity at DP 1



Prob. of liquidity at DP 2



Prob. of liquidity at DP 3



DARK POOL PROBLEM

Every timestep, client gives us an order to split among dark pools

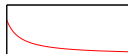
- 1 optimal policy when distributions are *known*
- 2 learn optimal policy when distributions are *unknown*

⇒ Buy 20,000 shares

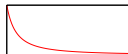
BROKER

DARK POOLS

Prob. of liquidity at DP 1



Prob. of liquidity at DP 2



Prob. of liquidity at DP 3



DARK POOL PROBLEM

Every timestep, client gives us an order to split among dark pools

- 1 optimal policy when distributions are *known*
- 2 learn optimal policy when distributions are *unknown*

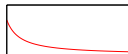
⇒ Buy 20,000 shares

BROKER

DARK POOLS

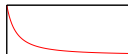
Submit 10,000

Prob. of liquidity at DP 1



Submit 7,000

Prob. of liquidity at DP 2



Submit 3,000

Prob. of liquidity at DP 3



DARK POOL PROBLEM

Every timestep, client gives us an order to split among dark pools

- 1 optimal policy when distributions are *known*
- 2 learn optimal policy when distributions are *unknown*

⇒ Buy 20,000 shares

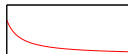
BROKER

DARK POOLS

Submit 10,000

5,000 shares available

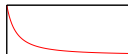
Prob. of liquidity at DP 1



Submit 7,000

0 shares available

Prob. of liquidity at DP 2



Submit 3,000

10,000 shares available

Prob. of liquidity at DP 3



DARK POOL PROBLEM

Every timestep, client gives us an order to split among dark pools

- 1 optimal policy when distributions are *known*
- 2 learn optimal policy when distributions are *unknown*

⇒ Buy 20,000 shares

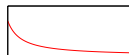
BROKER

DARK POOLS

Submit 10,000

5,000 shares available

Prob. of liquidity at DP 1



Submit 7,000

0 shares available

Prob. of liquidity at DP 2



Submit 3,000

10,000 shares available

Prob. of liquidity at DP 3



DARK POOL PROBLEM

Every timestep, client gives us an order to split among dark pools

- 1 optimal policy when distributions are *known*
- 2 learn optimal policy when distributions are *unknown*

⇒ Buy 20,000 shares

BROKER

DARK POOLS

Submit 10,000

5,000 shares available
 ← liquidity = 5,000

Prob. of liquidity at DP 1



Submit 7,000

0 shares available
 ← liquidity = 0

Prob. of liquidity at DP 2



Submit 3,000

10,000 shares available
 ← liquidity ≥ 3,000

Prob. of liquidity at DP 3



MAIN RESULTS

Algorithm for censored exploration/exploitation.

MAIN RESULTS

Algorithm for censored exploration/exploitation.

- Theoretical Results
 - efficient between-venue exploration
 - polynomial time convergence to near optimal policy
 - simple allocation/reestimation loop

MAIN RESULTS

Algorithm for censored exploration/exploitation.

- Theoretical Results
 - efficient between-venue exploration
 - polynomial time convergence to near optimal policy
 - simple allocation/reestimation loop
- Empirical Results
 - based on real order flow from major US broker-dealer
 - found simple parametric form for liquidity distribution
 - algorithm competitive with strategy used by real broker
 - algorithm converges to optimal performance in practice

PRELIMINARIES

Tail probabilities: $T_i(v) = \Pr$ at least v shares available at i

PRELIMINARIES

Tail probabilities: $T_i(v) = \Pr$ at least v shares available at i

- Allocation algorithm: *Greedy*
 - maximizes expected number of shares traded in one step
 - greedy algorithm:
 - allocate each share to maximize marginal increase in expected number of shares traded

