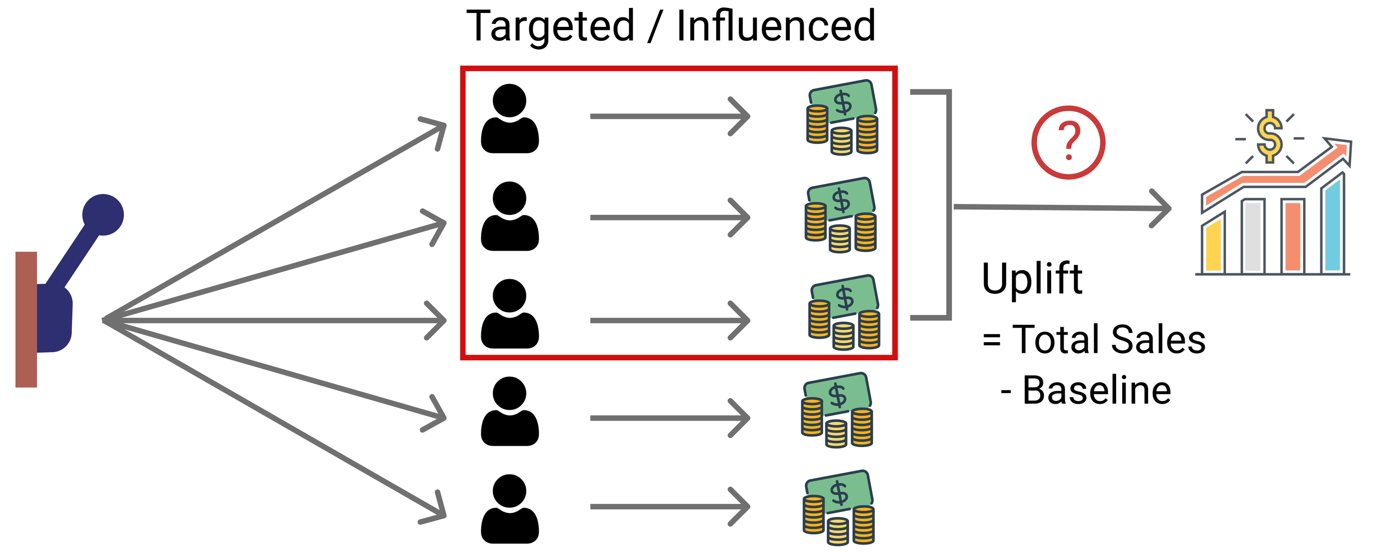
Uplifting Bandits

Yu-Guan Hsieh[1], Shiva Kasiviswanathan[2], Branislav Kveton[2] ([1]Université Grenoble Alpes [2]AWS AI Labs)

Multi-Armed Bandits and Uplift Modeling

- Multi-armed bandits **[Online]**: Learner repeatedly takes actions (pulls arms) and receives rewards from the chosen actions, with the goal of maximizing the cumulative rewards
- Uplift modelling **[Offline]** : Prediction of the incremental impact (uplift) of each action for better decision making
- In both problems, we aim to find good actions
- Applications: Marketing, Online advertisements, Clinical trials . . .

