

Convex Consensus with Asynchronous Fallback

Andrei Constantinescu

Diana Ghinea

Roger Wattenhofer

Floris Westermann



Distributed Computing Group
ETH zürich



1.

Why Convex Consensus?

(Almost) everything is convex if you believe!



Consensus (Byzantine Agreement)



Consensus (Byzantine Agreement)

n parties



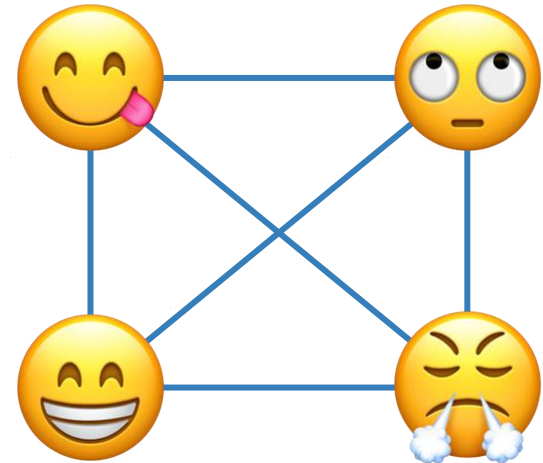
Consensus (Byzantine Agreement)

n parties



Consensus (Byzantine Agreement)

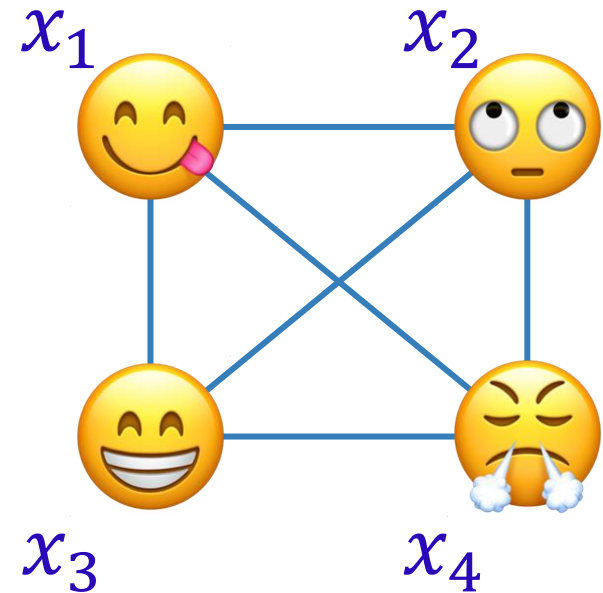
n parties



(authenticated bidirectional channels)

Consensus (Byzantine Agreement)

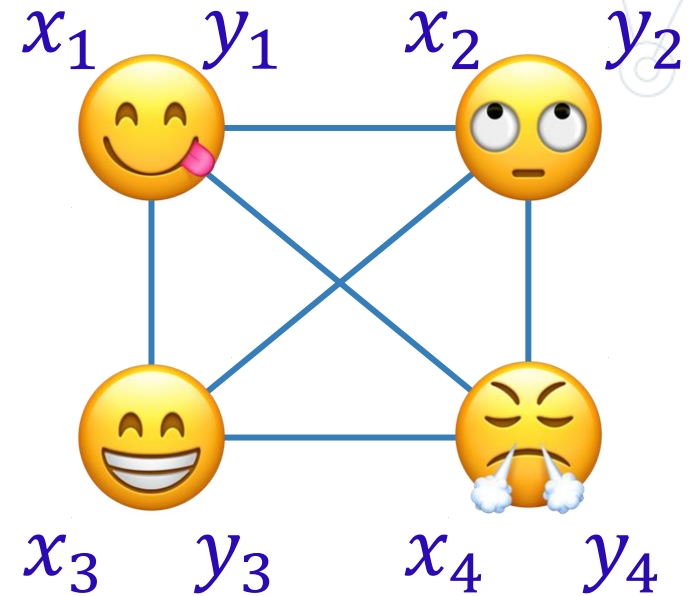
n parties with inputs x_1, \dots, x_n



(authenticated bidirectional channels)

Consensus (Byzantine Agreement)

