Multi-Armed Bandits

Lucas Janson and Sham Kakade

CS/Stat 184: Introduction to Reinforcement Learning

Fall 2023

Today

- Feedback from last lecture
- Recap
- Multi-armed bandit problem statement
- Baseline approaches: pure exploration and pure greedy
- Explore-then-commit

Feedback from feedback forms

1. Thank you to everyone who filled out the forms!
2.

Today

- ✓ Feedback from last lecture
- Recap
- Multi-armed bandit problem statement
- Baseline approaches: pure exploration and pure greedy
- Explore-then-commit

Iterative LQR (iLQR)

per iter: $H(d^2k + k^3)$

Recall $x_0 \sim \mu_0$; denote $\mathbb{E}_{x_0 \sim \mu_0}[x_0] = \bar{x}_0$

Initialize $\bar{u}_0^0, \ldots, \bar{u}_{H-1}^0$, (how might we do this?)

Generate nominal trajectory: $\bar{x}_0^0 = \bar{x}_0, \bar{u}_0^0, \ldots, \bar{u}_h^0, \bar{x}_{h+1}^0 = f(\bar{x}_h^0, \bar{u}_h^0), \ldots, \bar{x}_{H-1}^0, \bar{u}_{H-1}^0$

For $i = 0, 1, \ldots$

For each h, linearize $f(x, u)$ at $(\bar{x}_h^i, \bar{u}_h^i)$:

Note that although true f is stationary, its approximation f_h is not

$$f_h(x, u) \approx f(\bar{x}_h^i, \bar{u}_h^i) + \nabla_x f(\bar{x}_h^i, \bar{u}_h^i)(x - \bar{x}_h^i) + \nabla_u f(\bar{x}_h^i, \bar{u}_h^i)(u - \bar{u}_h^i)$$

For each h, quadratize $c_h(x, u)$ at $(\bar{x}_h^i, \bar{u}_h^i)$:

$$c_h(x, u) \approx \frac{1}{2} \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}^\top \begin{bmatrix} \nabla_x^2 c(\bar{x}_h^i, \bar{u}_h^i) & \nabla_{x,u}^2 c(\bar{x}_h^i, \bar{u}_h^i) \\ \nabla_{u,x}^2 c(\bar{x}_h^i, \bar{u}_h^i) & \nabla_u^2 c(\bar{x}_h^i, \bar{u}_h^i) \end{bmatrix} \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}$$

$$+ \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}^\top \begin{bmatrix} \nabla_x c(\bar{x}_h^i, \bar{u}_h^i) \\ \nabla_u c(\bar{x}_h^i, \bar{u}_h^i) \end{bmatrix} + c(\bar{x}_h^i, \bar{u}_h^i)$$

Formulate **time-dependent** LQR and compute its optimal control $\pi_0^i, \ldots, \pi_{H-1}^i$

Set new nominal trajectory: $\bar{x}_0^{i+1} = \bar{x}_0, \ \bar{u}_h^{i+1} = \pi_h^i(\bar{x}_h^{i+1}), \text{ and } \bar{x}_{h+1}^{i+1} = f(\bar{x}_h^{i+1}, \bar{u}_h^{i+1})$

Note this is true f, not approximation

Practical Considerations of Iterative LQR:

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{u}^i_0, \ldots, \bar{u}^i_{H-1}$, and the latest computed controls $\bar{u}_0, \ldots, \bar{u}_{H-1}$

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{u}_0^i, \ldots, \bar{u}_{H-1}^i$, and the latest computed controls $\bar{u}_0, \ldots, \bar{u}_{H-1}$

We want to find $\alpha \in [0,1]$ such that $\bar{u}_h^{i+1} := \alpha \, \bar{u}_h^i + (1-\alpha)\bar{u}_h$ has the smallest cost,

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{u}_0^i, \ldots, \bar{u}_{H-1}^i$, and the latest computed controls $\bar{u}_0, \ldots, \bar{u}_{H-1}$

We want to find $\alpha \in [0,1]$ such that $\bar{u}_h^{i+1} := \alpha\, \bar{u}_h^i + (1-\alpha)\bar{u}_h$ has the smallest cost,

$$\min_{\alpha \in [0,1]} \sum_{h=0}^{H-1} c(x_h, \bar{u}_h^{i+1})$$

s.t. $\quad x_{h+1} = f(x_h, \bar{u}_h^{i+1}), \quad \bar{u}_h^{i+1} = \alpha \bar{u}_h^i + (1-\alpha)\bar{u}_h, \quad x_0 = \bar{x}_0$

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{u}_0^i, \ldots, \bar{u}_{H-1}^i$, and the latest computed controls $\bar{u}_0, \ldots, \bar{u}_{H-1}$

We want to find $\alpha \in [0,1]$ such that $\bar{u}_h^{i+1} := \alpha\, \bar{u}_h^i + (1-\alpha)\bar{u}_h$ has the smallest cost,

$$\min_{\alpha \in [0,1]} \sum_{h=0}^{H-1} c(x_h, \bar{u}_h^{i+1})$$

s.t. $\quad x_{h+1} = f(x_h, \bar{u}_h^{i+1}), \quad \bar{u}_h^{i+1} = \alpha \bar{u}_h^i + (1-\alpha)\bar{u}_h, \quad x_0 = \bar{x}_0$

Why is this tractable?

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{u}_0^i, \ldots, \bar{u}_{H-1}^i$, and the latest computed controls $\bar{u}_0, \ldots, \bar{u}_{H-1}$

We want to find $\alpha \in [0,1]$ such that $\bar{u}_h^{i+1} := \alpha\, \bar{u}_h^i + (1-\alpha)\bar{u}_h$ has the smallest cost,

$$\min_{\alpha \in [0,1]} \sum_{h=0}^{H-1} c(x_h, \bar{u}_h^{i+1})$$

s.t. $\quad x_{h+1} = f(x_h, \bar{u}_h^{i+1}), \quad \bar{u}_h^{i+1} = \alpha \bar{u}_h^i + (1-\alpha)\bar{u}_h, \quad x_0 = \bar{x}_0$

Why is this tractable? because it is 1-dimensional!

Summary of LQR extended to nonlinear control:

Summary of LQR extended to nonlinear control:

Local Linearization:

Approximate an LQR at the balance (goal) position $(x^\star, u^\star)$ and then solve the approximated LQR

Summary of LQR extended to nonlinear control:

Local Linearization:

Approximate an LQR at the balance (goal) position $(x^\star, u^\star)$ and then solve the approximated LQR

Computes an approximately globally optimal solution for a small class of nonlinear control problems

Summary of LQR extended to nonlinear control:

Local Linearization:

Approximate an LQR at the balance (goal) position $(x^\star, u^\star)$ and then solve the approximated LQR

Computes an approximately globally optimal solution for a small class of nonlinear control problems

Iterative LQR

Iterate between:

(1) forming an LQR around the current nominal trajectory,

(2) computing a new nominal trajectory using the optimal policy of the LQR

Summary of LQR extended to nonlinear control:

Local Linearization:

Approximate an LQR at the balance (goal) position $(x^\star, u^\star)$ and then solve the approximated LQR

Computes an approximately globally optimal solution for a small class of nonlinear control problems

Iterative LQR

Iterate between:

(1) forming an LQR around the current nominal trajectory,

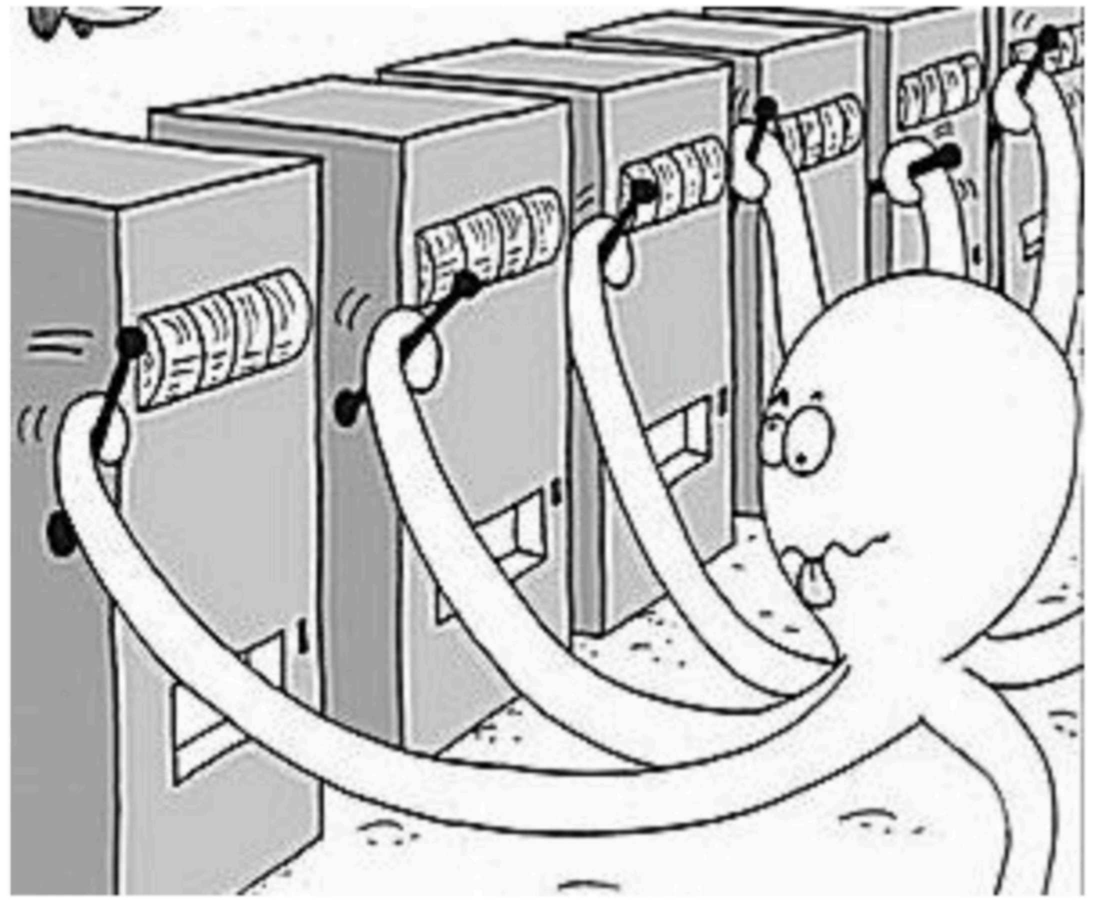
(2) computing a new nominal trajectory using the optimal policy of the LQR

