Byzantine Fault-Tolerant Distributed Set Intersection with Redundancy and its Relationship with Byzantine Optimization

Shuo Liu & Nitin Vaidya

Georgetown University

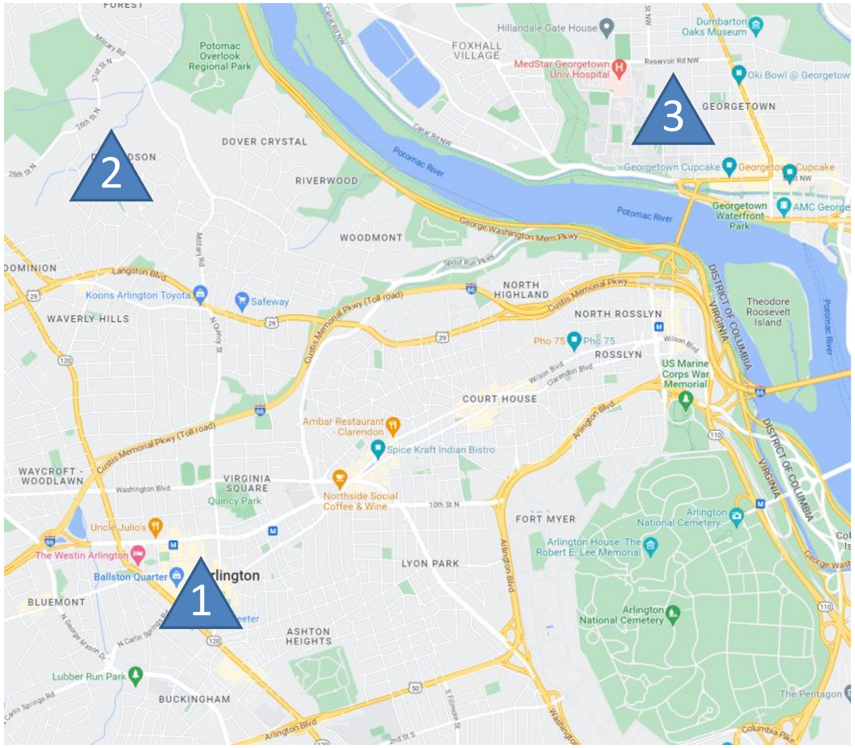
Distributed optimization

- n agents
- each agent i has $Q_i(x)$

$$\arg\min_x \sum_i Q_i(x)$$

- Many applications: machine learning, distributed sensing, …

Distributed optimization

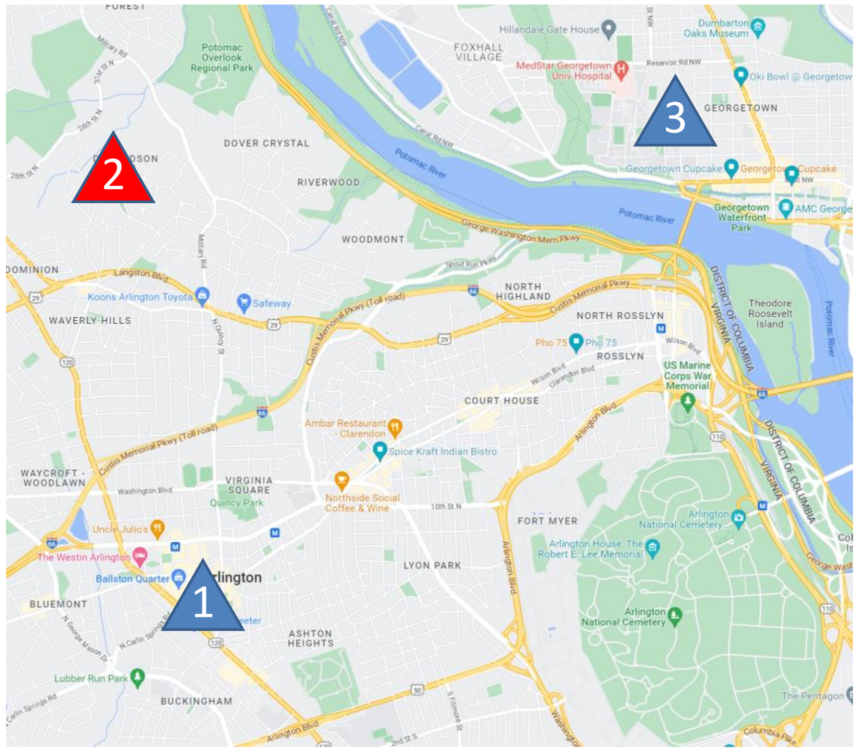


Cost examples

Money, fuel, energy…

Minimize *aggregate* cost

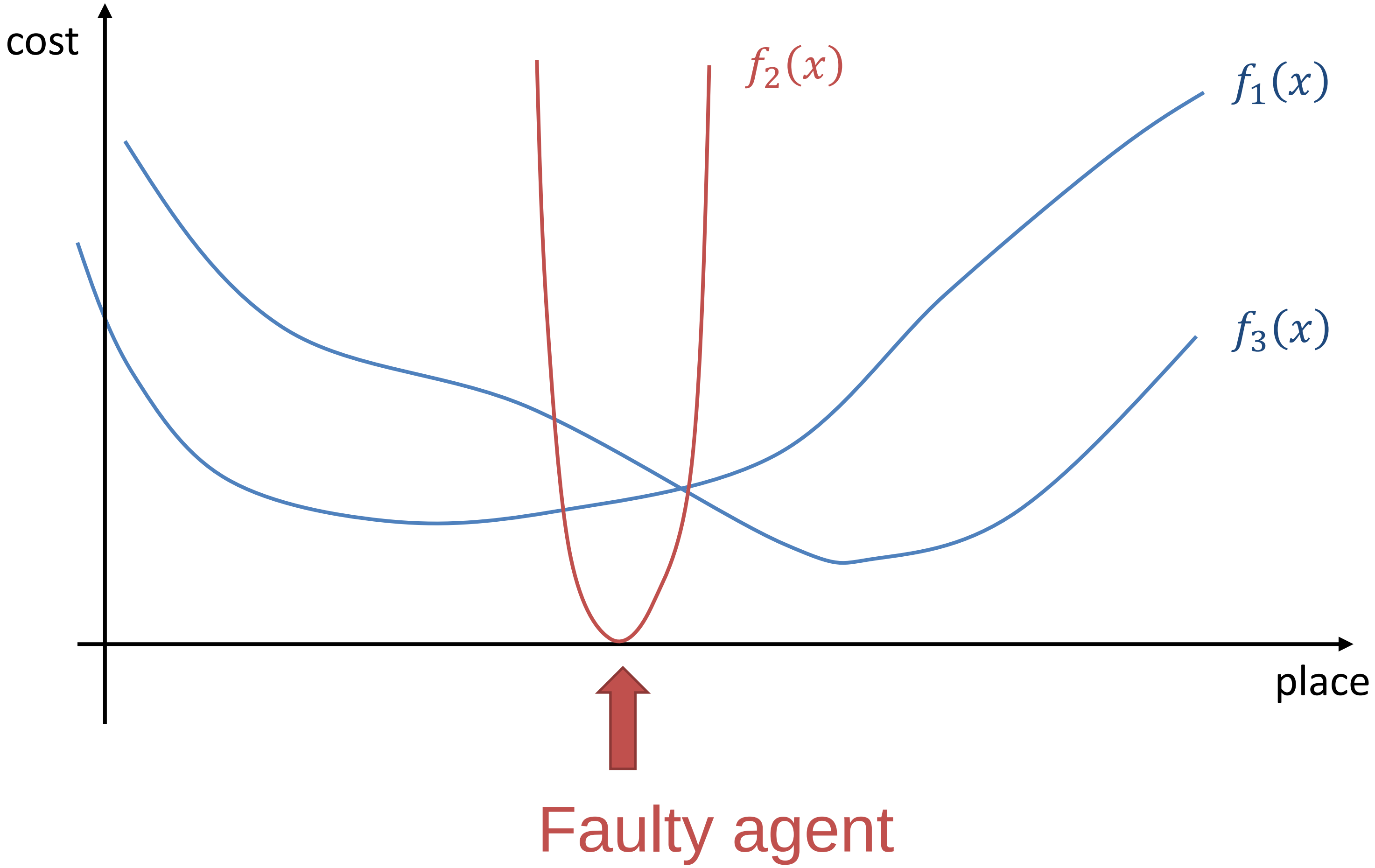
Distributed optimization



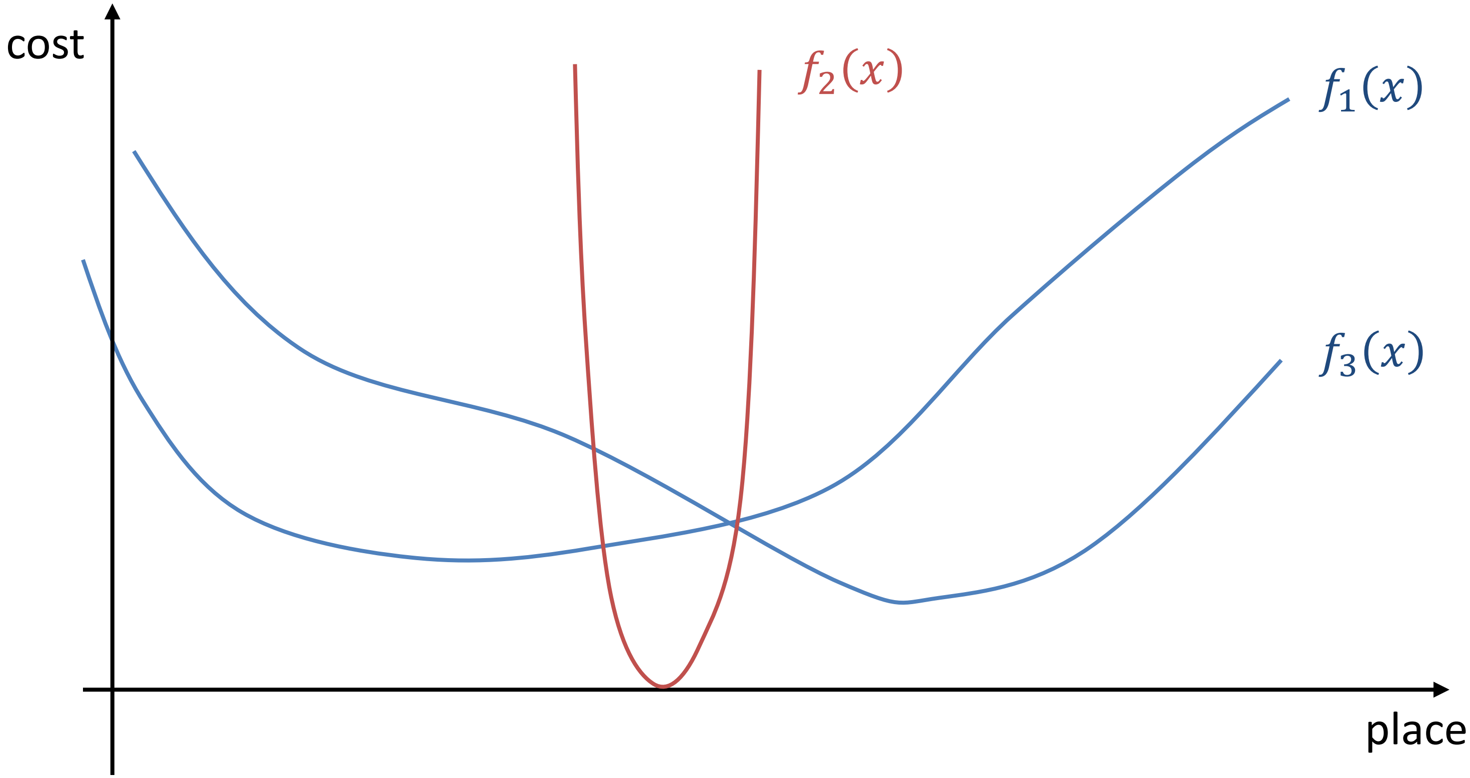
Adversarial agents

Minimize *aggregate* cost

Impact of Byzantine agents



Impact of Byzantine agents



Faulty agents can tamper the computation

[Su and Vaidya, 2016]

