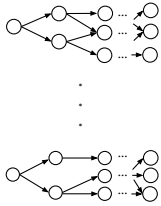
Prophet Inequalities for Bandits, Cabinets, and DAGs



Robin Bowers, Elias Lindgren, and **Bo Waggoner**
University of Colorado, Boulder

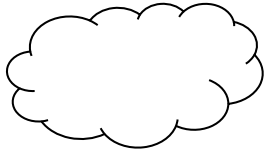
Columbia U.
May 2025

Introduction

Alternatives

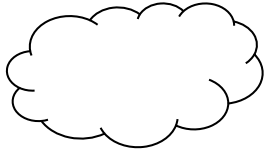
Decisionmaker

1



.
.
.

n



Potential products

Manufacturer

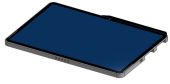
1

.

.

.

n



Potential products

1

15”

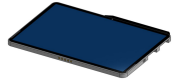
13”

fingerprint reader

none

.
.
.

n



11”

9”

second camera

none

Manufacturer

can interactively research & develop…

Potential products

1

15”

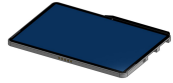
13”

fingerprint reader

none

.
.
.

n



11”

9”

second camera

none

Manufacturer

can interactively research & develop…

Potential products

1

15”

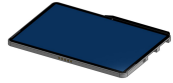
13”

fingerprint reader

none

.
.
.

n



11”

9”

second camera

none

Manufacturer

can interactively research & develop…

Potential products

1

fingerprint reader

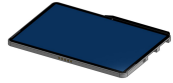
15”

none

13”

.
.
.

n



second camera

11”

none

9”

Manufacturer

can interactively research & develop…

Potential products

1

fingerprint reader

15”

none

13”

complete

.
.
.

n

second camera

11”

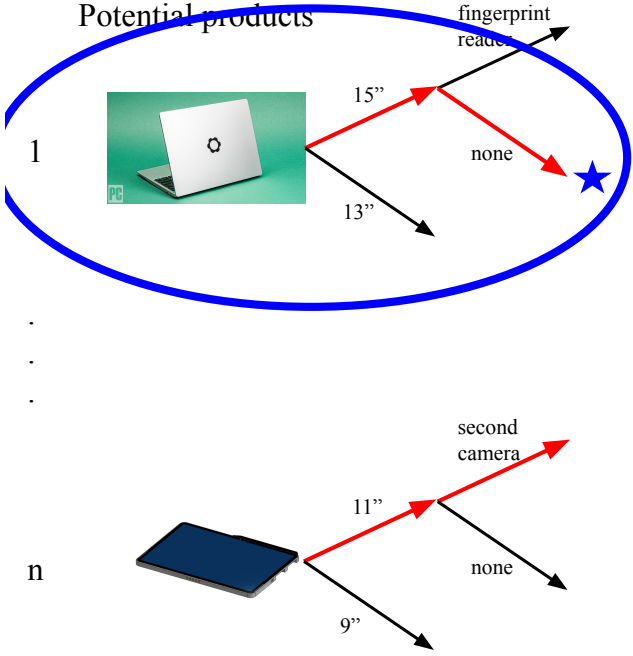
none

9”

complete

Manufacturer

can interactively research & develop…



Manufacturer

can interactively research & develop…

…and eventually produce **one**.

Potential products

fingerprint reader

15”

1

Manufacturer

Challenge:

- Significantly generalizes NP-hard problems
- Variant of 50-year open problem

Questions for today:

- What can be achieved in polynomial time?
- Can limited adaptivity ever suffice?

.
.
.

nteractively
rch &
op…

…and eventually produce **one**.

11”

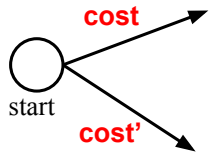
n

none

9”

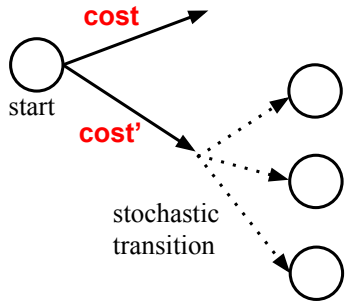
- **Model, related work**
- Results
- Methods: adaptivity
- Methods: polynomial-time results
- Conclusion

Markov Search Process: definition



- Begin in the “start” state
- Decide among actions with cost C(state, action)

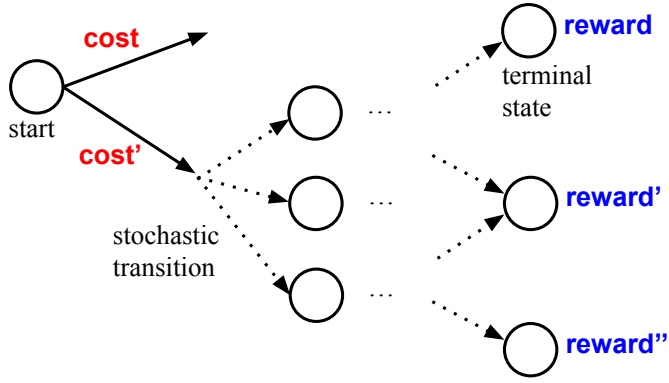
Markov Search Process: definition



- Begin in the “start” state
- Decide among actions with cost C(state, action)
- Undergo stochastic transition P(state, action)

DAG structure

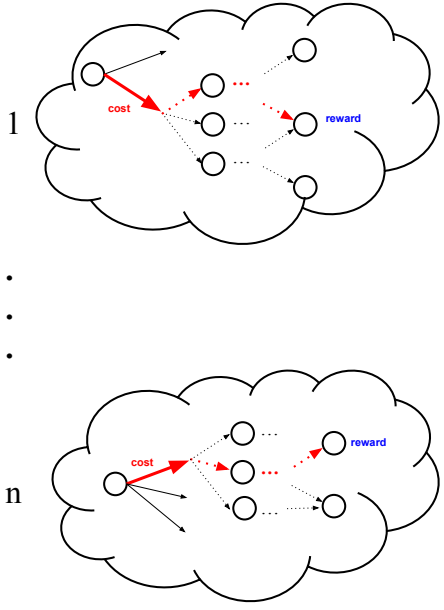
Markov Search Process: definition



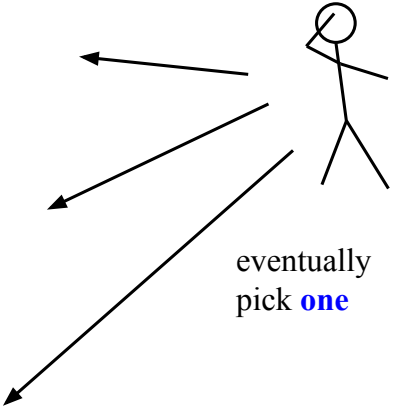
- Begin in the "start" state
- Decide among actions with cost C(state, action)
- Undergo stochastic transition P(state, action) *DAG structure*
- …
- Reach a terminal state with an *available* reward.