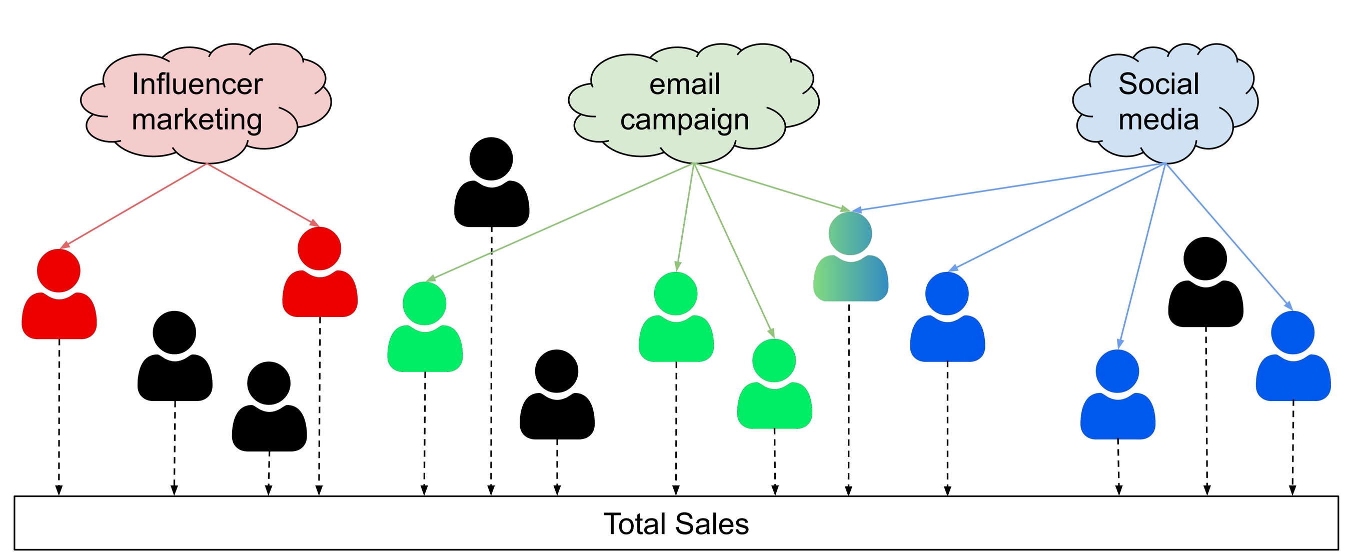
Uplifting Bandits



- Consider actions that affect the rewards through multiple intermediate variables
- The effect of each action is sparse: limited # of affected variables
- All the individual payoffs are observed

Formalisation and Motivating Example

- $\mathcal{A}$ set of K actions (marketing strategies) and $\mathcal{V}$ set of m variables (customers)
- In round t, take action a_t and observes the variables' payoffs $y_t = (y_t(i))_{i\in\mathcal{V}}$ drawn from a distribution $\mathcal{D}^{a_t}$
- Reward $r_t = \sum_{i\in\mathcal{V}} y_t(i)$ is summed over all the variables
- Each action a only affects a set $\mathcal{V}^a$ of $L^a \ll m$ variables
- The unaffected variables follow a baseline distribution $\mathcal{D}^0$



TL;DR

We introduce a new **multi-armed bandit** problem in which each action only affects the reward through **a sparse set of intermediate variables**, and show that for this problem estimating the **uplift** helps in significantly reducing the regret.

Result Overview

- Regret: performance gap between an algorithm and the algorithm that consistently takes the best action

$$\mathrm{Reg}_T = r_\star T - \sum_{t=1}^T r^{a_t} = \sum_{a\in\mathcal{A}} \underbrace{\sum_{t=1}^T \mathbb{1}\{a_t = a\}}_{N_T^a} \underbrace{(r_\star - r^a)}_{\Delta^a},$$

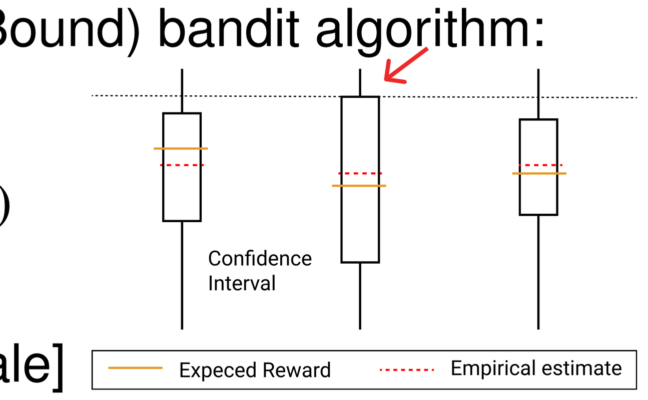
[r^a: expected reward of a; $r_\star$: highest expected reward]