PRELIMINARIES

Tail probabilities: $T_i(v) = \Pr$ at least v shares available at i

- Allocation algorithm: *Greedy*
 - maximizes expected number of shares traded in one step
 - greedy algorithm:
 - allocate each share to maximize marginal increase in expected number of shares traded
- Estimation algorithm: *OptimisticKM*
 - Kaplan-Meier nonparametric maximum likelihood estimator
 - slight modification required

ALGORITHM SKETCH

Input: Volume sequence V^1, V^2, V^3, \dots

Initialize \hat{T}_i to uniform $\forall i$;

for $t \leftarrow 1, 2, 3, \dots$ **do**

end

ALGORITHM SKETCH

Input: Volume sequence V^1, V^2, V^3, \dots
Initialize \hat{T}_i to uniform $\forall i$;
for $t \leftarrow 1, 2, 3, \dots$ **do**
 % Allocation Step:

 % Reestimation Step:

end

ALGORITHM SKETCH

Input: Volume sequence V^1, V^2, V^3, \dots
Initialize \hat{T}_i to uniform $\forall i$;
for $t \leftarrow 1, 2, 3, \dots$ **do**
 % Allocation Step:
 compute & submit optimal allocation w.r.t. \hat{T} ;
 % Reestimation Step:

end

ALGORITHM SKETCH

Input: Volume sequence V^1, V^2, V^3, \dots
Initialize \hat{T}_i to uniform $\forall i$;
for $t \leftarrow 1, 2, 3, \dots$ **do**
 % Allocation Step:
 compute & submit optimal allocation w.r.t. \hat{T} ;
 % Reestimation Step:
 use *OptimisticKM* to reestimate \hat{T}_i ;
end

STATEMENT OF THEORETICAL RESULT

MAIN THEOREM

For any $\epsilon > 0$ and $\delta > 0$, with probability $1 - \delta$, after running for a polynomial time, our algorithm makes an ϵ -optimal allocation on each subsequent time step with probability at least $1 - \epsilon$.

STATEMENT OF THEORETICAL RESULT

MAIN THEOREM

For any $\epsilon > 0$ and $\delta > 0$, with probability $1 - \delta$, after running for a polynomial time, our algorithm makes an ϵ -optimal allocation on each subsequent time step with probability at least $1 - \epsilon$.

- randomness is over draws from liquidity distributions, volumes

STATEMENT OF THEORETICAL RESULT

MAIN THEOREM

For any $\epsilon > 0$ and $\delta > 0$, with probability $1 - \delta$, after running for a polynomial time, our algorithm makes an ϵ -optimal allocation on each subsequent time step with probability at least $1 - \epsilon$.

- randomness is over draws from liquidity distributions, volumes
- polynomial in:
 - $1/\epsilon$
 - $\ln(1/\delta)$
 - number of venues (number of distributions we learn)
 - maximum volume (complexity of distributions we learn)

4 STEP PROOF SKETCH

- *OptimisticKM*: convergence bound, interaction with *Greedy*

4 STEP PROOF SKETCH

- *OptimisticKM*: convergence bound, interaction with *Greedy*
- Exploitation Lemma:
- Exploration Lemma:

4 STEP PROOF SKETCH

- *OptimisticKM*: convergence bound, interaction with *Greedy*
- Exploitation Lemma: if we are in a “known state”, allocation is ϵ -optimal
- Exploration Lemma:

4 STEP PROOF SKETCH

- *OptimisticKM*: convergence bound, interaction with *Greedy*
- Exploitation Lemma: if we are in a “known state”, allocation is ϵ -optimal
- Exploration Lemma: if we are not ϵ -optimal, then we explore

4 STEP PROOF SKETCH

- *OptimisticKM*: convergence bound, interaction with *Greedy*
- Exploitation Lemma: if we are in a “known state”, allocation is ϵ -optimal
- Exploration Lemma: if we are not ϵ -optimal, then we explore
- Main theorem: tying exploration and exploitation.

4 STEP PROOF SKETCH

- *OptimisticKM*: convergence bound, interaction with *Greedy*
- Exploitation Lemma: if we are in a “known state”, allocation is ϵ -optimal
- Exploration Lemma: if we are not ϵ -optimal, then we explore
- Main theorem: tying exploration and exploitation.