Markov Search Processes

independent, known structure



Decisionmaker



eventually
pick **one**

Formal (ish) problem statement

Combinatorial Markov Search:

- Given n Markov Search Processes[1] $\mathcal{M}_1, \ldots, \mathcal{M}_n$ *all parameters are known*
- Interact arbitrarily, incurring all costs
- Eventually halt and claim some i
  $\rightarrow$ Receive reward reached in $\mathcal{M}_i$
- Maximize expected reward - costs
- Benchmark: fully adaptive optimal algorithm.

[1] each $\mathcal{M}_i = (S_i, A_I, P_i, C_i, V_i) =$ (States, Actions, Transition Function, Costs, Rewards).

Formal (ish) problem statement

Combinatorial Markov Search:

- Given n Markov Search Processes[1] $\mathcal{M}_1, \ldots, \mathcal{M}_n$ *all parameters are known*
- Interact arbitrarily, incurring all costs
- Eventually halt and claim **a feasible subset** $F \in \mathcal{F}$ *$\mathcal{F} \subseteq 2^n$ downward-closed*
  $\rightarrow$ Receive rewards reached in **each** $\mathcal{M}_i$ **where** $i \in F$
- Maximize expected **sum of rewards** - costs
- Benchmark: fully adaptive optimal algorithm.

[1] each $\mathcal{M}_i = (S_i, A_I, P_i, C_i, V_i) =$ (States, Actions, Transition Function, Costs, Rewards).

Formal (ish) problem statement

Combinatorial Markov Search:

- Given n Markov Search Processes[1] $\mathcal{M}_1, \ldots, \mathcal{M}_n$ *all parameters are known*
- Interact arbitrarily, incurring all costs
- Eventually halt and claim **a feasible subset** $F \in \mathcal{F}$ $\mathcal{F} \subseteq 2^n$ *downward-closed*
  $\rightarrow$ Receive rewards reached in **each** $\mathcal{M}_i$ **where** $i \in F$
- Maximize expected **sum of rewards** - costs
- Benchmark: fully adaptive optimal algorithm.

Questions:

- What can be achieved in **polynomial time**?

[1] each $\mathcal{M}_i = (S_i, A_I, P_i, C_i, V_i) =$ (States, Actions, Transition Function, Costs, Rewards).

Formal (ish) problem statement

Combinatorial Markov Search:

- Given n Markov Search Processes[1] $\mathcal{M}_1, \ldots, \mathcal{M}_n$ *all parameters are known*
- Interact arbitrarily, incurring all costs
- Eventually halt and claim **a feasible subset** $F \in \mathcal{F}$ *$\mathcal{F} \subseteq 2^n$ downward-closed*
  $\rightarrow$ Receive rewards reached in **each** $\mathcal{M}_i$ **where** $i \in F$
- Maximize expected **sum of rewards** - costs
- Benchmark: fully adaptive optimal algorithm.

Questions:

- What can be achieved in **polynomial time**?
- What level of **adaptivity** is necessary?

[1] each $\mathcal{M}_i = (S_i, A_I, P_i, C_i, V_i)$ = (States, Actions, Transition Function, Costs, Rewards).

Formal (ish) problem statement

Combinatorial Markov Search:

- Given n Markov Search Processes[1] $\mathcal{M}_1, \ldots, \mathcal{M}_n$ *all parameters are known*
- Interact arbitrarily, incurring all costs
- Eventually halt and claim **a feasible subset** $F \in \mathcal{F}$ *$\mathcal{F} \subseteq 2^n$ downward-closed*
  $\rightarrow$ Receive rewards reached in **each** $\mathcal{M}_i$ **where** $i \in F$
- Maximize expected **sum of rewards** - costs
- Benchmark: fully adaptive optimal algorithm.

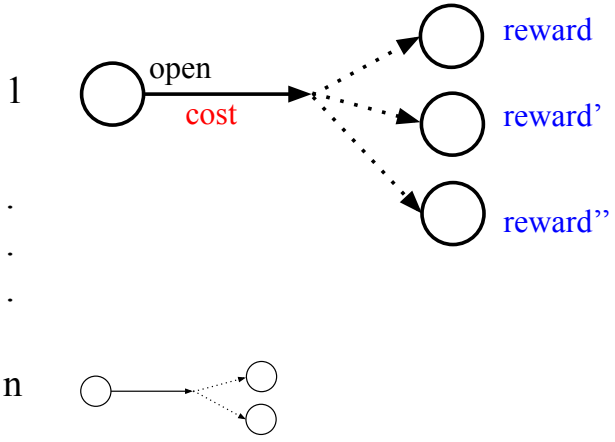
Questions:

- What can be achieved in **polynomial time**?
- What level of **adaptivity** is necessary?
- (bonus) What if each MSP is controlled by a strategic agent?

[1] each $\mathcal{M}_i = (S_i, A_I, P_i, C_i, V_i)$ = (States, Actions, Transition Function, Costs, Rewards).

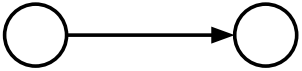
Prior work: Pandora's Box

Weitzman 1979:



Prior work: Pandora's Box

Weitzman 1979:

1 

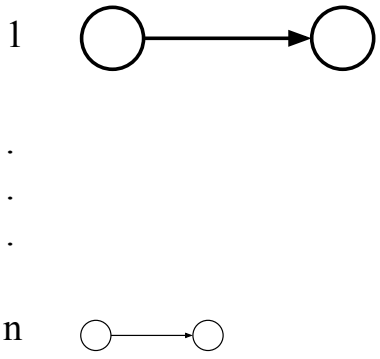
.
.
.

n 

From now on: omit stochastic transitions, only draw decision structure.

Prior work: Pandora's Box

Weitzman 1979: **index theorem**, polytime optimal.

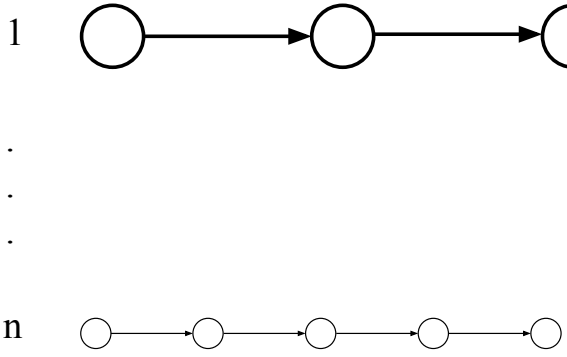


Index theorem:

- Index is a function only of a box
- Global policy as function of indices is **optimal**

Prior work: Multistage Pandora's Box

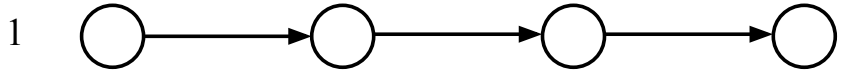
Multistage or "nested" Pandora's Box:[2]: **index theorem**, polytime optimal.



