Optimizing Submodular Functions

CS246: Mining Massive Datasets

Jure Leskovec, Stanford University
Charilaos Kanatsoulis, Stanford University

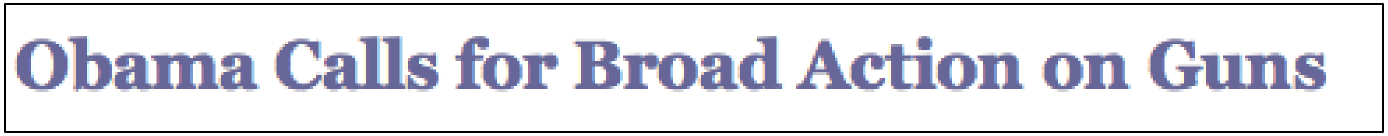
http://cs246.stanford.edu

Announcements

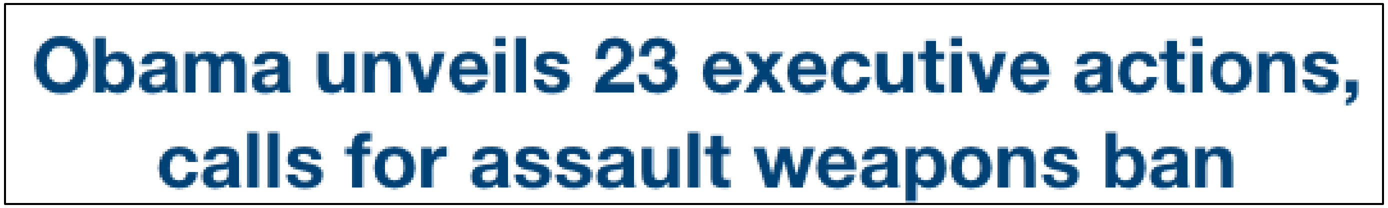
- HW 4 and Colab 8 due today (3/6 at 11:59 pm)
- HW 3, Colab 5, and Colab 6 grades released today (3/6)
- Colab 9 released today (3/6)

Recommendations: Diversity

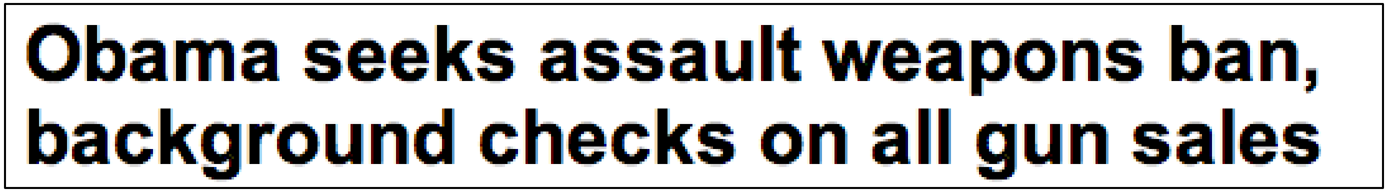
- Redundancy leads to a bad user experience



Obama Calls for Broad Action on Guns



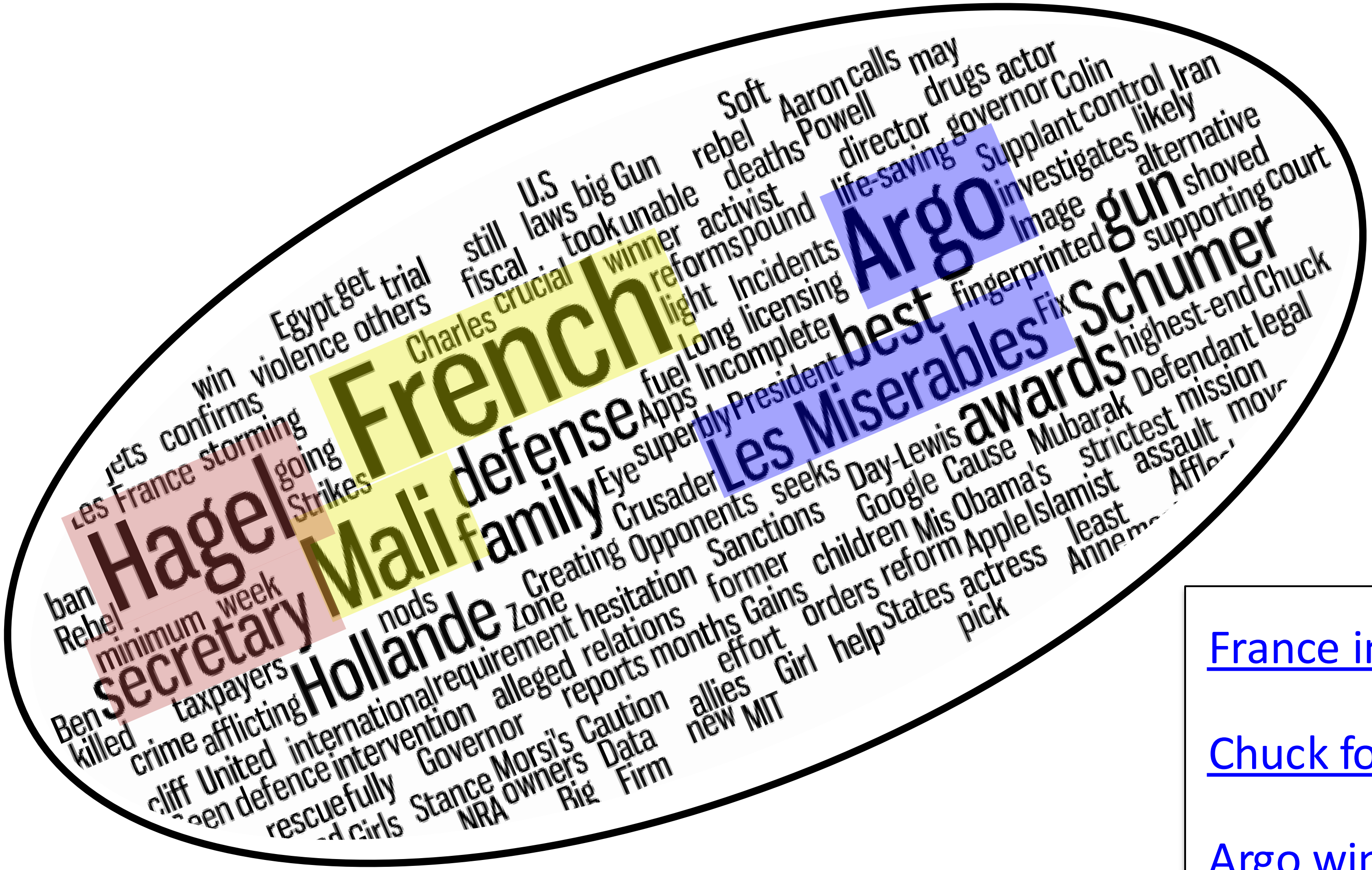
Obama unveils 23 executive actions, calls for assault weapons ban



Obama seeks assault weapons ban, background checks on all gun sales

  - Uncertainty around information need => don't put all eggs in one basket
- How do we optimize for diversity directly?