DETAILS..

- cutoff point (volume) for each venue
 \Rightarrow below cutoff, know tail probabilities well.
- known state \Leftrightarrow each allocation is below cutoff

EMPIRICAL RESULTS

DATA SET

- real order flow of major US broker-dealer
- 12 stocks
- 4 dark pools

EMPIRICAL RESULTS

DATA SET

- real order flow of major US broker-dealer
- 12 stocks
- 4 dark pools

EXPERIMENTS

EMPIRICAL RESULTS

DATA SET

- real order flow of major US broker-dealer
- 12 stocks
- 4 dark pools

EXPERIMENTS

- Simple parametric model for liquidity distribution
 - explicit probability of zero liquidity
 - Power Law for the rest

EMPIRICAL RESULTS

DATA SET

- real order flow of major US broker-dealer
- 12 stocks
- 4 dark pools

EXPERIMENTS

- Simple parametric model for liquidity distribution
 - explicit probability of zero liquidity
 - Power Law for the rest
- Simulator based on real data

SIMULATION RESULTS

COMPARE 4 STRATEGIES

- our algorithm
- naive bandits
- uniform allocation
- ideal allocation

* cheating *

SIMULATION RESULTS

COMPARE 4 STRATEGIES

- our algorithm
- naive bandits *similar to real-world strategy *
- uniform allocation
- ideal allocation * cheating *

SIMULATION RESULTS

COMPARE 4 STRATEGIES

- our algorithm
- naive bandits *similar to real-world strategy *
- uniform allocation
- ideal allocation * cheating *

NAIVE BANDITS

- maintain weight for each venue
- distribute proportional to weight
- increase weights when trade happens

SIMULATION RESULTS

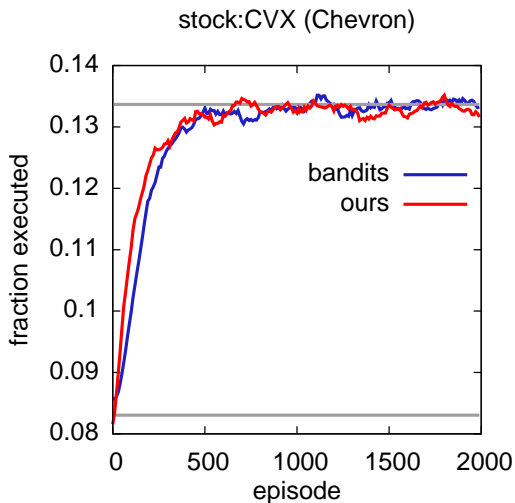
COMPARE 4 STRATEGIES

- our algorithm
- naive bandits *similar to real-world strategy *
- uniform allocation
- ideal allocation * cheating *

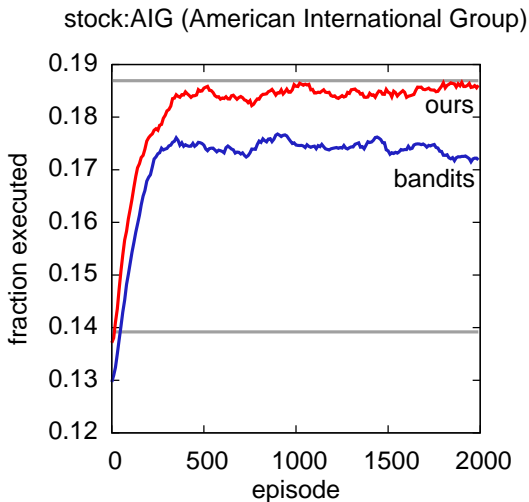
NAIVE BANDITS * 0/1 liquidity model! *

- maintain weight for each venue
- distribute proportional to weight
- increase weights when trade happens