[2]Dumitriua et al. 2003; Guha and Munagala 2008; Kleinberg et al. 2016; Gupta et al. 2019; Bowers and Waggoner 2024; Chawla et al. 2024

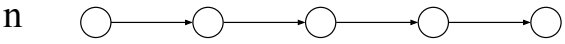
Prior work: Multistage Pandora's Box

Multistage or "nested" Pandora's Box:[2]: **index theorem**, polytime optimal.



.
.
.

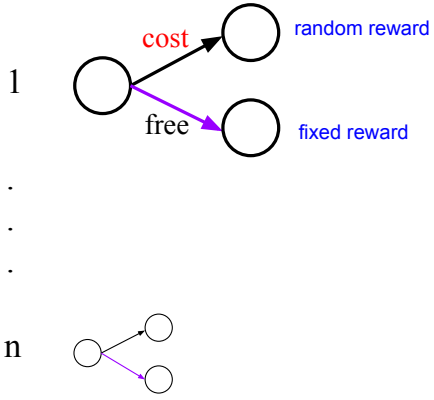
"**Bandit** Markov Search Processes" or "**bandits**"



[2]Dumitriua et al. 2003; Guha and Munagala 2008; Kleinberg et al. 2016; Gupta et al. 2019; Bowers and Waggoner 2024; Chawla et al. 2024

Prior work: nonobligatory Pandora's Box

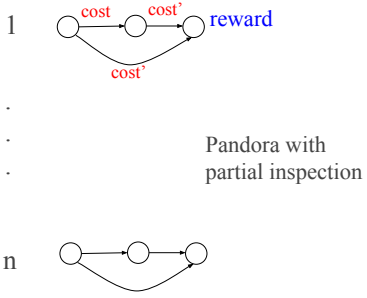
Pandora's Box with nonobligatory inspection[3]: NP-hard, PTAS



[3]Doval 2018; Beyhaghi and Kleinberg 2018; Beyhaghi and Cai 2023; Fu et al. 2023

More general MSP structures: highly open

Pandora's Box with partial inspection:[4] constant-factor approx.



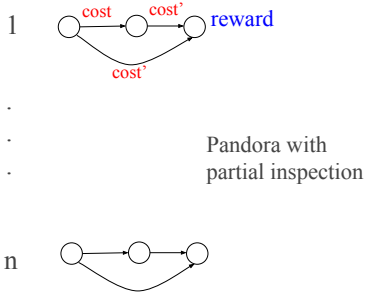
[4] Aouad et al. 2020; Chawla et al. 2024 (indep.)
[5] Doval and Scully 2024; Chawla et al. 2024 (indep.)

More general MSP structures: highly open

Pandora's Box with partial inspection:[4] constant-factor approx.

Local hedging technique:[5] extends Whittle condition for some MSPs

logarithmic approx for weighing scale problem

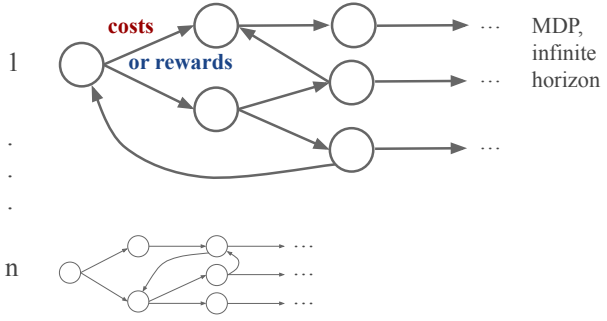


[4] Aouad et al. 2020; Chawla et al. 2024 (indep.)

[5] Doval and Scully 2024; Chawla et al. 2024 (indep.)

Related: superprocesses, Bayesian Bandits

Superprocess[6]*: analogue with MDPs where we* ***never stop and claim***



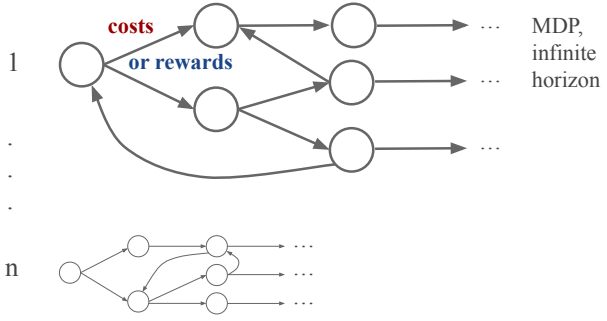
[6] *P. Nash 1973; Gittins 1979; Whittle 1980; Glazebrook 1982, 1993; Ma 2018*

Related: superprocesses, Bayesian Bandits

Superprocess[6]*: analogue with MDPs where we* ***never stop and claim***

If each is a bandit: ***Gittins index theorem****; slightly more generally,* ***Whittle index theorem***

Without the Whittle condition: little work (see Ma 2018)



[6] *P. Nash 1973; Gittins 1979; Whittle 1980; Glazebrook 1982, 1993; Ma 2018*

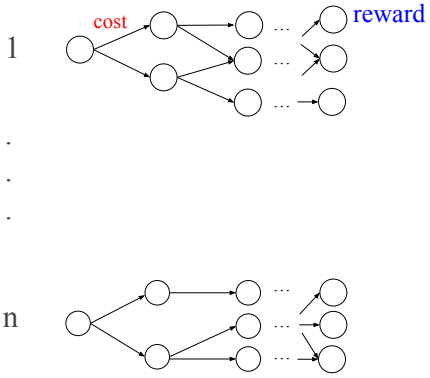
Our results

Combinatorial Markov Search: our results

Combinatorial Markov Search: our results

Approach: online algorithms.

- Face $\mathcal{M}_1, \ldots, \mathcal{M}_n$ one by one in arbitrary order.
- For each, "take it or leave it" before moving on.



Combinatorial Markov Search: our results

Approach: online algorithms.

- Face $\mathcal{M}_1, \ldots, \mathcal{M}_n$ one by one in arbitrary order.
- For each, "take it or leave it" before moving on.

Prophet inequality[7]:

- *special case where each $\mathcal{M}_i$ is just a random reward.*
- *constant-factor approx, namely $\frac{1}{2}$, including with matroid constraints.*

[7] *Samuel-Cahn 1986; Kleinberg, Weinberg 2012*

Combinatorial Markov Search: our results

Approach: online algorithms.

- Face $\mathcal{M}_1, \ldots, \mathcal{M}_n$ one by one in arbitrary order.
- For each, "take it or leave it" before moving on.