Computes a locally optimal (in policy space) solution for a large class of nonlinear control problems

Today

- ✓ Feedback from last lecture
- ✓ Recap
- Multi-armed bandit problem statement
- Baseline approaches: pure exploration and pure greedy
- Explore-then-commit

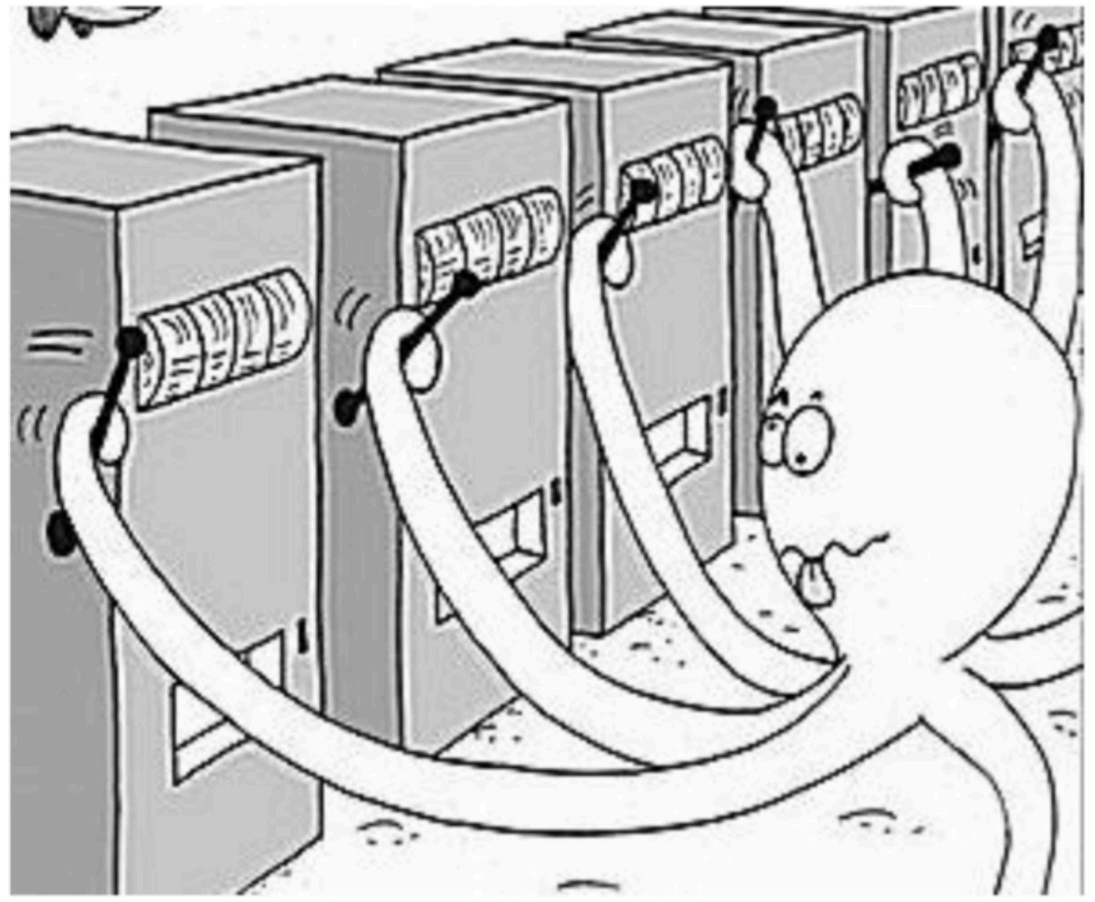
Intro to Multi-armed bandits (MAB)



Setting:

We have K many arms; label them $1, \ldots, K$

Intro to Multi-armed bandits (MAB)

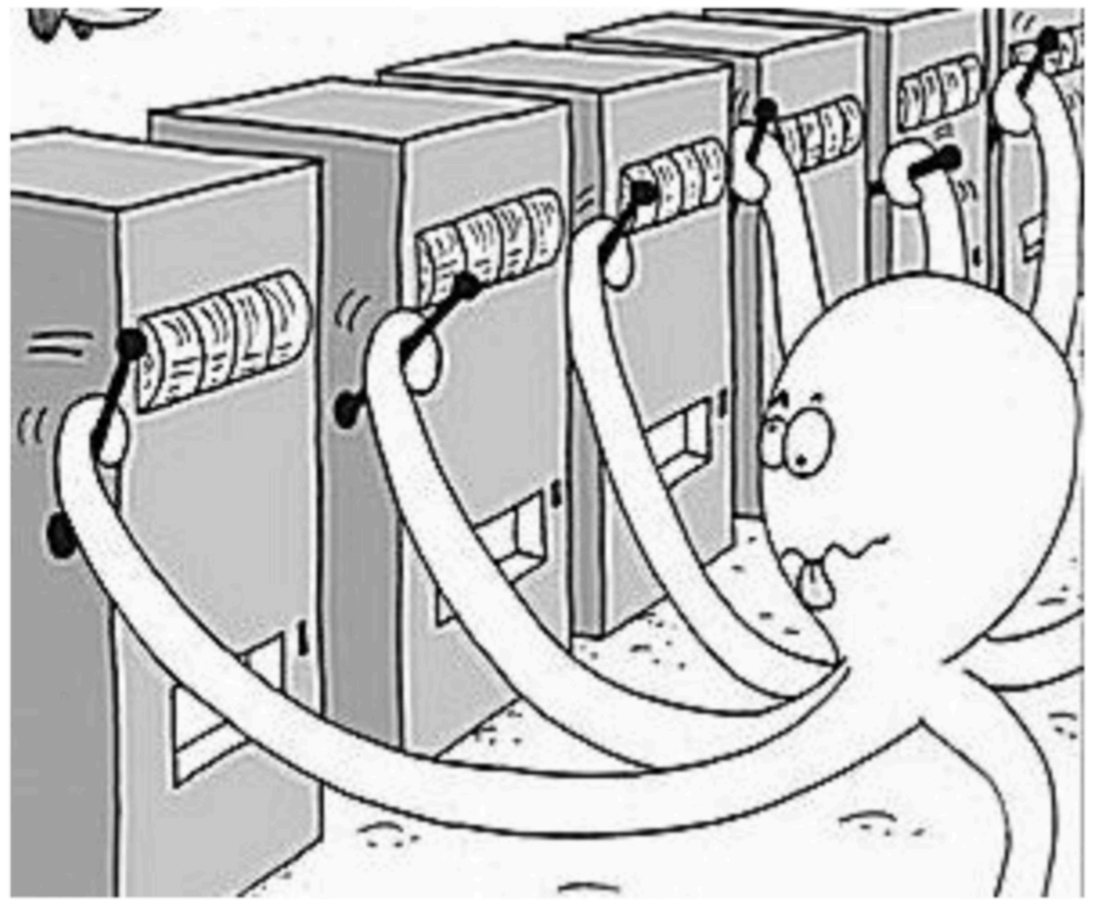


Setting:

We have K many arms; label them $1,\ldots,K$

Each arm has a unknown reward distribution, i.e., $\nu_k \in \Delta([0,1])$,
w/ mean $\mu_k = \mathbb{E}_{r\sim\nu_k}[r]$

Intro to Multi-armed bandits (MAB)