Byzantine optimization

- $\arg\min \sum_{\text{all}} Q_i(x)$ not useful

- Ideal goal

$$\arg\min \sum_{\text{honest } i} Q_i(x)$$

Exact Byzantine optimization

- There exist algorithms, that can solve

$$\arg\min \sum_{\text{honest } i} f_i(x)$$

exactly with redundancy in cost functions

[Gupta and Vaidya, 2020]

Exact Byzantine optimization

- With sufficient **redundancy**,
  $\arg\min \sum_{\text{honest } i} f_i(x)$ can be solved exactly

Exact Byzantine optimization

- With sufficient **redundancy**, $\arg\min \sum_{\text{honest } i} f_i(x)$ can be solved exactly

$2f$-redundancy

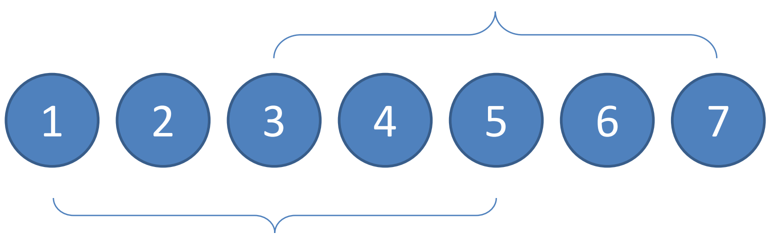
Aggregate of every $n - f$ functions has the same minimum set as aggregate of every $n - 2f$ functions

[Gupta and Vaidya, 2020]

$2f$-redundancy