- Define $L = \max_{a\in\mathcal{A}} L^a$, $\Delta = \min_{a\in\mathcal{A}} \Delta^a$, $\mu^a = \mathbb{E}_{y^a\sim\mathcal{D}^a}[y^a]$, and assume that the noise in each payoff to be 1-sub-Gaussian
- Consider various setups differing in the learner's knowledge on
  1. Baseline payoffs $\mu^0 = (\mu^0(i))_{i\in\mathcal{V}} = \mathbb{E}_{y^0\sim\mathcal{D}^0}[y^0]$
  2. The sets of affected variables $(\mathcal{V}^a)_{a\in\mathcal{A}}$

Algorithm	UCB	UpUCB (b)	UpUCB	UpUCB-nAff (b)	UpUCB-nAff
Affected known	No	Yes	Yes	No	No
Baseline known	No	Yes	No	Yes	No
Regret Bound	$\frac{Km^2}{\Delta}$	$\frac{KL^2}{\Delta}$		$\frac{KL^2}{\Delta}$	

From UCB to UpUCB (b)

- Standard UCB (Upper Confidence Bound) bandit algorithm:
  - Reward estimate
    $\hat{r}_t^a = \sum_{s=1}^t r_s \mathbb{1}\{a_s = a\}/\max(1, N_t^a)$
  - Width of confidence interval
    $c_t^a = \sigma\sqrt{2\log(1/\delta')/N_t^a}$ [σ:noise scale]
  - Take action with the highest UCB index: $U_t^a = \hat{r}_t^a + c_t^a$
  - The noise scales in m and the regret is $\mathcal{O}(Km^2 \log T/\Delta)$



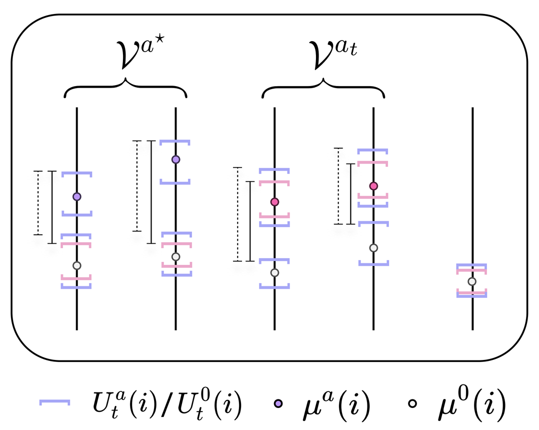
- **UpUCB (b)** [Known baseline and known affected variables]
  - Apply UCB to transformed rewards $r_t' = \sum_{i\in\mathcal{V}_{a_t}}(y_t(i) - \mu^0(i))$
  - This estimates the uplift $r_{\mathrm{up}}^a = r^a - r^0 = \sum_{i\in\mathcal{V}^a}(\mu^a(i) - \mu^0(i))$
  - r_t' is L-sub-Gaussian and thus the regret is in $\mathcal{O}(KL^2 \log T/\Delta)$

Main Result: Handling Unknown Baseline and Unknown Affected Variables

With Unknown Baseline: UpUCB

- Key Takeway: **compute the differences of the UCB indices**
- For $i \in \mathcal{V}^a$, $U_t^a(i)$ computed from the observed payoffs of i whenever a is pulled
- For baseline, $U_0^a(i)$ is estimated with the rounds that i is not affected (i.e., $i \notin \mathcal{V}^{a_t}$)
- Pick action with highest uplifting index

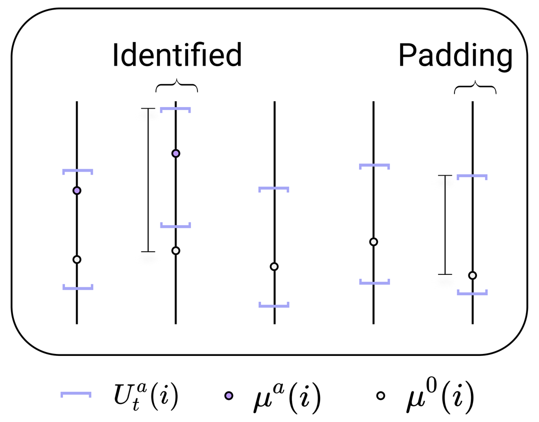
$$\tau_t^a = \sum_{i\in\mathcal{V}^a}(U_t^a(i) - U_t^0(i))$$