n parties with inputs x_1, \dots, x_n giving outputs y_1, \dots, y_n

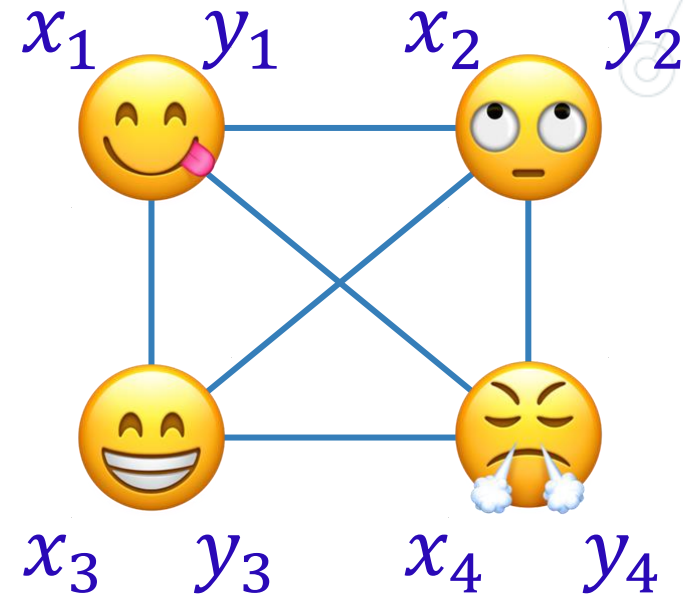


(authenticated bidirectional channels)

Consensus (Byzantine Agreement)

n parties with inputs x_1, \dots, x_n giving outputs y_1, \dots, y_n

Unknown $\leq t$ parties are *byzantine*

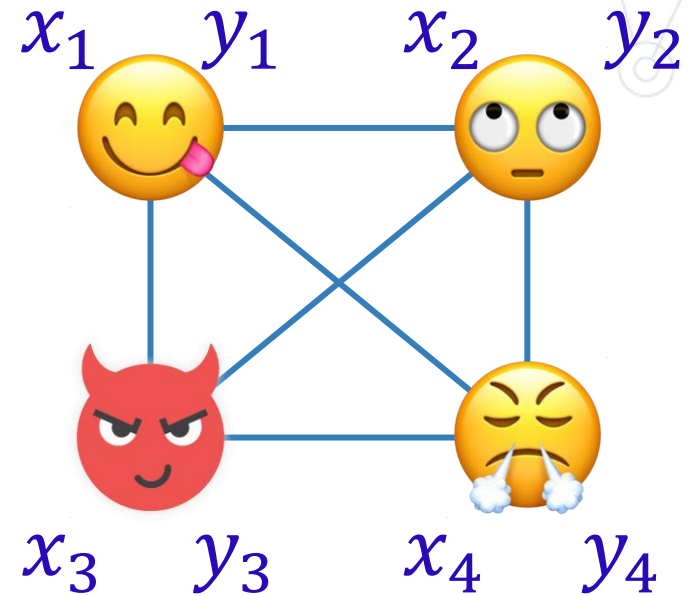


(authenticated bidirectional channels)

Consensus (Byzantine Agreement)

n parties with inputs x_1, \dots, x_n giving outputs y_1, \dots, y_n

Unknown $\leq t$ parties are *byzantine*

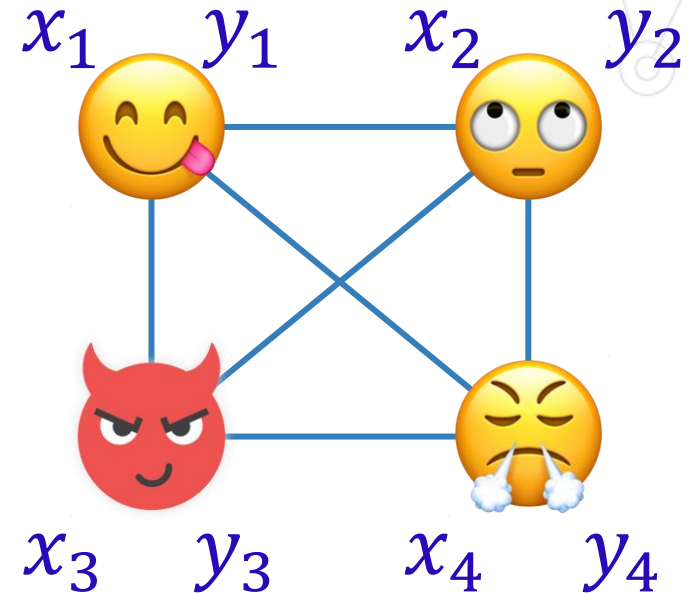


(authenticated bidirectional channels)

Consensus (Byzantine Agreement)

n parties with inputs x_1, \dots, x_n giving outputs y_1, \dots, y_n

Unknown $\leq t$ parties are *byzantine*
(other $\geq n - t$ are *honest*)