Aggregate of all n functions has the same minimum set as aggregate of every $n - 2f$ functions

$$X_1 = \arg\min \sum_{i=3}^{7} f_i(x)$$



$$X_2 = \arg\min \sum_{i=1}^{5} f_i(x)$$

$$X = \arg\min \sum_{i=1}^{7} f_i(x)$$

$n = 7$
$f = 1$

$$X = X_1 = X_2 = \cdots$$

[Gupta and Vaidya, 2020]

$2f$-redundancy

Aggregate of all n functions has the same minimum set as aggregate of every $n - 2f$ functions

$2f$-redundancy $\Rightarrow$ Exact fault-tolerance

$\arg\min \sum_{\text{honest } i} Q_i(x)$
can be computed

[Gupta and Vaidya, 2020]

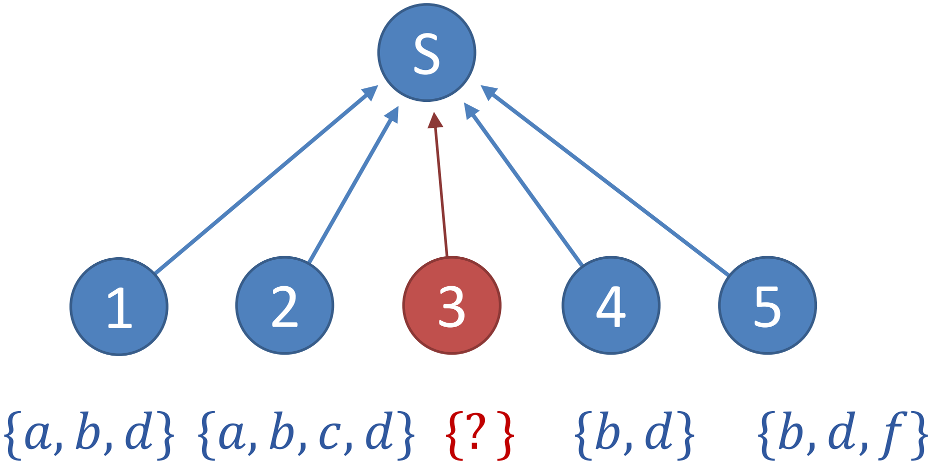
Byzantine Optimization → Byzantine Set Intersection

Byzantine set intersection

- Each agent i has an input set X_i
- Up to f agents may be Byzantine
- Output $\bigcap_{\text{honest } i} X_i$