Covering the day's news



France intervenes

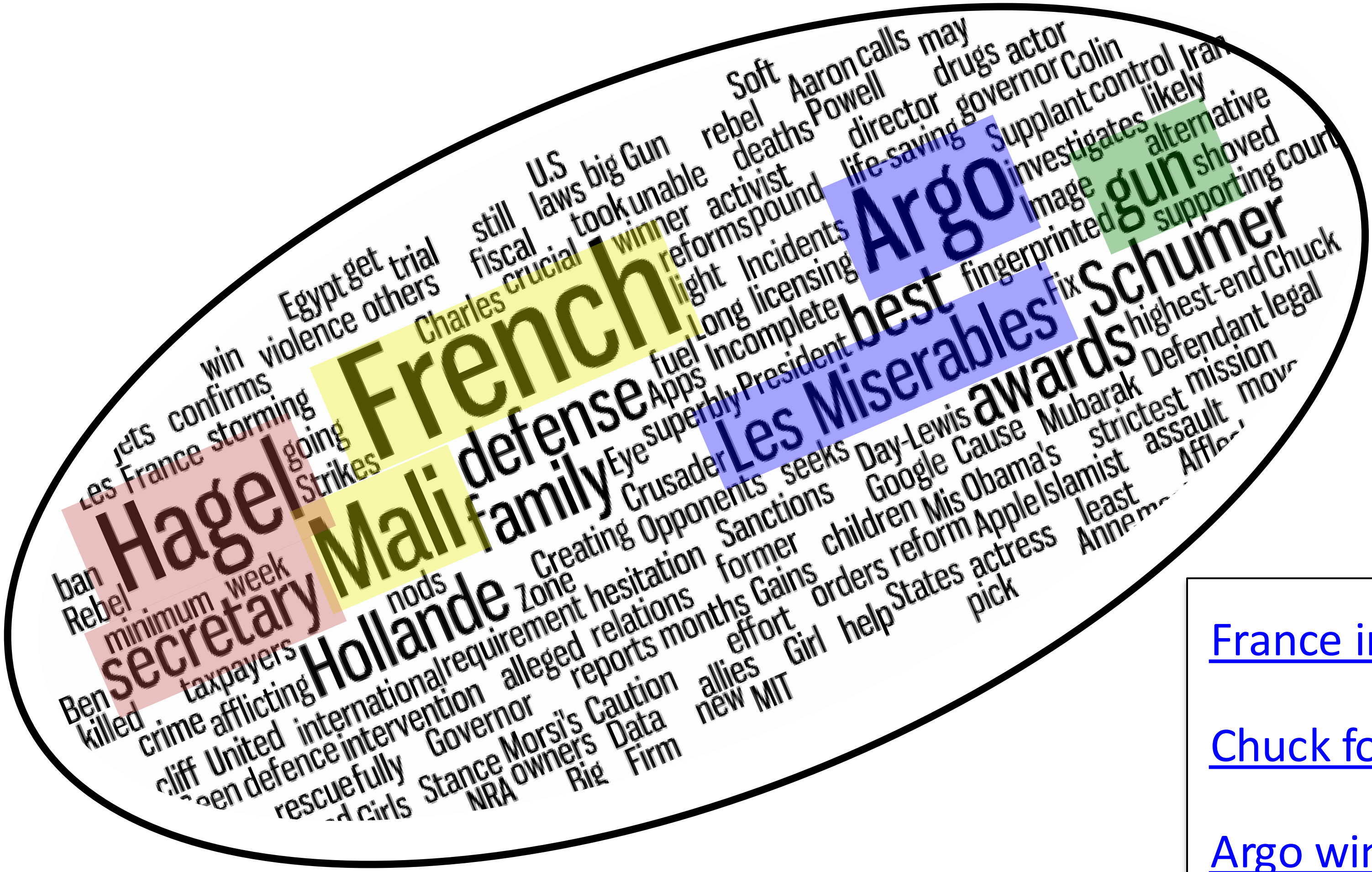
Chuck for Defense

Argo wins big

Hagel expects fight

Monday, January 14

Covering the day's news



France intervenes

Chuck for Defense

Argo wins big

New gun proposals

Monday, January 14

Encode Diversity as Coverage

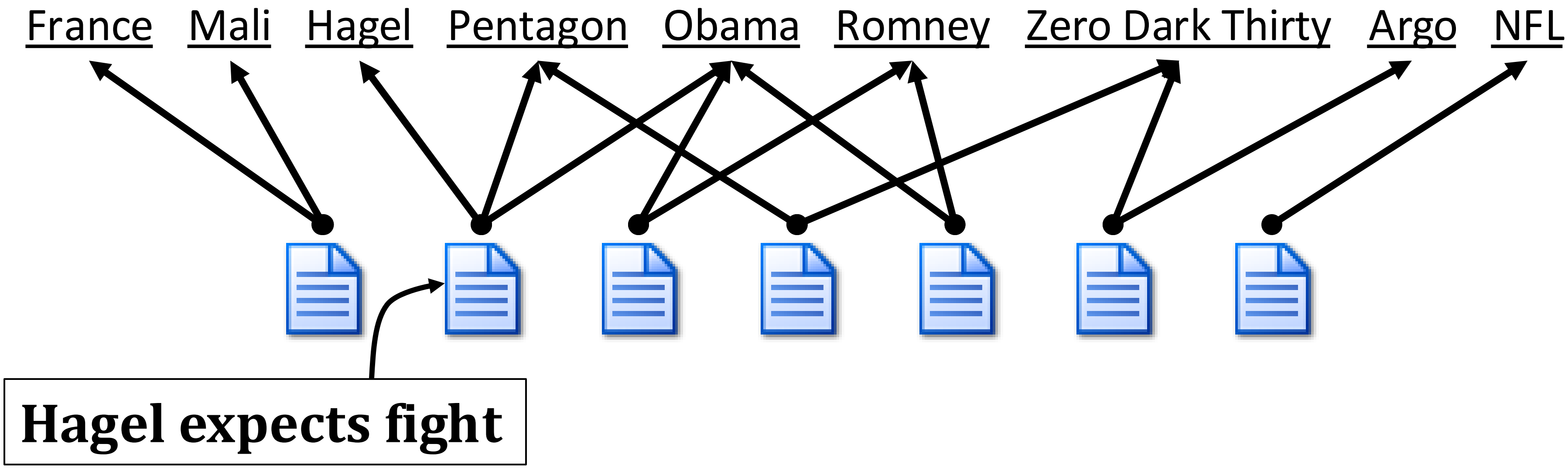
- **Idea:** Encode diversity as coverage problem
- **Example:** Word cloud of news for a single day
  - Want to select articles so that most words are "covered"

Egypt get trial still U.S laws big Gun rebel Soft Aaron calls may
win violence others fiscal took unable deaths Powell drugs actor
jets confirms Charles crucial winner activist director governor Colin
Les France storming French reforms pound life-saving
Hagel going light Incidents Argo Supplant control Iran
ban Strikes Long licensing investigates likely
Rebel fuel alternative
minimum week defense Apps Incomplete Image gun shoved
Ben secretary Mali family Eye superbly President best fingerprinted supporting court
killed taxpayers nods Crusader Les Miserables Fix Schumer
crime afflicting Hollande Zone Creating Opponents seeks Day-Lewis awards highest-end Chuck
cliff United international requirement hesitation Sanctions Google Cause Mubarak Defendant legal
Seen defence intervention alleged relations former children Mis Obama's strictest mission
rescuefully Governor reports months Gains orders reform Apple Islamist assault move
lead Girls Stance Morsi's Caution effort States actress least Affleck
NRA owners Data allies Girl help pick Anne meet
Big Firm new MIT

Diversity as Coverage

What is being covered?

- **Q: What is being covered?**
- **A: Concepts** (In our case: Named entities)



- **Q: Who is doing the covering?**
- **A: Documents**

Simple Abstract Model

- **Suppose we are given a set of documents D**
  - Each document **d** covers a set X_d of words/topics/named entities **W**
- **For a set of documents *A* $\subseteq$ *D* we define**

$$F(A) = \left| \bigcup_{i \in A} X_i \right|$$

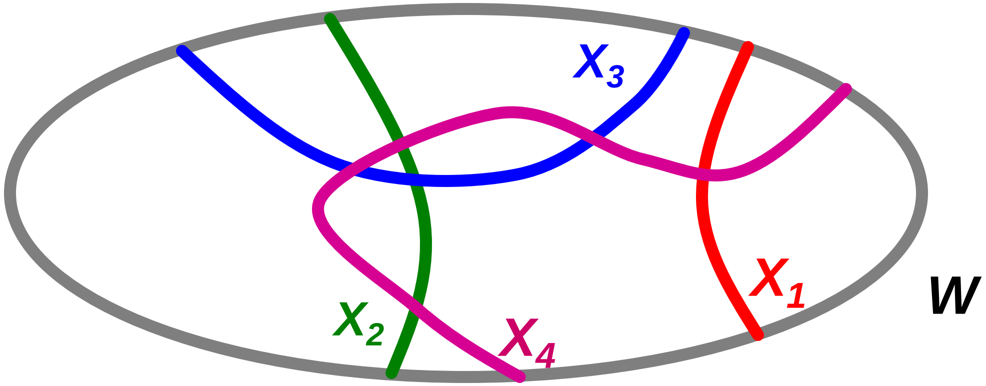
- **Goal: We want to**

$$\max_{|A| \leq k} F(A)$$

- **Note: *F(A)* is a set function:** $F(A)$: **Sets** $\rightarrow \mathbb{N}$

Maximum Coverage Problem

- Given universe of elements $W = \{w_1, \dots, w_n\}$ and sets $X_1, \dots, X_m \subseteq W$



- Goal: Find *k* sets X_i that cover the most of *W*
  - More precisely: Find *k* sets X_i whose size of the union is the largest
  - **Bad news:** A known NP-complete problem

Simple Greedy Heuristic

Simple Heuristic: Greedy Algorithm:

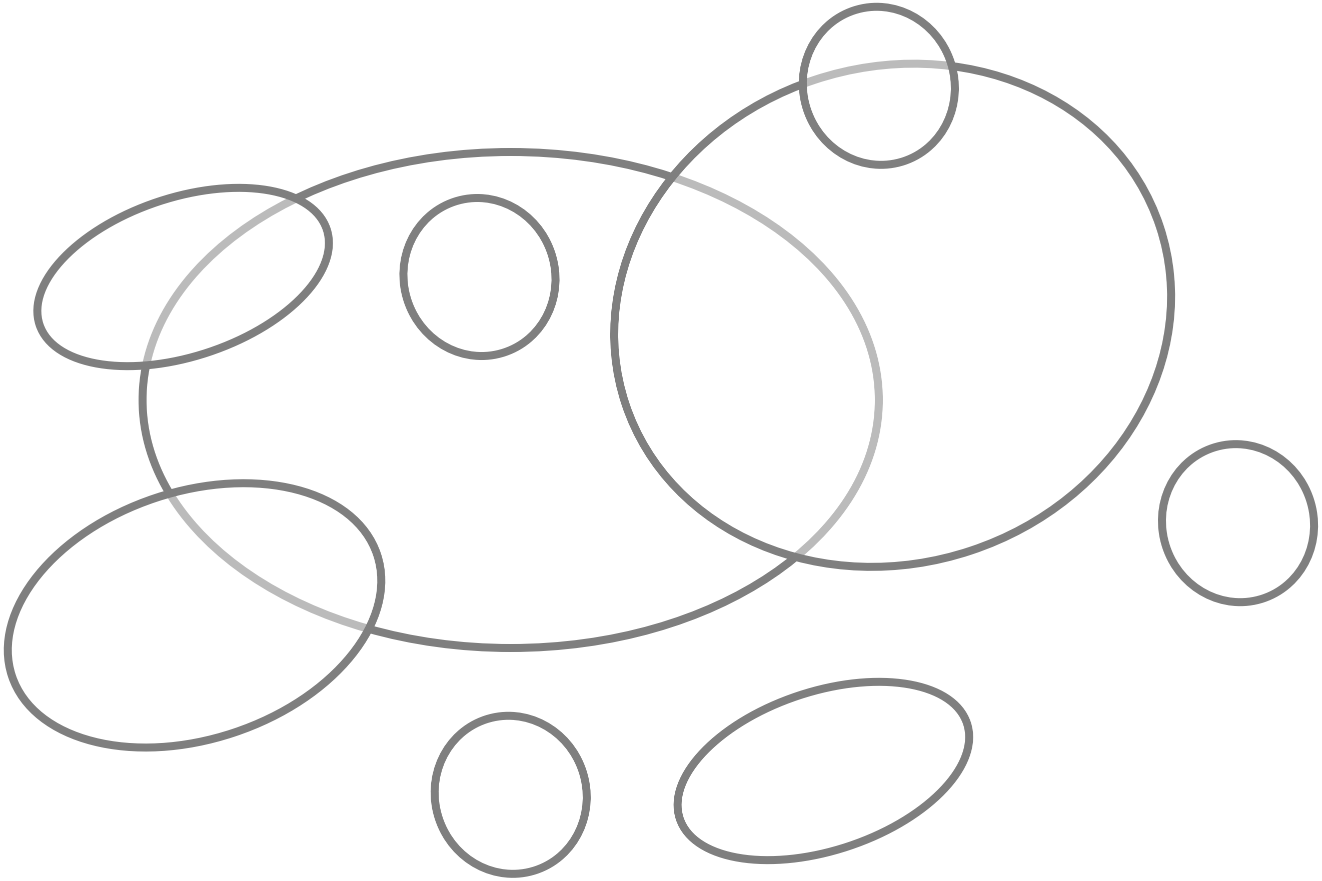
- Start with $A_0 = \{\,\}$
- For $i = 1 \dots k$
  - Find set d that $\max F(A_{i-1} \cup \{d\})$
  - Let $A_i = A_{i-1} \cup \{d\}$

$$F(A) = \left|\bigcup_{d \in A} X_d\right|$$

- **Example:**
  - Eval. $F(\{d_1\}), \dots, F(\{d_m\})$, pick best (say d_1)
  - Eval. $F(\{d_1\} \cup \{d_2\}), \dots, F(\{d_1\} \cup \{d_m\})$, pick best (say d_2)
  - Eval. $F(\{d_1, d_2\} \cup \{d_3\}), \dots, F(\{d_1, d_2\} \cup \{d_m\})$, pick best
  - And so on…