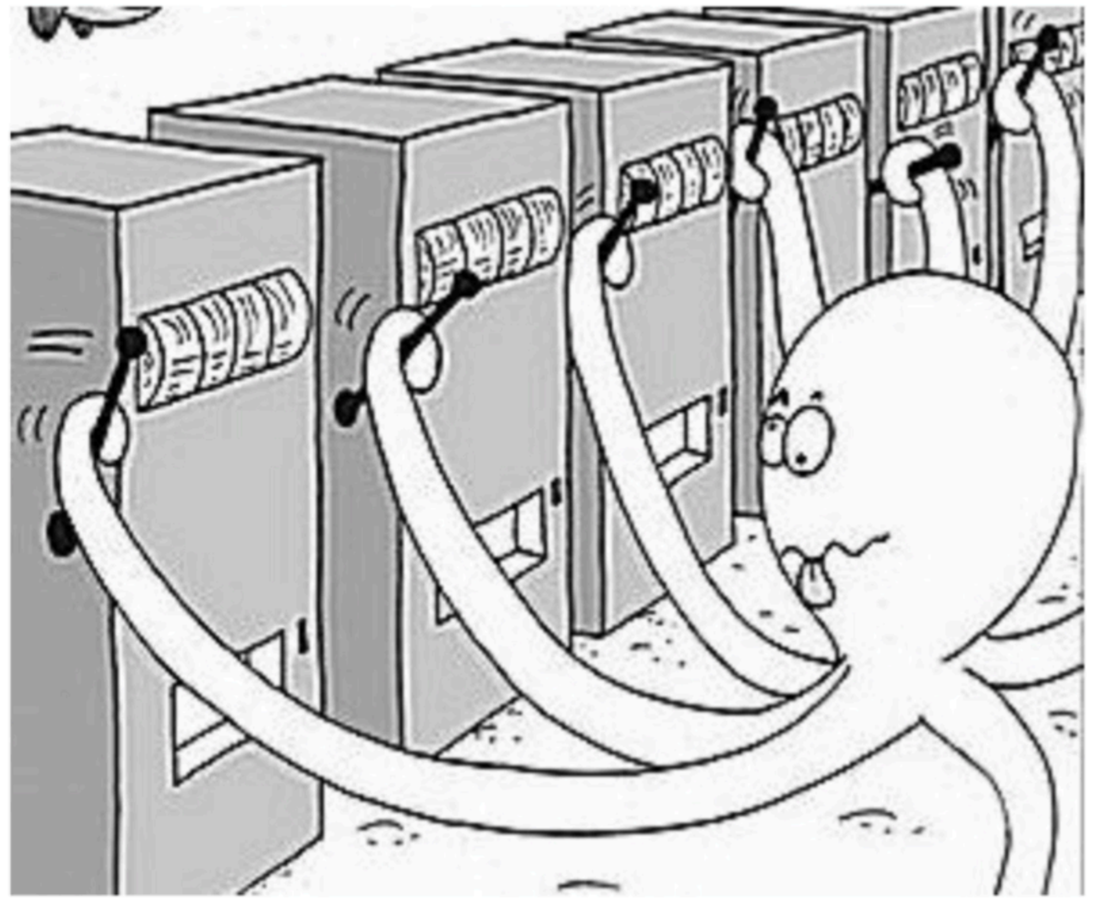
Setting:

We have K many arms; label them $1,\ldots,K$

Each arm has a unknown reward distribution, i.e., $\nu_k \in \Delta([0,1])$,
w/ mean $\mu_k = \mathbb{E}_{r\sim\nu_k}[r]$

Example: ν_k is a Bernoulli distribution w/ mean $\mu_k = \mathbb{P}_{r\sim\nu_k}(r = 1)$

Intro to Multi-armed bandits (MAB)



Setting:

We have K many arms; label them $1, \ldots, K$

Each arm has a unknown reward distribution, i.e., $\nu_k \in \Delta([0,1])$,
w/ mean $\mu_k = \mathbb{E}_{r \sim \nu_k}[r]$

Example: ν_k is a Bernoulli distribution w/ mean $\mu_k = \mathbb{P}_{r \sim \nu_k}(r = 1)$

Every time we pull arm k, we observe an i.i.d reward $r = \begin{cases} 1 & \text{w/ prob } \mu_k \\ 0 & \text{w/ prob } 1 - \mu_k \end{cases}$

Application: online advertising

Arms correspond to Ads

Reward is 1 if user clicks on ad

Application: online advertising

A learning system aims to maximize clicks in the long run:

Arms correspond to Ads

Reward is 1 if user clicks on ad

Application: online advertising

Arms correspond to Ads

Reward is 1 if user clicks on ad

A learning system aims to maximize clicks in the long run:

1. **Try** an Ad (pull an arm)

Application: online advertising

Arms correspond to Ads

Reward is 1 if user clicks on ad

A learning system aims to maximize clicks in the long run:

1. **Try** an Ad (pull an arm)

2. **Observe** if it is clicked (see a zero-one **reward**)

Application: online advertising

Arms correspond to Ads

Reward is 1 if user clicks on ad

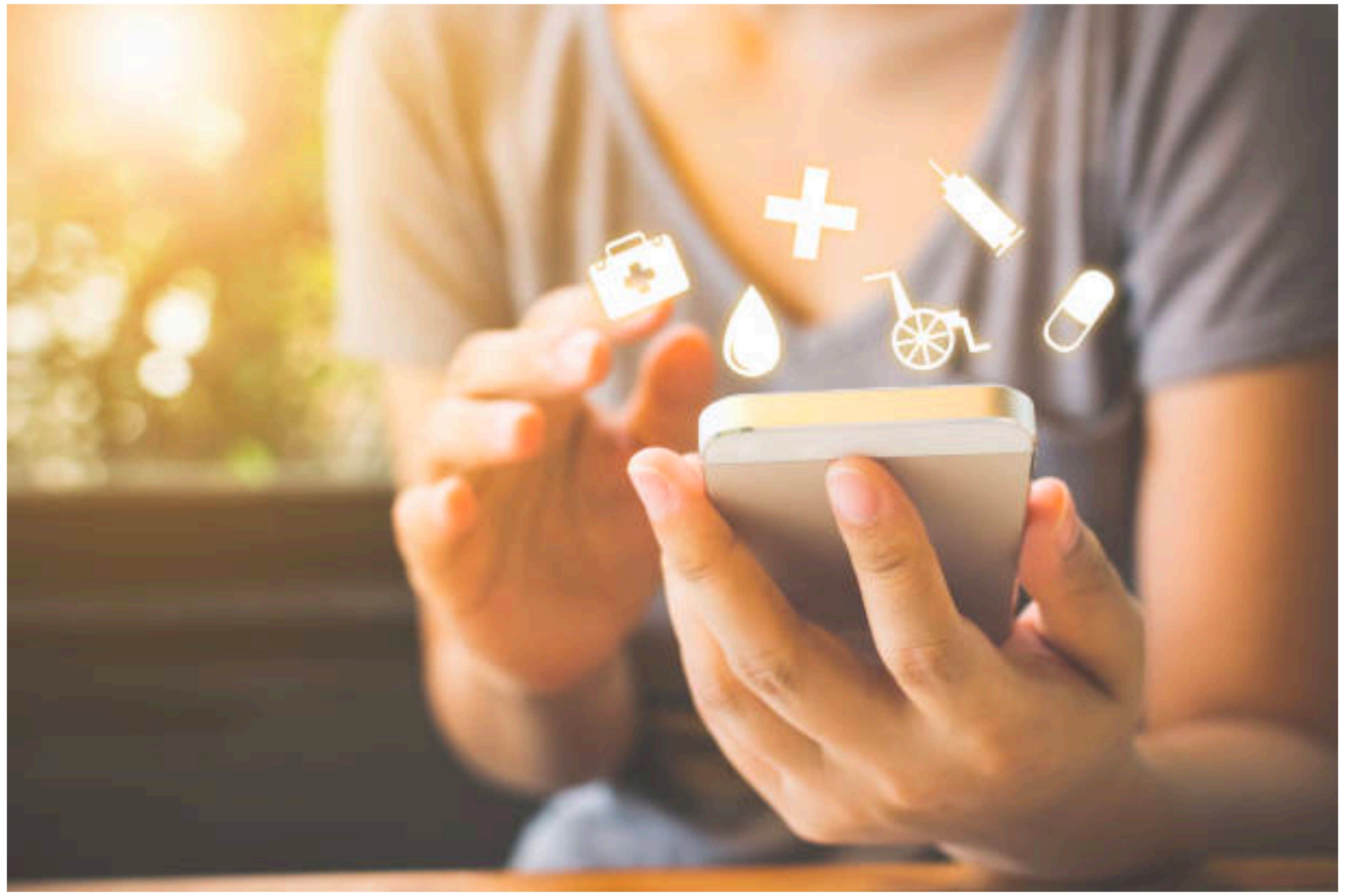
A learning system aims to maximize clicks in the long run:

1. **Try** an Ad (pull an arm)

2. **Observe** if it is clicked (see a zero-one **reward**)