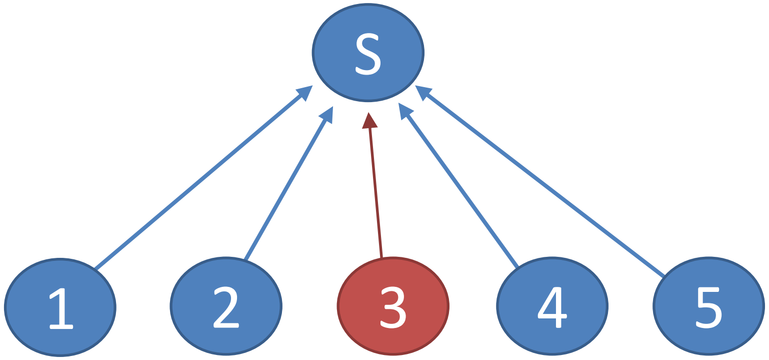
Byzantine set intersection

$n = 5$
$f = 1$



Byzantine set intersection

$n = 5$
$f = 1$



$\{a, b, d\}$ $\{a, b, c, d\}$ $\{?\}$ $\{b, d\}$ $\{b, d, f\}$

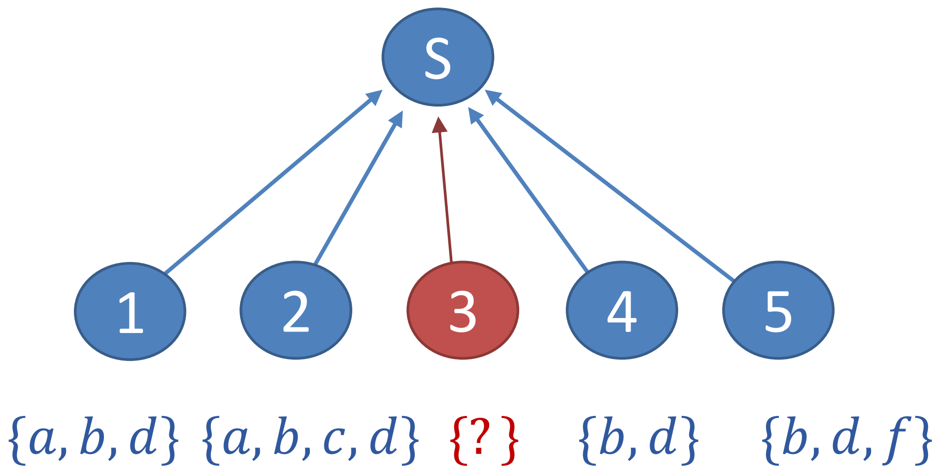
$X_3 = \{a, b\}$ $\quad \bigcap_{\text{all } i} X_i = \{b\}$

$X_3 = \{a, e\}$ $\quad \bigcap_{\text{all } i} X_i = \emptyset$

Byzantine set intersection

$n = 5$

$f = 1$



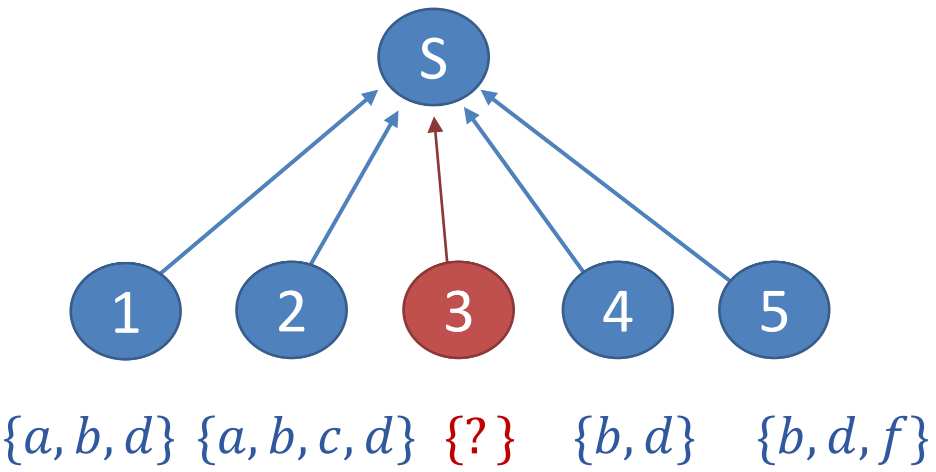
$$\bigcap_{\text{honest } i} X_i = \{b, d\}$$

$X_3 = \{a, b\}$ $\quad \bigcap_{\text{all } i} X_i = \{b\}$

$X_3 = \{a, e\}$ $\quad \bigcap_{\text{all } i} X_i = \emptyset$

Byzantine set intersection

$n = 5$
$f = 1$



$$\bigcap_{\text{honest } i} X_i = \{b, d\}$$

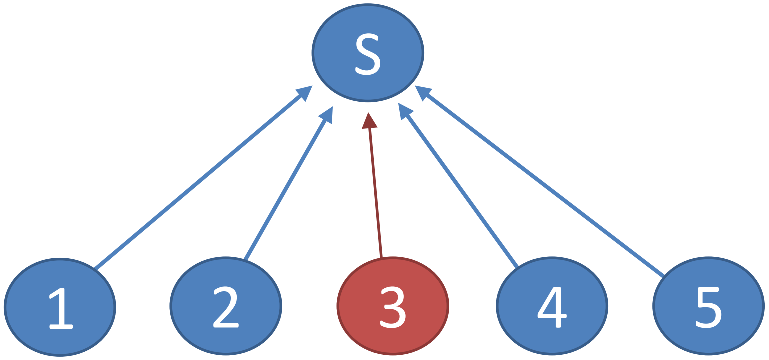
$X_3 = \{a, b\}$ $\quad \bigcap_{\text{all } i} X_i = \{b\}$