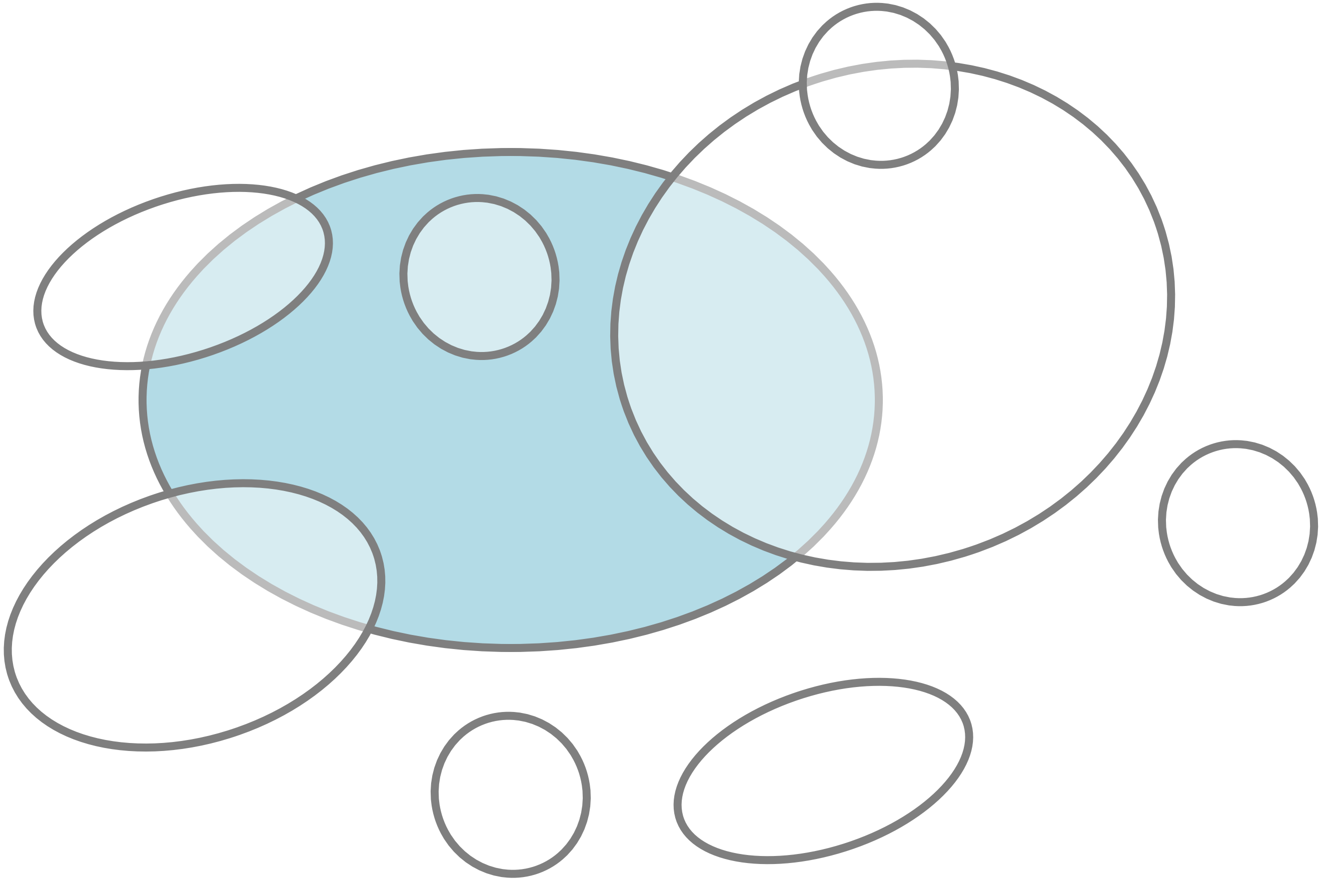
Simple Greedy Heuristic

- **Goal: Maximize the covered area**



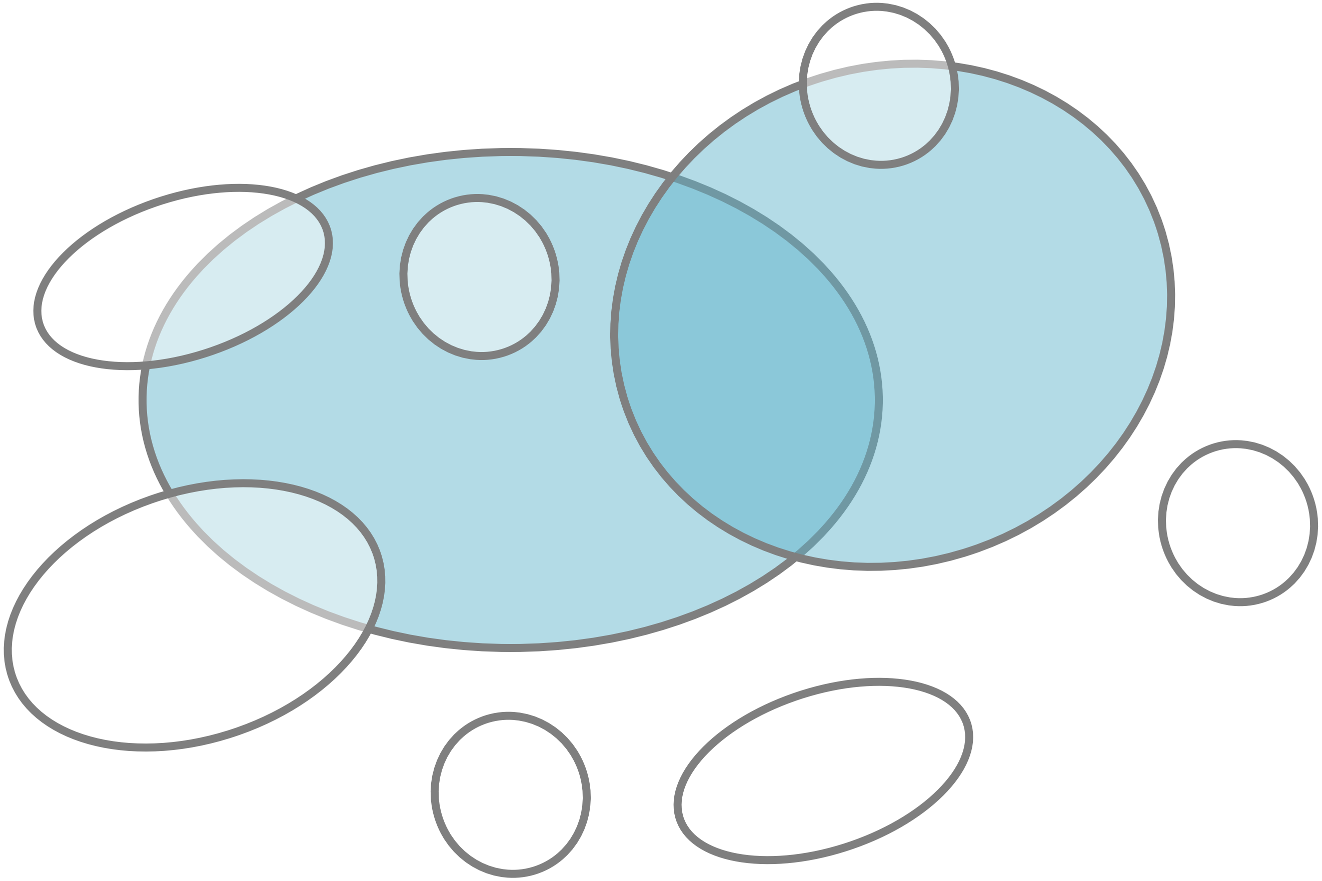
Simple Greedy Heuristic

- **Goal: Maximize the covered area**



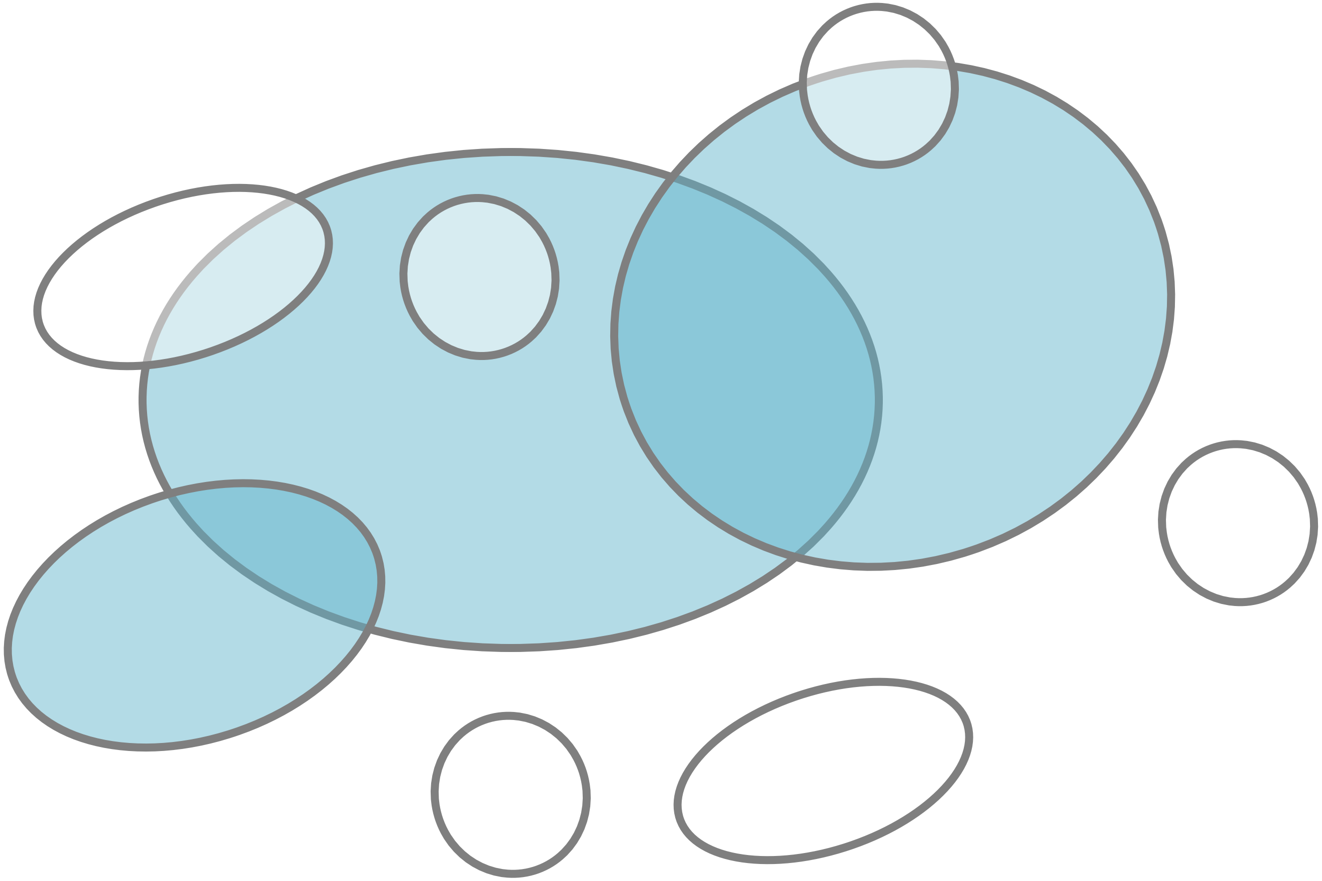
Simple Greedy Heuristic

- **Goal: Maximize the covered area**



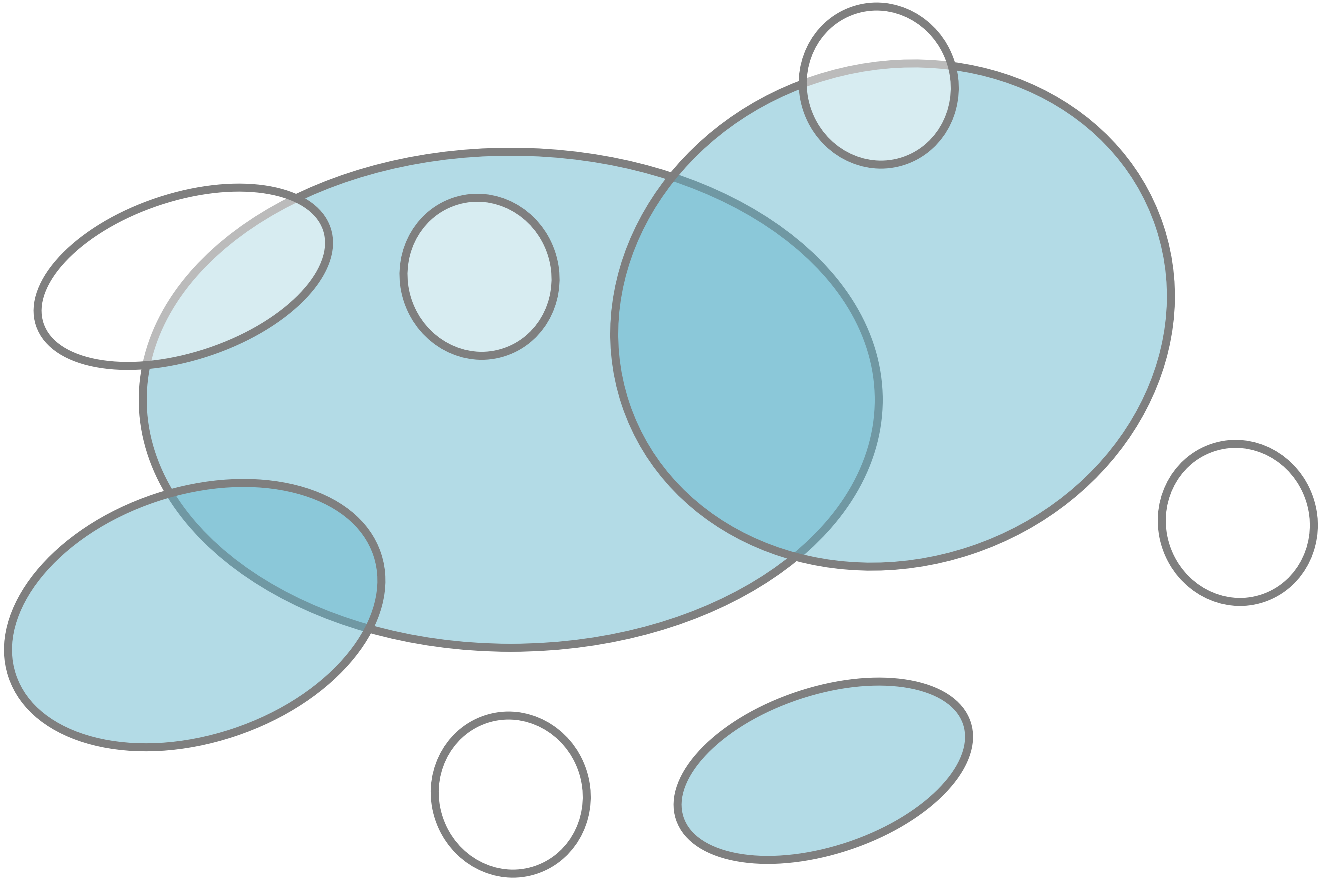
Simple Greedy Heuristic

- **Goal: Maximize the covered area**

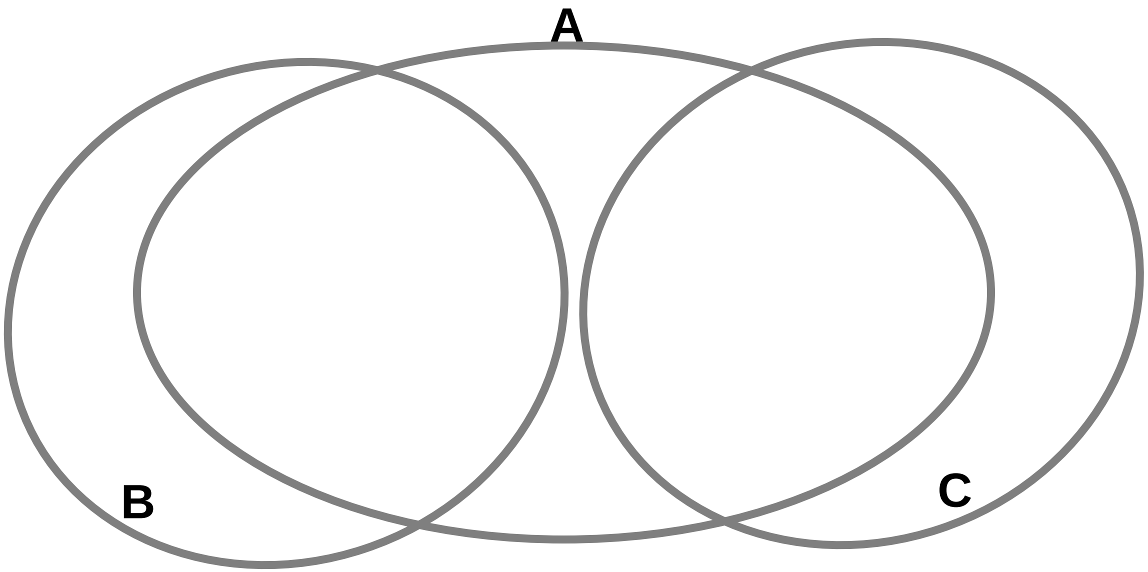


Simple Greedy Heuristic

- **Goal: Maximize the covered area**



When Greedy Heuristic Fails?



- Goal: Maximize the size of the covered area
- Greedy first picks A and then C
- But the optimal way would be to pick B and C

Approximation Guarantee

- **Greedy produces a solution *A***
  where: $F(A) \geq (1-1/e) * OPT$ $(F(A) \geq 0.63 * OPT)$
  [Nemhauser, Fisher, Wolsey '78]

- **Claim holds for functions F*(·)* with 2 properties:**