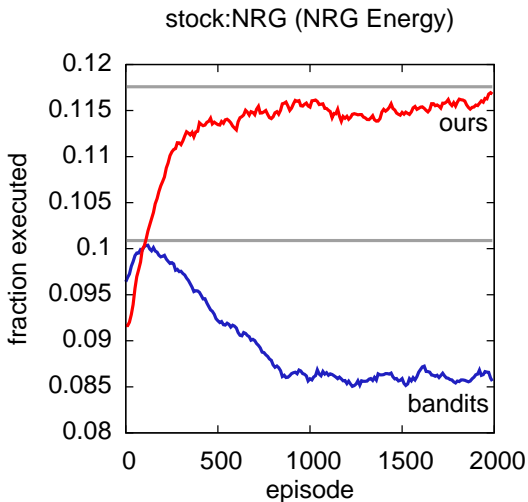
SAMPLE LEARNING CURVES



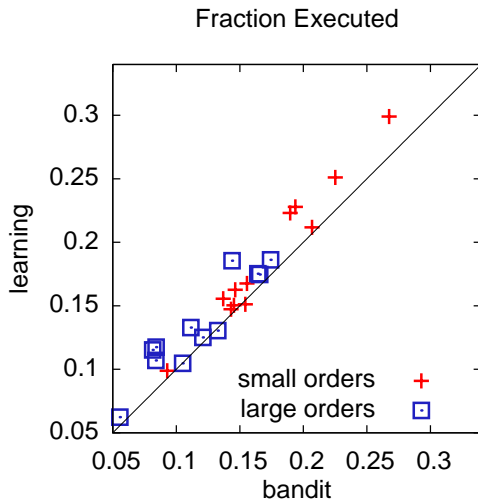
SAMPLE LEARNING CURVES



SAMPLE LEARNING CURVES



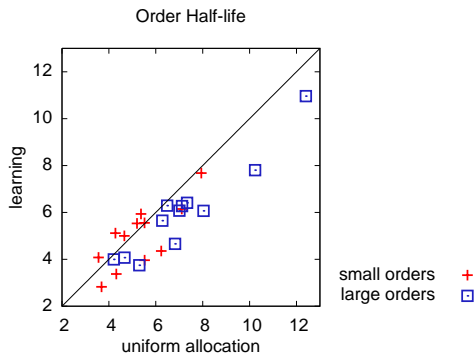
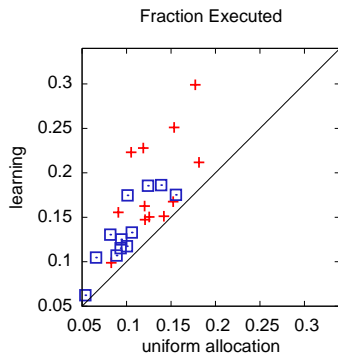
COMPARISON



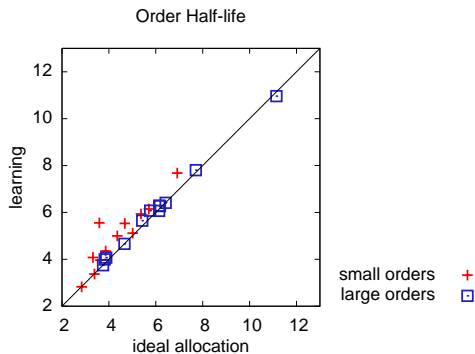
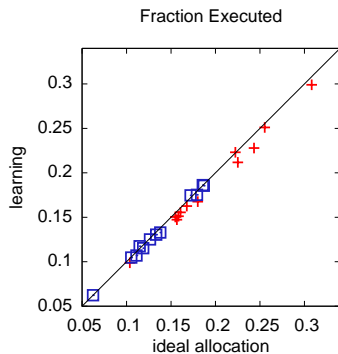
CONCLUSION

- Proposed a new algorithm for censored exploration/exploitation
- motivation from Dark Pool problem
- Theoretical analysis
- Empirical evaluation based on real data

COMPARISON



COMPARISON



COMPARISON