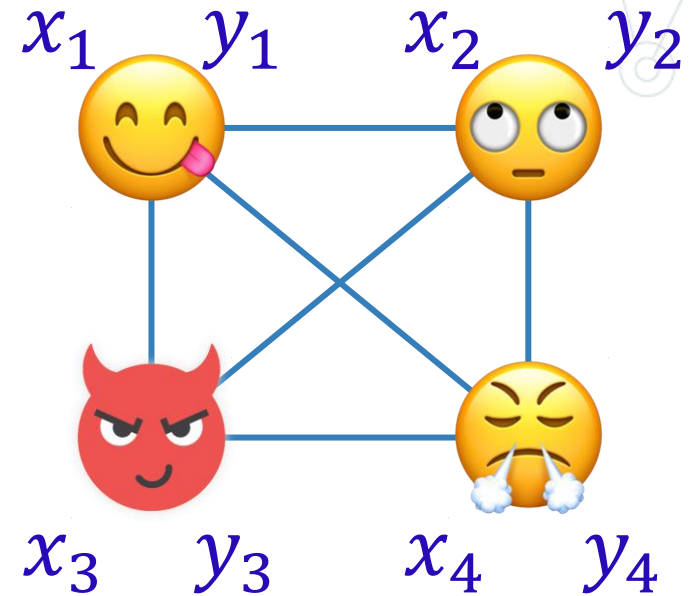
(authenticated bidirectional channels)

Consensus (Byzantine Agreement)

n parties with inputs x_1, \dots, x_n giving outputs y_1, \dots, y_n

Unknown $\leq t$ parties are *byzantine*
(other $\geq n - t$ are *honest*)

Agreement: *Honests* output same y



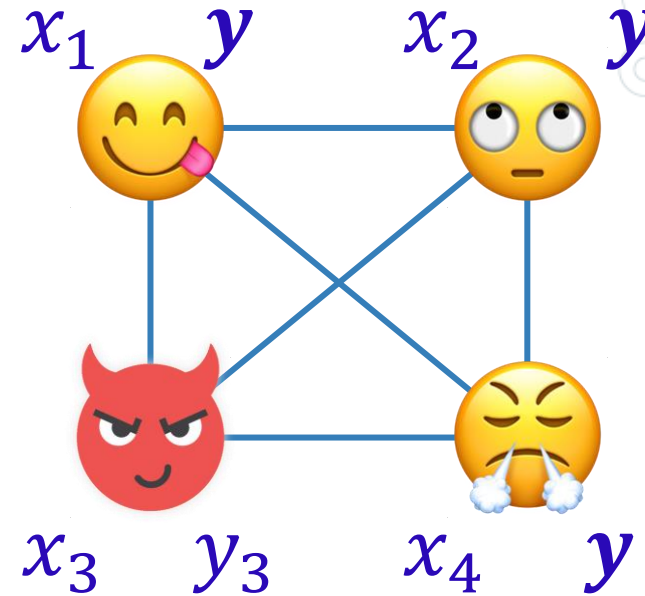
(authenticated bidirectional channels)

Consensus (Byzantine Agreement)

n parties with inputs x_1, \dots, x_n giving outputs y_1, \dots, y_n

Unknown $\leq t$ parties are *byzantine*
(other $\geq n - t$ are *honest*)

Agreement: *Honests* output same y



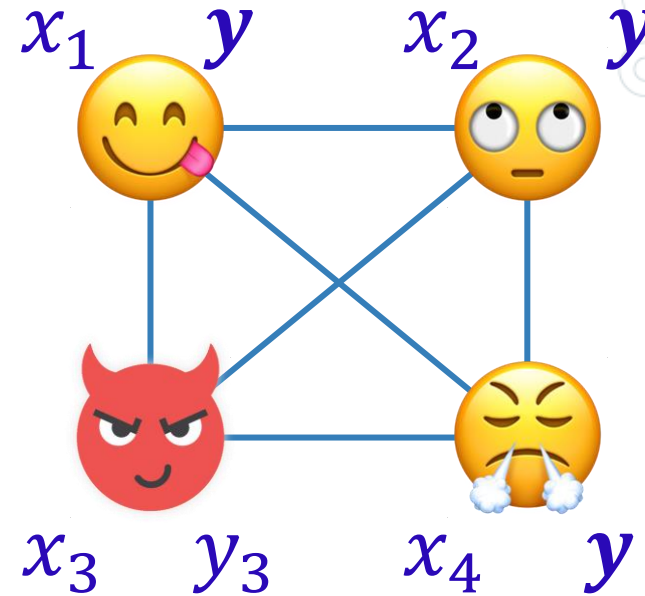
(authenticated bidirectional channels)

Consensus (Byzantine Agreement)

n parties with inputs x_1, \dots, x_n giving outputs y_1, \dots, y_n

Unknown $\leq t$ parties are *byzantine*
(other $\geq n - t$ are *honest*)

Agreement: *Honests* output same y
Validity: y is *meaningful*



(authenticated bidirectional channels)