Prophet inequality[7]:

- *special case where each $\mathcal{M}_i$ is just a random reward.*
- *constant-factor approx, namely $\frac{1}{2}$, including with matroid constraints.*

First question: ignoring efficiency, when can online algs yield nontrivial results?

[7] *Samuel-Cahn 1986; Kleinberg, Weinberg 2012*

Combinatorial Markov Search: our results

Approach: online algorithms.

- Face $\mathcal{M}_1, \ldots, \mathcal{M}_n$ one by one in arbitrary order.
- For each, "take it or leave it" before moving on.

Theorem (Adaptivity gap, inefficiently)

Combinatorial Markov Search: our results

Approach: online algorithms.

- Face $\mathcal{M}_1, \ldots, \mathcal{M}_n$ one by one in arbitrary order.
- For each, "take it or leave it" before moving on.

Theorem (Adaptivity gap, inefficiently)

For arbitrary MSPs and matroid constraints, there exists an online $\frac{1}{2}$ approximation.

Combinatorial Markov Search: our results

Approach: online algorithms.

- Face $\mathcal{M}_1, \ldots, \mathcal{M}_n$ one by one in arbitrary order.
- For each, "take it or leave it" before moving on.

Theorem (Adaptivity gap, inefficiently)

For arbitrary MSPs and matroid constraints, there exists an online $\frac{1}{2}$ approximation.

Theorem (Polytime approx)

Combinatorial Markov Search: our results

Approach: online algorithms.

- Face $\mathcal{M}_1, \ldots, \mathcal{M}_n$ one by one in arbitrary order.
- For each, "take it or leave it" before moving on.

Theorem (Adaptivity gap, inefficiently)

For arbitrary MSPs and matroid constraints, there exists an online $\frac{1}{2}$ approximation.

Theorem (Polytime approx)

For arbitrary MSPs and matroid constraints, there exists an online poly(input, $\frac{1}{\epsilon}$)-time algorithm that, for any $\epsilon > 0$, guarantees a $\frac{1}{2} - \epsilon$ approximation.

Adaptivity gap

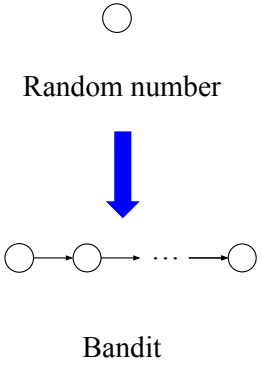
Prophet inequality reductions

Consider our problem under special cases of the Markov Search Process structure:

Random number

Prophet inequality reductions

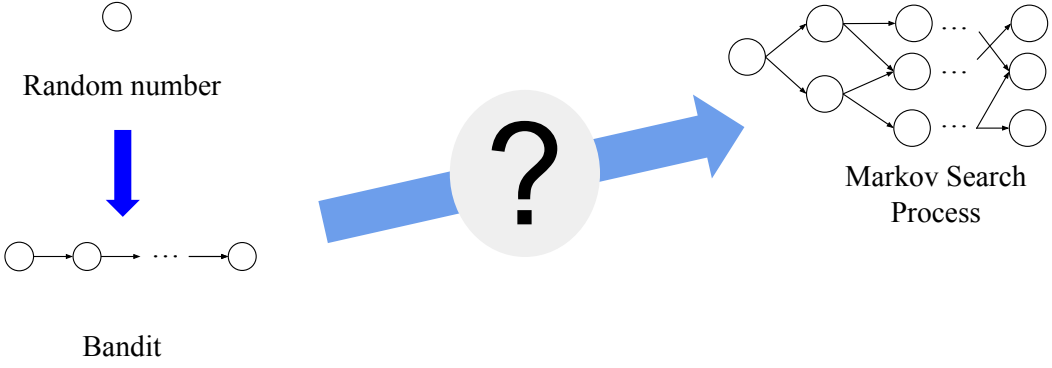
Consider our problem under special cases of the Markov Search Process structure:



Weitzman with matroid constraints and/or bandits: Singla 2018; Esfandiari 2019; Gupta et al. 2019

Prophet inequality reductions

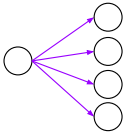
Consider our problem under special cases of the Markov Search Process structure:



Prophet inequality reductions

Consider our problem under special cases of the Markov Search Process structure:

Random number



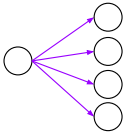
Cabinet

no costs, i.e. open drawers for free

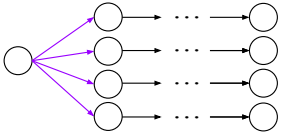
Prophet inequality reductions

Consider our problem under special cases of the Markov Search Process structure:

Random number



Cabinet

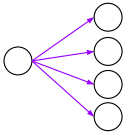


Bandits in Cabinet

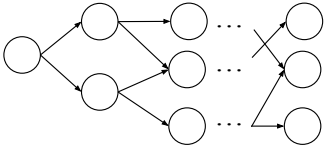
Prophet inequality reductions

Consider our problem under special cases of the Markov Search Process structure:

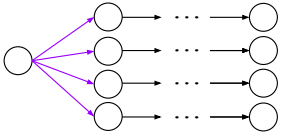
Random number



Cabinet



Markov Search Process

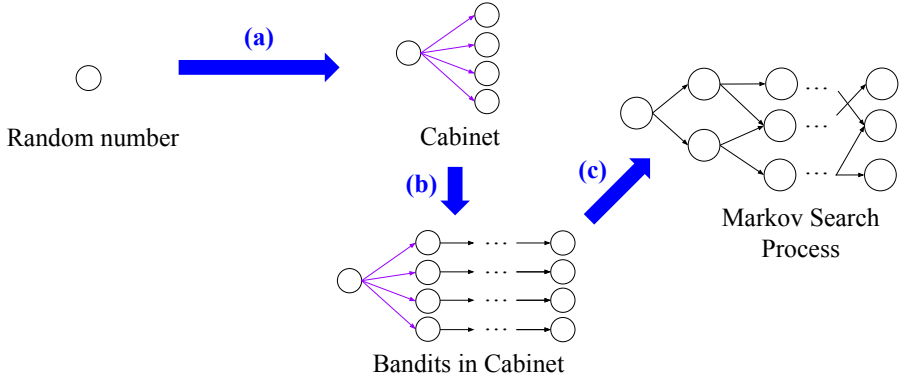


Bandits in Cabinet

Combinatorial Markov Search: main result 1

Theorem (Adaptivity gap)