$X_3 = \{a, e\}$ $\quad \bigcap_{\text{all } i} X_i = \emptyset$

Faulty agents can make intersection smaller

Byzantine set intersection

$n = 5$
$f = 1$



$\{a, b, d\}$ $\{a, b, c, d\}$ $\{?\}$ $\{b, d\}$ $\{b, d, f\}$

$$\bigcap_{\text{honest } i} X_i = \{b, d\}$$

$X_3 = \{a, b\}$ $\quad \bigcap_{\text{all } i} X_i = \{b\}$

$X_3 = \{a, e\}$ $\quad \bigcap_{\text{all } i} X_i = \emptyset$

Make each value redundant enough
so that we can avoid removing it

Optimization → Set Intersection

- $2f$-redundancy

[Gupta & Vaidya, 2020]

Aggregate of all n functions has the same minimum set as aggregate of every $n - 2f$ functions

Optimization → Set Intersection

- $2f$-redundancy

[Gupta & Vaidya, 2020]

Aggregate of all n functions has the same minimum set as aggregate of every $n - 2f$ functions

- Equivalent to $2f$-set-redundancy

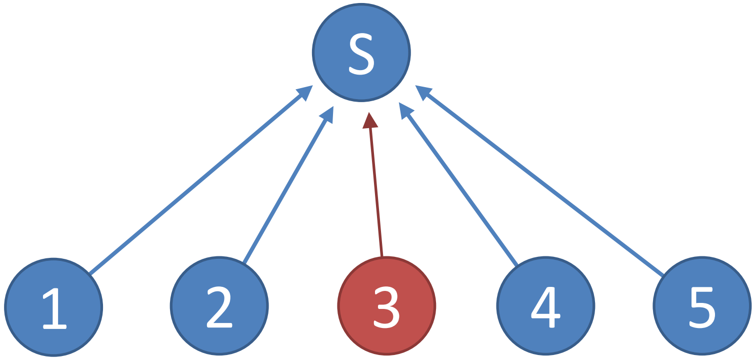
The **intersections** of sets $\bigcap_{i \in S} X_i$ of every $\geq n - 2f$ agents S are the same as $\bigcap_{i \in [n]} X_i$ of all n agents

$2f$-set-redundancy

The intersections of sets $\bigcap_{i \in S} X_i$ of every $\geq n - 2f$ agents S are the same as $\bigcap_{i \in [n]} X_i$ of all n agents

Server-based system

$2f$-set-redundancy is sufficient



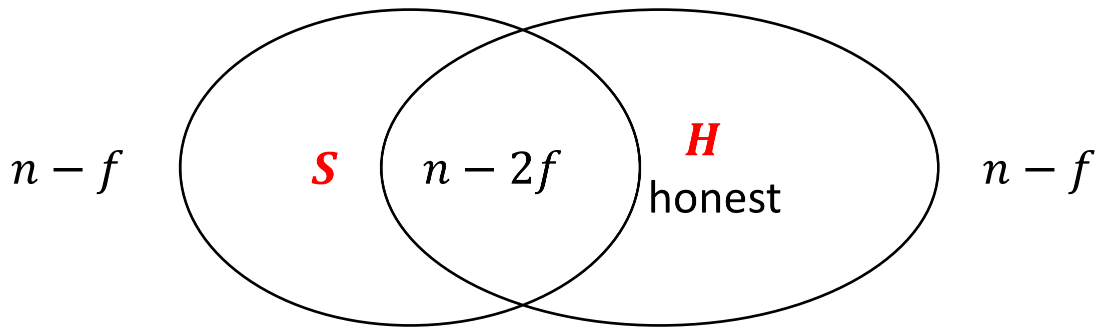
Server-based algorithm with $2f$-set-redundancy

- Find a subset of $n - f$ agents S such that the intersection of the input sets of any $n - 2f$ agents in S is the same

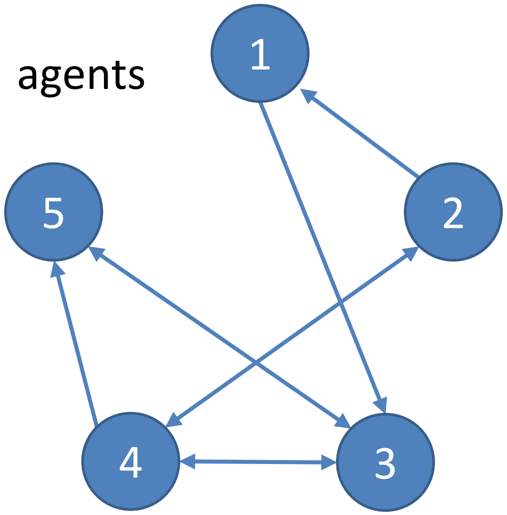
- Output the intersection of the input sets of agents in set S

Server-based algorithm with $2f$-set-redundancy

- Find a subset of $n - f$ agents S such that the intersection of the input sets of any $n - 2f$ agents in S is the same
- Output the intersection of the input sets of agents in set S



Decentralized system



- Relationships between communication graphs and redundancy

Decentralized system

- Find relationship between communication graphs and redundancy
  - Given $2f$-set-redundancy, what communication graph?
  - Given communication graph, what redundancy?