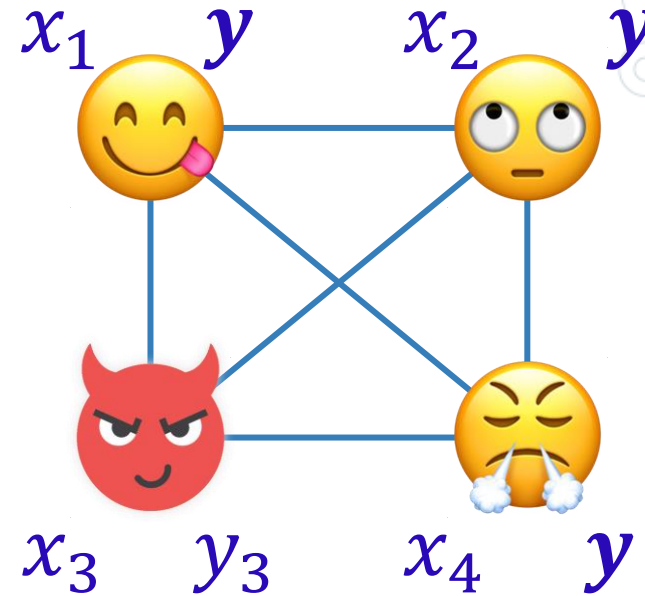
Consensus (Byzantine Agreement)

n parties with inputs x_1, \dots, x_n giving outputs y_1, \dots, y_n

Unknown $\leq t$ parties are *byzantine*
(other $\geq n - t$ are *honest*)

Agreement: *Honest* output same y
Validity: y is *meaningful*

Q: What does *meaningful* mean?



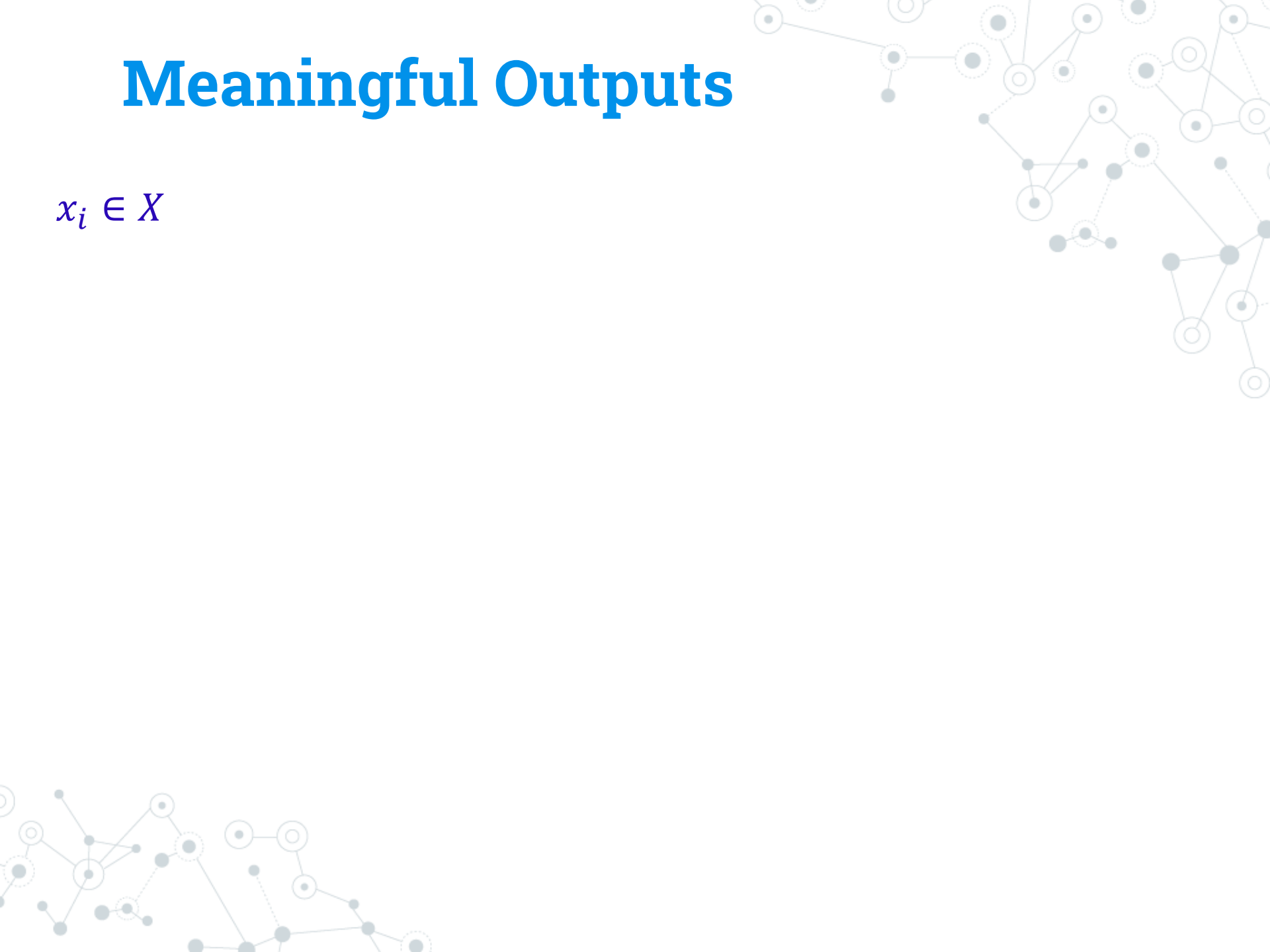
(authenticated bidirectional channels)