3. **Update**: Decide what ad to recommend for next round

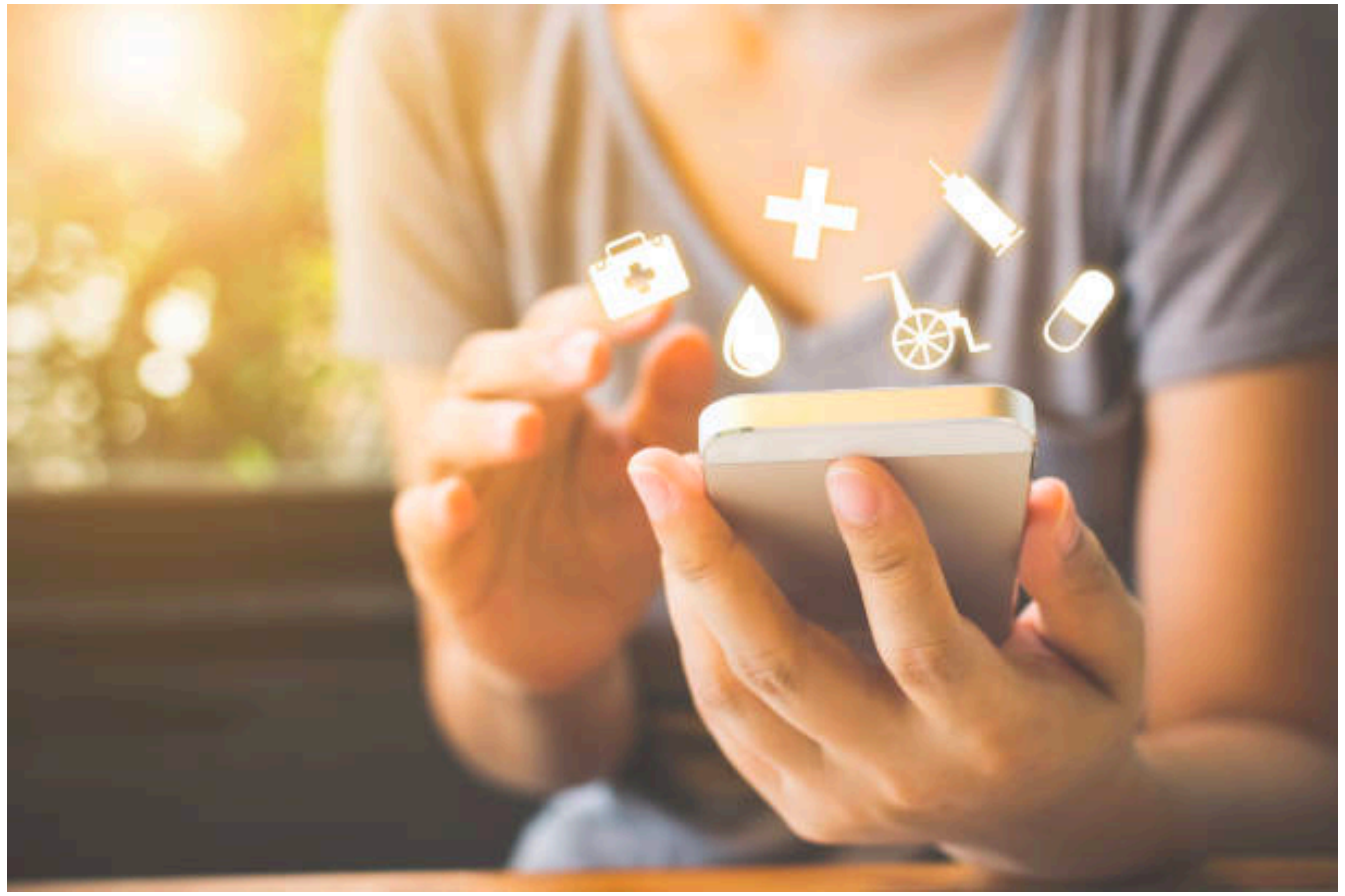
Application: mobile health



Arms correspond to messages sent to users

Reward is, e.g., 1 if user exercised after seeing message

Application: mobile health

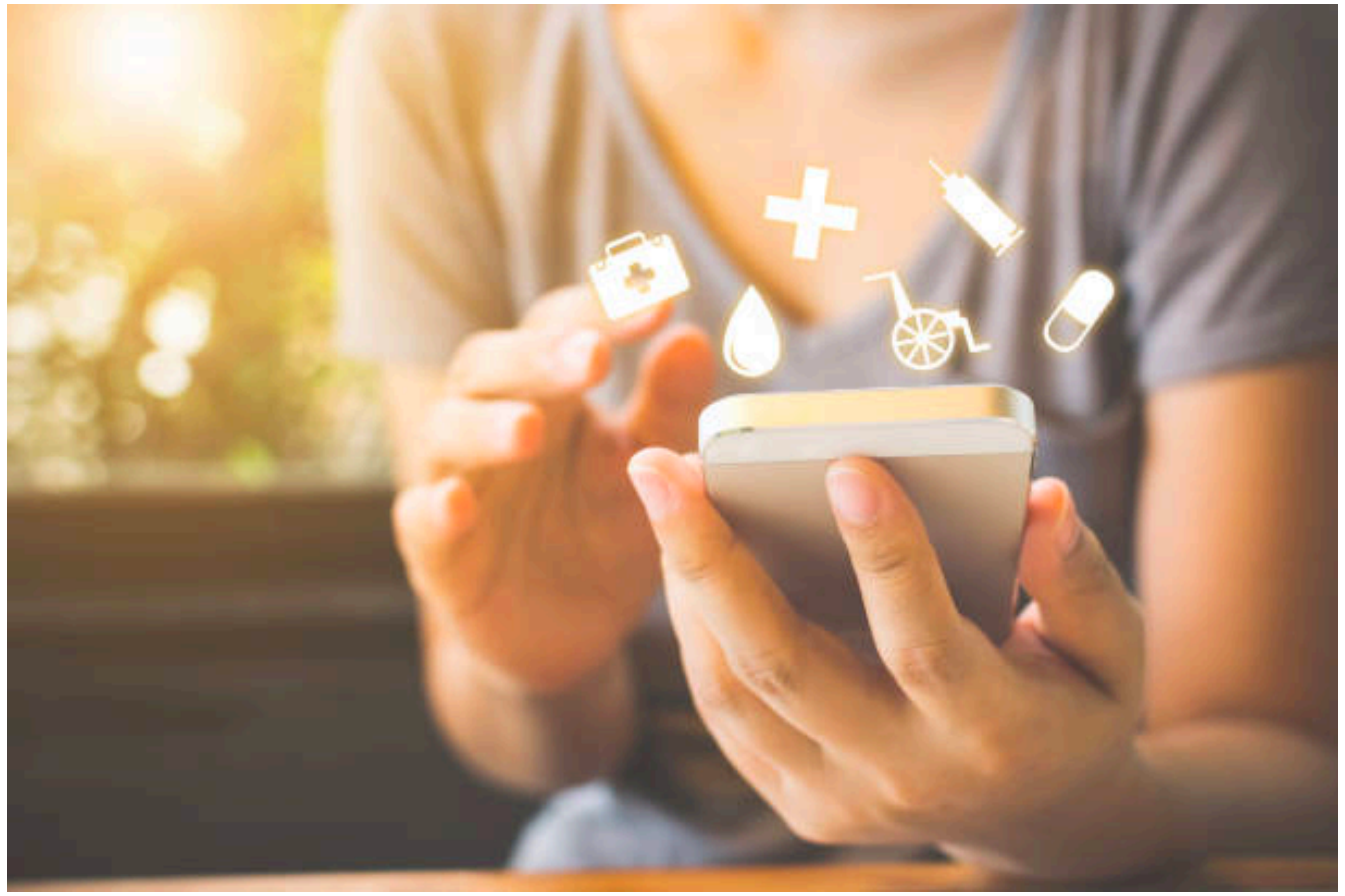


A learning system aims to maximize fitness in the long run:

Arms correspond to messages sent to users

Reward is, e.g., 1 if user exercised after seeing message

Application: mobile health



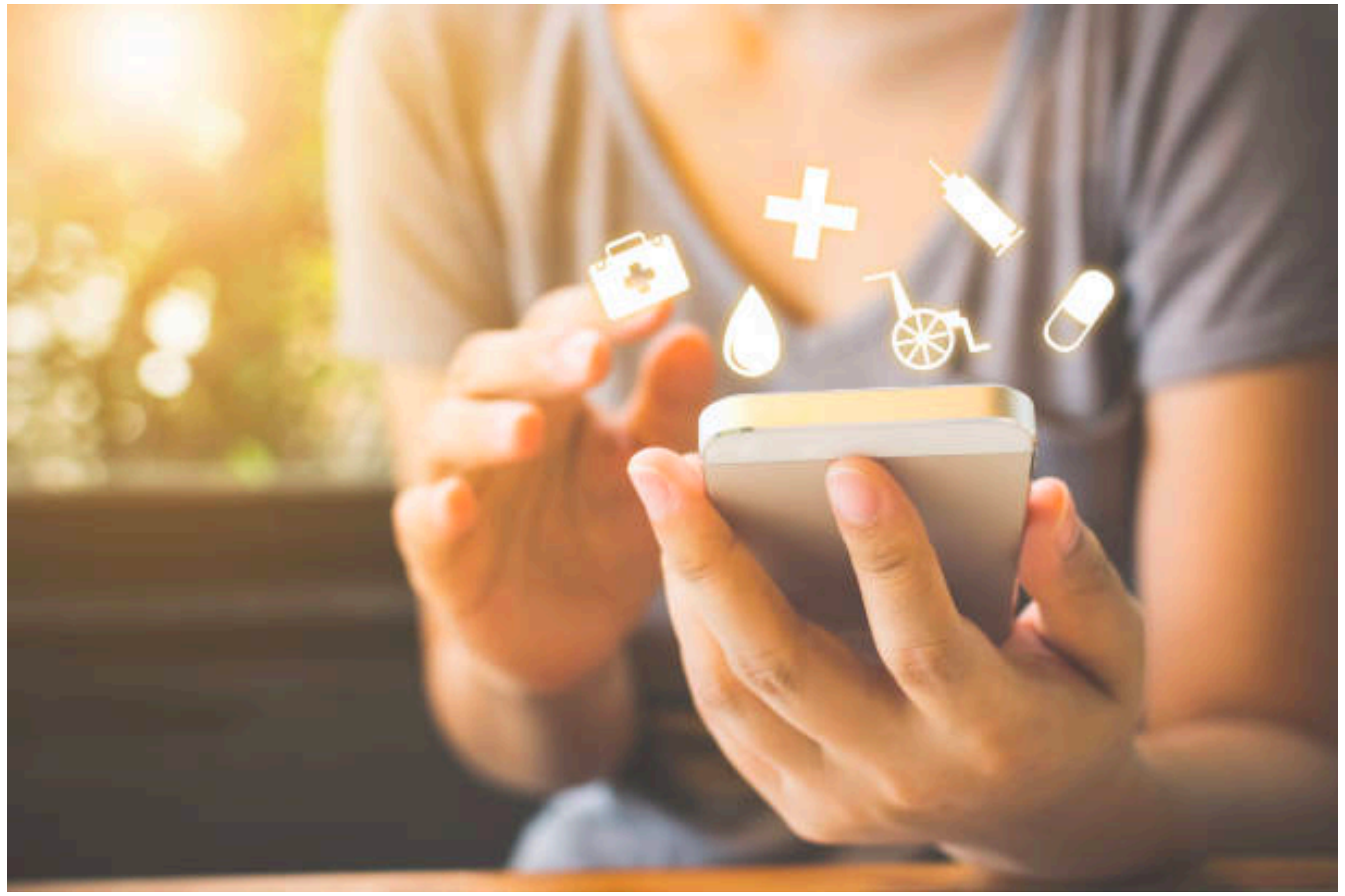
Arms correspond to messages sent to users