  - ***F* is monotone:** (adding more docs doesn't decrease coverage)
    if **A** $\subseteq$ ***B*** then ***F*(A)** $\leq$ ***F*(B)** and **F*({})*=0**
  - ***F* is submodular:**
    adding an element to a set gives less or equal improvement than adding it to one of its subsets

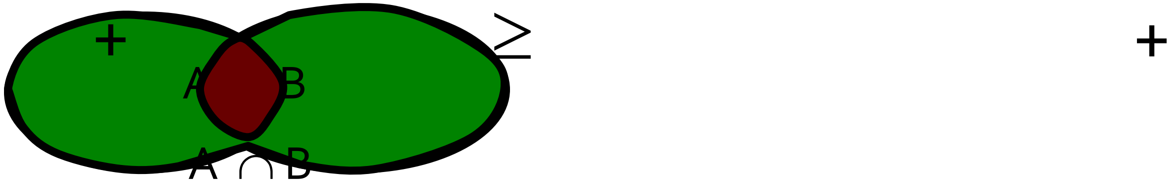
Submodularity: Definition

Definition:

- Set function ***F*(·)** is called **submodular** if:

  For all $\boldsymbol{A,B \subseteq W}$:

$$F(A) + F(B) \geq F(A \cup B) + F(A \cap B)$$



Submodularity: Or equivalently

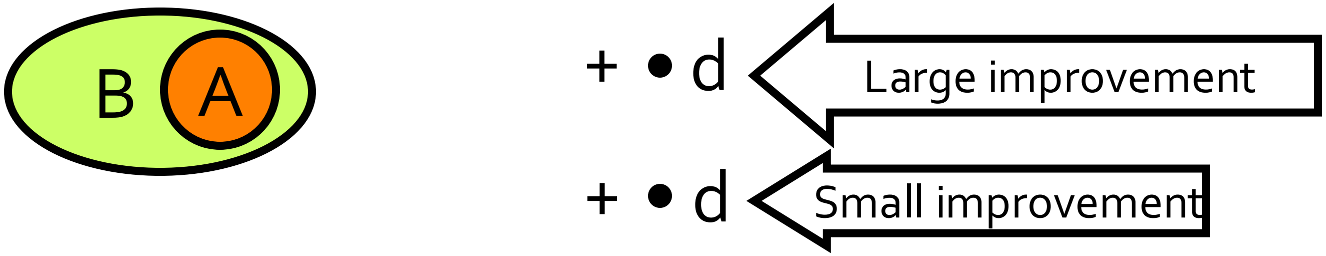
- **Diminishing returns characterization**

Equivalent definition:

- Set function ***F(·)*** is called **submodular** if:

  For all ***A*** $\subseteq$ ***B***:

$$\underbrace{F(A \cup \{d\}) - F(A)}_{\text{Gain of adding } \boldsymbol{d} \text{ to a small set}} \geq \underbrace{F(B \cup \{d\}) - F(B)}_{\text{Gain of adding } \boldsymbol{d} \text{ to a large set}}$$