Two types of algorithms

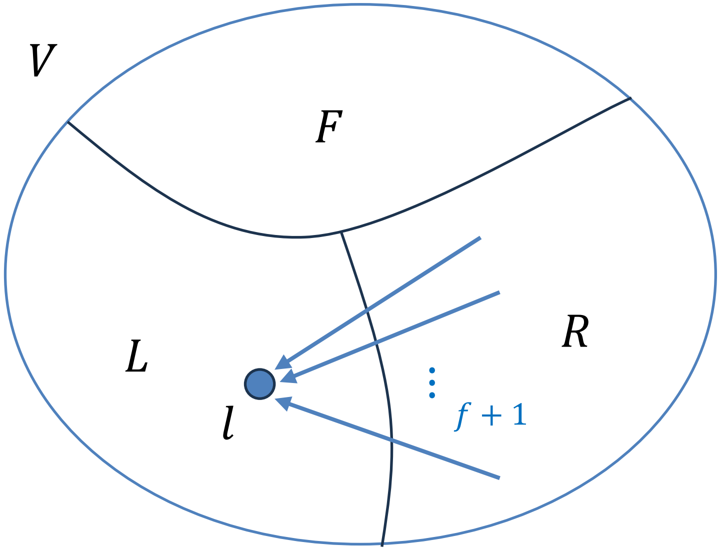
- Constrained algorithms
- Unconstrained algorithms

Constrained algorithms

- Iterative algorithms
- Each agent can only maintain a local set
- Each iteration can only send, receive, and update the local set

Necessary condition with $2f$-set-redundancy

For node partition L, R, F of V with $|F| \leq f$ if $|R| \geq f+1$, there exists $l \in L$ with $\geq f+1$ incoming neighbors in R



$|F| \leq f$

$|R| \geq f+1$

Necessary condition with $2f$-set-redundancy

The necessary condition can also be derived using previous results for *certified propagation*

[Tseng et al., 2015]

Sufficiency:
Constrained algorithm with $2f$-set-redundancy

- In each iteration, agents send their sets to outgoing neighbors

- Receive sets from neighbors

- Remove y local set if at least $f+1$ sets don't include y

Sufficiency:
Constrained algorithm with $2f$-set-redundancy

- In each iteration, agents send their sets to outgoing neighbors

This algorithm only practical for **finite** sets

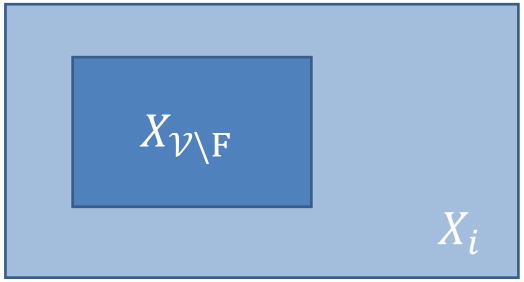
- Remove y local set if at least $f+1$ sets don't include y

Algorithm for a special case

- Input sets X_i's are closed hyperrectangles
- X_i can be represented by two points

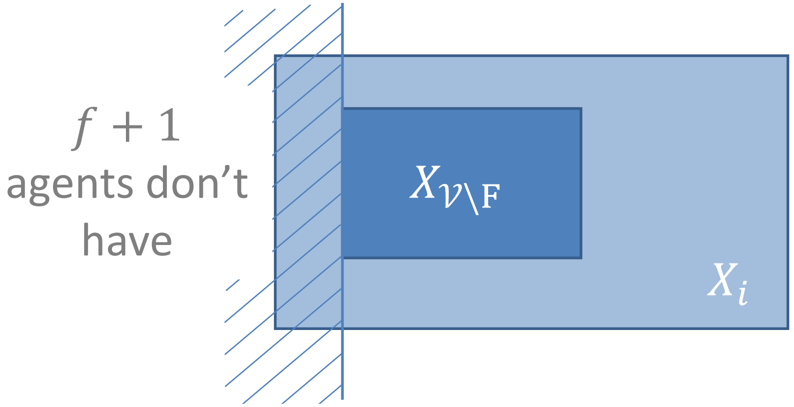
Algorithm for a special case

- $X_{\mathcal{V}\setminus\mathrm{F}}$ is also closed hyperrectangle



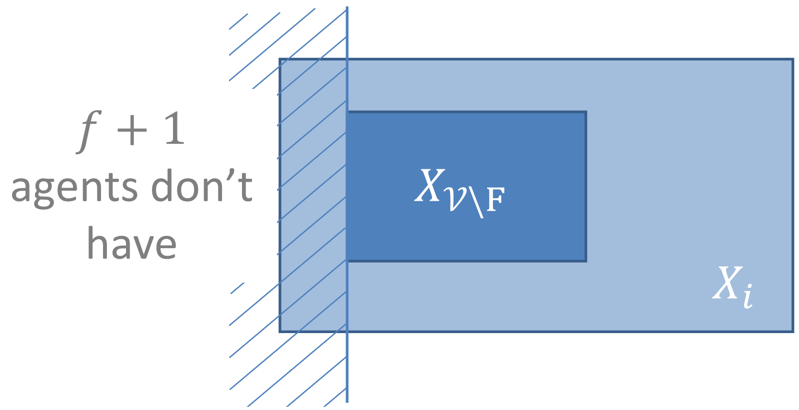
Algorithm for a special case

- $X_{\mathcal{V}\backslash \mathrm{F}}$ is also closed hyperrectangle
- $2f$-set-redundancy implies $\geq f+1$ honest agents don't have points outside each surface of $X_{\mathcal{V}\backslash \mathrm{F}}$



Algorithm for a special case

- $X_{\mathcal{V}\setminus\mathrm{F}}$ is also closed hyperrectangle
- $2f$-set-redundancy implies $\geq f+1$ honest agents don't have points outside each surface of $X_{\mathcal{V}\setminus\mathrm{F}}$
- Each agent that has points in this region can remove them in finite iterations



Constraints on the sets can be exploited to improve efficiency

Byzantine set intersection
→ Byzantine optimization

Set intersection → optimization

In a decentralized system, conditions for **Byzantine set intersection** are also