Meaningful Outputs



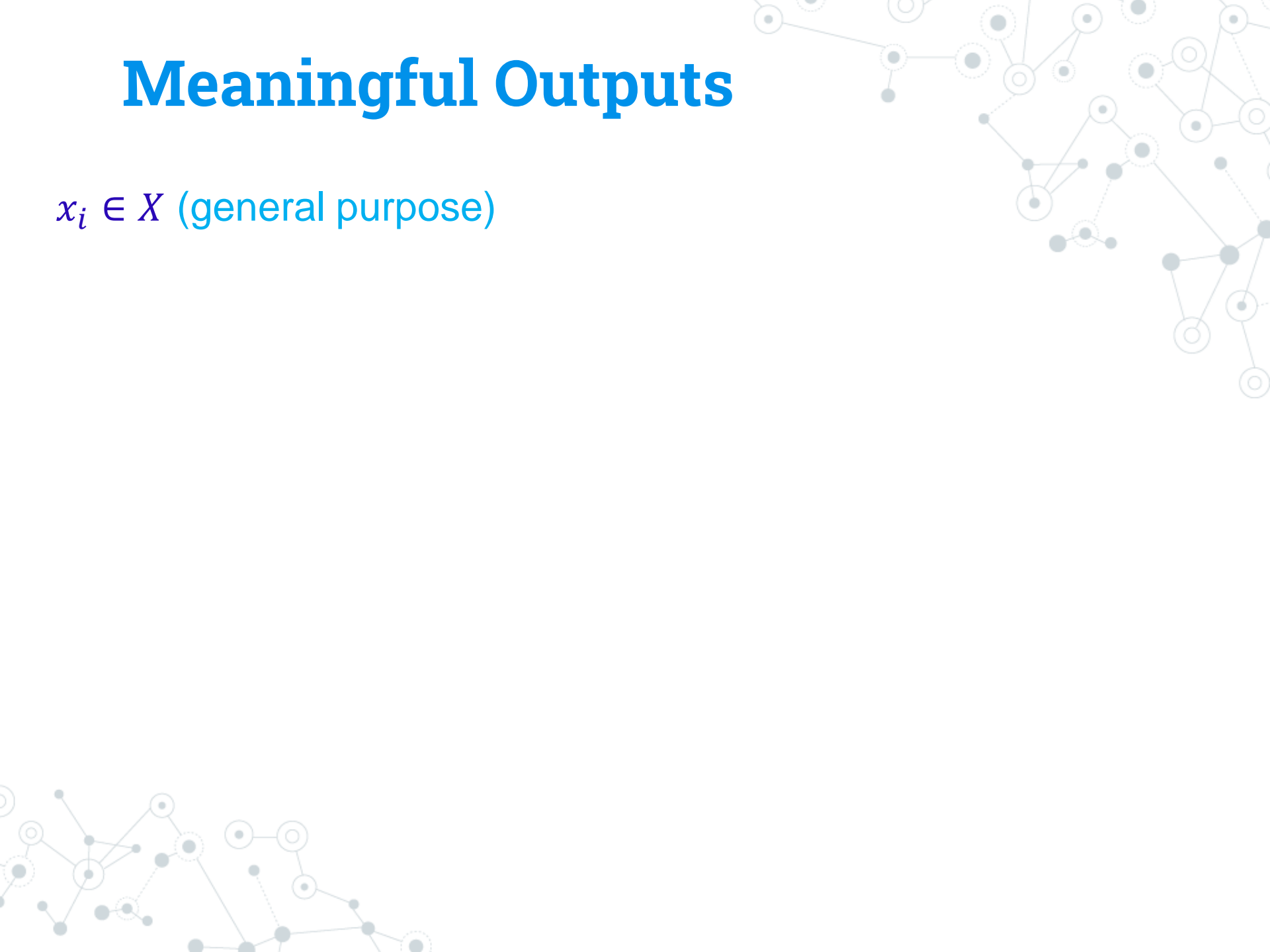
Meaningful Outputs

$x_i \in X$



Meaningful Outputs

$x_i \in X$ (general purpose)



Meaningful Outputs

A decorative network diagram in the top right corner, consisting of various nodes (some solid grey circles, some hollow white circles) connected by thin grey lines. The nodes are arranged in a complex, interconnected pattern.

$x_i \in X$ (general purpose)

- **Strong Validity:** If all **honest**s have the same input **x**, then the output is **y = x** (otherwise no constraints).