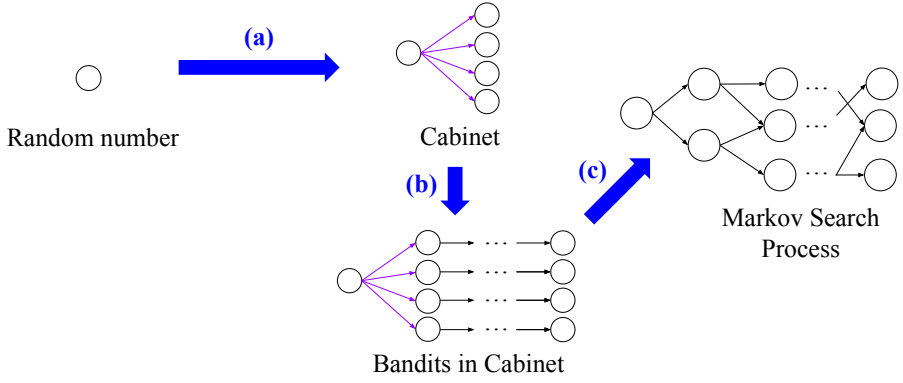
If $\mathcal{F}$ is downward-closed and there exists an α prophet inequality for $\mathcal{F}$, then there is an online α-approx for CMS where the alternatives are: (a) Cabinets, (b) Bandits in Cabinets, (c) arbitrary Markov Search Processes.



Combinatorial Markov Search: main result 1

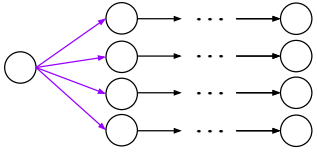
Theorem (Adaptivity gap)

If $\mathcal{F}$ is downward-closed and there exists an α prophet inequality for $\mathcal{F}$ ***to the ex-ante relaxation****, then there is an online α-approx for CMS where the alternatives are: (a) Cabinets, (b) Bandits in Cabinets, (c) arbitrary Markov Search Processes.*

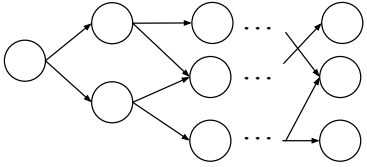


Last things first

Lemma (c): Online α-approx for Bandits in Cabinets $\implies$ online α-approx for arbitrary MSPs.



Bandits in cabinets

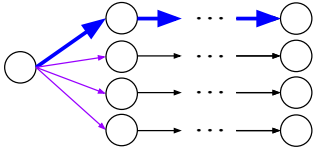


Markov Search Processes

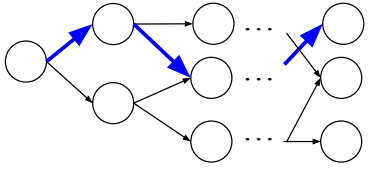
Last things first

Lemma (c): Online α-approx for Bandits in Cabinets $\implies$ online α-approx for arbitrary MSPs.

Idea[7]*: each fixed policy in the MSP induces a bandit.* *exponentially many*



Bandits in cabinets



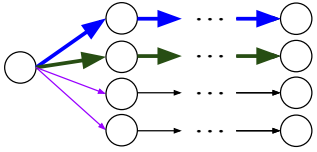
Markov Search Processes

[7] Chawla et al. 2024, independently

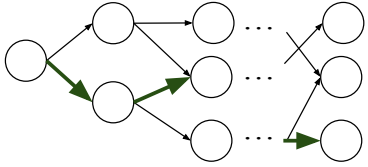
Last things first

Lemma (c): Online α-approx for Bandits in Cabinets $\implies$ online α-approx for arbitrary MSPs.

Idea: each fixed policy in the MSP induces a bandit. *exponentially many*



Bandits in cabinets



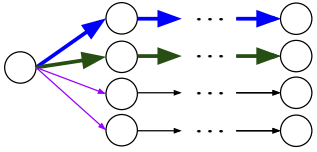
Markov Search Processes

Last things first

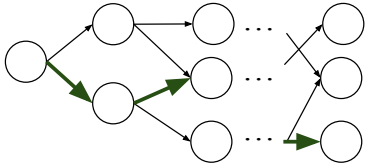
Lemma (c): Online α-approx for Bandits in Cabinets $\implies$ online α-approx for arbitrary MSPs.

Idea: each fixed policy in the MSP induces a bandit. *exponentially many*

Lemma: $\mathsf{OPT}(\mathcal{M}_i) = \mathsf{OPT}(\mathsf{cabinet}_i)$.



Bandits in cabinets



Markov Search Processes

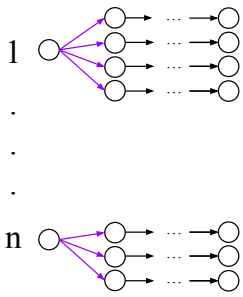
Last things first

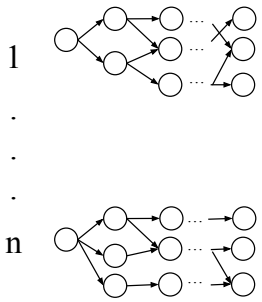
Lemma (c): Online α-approx for Bandits in Cabinets $\implies$ online α-approx for arbitrary MSPs.

Idea: each fixed policy in the MSP induces a bandit. *exponentially many*

Lemma: $\mathsf{OPT}(\mathcal{M}_i) = \mathsf{OPT}(\mathsf{cabinet}_i)$.

Lemma: $\mathsf{OPT}(\mathcal{M}_1, \ldots, \mathcal{M}_n) \leq$?





Last things first

Lemma (c): Online α-approx for Bandits in Cabinets $\implies$ online α-approx for arbitrary MSPs.

Idea: each fixed policy in the MSP induces a bandit. *exponentially many*

Lemma: $\mathsf{OPT}(\mathcal{M}_i) = \mathsf{OPT}(\mathsf{cabinet}_i)$.

Lemma: $\mathsf{OPT}(\mathcal{M}_1, \ldots, \mathcal{M}_n) \leq \text{ex-ante-}\mathsf{OPT}(\mathsf{cabinet}_1, \ldots, \mathsf{cabinet}_n)$.

Last things first

Lemma (c): Online α-approx for Bandits in Cabinets $\implies$ online α-approx for arbitrary MSPs.

Idea: each fixed policy in the MSP induces a bandit. *exponentially many*

Lemma: $\mathsf{OPT}(\mathcal{M}_i) = \mathsf{OPT}(\mathsf{cabinet}_i)$.

Lemma: $\mathsf{OPT}(\mathcal{M}_1, \ldots, \mathcal{M}_n) \leq \text{ex-ante-}\mathsf{OPT}(\mathsf{cabinet}_1, \ldots, \mathsf{cabinet}_n)$.

ex-ante feasible: $\Pr[\textit{choose } 1] + \cdots + \Pr[\textit{choose } n] \leq 1$

Last things first

Lemma (c): Online α-approx for Bandits in Cabinets $\implies$ online α-approx for arbitrary MSPs.