Reward is, e.g., 1 if user exercised
after seeing message

A learning system aims to
maximize fitness in the long run:

1. **Send** a message (pull an arm)

Application: mobile health



Arms correspond to messages sent to users

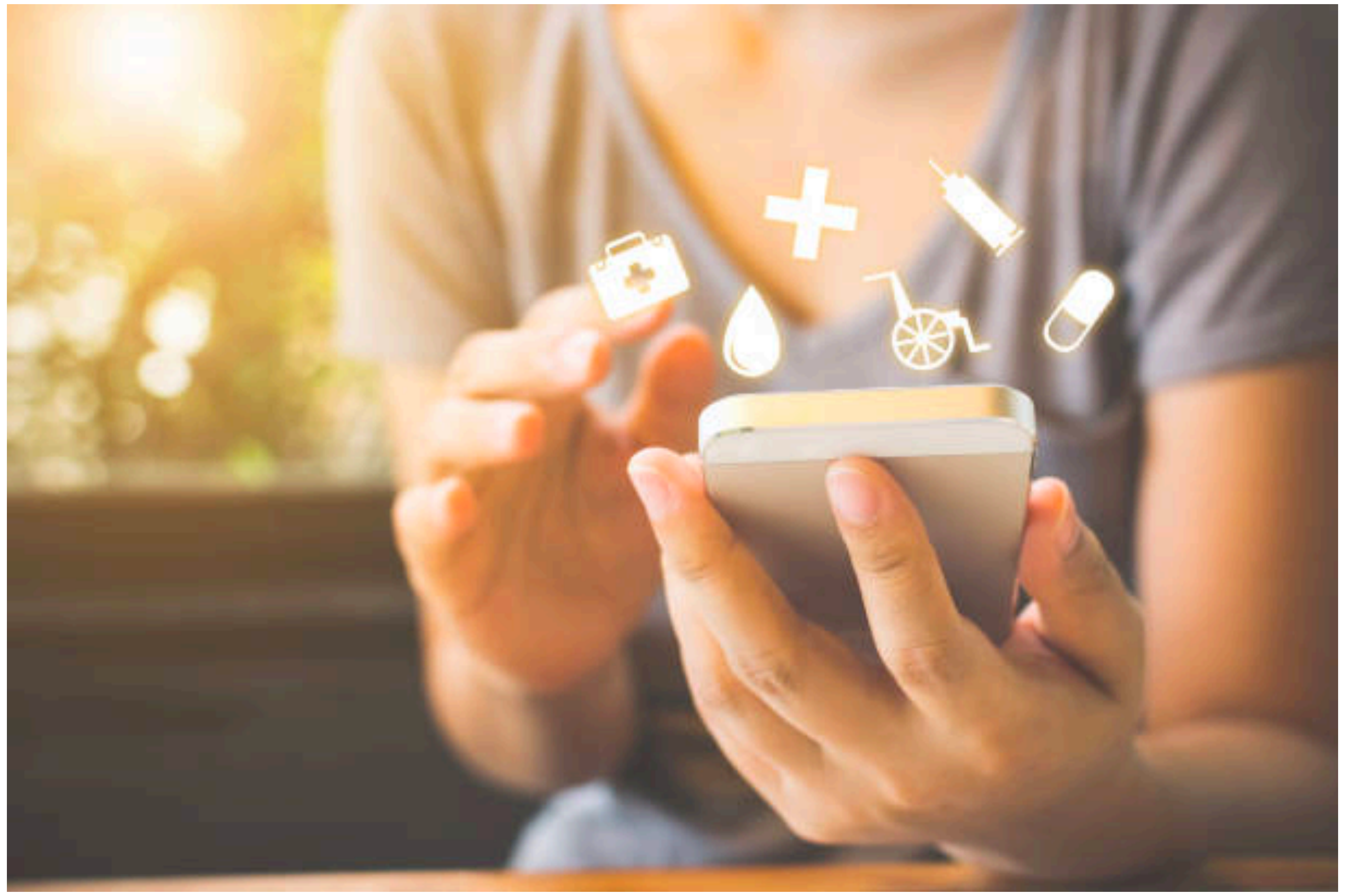
Reward is, e.g., 1 if user exercised
after seeing message

A learning system aims to
maximize fitness in the long run:

1. **Send** a message (pull an arm)

2. **Observe** if user exercises
(see a zero-one **reward**)

Application: mobile health



Arms correspond to messages sent to users

Reward is, e.g., 1 if user exercised after seeing message

A learning system aims to maximize fitness in the long run:

1. **Send** a message (pull an arm)

2. **Observe** if user exercises (see a zero-one **reward**)

3. **Update**: Decide what message to send next round

MAB sequential process

More formally, we have the following interactive learning process:

For $t = 0 \rightarrow T - 1$

MAB sequential process

More formally, we have the following interactive learning process:

For $t = 0 \rightarrow T - 1$

1. Learner pulls arm $a_t \in \{1, \ldots, K\}$

MAB sequential process

More formally, we have the following interactive learning process:

For $t = 0 \rightarrow T - 1$

(based on historical information)

1. Learner pulls arm $a_t \in \{1, \ldots, K\}$

MAB sequential process

More formally, we have the following interactive learning process:

For $t = 0 \rightarrow T - 1$

(based on historical information)

1. Learner pulls arm $a_t \in \{1,\ldots,K\}$

2. Learner observes an i.i.d reward $r_t \sim \nu_{a_t}$ of arm a_t

MAB sequential process

More formally, we have the following interactive learning process:

For $t = 0 \rightarrow T - 1$

(based on historical information)

1. Learner pulls arm $a_t \in \{1, \ldots, K\}$

2. Learner observes an i.i.d reward $r_t \sim \nu_{a_t}$ of arm a_t

Note: each iteration, we do not observe rewards of arms that we did not try

MAB sequential process

More formally, we have the following interactive learning process:

For $t = 0 \rightarrow T - 1$

(based on historical information)

1. Learner pulls arm $a_t \in \{1, \ldots, K\}$

2. Learner observes an i.i.d reward $r_t \sim \nu_{a_t}$ of arm a_t

Note: each iteration, we do not observe rewards of arms that we did not try

Note: there is no state s; rewards from a given arm are i.i.d. (data NOT i.i.d.!)

MAB learning objective

MAB learning objective

Optimal policy when reward distributions known is trivial: $\mu^\star := \max_{k \in [K]} \mu_k$

MAB learning objective

Optimal policy when reward distributions known is trivial: $\mu^\star := \max_{k \in [K]} \mu_k$

$$\text{Regret}_T = T\mu^\star - \sum_{t=0}^{T-1} \mu_{a_t}$$

MAB learning objective

Optimal policy when reward distributions known is trivial: $\mu^\star := \max_{k \in [K]} \mu_k$

$$\text{Regret}_T = T\mu^\star - \sum_{t=0}^{T-1} \mu_{a_t}$$

Total expected reward if we pulled best arm over T rounds

MAB learning objective

Optimal policy when reward distributions known is trivial: $\mu^\star := \max_{k \in [K]} \mu_k$

$$\text{Regret}_T = T\mu^\star - \sum_{t=0}^{T-1} \mu_{a_t}$$

Total expected reward if we pulled best arm over T rounds

Total expected reward of the arms we pulled over T rounds

MAB learning objective

Optimal policy when reward distributions known is trivial: $\mu^\star := \max_{k \in [K]} \mu_k$

$$\text{Regret}_T = T\mu^\star - \sum_{t=0}^{T-1} \mu_{a_t}$$

Total expected reward if we pulled best arm over T rounds

Total expected reward of the arms we pulled over T rounds

Goal: want Regret_T as small as possible

MAB learning objective

Optimal policy when reward distributions known is trivial: $\mu^\star := \max_{k \in [K]} \mu_k$

$$\text{Regret}_T = T\mu^\star - \sum_{t=0}^{T-1} \mu_{a_t}$$

Why not sum the r_t?

Total expected reward if we pulled best arm over T rounds

Total expected reward of the arms we pulled over T rounds

Goal: want Regret_T as small as possible

Why is MAB hard?

Exploration-Exploitation Tradeoff:

Why is MAB hard?

Exploration-Exploitation Tradeoff:

Every round, we need to ask ourselves:

Should we pull the arm that currently appears best now (**exploit**; immediate payoff)?
Or pull another arm, in order to potentially learn it is better (**explore**; payoff later)?

Today

- ✓ Feedback from last lecture
- ✓ Recap
- ✓ Multi-armed bandit problem statement
- Baseline approaches: pure exploration and pure greedy
- Explore-then-commit

Naive baseline: pure exploration

Algorithm: at each round choose an arm uniformly at random from among $\{1,\ldots,K\}$

Naive baseline: pure exploration

Algorithm: at each round choose an arm uniformly at random from among $\{1, \ldots, K\}$

Clearly no learning taking place!

Naive baseline: pure exploration

Algorithm: at each round choose an arm uniformly at random from among $\{1, \ldots, K\}$

Clearly no learning taking place!

grows linearly in T

$$\mathbb{E}[\text{Regret}_T] = \mathbb{E}\left[T\mu^\star - \sum_{t=0}^{T-1} \mu_{a_t}\right] = T\left(\mu^\star - \bar{\mu}\right) = \Omega(T)$$

$$\bar{\mu} = \frac{1}{K}\sum_{k=1}^{K} \mu_k$$

Baseline: pure greedy

Algorithm: try each arm once, and then commit to the one that has the **highest observed** reward

Baseline: pure greedy

Algorithm: try each arm once, and then commit to the one that has the **highest observed** reward

Q: what could go wrong?

Baseline: pure greedy

Algorithm: try each arm once, and then commit to the one that has the **highest observed** reward

Q: what could go wrong?