Meaningful Outputs

$x_i \in X$ (general purpose)

- **Strong Validity:** If all **honest** have the same input **x**, then the output is **y = x** (otherwise no constraints).
- **Honest-Input Validity:** **y** is the input of an **honest** party.

Meaningful Outputs

$x_i \in X$ (general purpose)

- **Strong Validity:** If all **honest** have the same input **x**, then the output is **y = x** (otherwise no constraints).
- **Honest-Input Validity:** **y** is the input of an **honest** party.

$x_i \in \mathbb{R}$

Meaningful Outputs



$x_i \in X$ (general purpose)

- **Strong Validity:** If all **honest** have the same input **x**, then the output is **y = x** (otherwise no constraints).
- **Honest-Input Validity:** **y** is the input of an **honest** party.

$x_i \in \mathbb{R}$ (temperature, altitude, price)



Meaningful Outputs

$x_i \in X$ (general purpose)

- **Strong Validity:** If all **honest** have the same input **x**, then the output is **y = x** (otherwise no constraints).
- **Honest-Input Validity:** **y** is the input of an **honest** party.

$x_i \in \mathbb{R}$ (temperature, altitude, price)

- **Honest-Range Validity:** **y** is between the smallest and largest **honest** inputs.

Meaningful Outputs

$x_i \in X$ (general purpose)

- **Strong Validity:** If all **honest** have the same input **x**, then the output is **y = x** (otherwise no constraints).
- **Honest-Input Validity:** **y** is the input of an **honest** party.

$x_i \in \mathbb{R}$ (temperature, altitude, price)

- **Honest-Range Validity:** **y** is between the smallest and largest **honest** inputs.

$x_i \in \mathbb{R}^D$

Meaningful Outputs

$x_i \in X$ (general purpose)

- **Strong Validity:** If all **honest** have the same input \mathbf{x} , then the output is $\mathbf{y} = \mathbf{x}$ (otherwise no constraints).
- **Honest-Input Validity:** \mathbf{y} is the input of an **honest** party.

$x_i \in \mathbb{R}$ (temperature, altitude, price)

- **Honest-Range Validity:** \mathbf{y} is between the smallest and largest **honest** inputs.

$x_i \in \mathbb{R}^D$ (meeting point on a map)

Meaningful Outputs

$x_i \in X$ (general purpose)

- **Strong Validity:** If all **honest** have the same input \mathbf{x} , then the output is $\mathbf{y} = \mathbf{x}$ (otherwise no constraints).
- **Honest-Input Validity:** \mathbf{y} is the input of an **honest** party.

$x_i \in \mathbb{R}$ (temperature, altitude, price)

- **Honest-Range Validity:** \mathbf{y} is between the smallest and largest **honest** inputs.

$x_i \in \mathbb{R}^D$ (meeting point on a map)

- **Convex Validity:** \mathbf{y} is in the convex hull of **honest** inputs.

Meaningful Outputs

$x_i \in X$ (general purpose)

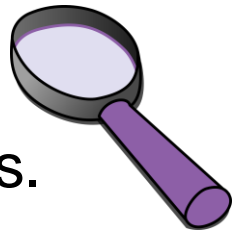
- **Strong Validity:** If all **honest** have the same input \mathbf{x} , then the output is $\mathbf{y} = \mathbf{x}$ (otherwise no constraints).
- **Honest-Input Validity:** \mathbf{y} is the input of an **honest** party.

$x_i \in \mathbb{R}$ (temperature, altitude, price)