With Unknown Affected Variables: UpUCB-nAff (b)

- Regret bound depends on a given $L > \arg\max_{a\in\mathcal{A}} L^a$
- Key Takeaway: **identify the affected variables on the fly**
- Construct the uplifting index in two steps
  1. Identification of affected $\widehat{\mathcal{V}}_t^a$
  2. Optimistic padding with set $\mathcal{L}_t^a$
- Pick action with highest uplifting index

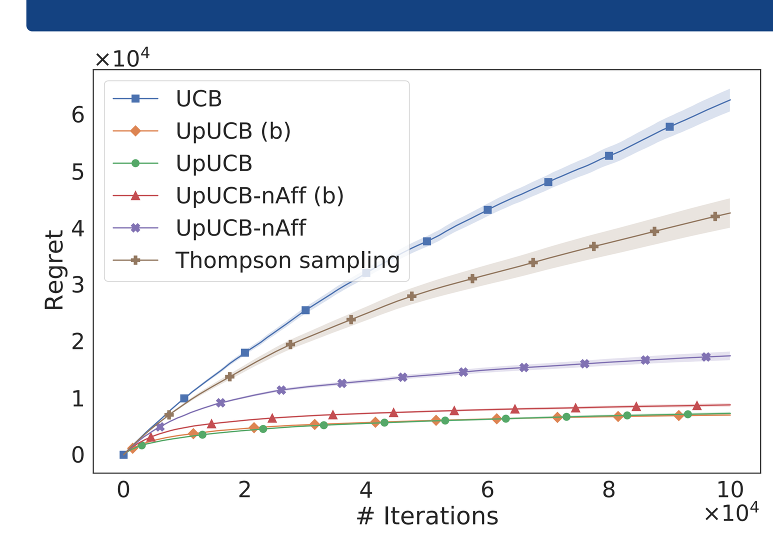
$$\tau_t^a = \sum_{i\in\widehat{\mathcal{V}}_t^a \cup \mathcal{L}_t^a}(U_t^a(i) - \mu^0(i))$$



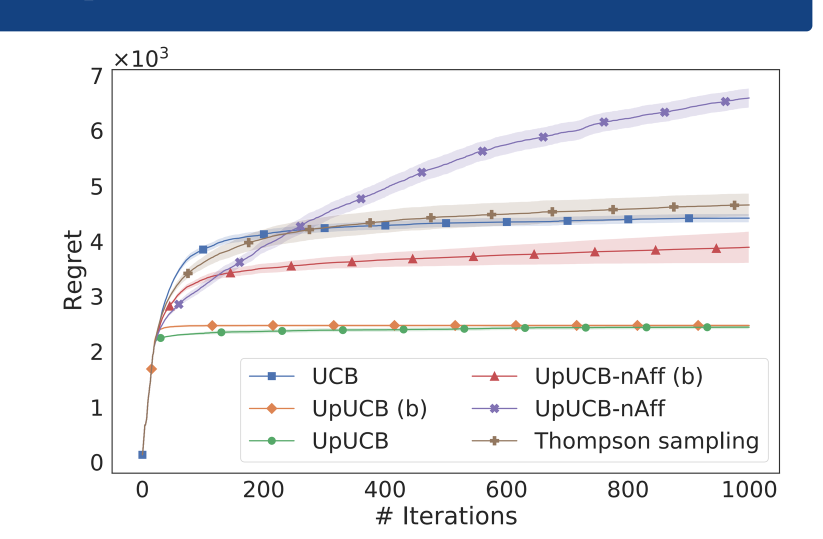
The General Case, Lower Bounds, and Extensions

- UpUCB-nAff combines UpUCB and UpUCB-nAff (b) to tackle the situation where both baseline and affected variables are unknown
- Matching lower bounds justify the necessity of our assumptions
- We also extend the setup to the contextual case where we associate with each variable a feature vector $x_t(i)$

Numerical Experiments



Synthetic data
Gaussian noises
$K = 10, m = 100, L^a \equiv 10, \Delta \sim 0.2$



Constructed with Criteo Uplift Dataset
Independent Bernoulli noises
$K = 20, m = 10^5, L = 12654, \Delta \sim 30$