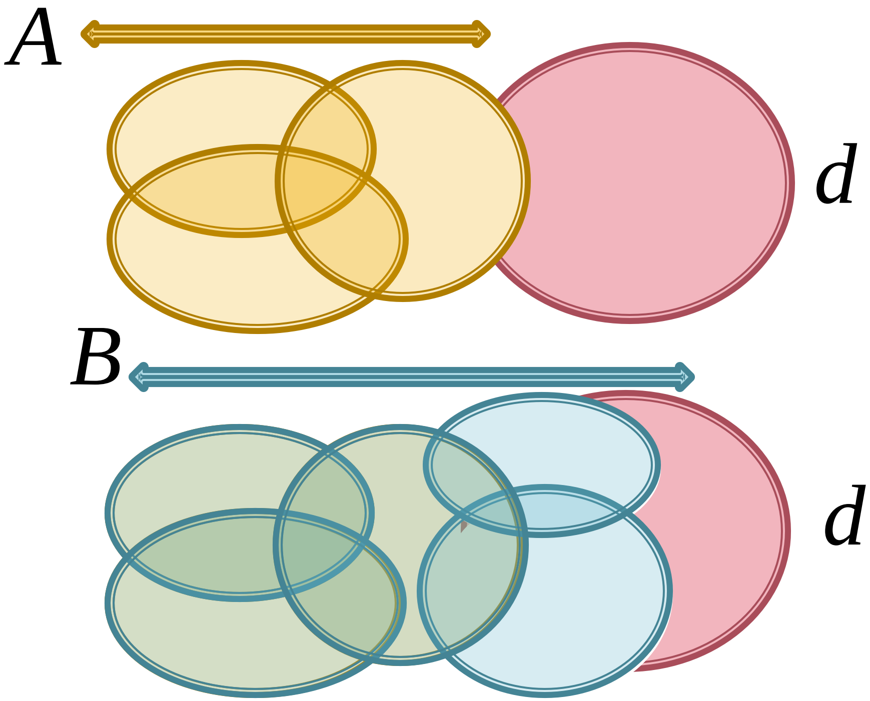


Example: Set Cover

- ***F(·)*** is **submodular**: ***A ⊆ B***

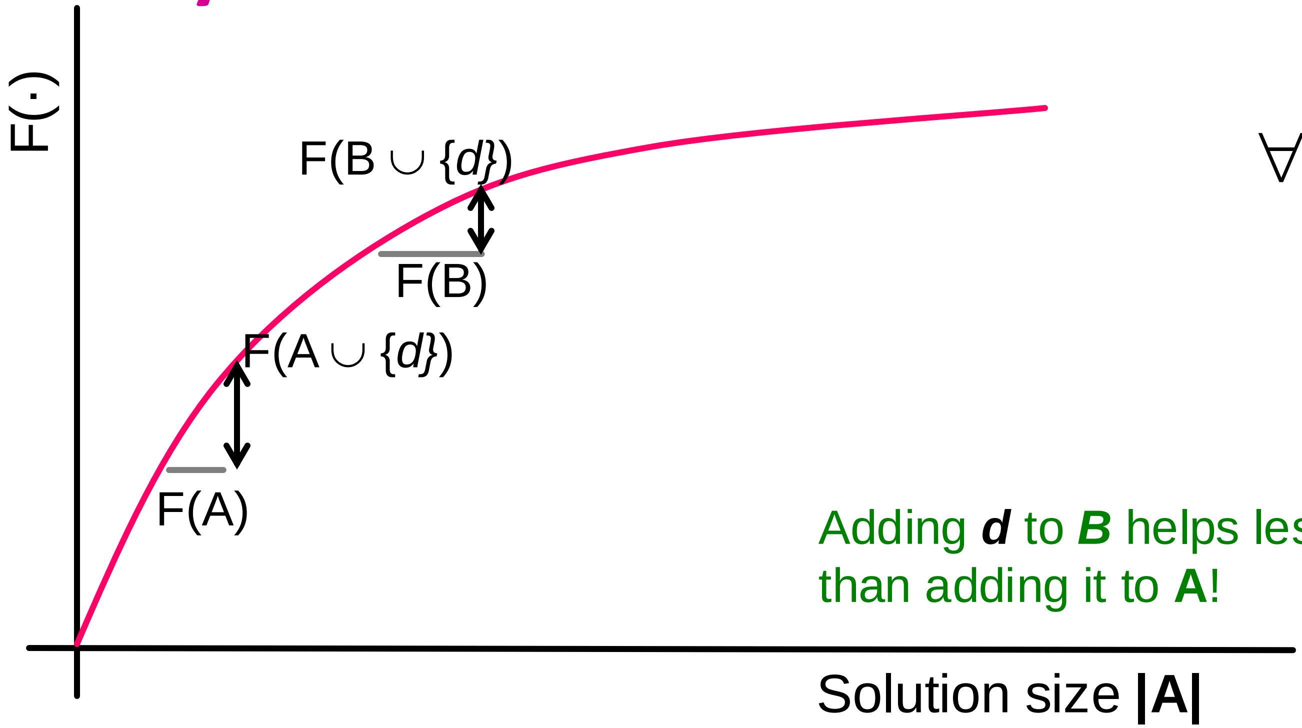
$$\underbrace{F(A \cup \{d\}) - F(A)}_{\text{Gain of adding } \mathbf{d} \text{ to a small set}} \geq \underbrace{F(B \cup \{d\}) - F(B)}_{\text{Gain of adding } \mathbf{d} \text{ to a large set}}$$

- **Natural example:**
  - Sets $d_1, \ldots, d_m$
  - $F(A) = |\bigcup_{i \in A} d_i|$
    (size of the covered area)
  - **<u>Claim:</u>**
    $\boldsymbol{F(A)}$ **is submodular!**



Submodularity– Diminishing returns

- **Submodularity is discrete analogue of concavity**



$\forall A \subseteq B$

Adding ***d*** to ***B*** helps less than adding it to **A**!

$$\underbrace{F(A \cup \{d\}) - F(A)}_{\text{Gain of adding } X_d \text{ to a small set}} \geq \underbrace{F(B \cup \{d\}) - F(B)}_{\text{Gain of adding } X_d \text{ to a large set}}$$

Submodularity & Concavity

- **Marginal gain:**

$$\Delta_F(d|A) = F(A \cup \{d\}) - F(A)$$

- **Submodular:**

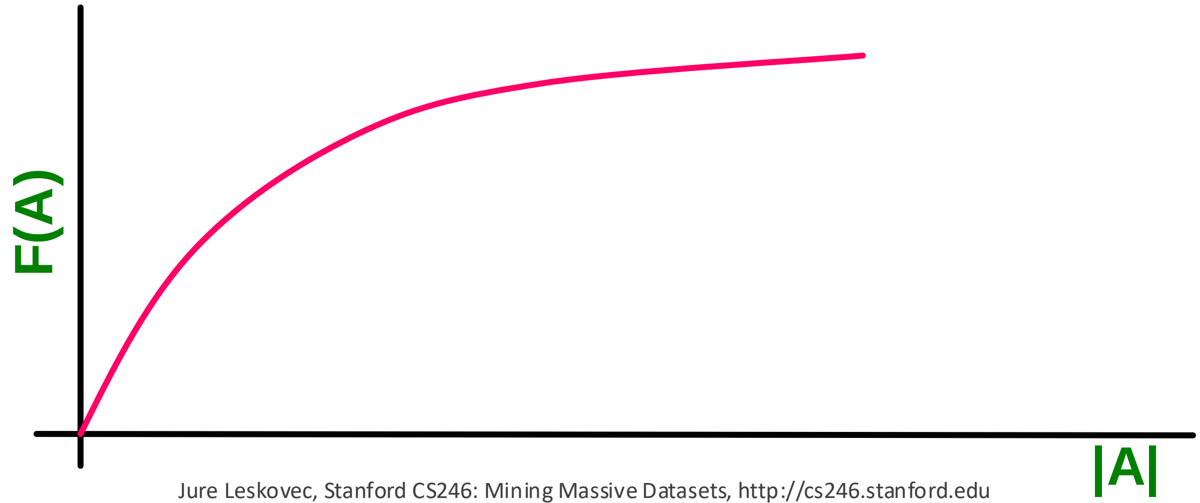
$A \subseteq B$

$$F(A \cup \{d\}) - F(A) \geq F(B \cup \{d\}) - F(B)$$

- **Concavity:**

$a \leq b$

$$f(a + d) - f(a) \geq f(b + d) - f(b)$$



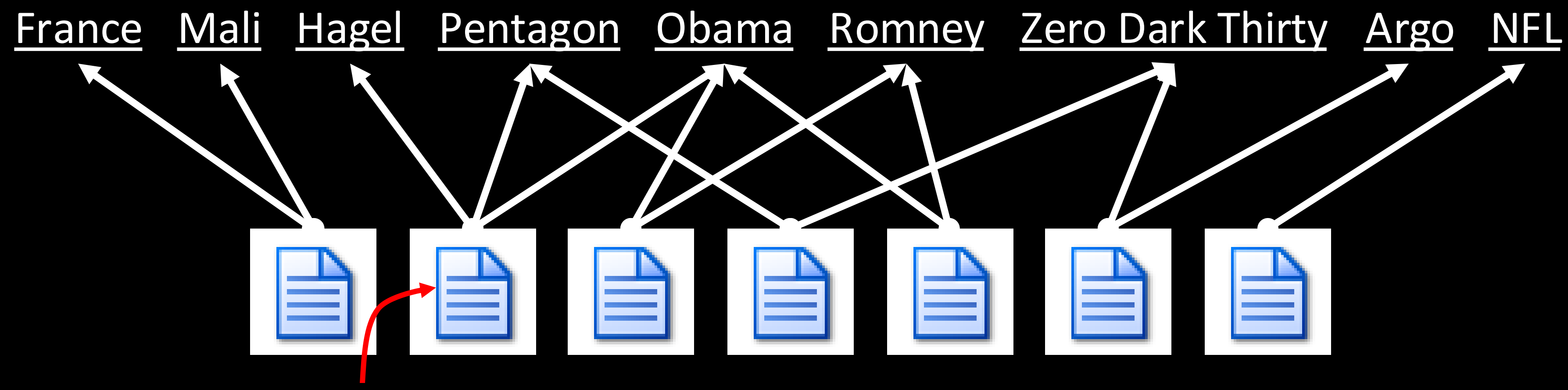
Submodularity: Useful Fact

- Let $F_1 \dots F_m$ be **submodular** and $\lambda_1 \dots \lambda_m > 0$ then $F(A) = \sum_{i=1}^{m} \lambda_i F_i(A)$ **is submodular**
  - **Submodularity is closed under non-negative linear combinations!**

- **This is an extremely useful fact:**
  - **Average of submodular functions is submodular:** $F(A) = \sum_i P(i) \cdot F_i(A)$
  - **Multicriterion optimization:** $F(A) = \sum_i \lambda_i F_i(A)$

Back to our problem

- Q: What is being covered?
- **A: Concepts** (In our case: Named entities)

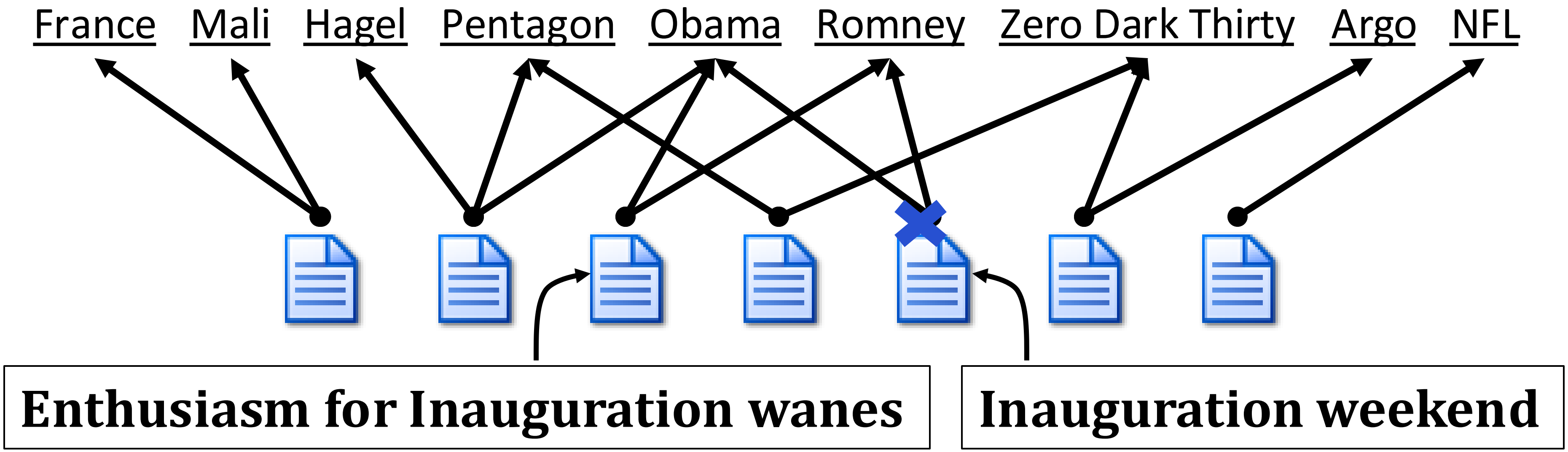


Hagel expects fight

- Q: Who is doing the covering?
- **A: Documents**

Back to our Concept Cover Problem

- **Objective:** **pick *k* docs that cover most concepts**



France Mali Hagel Pentagon Obama Romney Zero Dark Thirty Argo NFL

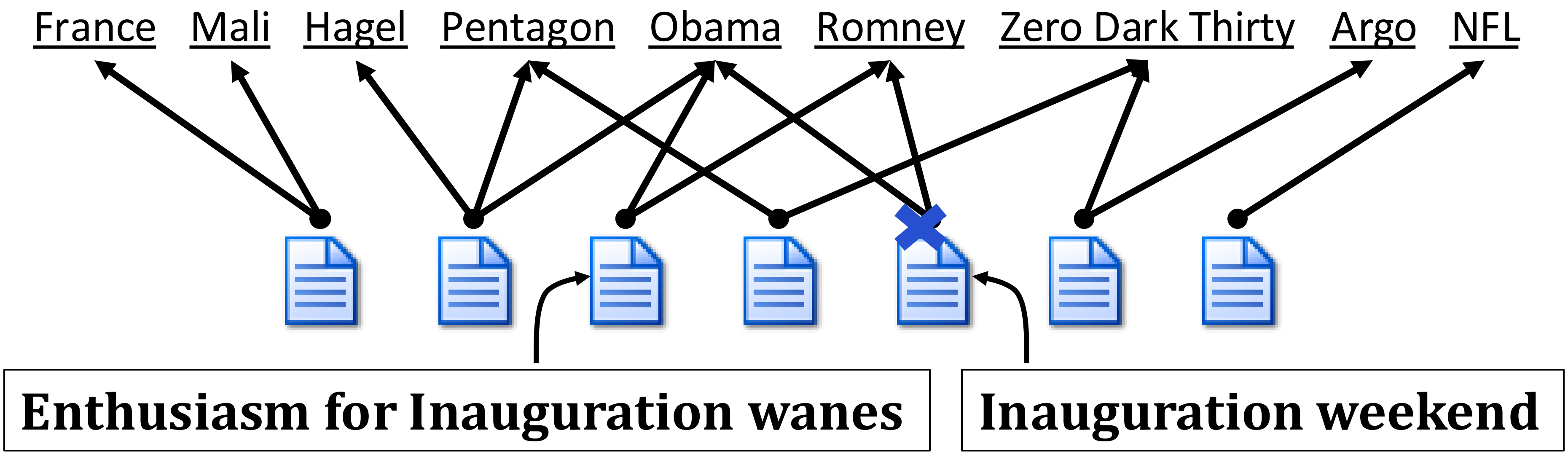
Enthusiasm for Inauguration wanes

Inauguration weekend

- ***F(A):* the number of concepts covered by *A***
  - ***Elements…concepts, Sets … concepts in docs***
  - ***F(A)*** is submodular and monotone!
  - We can use **greedy algorithm** to optimize ***F***