Idea: each fixed policy in the MSP induces a bandit. *exponentially many*

Lemma: $\mathsf{OPT}(\mathcal{M}_i) = \mathsf{OPT}(\mathsf{cabinet}_i)$.

Lemma: $\mathsf{OPT}(\mathcal{M}_1, \ldots, \mathcal{M}_n) \leq \text{ex-ante-}\mathsf{OPT}(\mathsf{cabinet}_1, \ldots, \mathsf{cabinet}_n)$.

ex-ante feasible: $(\Pr[1 \in F], \ldots, \Pr[n \in F]) \in \mathit{conv}(\{1_S : S \in \mathcal{F}\})$

Last things first

Lemma (c): Online α-approx for Bandits in Cabinets $\implies$ online α-approx for arbitrary MSPs.

Idea: each fixed policy in the MSP induces a bandit. *exponentially many*

Lemma: $\mathsf{OPT}(\mathcal{M}_i) = \mathsf{OPT}(\mathsf{cabinet}_i)$.

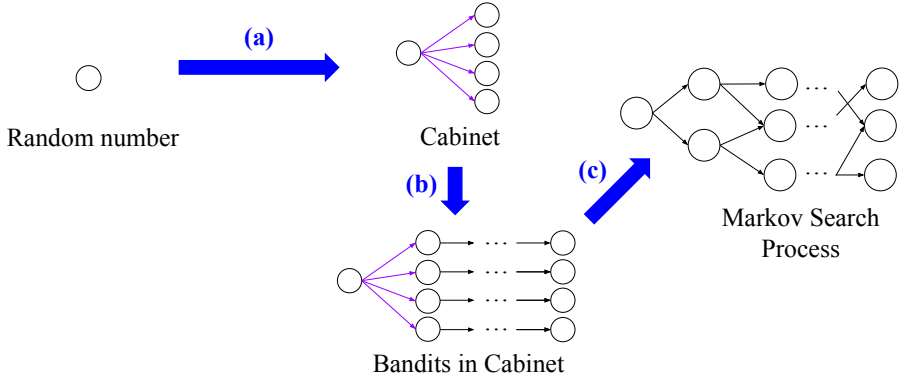
Lemma: $\mathsf{OPT}(\mathcal{M}_1, \ldots, \mathcal{M}_n) \leq \text{ex-ante-}\mathsf{OPT}(\mathsf{cabinet}_1, \ldots, \mathsf{cabinet}_n)$.

Lemma: $\mathsf{ALG}(\mathcal{M}_1, \ldots, \mathcal{M}_n) = \mathsf{ALG}(\mathsf{cabinet}_1, \ldots, \mathsf{cabinet}_n)$. □

Reminder: reductions

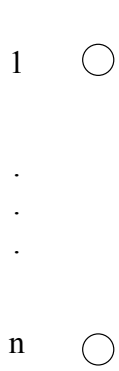
Theorem (Adaptivity gap)

If $\mathcal{F}$ is a downward-closed constraint and there exists an α prophet inequality for $\mathcal{F}$ to the ex-ante relaxation, then there is an online α-approx for CMS where the alternatives are: (a) Cabinets, (b) Bandits in Cabinets, (c) arbitrary Markov Search Processes.

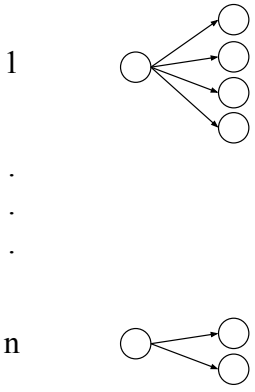


From Prophets to Cabinets-Prophets

Lemma (a): ex-ante Prophet Inequality $\alpha \implies$ ex-ante online α-approx for Cabinets.



Prophet inequality

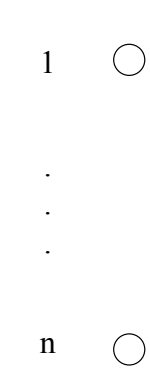


Cabinets

Note: there exists a matroid prophet inequality of $\frac{1}{2}$ to ex-ante-OPT (Lee, Singla 2018).

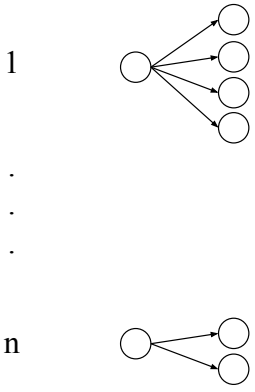
From Prophets to Cabinets-Prophets

Lemma (a): ex-ante Prophet Inequality $\alpha \implies$ ex-ante online α-approx for Cabinets.



Prophet inequality

Observation: ex-ante-OPT is nonadaptive! (WLOG)

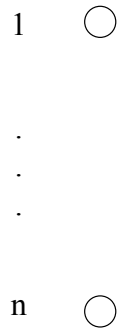


Cabinets

Note: there exists a matroid prophet inequality of $\frac{1}{2}$ to ex-ante-OPT (Lee, Singla 2018).

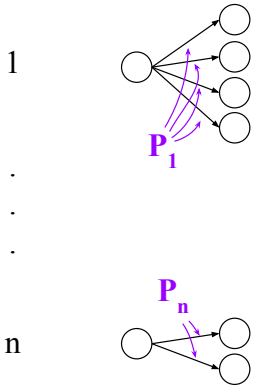
From Prophets to Cabinets-Prophets

Lemma (a): ex-ante Prophet Inequality $\alpha \implies$ ex-ante online α-approx for Cabinets.



Prophet inequality

- ex-ante-OPT decides $\sim \mathbf{P_i}$ non-adaptively

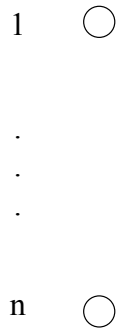


Cabinets

Note: there exists a matroid prophet inequality of $\frac{1}{2}$ to ex-ante-OPT (Lee, Singla 2018).

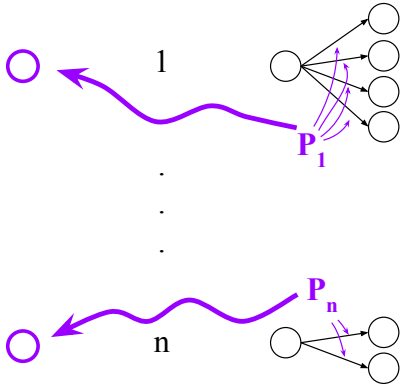
From Prophets to Cabinets-Prophets

Lemma (a): ex-ante Prophet Inequality $\alpha \implies$ ex-ante online α-approx for Cabinets.



Prophet inequality

- ex-ante-OPT decides $\sim \mathbf{P_i}$ non-adaptively
- so it sees a set of independent random vars

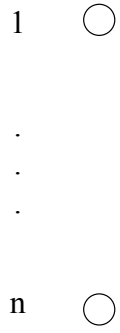


Cabinets

Note: there exists a matroid prophet inequality of $\frac{1}{2}$ to ex-ante-OPT (Lee, Singla 2018).