A bad arm (i.e., low μ_k) may generate a high reward by chance (or vice versa)!

Example: pure greedy

More concretely, let's say we have two arms:

Reward distribution for arm 1: ν_1 = Bernoulli($\mu_1 = 0.6$)

Reward distribution for arm 2: ν_2 = Bernoulli($\mu_2 = 0.4$)

Example: pure greedy

More concretely, let's say we have two arms:

Reward distribution for arm 1: ν_1 = Bernoulli$(\mu_1 = 0.6)$

Reward distribution for arm 2: ν_2 = Bernoulli$(\mu_2 = 0.4)$

Clearly the first arm is better!

Example: pure greedy

More concretely, let's say we have two arms:

Reward distribution for arm 1: ν_1 = Bernoulli($\mu_1 = 0.6$)

Reward distribution for arm 2: ν_2 = Bernoulli($\mu_2 = 0.4$)

Clearly the first arm is better!

First $a_0 = 1$, $a_1 = 2$:

with probability 16%, we observe reward pair $(r_0, r_1) = (0,1)$

Example: pure greedy

More concretely, let's say we have two arms:

Reward distribution for arm 1: ν_1 = Bernoulli$(\mu_1 = 0.6)$

Reward distribution for arm 2: ν_2 = Bernoulli$(\mu_2 = 0.4)$

Clearly the first arm is better!

$(1 - \mu_1)\mu_2 = (1 - 0.6) \times 0.4$

First $a_0 = 1, a_1 = 2$:

with probability 16%, we observe reward pair $(r_0, r_1) = (0,1)$

Example: pure greedy

More concretely, let's say we have two arms:

Reward distribution for arm 1: ν_1 = Bernoulli($\mu_1 = 0.6$)

Reward distribution for arm 2: ν_2 = Bernoulli($\mu_2 = 0.4$)

Clearly the first arm is better!

$(1 - \mu_1)\mu_2 = (1 - 0.6) \times 0.4$

First $a_0 = 1, a_1 = 2$:

with probability 16%, we observe reward pair $(r_0, r_1) = (0,1)$

$\mathbb{E}[\text{Regret}_T] \geq (T-2) \times \mathbb{P}(\text{select arm 2 for all } t > 1) \times (\text{regret of arm 2})$

Example: pure greedy

More concretely, let's say we have two arms:

Reward distribution for arm 1: ν_1 = Bernoulli($\mu_1 = 0.6$)

Reward distribution for arm 2: ν_2 = Bernoulli($\mu_2 = 0.4$)

Clearly the first arm is better!

$(1-\mu_1)\mu_2 = (1-0.6) \times 0.4$

First $a_0 = 1, a_1 = 2$:

with probability 16%, we observe reward pair $(r_0, r_1) = (0,1)$

$$\mathbb{E}[\text{Regret}_T] \geq (T-2) \times \mathbb{P}(\text{select arm 2 for all } t > 1) \times (\text{regret of arm 2})$$
$$= (T-2) \times .16 \times 0.2 = \Omega(T)$$

Example: pure greedy

More concretely, let's say we have two arms:

Reward distribution for arm 1: ν_1 = Bernoulli($\mu_1 = 0.6$)

Reward distribution for arm 2: ν_2 = Bernoulli($\mu_2 = 0.4$)

Clearly the first arm is better!

$(1 - \mu_1)\mu_2 = (1 - 0.6) \times 0.4$

First $a_0 = 1, a_1 = 2$:

with probability 16%, we observe reward pair $(r_0, r_1) = (0,1)$

$$\mathbb{E}[\text{Regret}_T] \geq (T-2) \times \mathbb{P}(\text{select arm 2 for all } t > 1) \times (\text{regret of arm 2})$$
$$= (T-2) \times .16 \times 0.2 = \Omega(T)$$

Same rate as pure exploration!

Today

- ✓ Feedback from last lecture
- ✓ Recap
- ✓ Multi-armed bandit problem statement
- ✓ Baseline approaches: pure exploration and pure greedy
- Explore-then-commit

Lessons learned

Lessons learned

Lesson from pure greedy: exploring each arm once is not enough

Lessons learned

Lesson from pure greedy: exploring each arm once is not enough

Lesson from pure exploration: exploring each arm too much is bad too

Lessons learned

Lesson from pure greedy: exploring each arm once is not enough
Lesson from pure exploration: exploring each arm too much is bad too

Let's allow both, and see how best to trade them off

Lessons learned

Lesson from pure greedy: exploring each arm once is not enough
Lesson from pure exploration: exploring each arm too much is bad too

Let's allow both, and see how best to trade them off

Plan: (1) try each arm multiple times, (2) compute the empirical mean of each arm, (3) commit to the one that has the highest empirical mean

Explore-Then-Commit (ETC)

Explore-Then-Commit (ETC)

Algorithm hyper parameter $N_{\text{e}} < T/K$ (we assume $T >> K$)

Explore-Then-Commit (ETC)

$N_{\text{e}} =$ Number of explorations

Algorithm hyper parameter $N_{\text{e}} < T/K$ (we assume T >> K)

Explore-Then-Commit (ETC)

N_e = Number of explorations

Algorithm hyper parameter $N_e < T/K$ (we assume $T >> K$)

For $k = 1, \ldots, K$: (Exploration phase)

Explore-Then-Commit (ETC)

N_e = Number of explorations

Algorithm hyper parameter $N_\text{e} < T/K$ (we assume T >> K)

For $k = 1,\ldots,K$: (Exploration phase)

Pull arm k N_e times to observe $\{r_i^{(k)}\}_{i=1}^{N_\text{e}} \sim \nu_k$

Explore-Then-Commit (ETC)

N_e = Number of explorations

Algorithm hyper parameter $N_e < T/K$ (we assume $T >> K$)

For $k = 1,\ldots,K$: (Exploration phase)

Pull arm k N_e times to observe $\{r_i^{(k)}\}_{i=1}^{N_e} \sim \nu_k$

Calculate arm k's empirical mean: $\hat{\mu}_k = \frac{1}{N_e} \sum_{i=1}^{N_e} r_i^{(k)}$

Explore-Then-Commit (ETC)

N_{e} = Number of explorations

Algorithm hyper parameter $N_{\text{e}} < T/K$ (we assume $T >> K$)

For $k = 1,\ldots,K$: (Exploration phase)

Pull arm k N_{e} times to observe $\{r_i^{(k)}\}_{i=1}^{N_{\text{e}}} \sim \nu_k$

Calculate arm k's empirical mean: $\hat{\mu}_k = \frac{1}{N_{\text{e}}} \sum_{i=1}^{N_{\text{e}}} r_i^{(k)}$

For $t = N_{\text{e}}K, \ldots, (T-1)$: (Exploitation phase)

Explore-Then-Commit (ETC)

N_e = Number of explorations

Algorithm hyper parameter $N_e < T/K$ (we assume $T >> K$)

For $k = 1,\ldots,K$: (Exploration phase)

Pull arm k N_e times to observe $\{r_i^{(k)}\}_{i=1}^{N_e} \sim \nu_k$

Calculate arm k's empirical mean: $\hat{\mu}_k = \frac{1}{N_e}\sum_{i=1}^{N_e} r_i^{(k)}$

For $t = N_e K, \ldots, (T-1)$: (Exploitation phase)

Pull the best empirical arm $a_t = \arg\max_{i\in[K]} \hat{\mu}_i$

Explore-Then-Commit (ETC)

N_{e} = Number of explorations

Algorithm hyper parameter $N_{\text{e}} < T/K$ (we assume $T >> K$)

For $k = 1,\ldots,K$: (Exploration phase)

Pull arm k N_{e} times to observe $\{r_i^{(k)}\}_{i=1}^{N_{\text{e}}} \sim \nu_k$

Calculate arm k's empirical mean: $\hat{\mu}_k = \frac{1}{N_{\text{e}}} \sum_{i=1}^{N_{\text{e}}} r_i^{(k)}$

For $t = N_{\text{e}}K,\ldots,(T-1)$: (Exploitation phase)

Pull the best empirical arm $a_t = \arg\max_{i \in [K]} \hat{\mu}_i$

Q: how to set N_{e}?

Regret Analysis Strategy

Regret Analysis Strategy

1. Calculate regret during exploration stage

Regret Analysis Strategy

1. Calculate regret during exploration stage
2. Quantify error of arm mean estimates at end of exploration stage

Regret Analysis Strategy

1. Calculate regret during exploration stage
2. Quantify error of arm mean estimates at end of exploration stage
3. Using step 2, calculate regret during exploitation stage

Regret Analysis Strategy

1. Calculate regret during exploration stage
2. Quantify error of arm mean estimates at end of exploration stage
3. Using step 2, calculate regret during exploitation stage

(Actually, will only be able to upper-bound total regret in steps 1-3)

Regret Analysis Strategy

1. Calculate regret during exploration stage
2. Quantify error of arm mean estimates at end of exploration stage
3. Using step 2, calculate regret during exploitation stage

   (Actually, will only be able to upper-bound total regret in steps 1-3)

4. Minimize our upper-bound over N_{e}

But First… An Important Inequality

<u>Hoeffding inequality</u>

But First… An Important Inequality

Hoeffding inequality

Given N i.i.d samples $\{r_i\}_{i=1}^N \sim \nu \in \Delta([0,1])$ with mean μ, let $\hat{\mu} := \frac{1}{N}\sum_{i=1}^{N} r_i$.

Then with probability at least $1 - \delta$,

$$\left| \hat{\mu} - \mu \right| \leq \sqrt{\frac{\ln(2/\delta)}{2N}}$$

But First… An Important Inequality

Hoeffding inequality

Given N i.i.d samples $\{r_i\}_{i=1}^N \sim \nu \in \Delta([0,1])$ with mean μ, let $\hat{\mu} := \frac{1}{N}\sum_{i=1}^{N} r_i$.

