# Online Algorithms

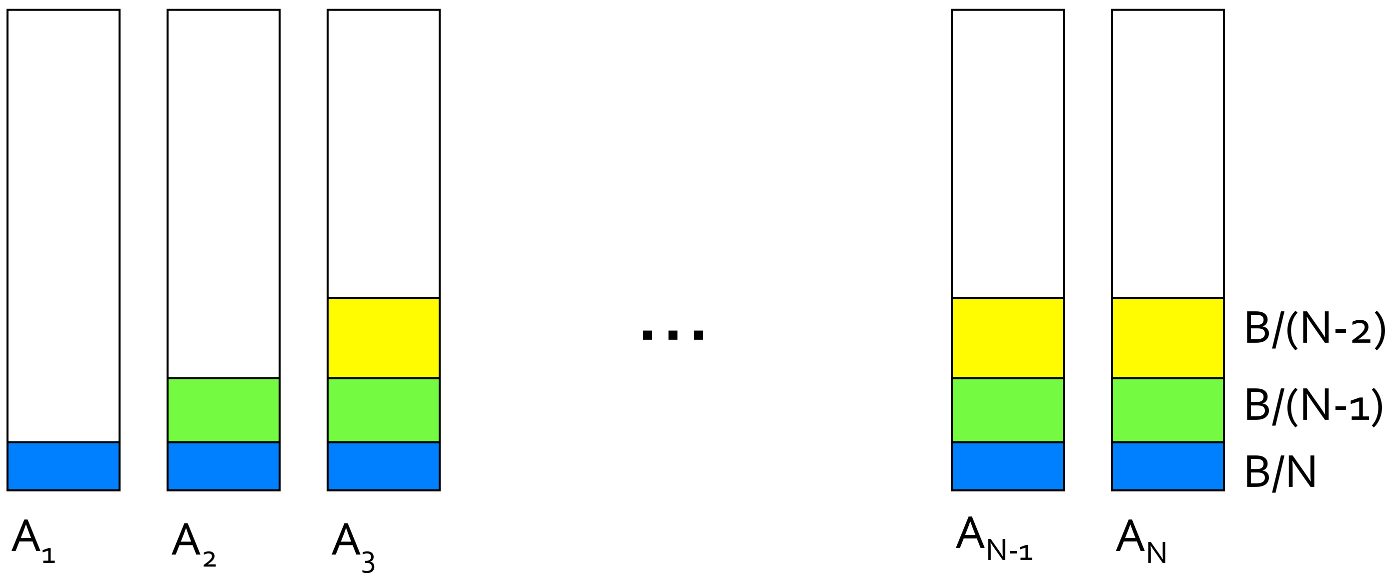
## Generalized BALANCE

**Mining of Massive Datasets**
**Leskovec, Rajaraman, and Ullman**
**Stanford University**

# Worst case for BALANCE

- ***N* advertisers:** $A_1, A_2, \ldots A_N$
  - Each with budget $B > N$
- **Queries:**
  - $N \cdot B$ queries appear in $N$ rounds of $B$ queries each
- **Bidding:**
  - Round **1** queries: bidders $A_1, A_2, \ldots, A_N$
  - Round **2** queries: bidders $A_2, A_3, \ldots, A_N$
  - Round $i$ queries: bidders $A_i, \ldots, A_N$
- **Optimum allocation:**
  Allocate round $i$ queries to $A_i$
  - Optimum revenue $N \cdot B$

# BALANCE Allocation




After k rounds, the allocation to advertiser k is:
$S_k = \sum_{1 \leq i \leq k} B/(N-i+1)$

**If we find the smallest *k* such that $S_k \geq B$, then after *k* rounds we cannot allocate any queries to any advertiser**

# BALANCE: Analysis

$B/1 \quad B/2 \quad B/3 \quad \ldots \quad B/(N-(k-1)) \ldots B/(N-1) \quad B/N$

$S_1$

$S_2$

$S_k$ **= B**

$1/1 \quad 1/2 \quad 1/3 \quad \ldots \quad 1/(N-(k-1)) \ldots 1/(N-1) \quad 1/N$

$S_1$

$S_2$

$S_k$ **= 1**

# BALANCE: Analysis

- **Fact:** for large ***n***
  - Result due to Euler

$$1/1 \quad 1/2 \quad 1/3 \quad \ldots \quad 1/(N-(k-1)) \ldots 1/(N-1) \quad 1/N$$

$\ln(N)$

$\ln(N)-1 = \ln(N-k)$ $\qquad$ $S_k = 1$

$\ln(N-k) = \ln(N) - 1$

$\ln(N/(N-k)) = 1$

$N/(N-k) = e$

$k = N(1 - 1/e)$

# BALANCE: Analysis

- So after the first $k=N(1-1/e)$ rounds, we cannot allocate a query to any advertiser
- **Revenue = $B \cdot N (1-1/e)$**
- **Competitive ratio = $1-1/e$**

# General Version of the Problem

- So far: all bids = 1, all budgets equal (=B)

- **In a general setting BALANCE can be terrible**
  - Consider query **q**, two advertisers $A_1$ and $A_2$
  - $A_1$: ***bid*** = **1**, ***budget*** = **110**
  - $A_2$: ***bid*** = **10**, ***budget*** = **100**
  - Suppose we see **10** instances of **q**
  - BALANCE always selects $A_1$ and earns **10**
  - Optimal earns **100**

# Generalized BALANCE

- Consider query ***q***, bidder ***i***
  - Bid = $\boldsymbol{x_i}$
  - Budget = $\boldsymbol{b_i}$
  - Amount spent so far = $\boldsymbol{m_i}$
  - Fraction of budget left over $\boldsymbol{f_i = 1 - m_i/b_i}$
  - Define $\boldsymbol{\psi_i(q) = x_i(1-e^{-f_i})}$
- Allocate query ***q*** to bidder ***i*** with largest value of $\boldsymbol{\psi_i(q)}$
- **Same competitive ratio (1-1/e)**