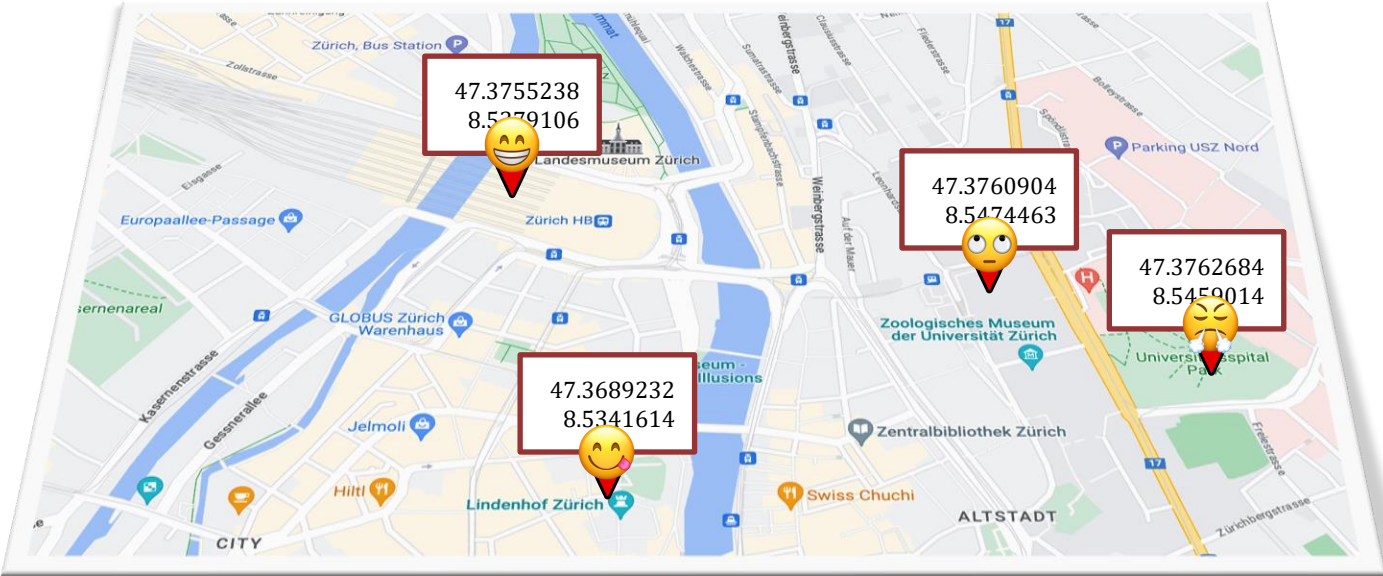
- **Honest-Range Validity:** \mathbf{y} is between the smallest and largest **honest** inputs.

$x_i \in \mathbb{R}^D$ (meeting point on a map)

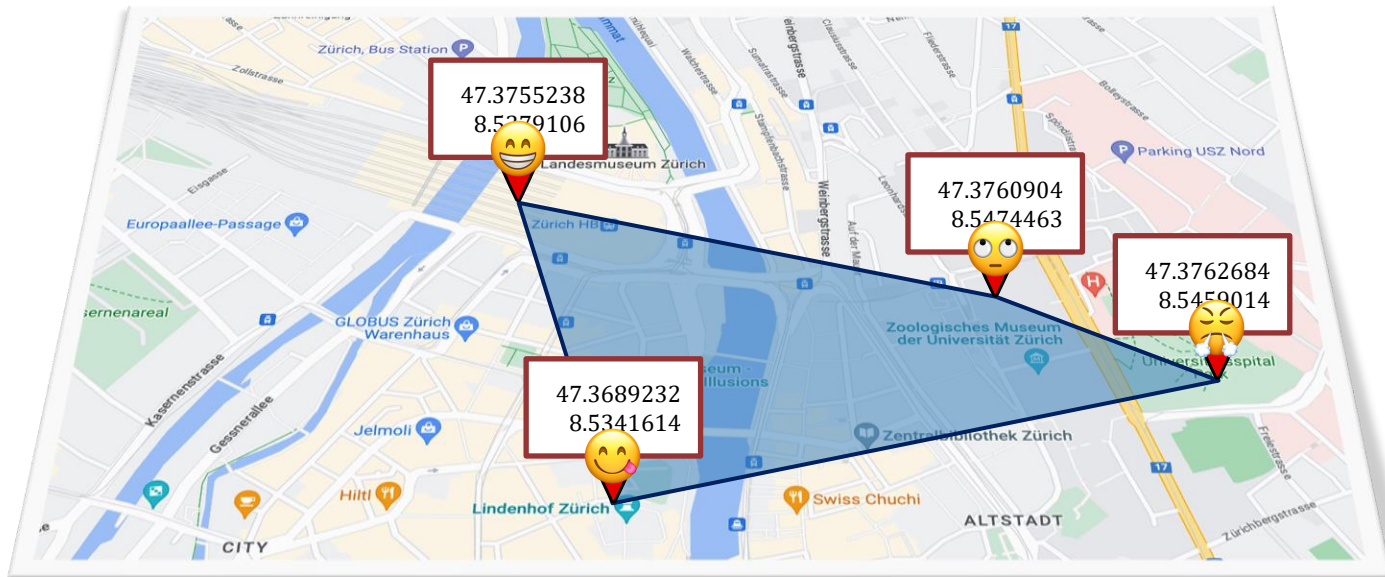
- **Convex Validity:** \mathbf{y} is in the convex hull of **honest** inputs.