Then with probability at least $1-\delta$,

$$\left|\hat{\mu}-\mu\right| \le \sqrt{\frac{\ln(2/\delta)}{2N}}$$

- Why is this useful? Quantify error of arm mean estimates at end of exploration stage (if all estimates are close, arm we commit to must be close to best)

But First… An Important Inequality

<u>Hoeffding inequality</u>

Given N i.i.d samples $\{r_i\}_{i=1}^N \sim \nu \in \Delta([0,1])$ with mean μ, let $\hat{\mu} := \frac{1}{N} \sum_{i=1}^{N} r_i$.

Then with probability at least $1 - \delta$,

$$\left| \hat{\mu} - \mu \right| \leq \sqrt{\frac{\ln(2/\delta)}{2N}}$$

- Why is this useful? Quantify error of arm mean estimates at end of exploration stage (if all estimates are close, arm we commit to must be close to best)
- Why is this true? Full proof beyond course scope, but intuition easier…

Intuition Behind Hoeffding

Hoeffding inequality: sample mean of N i.i.d. samples on $[0,1]$ satisfies

$$\left|\hat{\mu} - \mu\right| \leq \sqrt{\frac{\ln(2/\delta)}{2N}} \text{ w/p } 1 - \delta$$

Intuition Behind Hoeffding

Hoeffding inequality: sample mean of N i.i.d. samples on $[0,1]$ satisfies

$$\left|\hat{\mu} - \mu\right| \leq \sqrt{\frac{\ln(2/\delta)}{2N}} \text{ w/p } 1 - \delta$$

Think of as finite-sample (and conservative) version of Central Limit Theorem (CLT):

Intuition Behind Hoeffding

Hoeffding inequality: sample mean of N i.i.d. samples on $[0,1]$ satisfies

$$\left|\hat{\mu} - \mu\right| \leq \sqrt{\frac{\ln(2/\delta)}{2N}} \text{ w/p } 1 - \delta$$

Think of as finite-sample (and conservative) version of Central Limit Theorem (CLT):

- CLT $\Rightarrow \hat{\mu} - \mu \approx$ Gaussian w/ mean 0 and standard deviation $\propto \sqrt{1/N}$

Intuition Behind Hoeffding

Hoeffding inequality: sample mean of N i.i.d. samples on $[0,1]$ satisfies

$$\left|\hat{\mu} - \mu\right| \leq \sqrt{\frac{\ln(2/\delta)}{2N}} \text{ w/p } 1 - \delta$$

Think of as finite-sample (and conservative) version of Central Limit Theorem (CLT):

- CLT $\Rightarrow \hat{\mu} - \mu \approx$ Gaussian w/ mean 0 and standard deviation $\propto \sqrt{1/N}$
- CLT standard deviation explains the Hoeffding denominator

Intuition Behind Hoeffding

Hoeffding inequality: sample mean of N i.i.d. samples on $[0,1]$ satisfies

$$\left|\hat{\mu} - \mu\right| \leq \sqrt{\frac{\ln(2/\delta)}{2N}} \text{ w/p } 1 - \delta$$

Think of as finite-sample (and conservative) version of Central Limit Theorem (CLT):

- CLT $\Rightarrow \hat{\mu} - \mu \approx$ Gaussian w/ mean 0 and standard deviation $\propto \sqrt{1/N}$
- CLT standard deviation explains the Hoeffding denominator
- Numerator is because Gaussian has double-exponential tails, i.e., probability of a deviation from the mean by x scales roughly like e^{-x^2}, which, when inverted (i.e., set $\delta = e^{-x^2}$ and solve for x) gives $x = \sqrt{\ln(1/\delta)}$

Intuition Behind Hoeffding

Hoeffding inequality: sample mean of N i.i.d. samples on $[0,1]$ satisfies

$$\left|\hat{\mu} - \mu\right| \leq \sqrt{\frac{\ln(2/\delta)}{2N}} \text{ w/p } 1 - \delta$$

Think of as finite-sample (and conservative) version of Central Limit Theorem (CLT):

- CLT $\Rightarrow$ $\hat{\mu} - \mu \approx$ Gaussian w/ mean 0 and standard deviation $\propto \sqrt{1/N}$
- CLT standard deviation explains the Hoeffding denominator
- Numerator is because Gaussian has double-exponential tails, i.e., probability of a deviation from the mean by x scales roughly like e^{-x^2}, which, when inverted (i.e., set $\delta = e^{-x^2}$ and solve for x) gives $x = \sqrt{\ln(1/\delta)}$
- Don't worry too much about the extra 2's… CLT is only approximate!

Back to Regret Analysis of ETC

Back to Regret Analysis of ETC

1. Calculate regret during exploration stage

Back to Regret Analysis of ETC

1. Calculate regret during exploration stage

$$\text{Regret}_{N_e K} \leq N_e K \text{ with probability 1}$$

Back to Regret Analysis of ETC

1. Calculate regret during exploration stage

$$\text{Regret}_{N_e K} \leq N_e K \text{ with probability 1}$$

2. Quantify error of arm mean estimates at end of exploration stage

Back to Regret Analysis of ETC

1. Calculate regret during exploration stage

$$\text{Regret}_{N_\mathsf{e}K} \leq N_\mathsf{e}K \text{ with probability 1}$$

2. Quantify error of arm mean estimates at end of exploration stage

   a) Hoeffding $\Rightarrow \mathbb{P}\left(|\hat{\mu}_k - \mu_k| \leq \sqrt{\ln(2/\delta)/2N_\mathsf{e}} \right) \geq 1 - \delta$

Back to Regret Analysis of ETC

1. Calculate regret during exploration stage

$$\text{Regret}_{N_\text{e}K} \leq N_\text{e}K \text{ with probability } 1$$

2. Quantify error of arm mean estimates at end of exploration stage

   a) Hoeffding $\Rightarrow \mathbb{P}\left(|\hat{\mu}_k - \mu_k| \leq \sqrt{\ln(2/\delta)/2N_\text{e}} \right) \geq 1 - \delta$

   b) Recall Union/Boole/Bonferroni bound: $\mathbb{P}(\text{any of } A_1, \ldots, A_K) \leq \sum_{k=1}^{K} \mathbb{P}(A_k)$

Back to Regret Analysis of ETC

1. Calculate regret during exploration stage

$$\text{Regret}_{N_\text{e}K} \leq N_\text{e}K \text{ with probability 1}$$

2. Quantify error of arm mean estimates at end of exploration stage

   a) Hoeffding $\Rightarrow \mathbb{P}\left(|\hat{\mu}_k - \mu_k| \leq \sqrt{\ln(2/\delta)/2N_\text{e}}\right) \geq 1 - \delta$

   $\mathbb{P}(\forall k, A_1^c, \ldots, A_K^c) \geq 1 - \sum_{k=1}^{K} \mathbb{P}(A_k)$

   $\Updownarrow$

   b) Recall Union/Boole/Bonferroni bound: $\mathbb{P}(\text{any of } A_1, \ldots, A_K) \leq \sum_{k=1}^{K} \mathbb{P}(A_k)$

Back to Regret Analysis of ETC

1. Calculate regret during exploration stage

$$\text{Regret}_{N_e K} \leq N_e K \text{ with probability } 1$$

2. Quantify error of arm mean estimates at end of exploration stage

   a) Hoeffding $\Rightarrow \mathbb{P}\left(|\hat{\mu}_k - \mu_k| \leq \sqrt{\ln(2/\delta)/2N_e}\right) \geq 1 - \delta$

   $\mathbb{P}(\forall k, A_1^c, \ldots, A_K^c) \geq 1 - \sum_{k=1}^{K} \mathbb{P}(A_k)$ $\Updownarrow$

   b) Recall Union/Boole/Bonferroni bound: $\mathbb{P}(\text{any of } A_1, \ldots, A_K) \leq \sum_{k=1}^{K} \mathbb{P}(A_k)$

   c) $\delta \to \delta/K$, Union bound with $A_k = \left\{ |\hat{\mu}_k - \mu_k| > \sqrt{\ln(2K/\delta)/2N_e} \right\}$, and Hoeffding:

Back to Regret Analysis of ETC

1. Calculate regret during exploration stage

$$\text{Regret}_{N_e K} \leq N_e K \text{ with probability 1}$$

2. Quantify error of arm mean estimates at end of exploration stage

   a) Hoeffding $\Rightarrow \mathbb{P}\left(|\hat{\mu}_k - \mu_k| \leq \sqrt{\ln(2/\delta)/2N_e} \right) \geq 1 - \delta$

   $\mathbb{P}(\forall k, A_1^c, \ldots, A_K^c) \geq 1 - \sum_{k=1}^{K} \mathbb{P}(A_k)$

   $\Updownarrow$

   b) Recall Union/Boole/Bonferroni bound: $\mathbb{P}(\text{any of } A_1, \ldots, A_K) \leq \sum_{k=1}^{K} \mathbb{P}(A_k)$

   c) $\delta \to \delta/K$, Union bound with $A_k = \left\{ |\hat{\mu}_k - \mu_k| > \sqrt{\ln(2K/\delta)/2N_e} \right\}$, and Hoeffding:

$$\Rightarrow \mathbb{P}\left(\forall k, |\hat{\mu}_k - \mu_k| \leq \sqrt{\ln(2K/\delta)/2N_e} \right) \geq 1 - \delta$$

Regret Analysis of ETC (cont'd)

Regret Analysis of ETC (cont'd)

2. Quantify error of arm mean estimates at end of exploration stage:

Regret Analysis of ETC (cont'd)