The Set Cover Problem

- **Objective:** pick ***k*** docs that cover most concepts



France Mali Hagel Pentagon Obama Romney Zero Dark Thirty Argo NFL

Enthusiasm for Inauguration wanes

Inauguration weekend

The good:

Penalizes redundancy

Submodular

The bad:

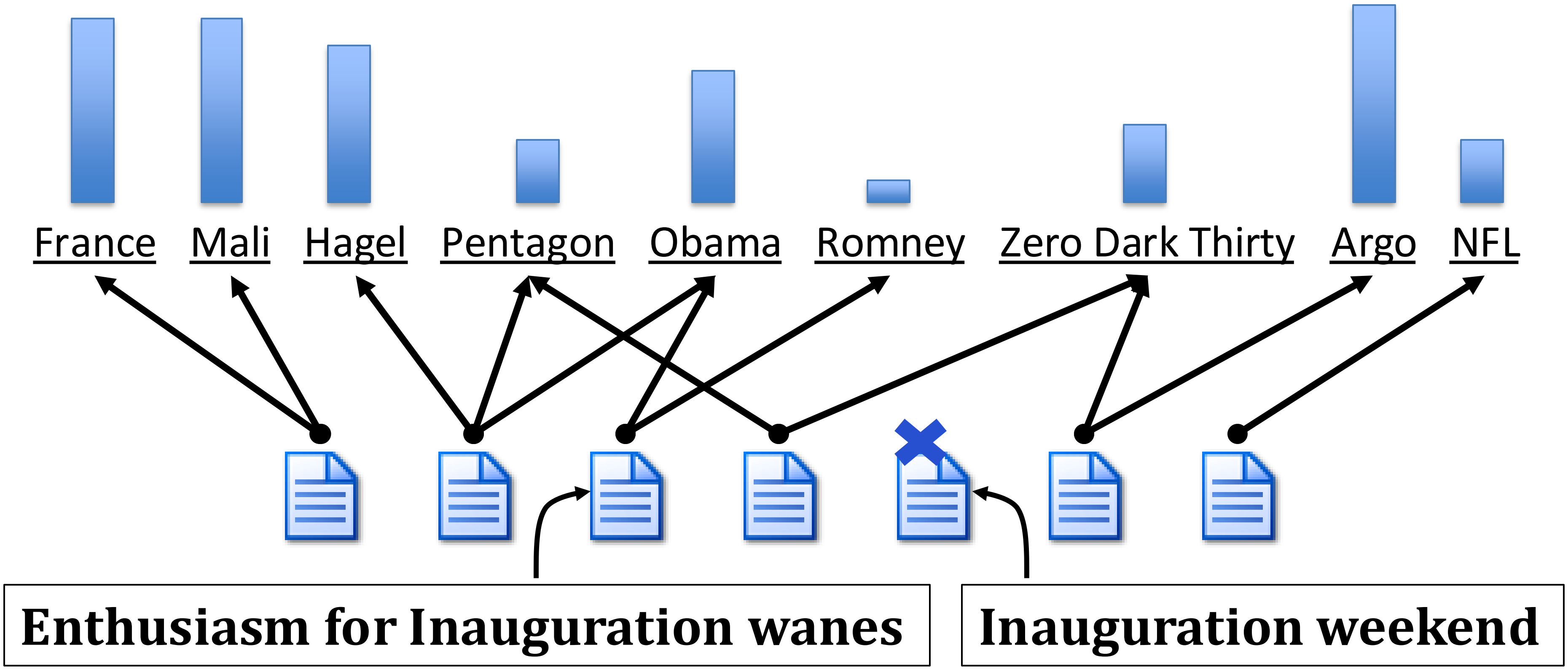
Concept importance?

All-or-nothing too harsh

Probabilistic Set Cover

Concept importance?

- **Objective:** pick ***k*** docs that cover most concepts



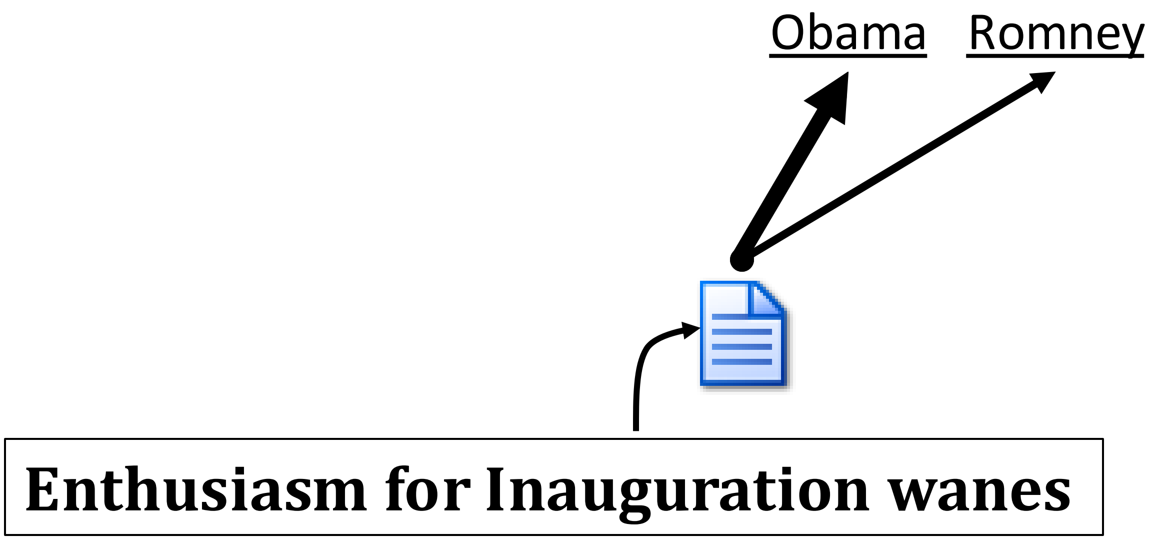
- **Each concept c has importance weight w_c**

All-or-nothing too harsh

- **Document coverage function**

$\text{cover}_d(c) =$ **probability** document **d** covers concept **c**

[e.g., how strongly **d** covers **c**]



Enthusiasm for Inauguration wanes

Probabilistic Set Cover

- **Document coverage function:**

  $\mathrm{cover}_d(c) =$ **probability** document **d** covers concept **c**

  - **Cover$_d$(c)** can also model how relevant is concept **c** for user **u**

- **Set coverage function:**

  $$\mathrm{cover}_{\mathcal{A}}(c) = 1 - \prod_{d \in \mathcal{A}} (1 - \mathrm{cover}_d(c))$$

  - Prob. that at least one document in **A** covers **c**

- **Objective:**

  concept weights

  $$\max_{\mathcal{A}:|\mathcal{A}|\leq k} F(\mathcal{A}) = \sum_c w_c \; \mathrm{cover}_{\mathcal{A}}(c)$$

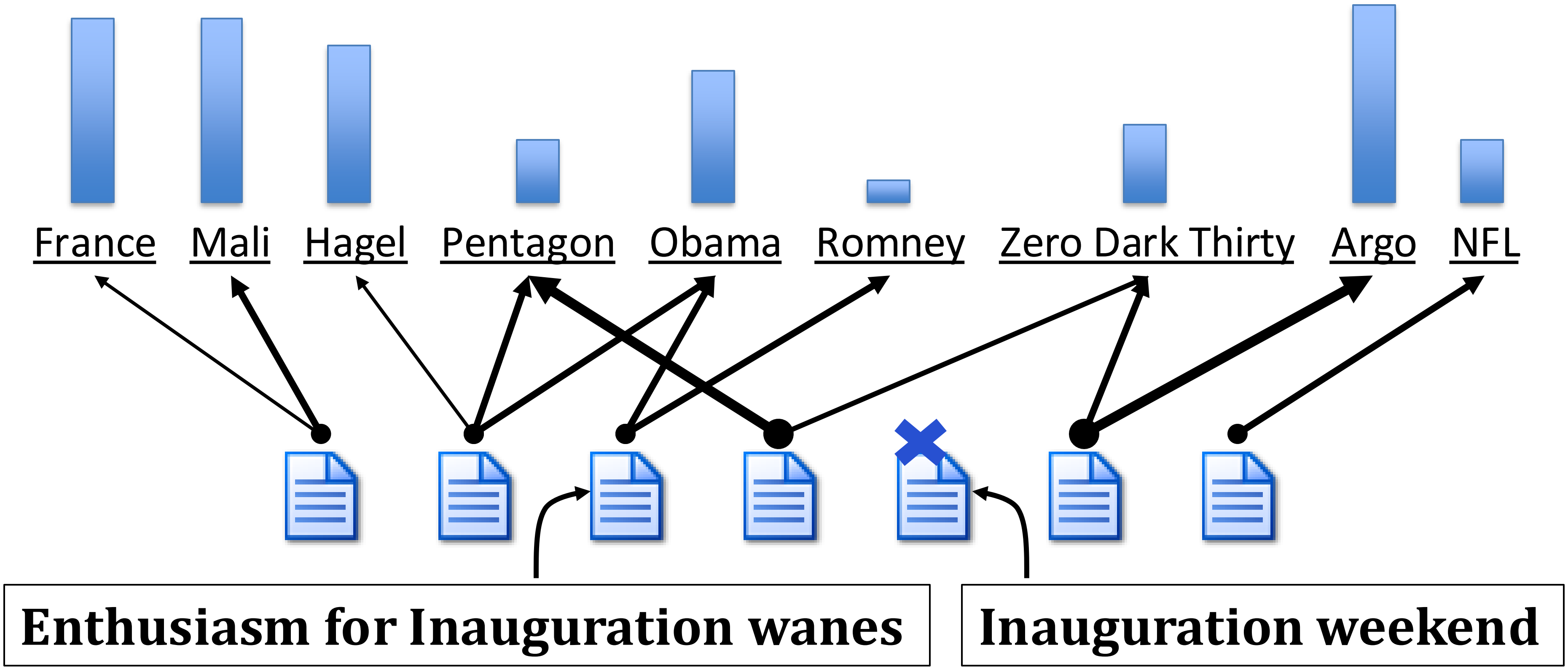
Optimizing F(A)

$$\max_{\mathcal{A}:|\mathcal{A}|\leq k} F(\mathcal{A}) = \sum_{c} w_c \ \mathrm{cover}_{\mathcal{A}}(c)$$