- Sufficient for **Byzantine optimization**

- Necessary when assuming unique minimum point

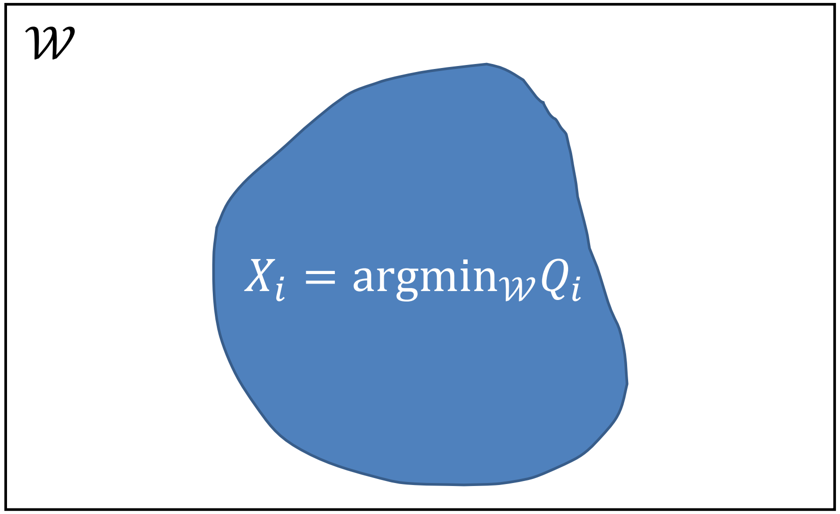
Also need to address **<u>infinite</u>** sets

Approximate algorithm over area $\mathcal{W}$

- Find points on ϵ-grid with gradients $\leq \mathcal{O}(\sqrt{d}\epsilon)$ in $\mathcal{W}$
- Byzantine set intersection on sampled points
- Output is $\mathcal{O}(\epsilon)$-bounded to true minimum

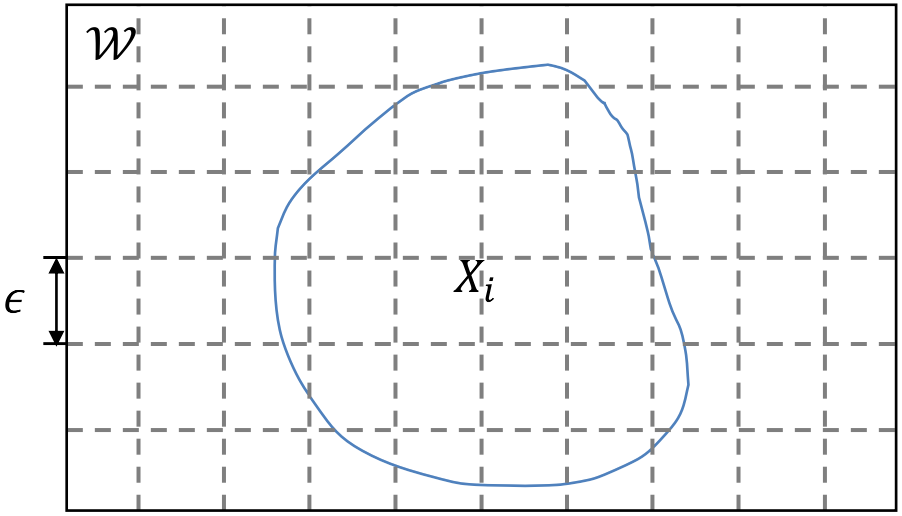
Approximate algorithm over area $\mathcal{W}$

- Find points on ϵ-grid with gradients $\leq \mathcal{O}(\sqrt{d}\epsilon)$ in $\mathcal{W}$
- Byzantine set intersection on sampled points
- Output is $\mathcal{O}(\epsilon)$-bounded to true minimum



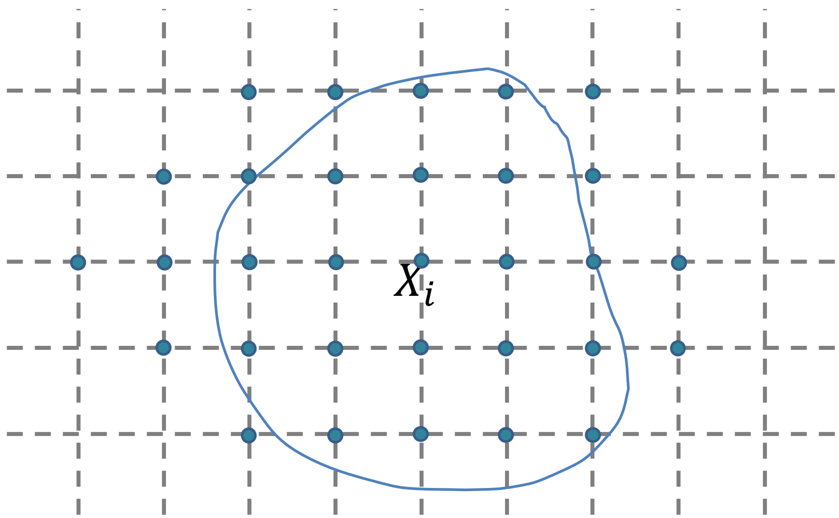
Approximate algorithm over area $\mathcal{W}$

- Find points on ϵ-grid with gradients $\leq \mathcal{O}(\sqrt{d}\epsilon)$ in $\mathcal{W}$
- Byzantine set intersection on sampled points
- Output is $\mathcal{O}(\epsilon)$-bounded to true minimum