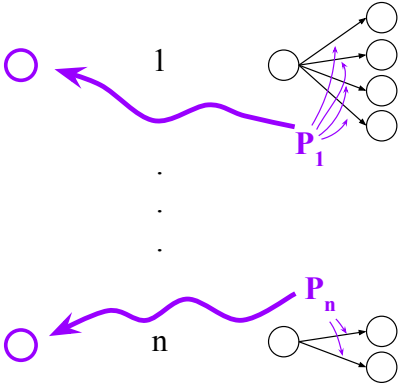
From Prophets to Cabinets-Prophets

Lemma (a): ex-ante Prophet Inequality $\alpha \implies$ ex-ante online α-approx for Cabinets.



Prophet inequality

- ex-ante-OPT decides $\sim \mathbf{P_i}$ non-adaptively
- so it sees a set of independent random vars
- Reduction: we also decide $\sim \mathbf{P_i}$ and apply prophets alg



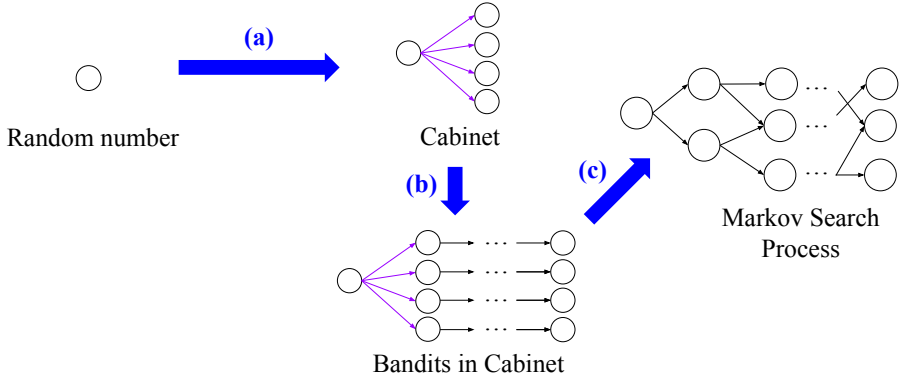
Cabinets

Note: there exists a matroid prophet inequality of $\frac{1}{2}$ to ex-ante-OPT (Lee, Singla 2018).

Reminder: reductions (redux)

Theorem (Adaptivity gap)

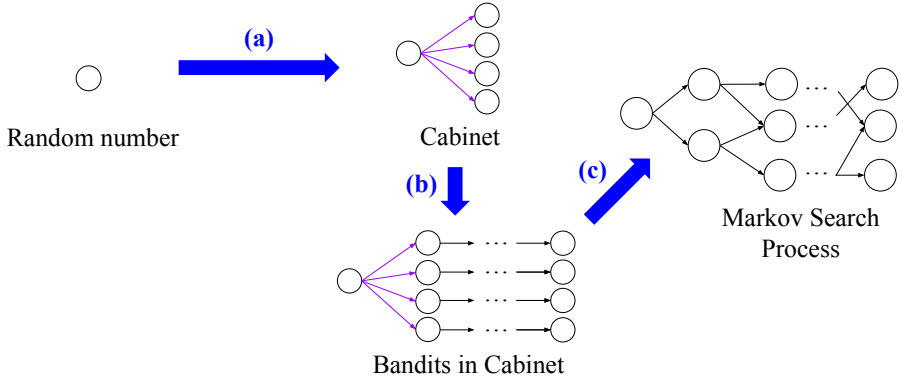
If $\mathcal{F}$ is a downward-closed constraint and there exists an α prophet inequality for $\mathcal{F}$ to the ex-ante relaxation, then there is an online α-approx for CMS where the alternatives are: (a) Cabinets, (b) Bandits in Cabinets, (c) arbitrary Markov Search Processes.



Polynomial-time approximation

Toward polynomial time

Problem: reduction **(c)** is exponential

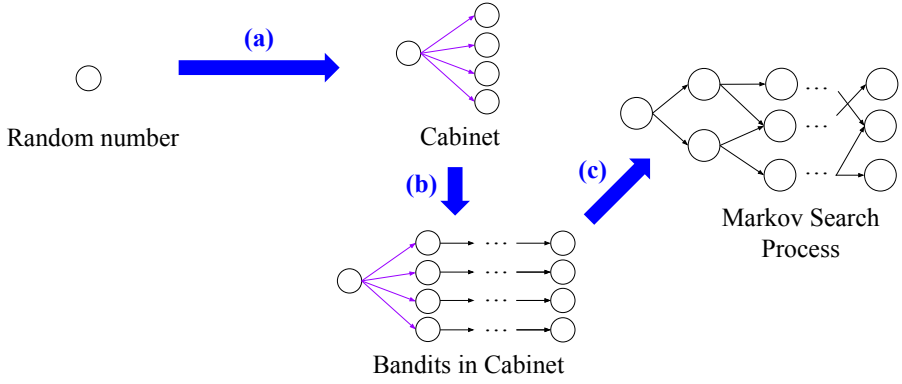


Toward polynomial time

Problem: reduction **(c)** is exponential

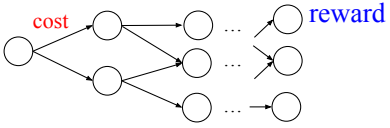
Solution:

- Modify prophet inequality **(a)** to interface with an oracle
- Implement the oracle for **(c)**



An oracle for Markov Search Processes

Single-Agent Utility Problem, SAUP$(\mathcal{M}, T)$:
interact with $\mathcal{M}$, take-it-or-leave-it, assuming a price of T for claiming a reward.

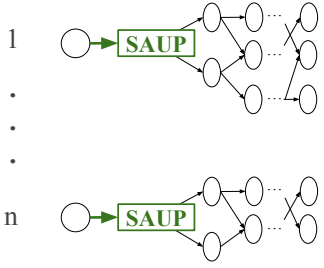


must pay T to
claim reward

An oracle for Markov Search Processes

Single-Agent Utility Problem, SAUP$(\mathcal{M}, T)$:
interact with $\mathcal{M}$, take-it-or-leave-it, assuming a price of T for claiming a reward.

SAUP-based online algorithm:
for each arrival i, set a threshold T_i, then optimally solve SAUP$(\mathcal{M}_i, T_i)$.

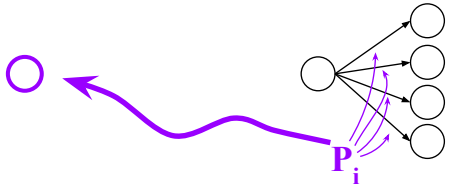


ALG: set thresholds T_i with custom prophet inequality; run SAUP on each arrival.