- The objective function is also **submodular**
  - Intuitively, it has a **diminishing returns** property
  - Greedy algorithm leads to a (1 – 1/e) ~ 63% approximation, i.e., a **near-optimal** solution

Summary: Probabilistic Set Cover

- **Objective:** pick ***k*** docs that cover most concepts



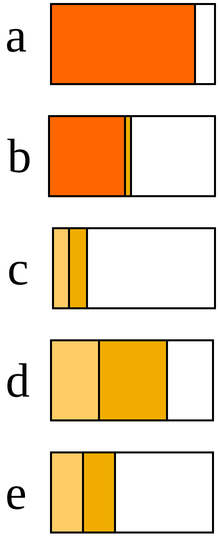
- Each concept c has importance weight w_c
- Documents partially cover concepts: $\mathbf{cover}_d(\boldsymbol{c})$

Lazy Optimization of Submodular Functions

Submodular Functions

Greedy

Marginal gain:
F(A∪x)-F(A)



Add document with highest marginal gain

- **Greedy** algorithm is **slow!**
  - At each iteration we need to re-evaluate marginal gains of **all remaining documents**
  - **Runtime $O(|D| \cdot K)$ for selecting K documents out of the set of D of them**

[Leskovec et al., KDD '07]

Speeding up Greedy

- **In round i:** So far we have $A_{i-1} = \{d_1, \dots, d_{i-1}\}$
  - Now we pick $\mathbf{d}_i = \arg\max_{d \in V} F(A_{i-1} \cup \{d\}) - F(A_{i-1})$
    - Greedy algorithm **maximizes the "marginal benefit"**

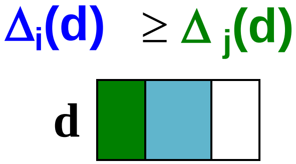
$$\Delta_i(d) = F(A_{i-1} \cup \{d\}) - F(A_{i-1})$$

- **By submodularity property:**

$$F(A_i \cup \{d\}) - F(A_i) \geq F(A_j \cup \{d\}) - F(A_j) \text{ for } i < j$$

- **Observation: By submodularity:**
  For every $d \in D$
  $\Delta_i(d) \geq \Delta_j(d)$ for $i < j$ since $A_i \subseteq A_j$

- **Marginal benefits $\Delta_i(d)$ only shrink!**
  (as i grows)



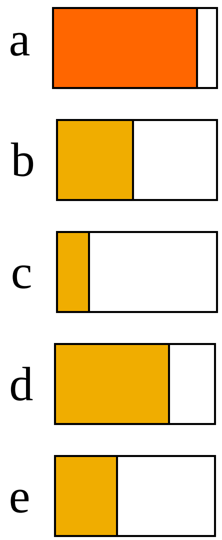
Selecting document ***d*** in step ***i*** covers more words than selecting ***d*** at step ***j*** (j>i)

[Leskovec et al., KDD '07]

Lazy Greedy

- **Idea:**
  - Use Δ_i as upper-bound on Δ_j ($j > i$)
- **Lazy Greedy**:
  - Keep an ordered list of marginal benefits Δ_i from previous iteration
  - Re-evaluate Δ_i only for top element
  - Re-sort and prune

(Upper bound on) Marginal gain Δ_1



$A_1=\{a\}$

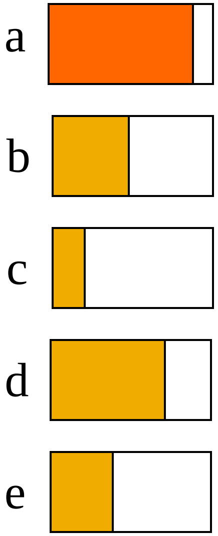
$$F(A \cup \{d\}) - F(A) \geq F(B \cup \{d\}) - F(B) \qquad A \subseteq B$$

[Leskovec et al., KDD '07]

Lazy Greedy

- **Idea:**
  - Use Δ_i as upper-bound on Δ_j ($j > i$)
- **Lazy Greedy**:
  - Keep an ordered list of marginal benefits Δ_i from previous iteration
  - Re-evaluate Δ_i only for top element
  - Re-sort and prune

Upper bound on Marginal gain Δ_2



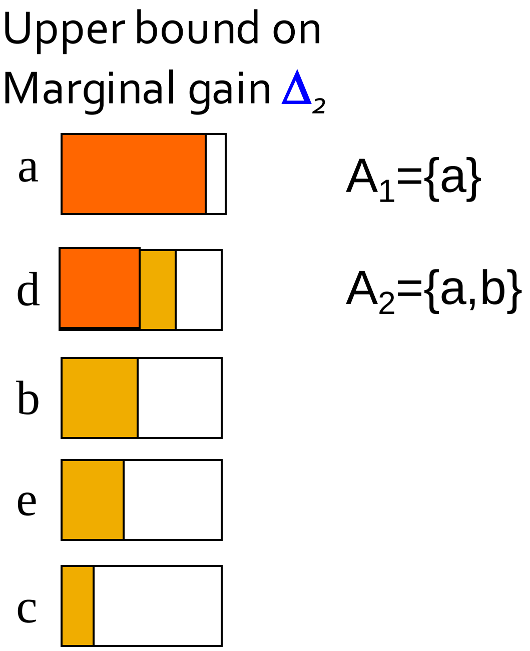
$A_1=\{a\}$

$$F(A \cup \{d\}) - F(A) \;\geq\; F(B \cup \{d\}) - F(B) \qquad A \subseteq B$$

[Leskovec et al., KDD '07]

Lazy Greedy

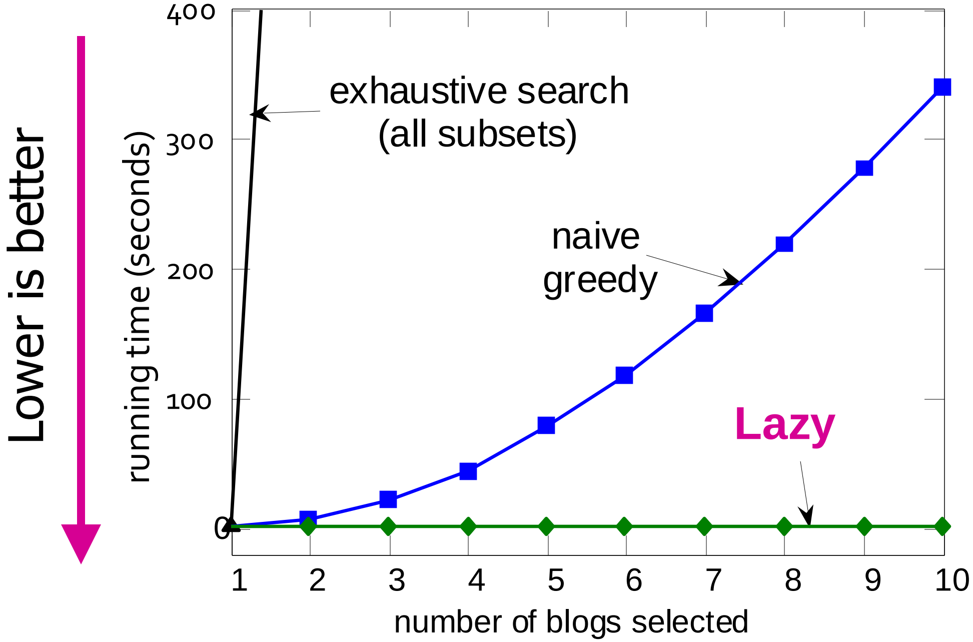
- **Idea:**
  - Use Δ_i as upper-bound on Δ_j ($j > i$)
- **Lazy Greedy:**
  - Keep an ordered list of marginal benefits Δ_i from previous iteration
  - Re-evaluate Δ_i only for top element
  - Re-sort and prune



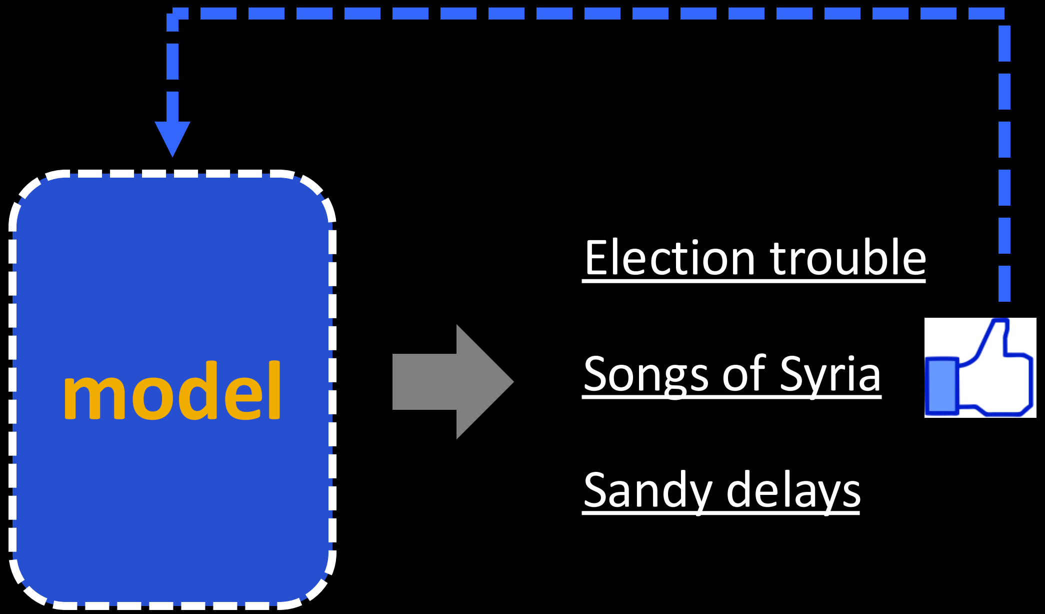
$$F(A \cup \{d\}) - F(A) \geq F(B \cup \{d\}) - F(B) \quad A \subseteq B$$

Summary so far

- **Summary so far:**
  - Diversity can be formulated as a set cover
  - Set cover is submodular optimization problem
  - Can be (approximately) solved using greedy algorithm
  - Lazy-greedy gives significant speedup



But what about **personalization**?