2. Quantify error of arm mean estimates at end of exploration stage:

$$\mathbb{P}\left(\forall k,\ |\hat{\mu}_k - \mu_k| \leq \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}}\right) \geq 1 - \delta$$

Regret Analysis of ETC (cont'd)

2. Quantify error of arm mean estimates at end of exploration stage:

$$\mathbb{P}\left(\forall k,\ |\hat{\mu}_k - \mu_k| \leq \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}}\right) \geq 1 - \delta$$

3. Using step 2, calculate regret during exploitation stage:

Regret Analysis of ETC (cont'd)

2. Quantify error of arm mean estimates at end of exploration stage:

$$\mathbb{P}\left(\forall k,\ |\hat{\mu}_k - \mu_k| \leq \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}}\right) \geq 1 - \delta$$

3. Using step 2, calculate regret during exploitation stage:

Denote (apparent) best arm after exploration stage by $\hat{k}$ and actual best arm by $k^\star$

Regret Analysis of ETC (cont'd)

2. Quantify error of arm mean estimates at end of exploration stage:

$$\mathbb{P}\left(\forall k,\ |\hat{\mu}_k - \mu_k| \le \sqrt{\ln(2K/\delta)/2N_\mathsf{e}}\right) \ge 1 - \delta$$

3. Using step 2, calculate regret during exploitation stage:

Denote (apparent) best arm after exploration stage by $\hat{k}$ and actual best arm by $k^\star$

$$\text{regret at each step of exploitation phase} = \mu_{k^\star} - \mu_{\hat{k}}$$

Regret Analysis of ETC (cont'd)

2. Quantify error of arm mean estimates at end of exploration stage:

$$\mathbb{P}\left(\forall k,\ |\hat{\mu}_k - \mu_k| \le \sqrt{\ln(2K/\delta)/2N_\mathsf{e}}\right) \ge 1 - \delta$$

3. Using step 2, calculate regret during exploitation stage:

Denote (apparent) best arm after exploration stage by $\hat{k}$ and actual best arm by $k^\star$

$$\text{regret at each step of exploitation phase} = \mu_{k^\star} - \mu_{\hat{k}}$$

$$= \mu_{k^\star} + (\hat{\mu}_{k^\star} - \hat{\mu}_{k^\star}) - \mu_{\hat{k}} + (\hat{\mu}_{\hat{k}} - \hat{\mu}_{\hat{k}})$$

Regret Analysis of ETC (cont'd)

2. Quantify error of arm mean estimates at end of exploration stage:

$$\mathbb{P}\left(\forall k,\ |\hat{\mu}_k - \mu_k| \leq \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}}\right) \geq 1 - \delta$$

3. Using step 2, calculate regret during exploitation stage:

Denote (apparent) best arm after exploration stage by $\hat{k}$ and actual best arm by $k^\star$

$$\text{regret at each step of exploitation phase} = \mu_{k^\star} - \mu_{\hat{k}}$$

$$= \mu_{k^\star} + (\hat{\mu}_{k^\star} - \hat{\mu}_{k^\star}) - \mu_{\hat{k}} + (\hat{\mu}_{\hat{k}} - \hat{\mu}_{\hat{k}})$$

$$= (\mu_{k^\star} - \hat{\mu}_{k^\star}) + (\hat{\mu}_{\hat{k}} - \mu_{\hat{k}}) + (\hat{\mu}_{k^\star} - \hat{\mu}_{\hat{k}})$$

Regret Analysis of ETC (cont'd)

2. Quantify error of arm mean estimates at end of exploration stage:

$$\mathbb{P}\left(\forall k,\ |\hat{\mu}_k - \mu_k| \leq \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}}\right) \geq 1 - \delta$$

3. Using step 2, calculate regret during exploitation stage:

Denote (apparent) best arm after exploration stage by $\hat{k}$ and actual best arm by $k^\star$

$$\text{regret at each step of exploitation phase} = \mu_{k^\star} - \mu_{\hat{k}}$$
$$= \mu_{k^\star} + (\hat{\mu}_{k^\star} - \hat{\mu}_{k^\star}) - \mu_{\hat{k}} + (\hat{\mu}_{\hat{k}} - \hat{\mu}_{\hat{k}})$$
$$= (\mu_{k^\star} - \hat{\mu}_{k^\star}) + (\hat{\mu}_{\hat{k}} - \mu_{\hat{k}}) + (\hat{\mu}_{k^\star} - \hat{\mu}_{\hat{k}})$$
$$\leq \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}} + \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}} + 0 \quad \text{w/p } 1 - \delta$$

Regret Analysis of ETC (cont'd)

2. Quantify error of arm mean estimates at end of exploration stage:

$$\mathbb{P}\left(\forall k,\ |\hat{\mu}_k - \mu_k| \leq \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}}\right) \geq 1-\delta$$

3. Using step 2, calculate regret during exploitation stage:

Denote (apparent) best arm after exploration stage by $\hat{k}$ and actual best arm by $k^\star$

$$\text{regret at each step of exploitation phase} = \mu_{k^\star} - \mu_{\hat{k}}$$

$$= \mu_{k^\star} + (\hat{\mu}_{k^\star} - \hat{\mu}_{k^\star}) - \mu_{\hat{k}} + (\hat{\mu}_{\hat{k}} - \hat{\mu}_{\hat{k}})$$

$$= (\mu_{k^\star} - \hat{\mu}_{k^\star}) + (\hat{\mu}_{\hat{k}} - \mu_{\hat{k}}) + (\hat{\mu}_{k^\star} - \hat{\mu}_{\hat{k}})$$

$$\leq \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}} + \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}} + 0 \quad \text{w/p } 1-\delta$$

$$= \sqrt{2\ln(2K/\delta)/N_{\mathsf{e}}}$$

Regret Analysis of ETC (cont'd)

2. Quantify error of arm mean estimates at end of exploration stage:

$$\mathbb{P}\left(\forall k,\ |\hat{\mu}_k - \mu_k| \le \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}}\right) \ge 1 - \delta$$

3. Using step 2, calculate regret during exploitation stage:

Denote (apparent) best arm after exploration stage by $\hat{k}$ and actual best arm by $k^\star$

$$\begin{aligned}
\text{regret at each step of exploitation phase} &= \mu_{k^\star} - \mu_{\hat{k}} \\
&= \mu_{k^\star} + (\hat{\mu}_{k^\star} - \hat{\mu}_{k^\star}) - \mu_{\hat{k}} + (\hat{\mu}_{\hat{k}} - \hat{\mu}_{\hat{k}}) \\
&= (\mu_{k^\star} - \hat{\mu}_{k^\star}) + (\hat{\mu}_{\hat{k}} - \mu_{\hat{k}}) + (\hat{\mu}_{k^\star} - \hat{\mu}_{\hat{k}}) \\
&\le \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}} + \sqrt{\ln(2K/\delta)/2N_{\mathsf{e}}} + 0 \quad \text{w/p } 1-\delta \\
&= \sqrt{2\ln(2K/\delta)/N_{\mathsf{e}}}
\end{aligned}$$

$$\Rightarrow \text{total regret during exploitation} \le T\sqrt{2\ln(2K/\delta)/N_{\mathsf{e}}} \quad \text{w/p } 1-\delta$$

Regret Analysis of ETC (cont'd)

Regret Analysis of ETC (cont'd)

4. From steps 1-3: with probability $1 - \delta$,

$$\text{Regret}_T \leq N_\text{e}K + T\sqrt{2\ln(2K/\delta)/N_\text{e}}$$

Regret Analysis of ETC (cont'd)

4. From steps 1-3: with probability $1 - \delta$,

$$\text{Regret}_T \leq N_{\text{e}}K + T\sqrt{2\ln(2K/\delta)/N_{\text{e}}}$$

Take any N_{e} so that $N_{\text{e}} \to \infty$ and $N_{\text{e}}/T \to 0$ (e.g., $N_{\text{e}} = \sqrt{T}$): sublinear regret!

Regret Analysis of ETC (cont'd)

4. From steps 1-3: with probability $1 - \delta$,

$$\text{Regret}_T \leq N_\text{e} K + T\sqrt{2\ln(2K/\delta)/N_\text{e}}$$

Take any N_e so that $N_\text{e} \to \infty$ and $N_\text{e}/T \to 0$ (e.g., $N_\text{e} = \sqrt{T}$): sublinear regret!

Minimize over N_e: (won't bore you with algebra)

$$\text{optimal } N_\text{e} = \left(\frac{T\sqrt{\ln(2K/\delta)/2}}{K}\right)^{2/3}$$

Regret Analysis of ETC (cont'd)

4. From steps 1-3: with probability $1 - \delta$,

$$\text{Regret}_T \leq N_\text{e} K + T\sqrt{2\ln(2K/\delta)/N_\text{e}}$$

Take any N_e so that $N_\text{e} \to \infty$ and $N_\text{e}/T \to 0$ (e.g., $N_\text{e} = \sqrt{T}$): sublinear regret!

Minimize over N_e: (won't bore you with algebra)

$$\text{optimal } N_\text{e} = \left(\frac{T\sqrt{\ln(2K/\delta)/2}}{K} \right)^{2/3}$$

(A bit more algebra to plug optimal N_e into Regret_T equation above)

$$\Rightarrow \text{Regret}_T \leq 3T^{2/3}(K\ln(2K/\delta)/2)^{1/3} = o(T)$$

Today

- Feedback from last lecture
- Recap
- Multi-armed bandit problem statement
- Baseline approaches: pure exploration and pure greedy
- Explore-then-commit

Summary:

- Multi-armed bandits (or MAB or just bandits)
  - Exemplify exploration vs exploitation
  - Pure greedy not much better than pure exploration (linear regret)
  - Explore then commit obtains sublinear regret

Attendance:
bit.ly/3RcTC9T

Feedback:
bit.ly/3RHtlxy