An oracle for Markov Search Processes

Single-Agent Utility Problem, SAUP$(\mathcal{M}, T)$:
interact with $\mathcal{M}$, take-it-or-leave-it, assuming a price of T for claiming a reward.

SAUP-based online algorithm:
for each arrival i, set a threshold T_i, then optimally solve SAUP$(\mathcal{M}_i, T_i)$.

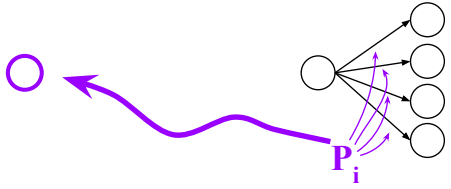


Not SAUP

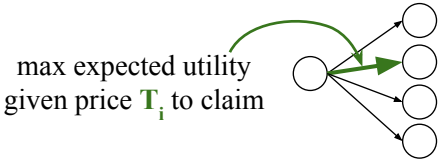
An oracle for Markov Search Processes

Single-Agent Utility Problem, SAUP$(\mathcal{M}, T)$:
interact with $\mathcal{M}$, take-it-or-leave-it, assuming a price of T for claiming a reward.

SAUP-based online algorithm:
for each arrival i, set a threshold T_i, then optimally solve SAUP$(\mathcal{M}_i, T_i)$.



Not SAUP



SAUP

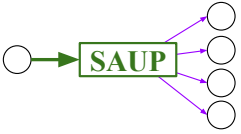
Polytime approx for Combinatorial Markov Search

Proposition

For arbitrary MSPs and any matroid constraint:

- *There is an online, polytime, SAUP-based, $\frac{1}{2}$-approx for Cabinets.* to ex-ante-OPT

$\rightarrow$ *we roll an original proof based on Lee and Singla 2018.*



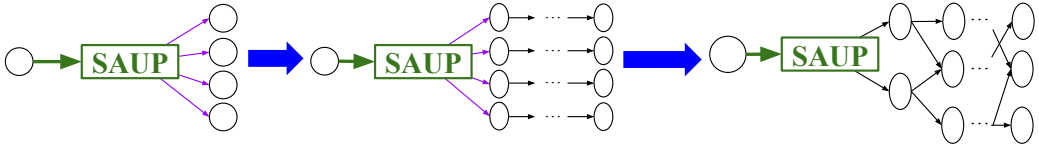
Polytime approx for Combinatorial Markov Search

Proposition

For arbitrary MSPs and any matroid constraint:

- *There is an online, polytime, SAUP-based, $\frac{1}{2}$-approx for Cabinets.* to ex-ante-OPT

$\rightarrow$ *we roll an original proof based on Lee and Singla 2018.*

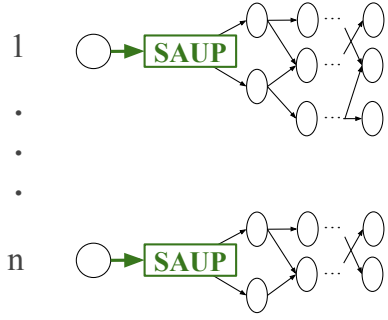


Polytime approx for Combinatorial Markov Search

Proposition

For arbitrary MSPs and any matroid constraint:

- *There is an online, polytime, SAUP-based, $\frac{1}{2}$-approx for Cabinets.* to ex-ante-OPT
- ***There is a polynomial-time algorithm for SAUP**$(\mathcal{M}_i, T_i)$.*



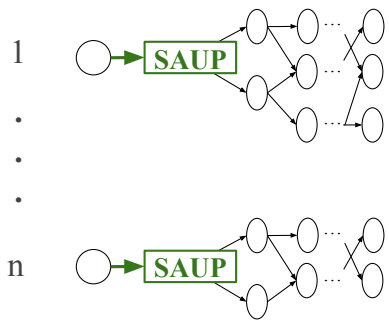
ALG: set thresholds T_i with custom prophet inequality; run SAUP on each arrival.

Polytime approx for Combinatorial Markov Search

Proposition

For arbitrary MSPs and any matroid constraint:

- *There is an online, polytime, SAUP-based, $\frac{1}{2}$-approx for Cabinets.* to ex-ante-OPT
- *There is a polynomial-time algorithm for SAUP$(\mathcal{M}_i, T_i)$.*
- ***There is an FPTAS for ex-ante-OPT**$(\mathcal{M}_1, \ldots, \mathcal{M}_n)$.* allows computing T_i



ALG: set thresholds T_i with custom prophet inequality; run SAUP on each arrival.

Polytime approx for Combinatorial Markov Search

Proposition

For arbitrary MSPs and any matroid constraint:

- *There is an online, polytime, SAUP-based, $\frac{1}{2}$-approx for Cabinets.* to ex-ante-OPT
- *There is a polynomial-time algorithm for SAUP$(\mathcal{M}_i, T_i)$.*
- *There is an FPTAS for ex-ante-OPT$(\mathcal{M}_1, \ldots, \mathcal{M}_n)$.* allows computing T_i

Theorem (Main result)

For CMS with any matroid constraint, there is an online poly(input, $\frac{1}{\epsilon}$)-time algorithm that, for any $\epsilon > 0$, guarantees a $\frac{1}{2} - \epsilon$ approx. uses add'l trick for ϵ

Bonus: strategic agents

Suppose:

- Each agent $i = 1, \ldots, n$ controls a Markov Search Process $\mathcal{M}_i$
- Mechanism can select any feasible subset of agents as "winners"
- i can collect the reward from $\mathcal{M}_i$ iff they are selected

Bonus: strategic agents

Suppose:

- Each agent $i = 1, \ldots, n$ controls a Markov Search Process $\mathcal{M}_i$
- Mechanism can select any feasible subset of agents as "winners"
- i can collect the reward from $\mathcal{M}_i$ iff they are selected

Corollary

For matroids, this problem admits a Price of Anarchy of $\frac{1}{2}$.

$\rightarrow$ *We sequentially offer agents prices (thresholds) from our algorithm; they run SAUP.*

Wrapup

Summary and Future Work

Summary:

- Model: n independent costly search processes, followed by selection of rewards
- Surprise 1: non-adaptive $\frac{1}{2}$ approx
- Surprise 2: polytime $\frac{1}{2} - \epsilon$ approx
- Bonus: Price of Anarchy $\frac{1}{2}$

Summary and Future Work

Summary:

- Model: n independent costly search processes, followed by selection of rewards
- Surprise 1: non-adaptive $\frac{1}{2}$ approx
- Surprise 2: polytime $\frac{1}{2} - \epsilon$ approx
- Bonus: Price of Anarchy $\frac{1}{2}$

Future work:

- Improved algorithms or hardness result
- Limits of "approximate index theorems" (see Chawla et al. 2025)

Conclusion