model

Election trouble

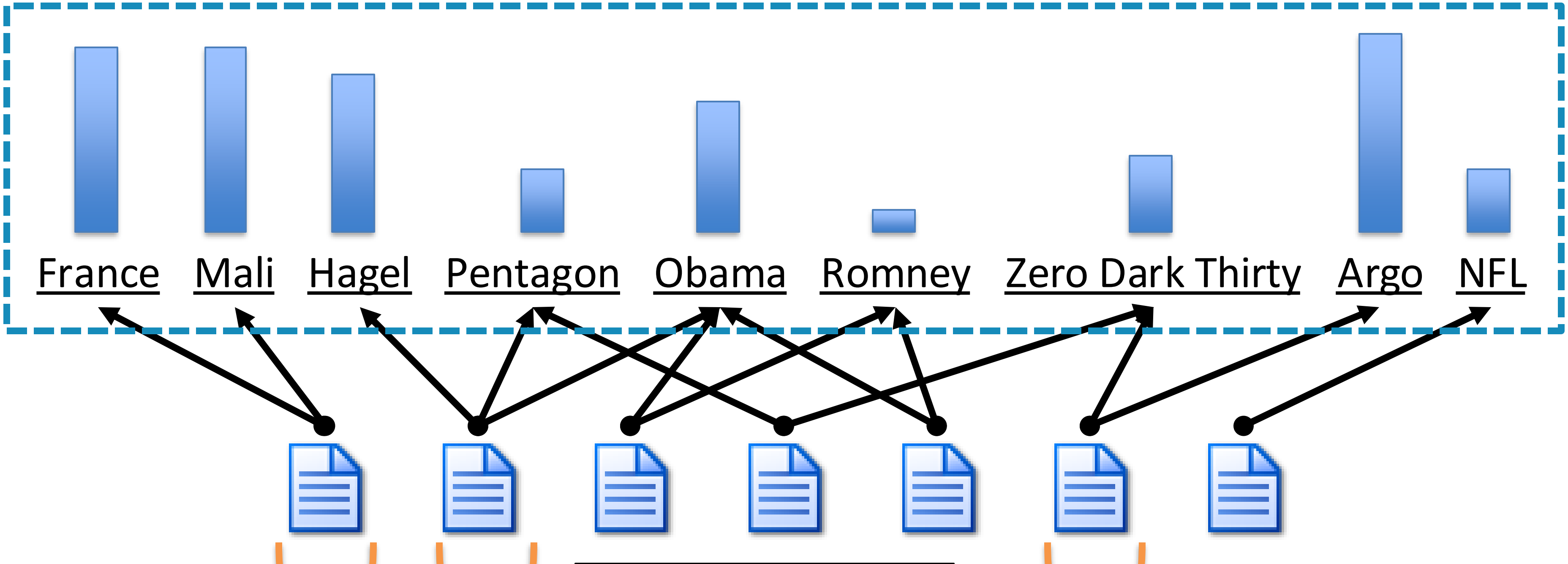
Songs of Syria

Sandy delays

Recommendations

Concept Coverage

We assumed same concept weighting for all users



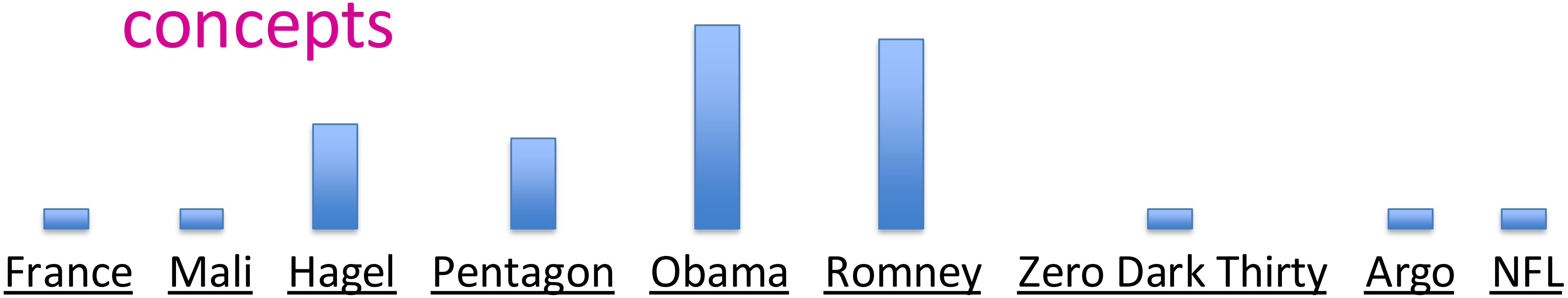
France intervenes

Chuck for Defense

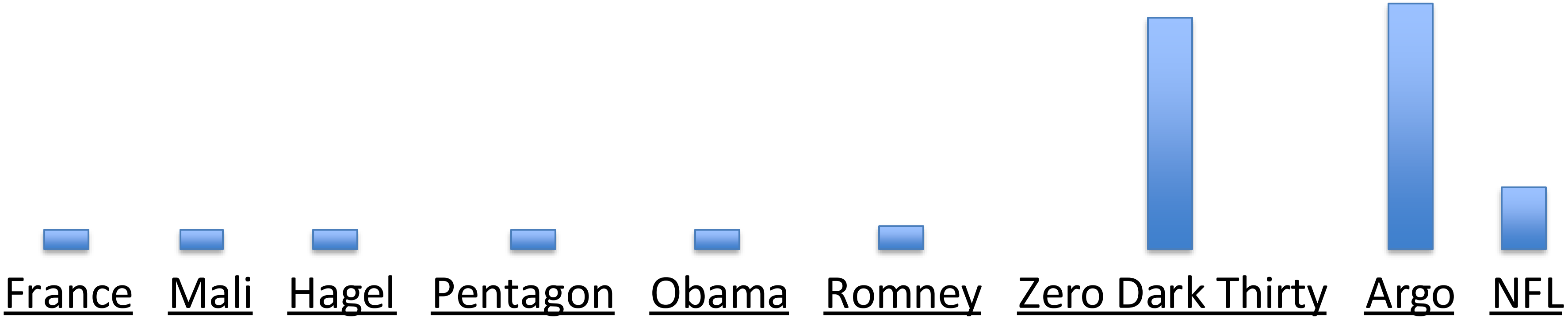
Argo wins big

Personal Concept Weights

- Each user has **different** preferences over concepts



politico



movie buff

Personal concept weights

- Assume each user ***u*** has **different** preference vector $\boldsymbol{w_c^{(u)}}$ over concepts ***c***

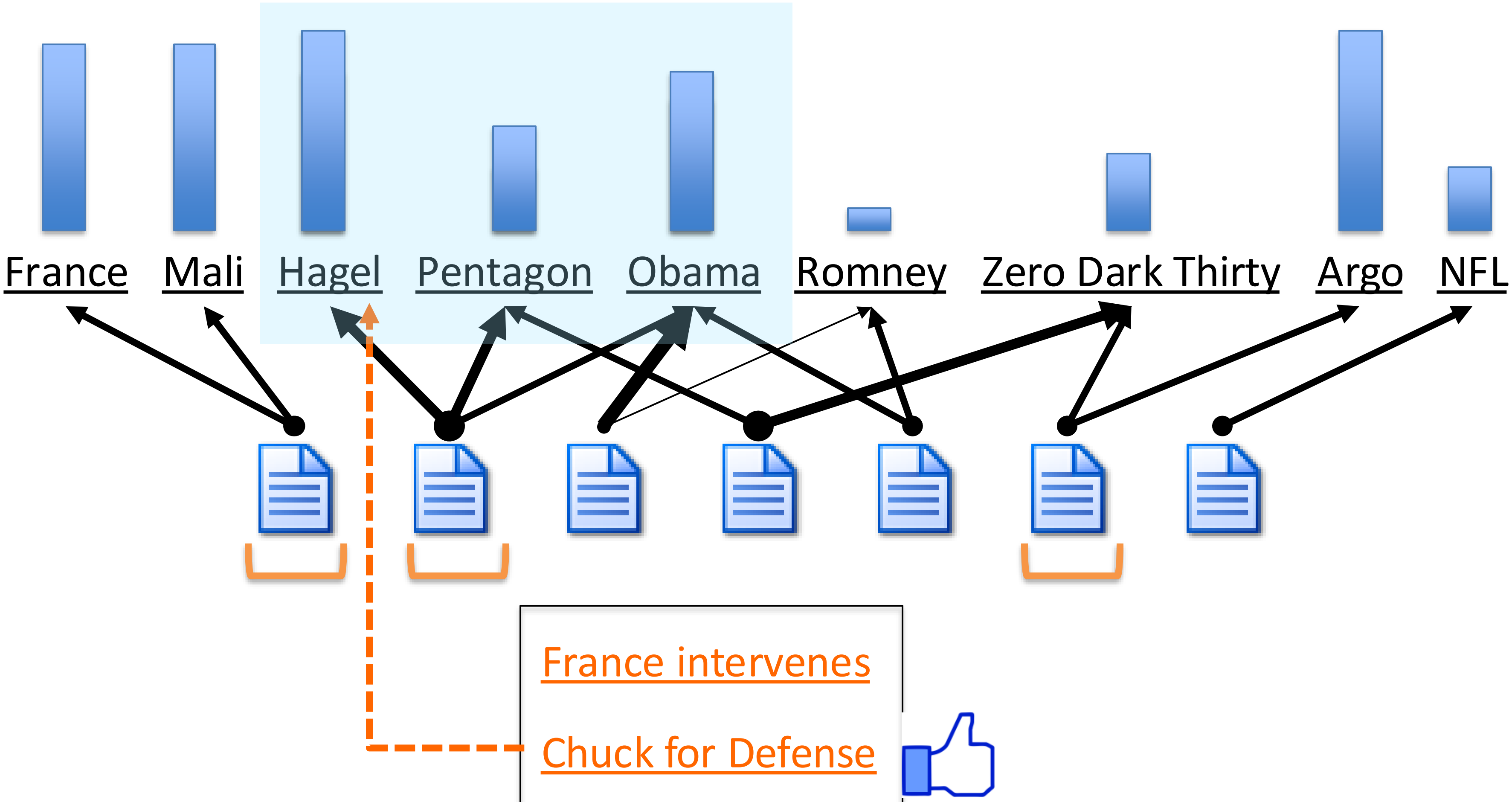
$$\max_{\mathcal{A}:|\mathcal{A}|\leq k} F(\mathcal{A}) = \sum_c w_c \;\text{cover}_{\mathcal{A}}(c)$$

$$\Big\downarrow$$

$$\max_{\mathcal{A}:|\mathcal{A}|\leq k} F(\mathcal{A}) = \sum_c w_c^{(u)} \;\text{cover}_{\mathcal{A}}(c)$$

- **Goal:** Learn personal concept weights from user feedback

Interactive Concept Coverage



France intervenes

Chuck for Defense

Argo wins big

Multiplicative Weights (MW)

- **Multiplicative Weights** algorithm
  - Assume each concept $\boldsymbol{c}$ has weight $\boldsymbol{w_c}$
  - We recommend document $\boldsymbol{d}$ and receive feedback, say $\boldsymbol{r} =$ **+1** or **-1**
  - **Update the weights:**
    - **For each $\boldsymbol{c \in X_d}$ set $\boldsymbol{w_c = \beta^r w_c}$**
      - If concept **c** appears in doc **d** and we received positive feedback **r=+1** then we increase the weight **w**$_{\mathbf{c}}$ by multiplying it by $\boldsymbol{\beta}$ $(\boldsymbol{\beta} > 1)$ otherwise we decrease the weight (divide by $\boldsymbol{\beta}$)
    - **Normalize weights so that $\sum_c \boldsymbol{w_c} = 1$**

Summary of the Algorithm

- **Steps of the algorithm:**
  1. Identify **items** to recommend from
  2. Identify **concepts** [what makes items redundant?]
  3. **Weigh** concepts by general importance
  4. Define **item-concept coverage function**
  5. **Select** items using probabilistic set cover
  6. Obtain **feedback**, **update** weights