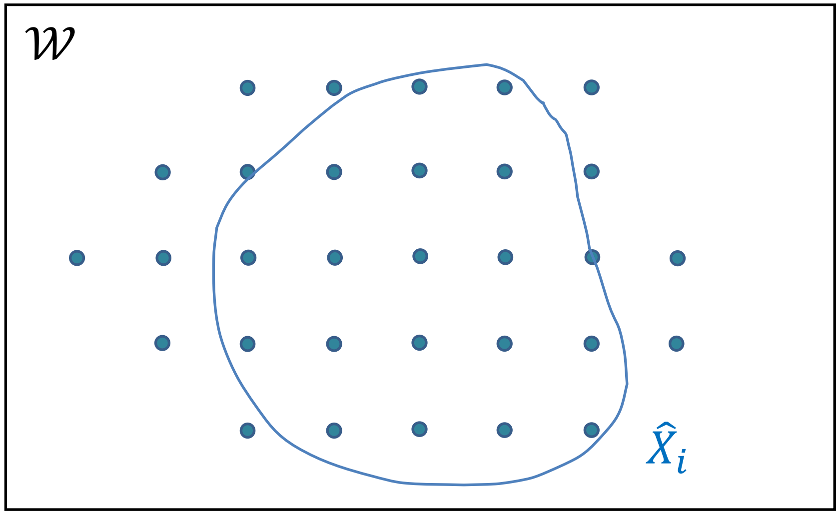
Approximate algorithm over area $\mathcal{W}$

- Find points on ϵ-grid with gradients $\leq \mathcal{O}(\sqrt{d}\epsilon)$ in $\mathcal{W}$
- Byzantine set intersection on sampled points
- Output is $\mathcal{O}(\epsilon)$-bounded to true minimum



Approximate algorithm over area $\mathcal{W}$

- Find points on ϵ-grid with gradients $\leq \mathcal{O}(\sqrt{d}\epsilon)$ in $\mathcal{W}$
- Byzantine set intersection on sampled points
- Output is $\mathcal{O}(\epsilon)$-bounded to true minimum

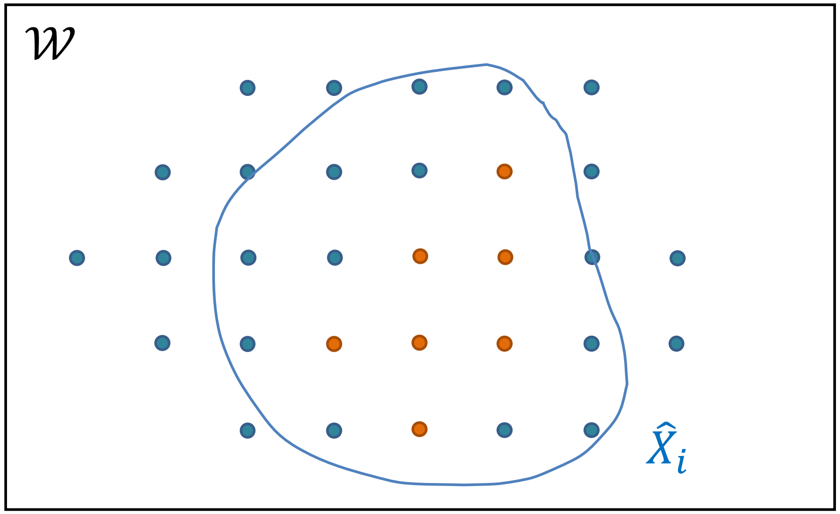


Bounded gradients

Finite points in $\hat{X}_i$

Approximate algorithm over area $\mathcal{W}$

- Find points on ϵ-grid with gradients $\leq \mathcal{O}(\sqrt{d}\epsilon)$ in $\mathcal{W}$
- Byzantine set intersection on sampled points
- Output is $\mathcal{O}(\epsilon)$-bounded to true minimum



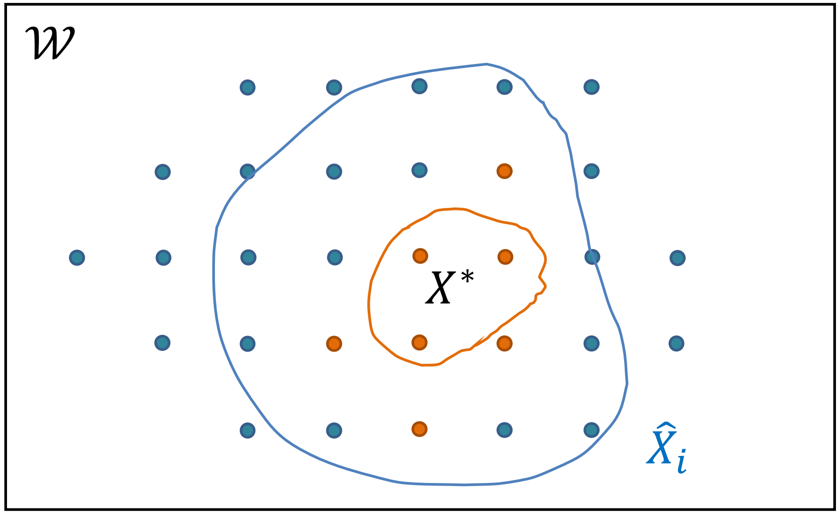
Bounded gradients

Finite points in $\hat{X}_i$

Approximate algorithm over area $\mathcal{W}$

- Find points on ϵ-grid with gradients $\leq \mathcal{O}(\sqrt{d}\epsilon)$ in $\mathcal{W}$
- Byzantine set intersection on sampled points
- Output is $\mathcal{O}(\epsilon)$-bounded to true minimum

assuming Lipschitz gradients and strongly convex aggregate functions



Bounded gradients

Finite points in $\hat{X}_i$

Summary

- Byzantine set intersection
  - Necessary and sufficient conditions
- Set intersection → Byzantine optimization
  - Algorithm using grid sampling