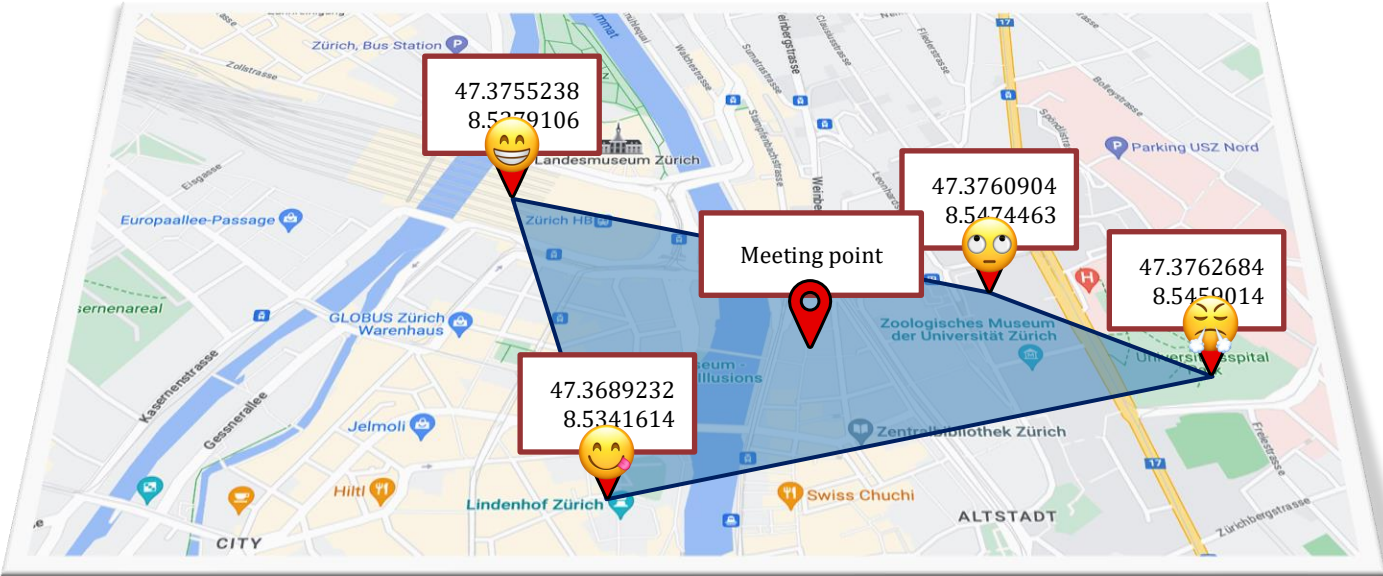
Convex Validity in \mathbb{R}^D



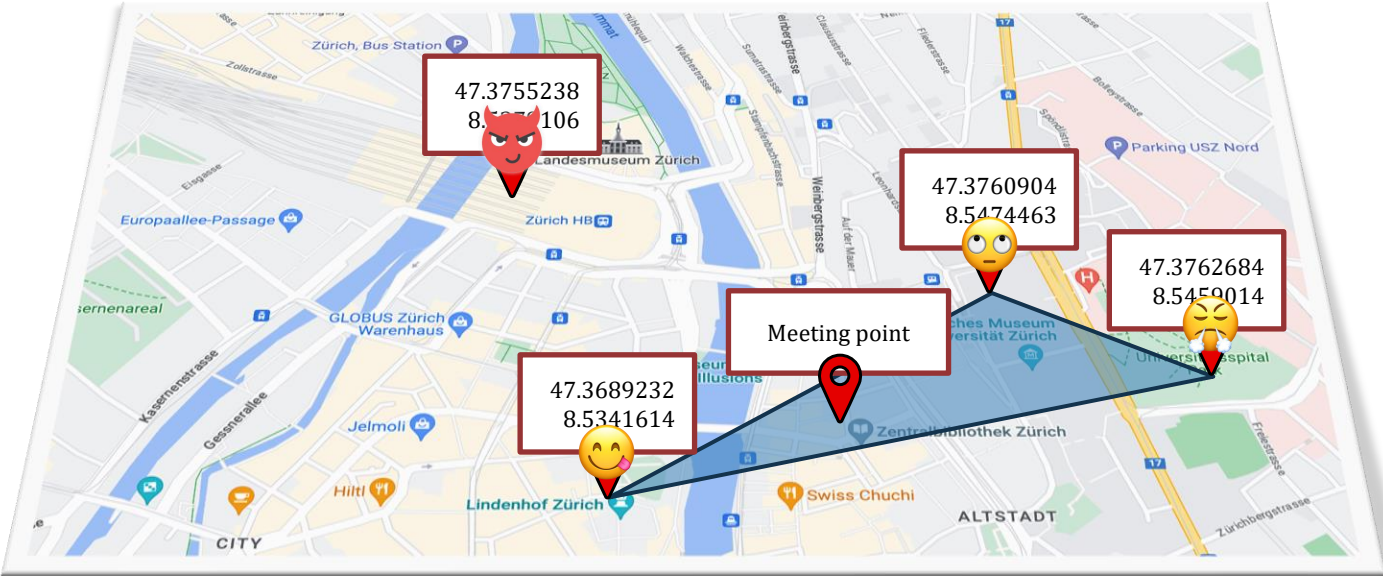
Convex Validity in \mathbb{R}^D



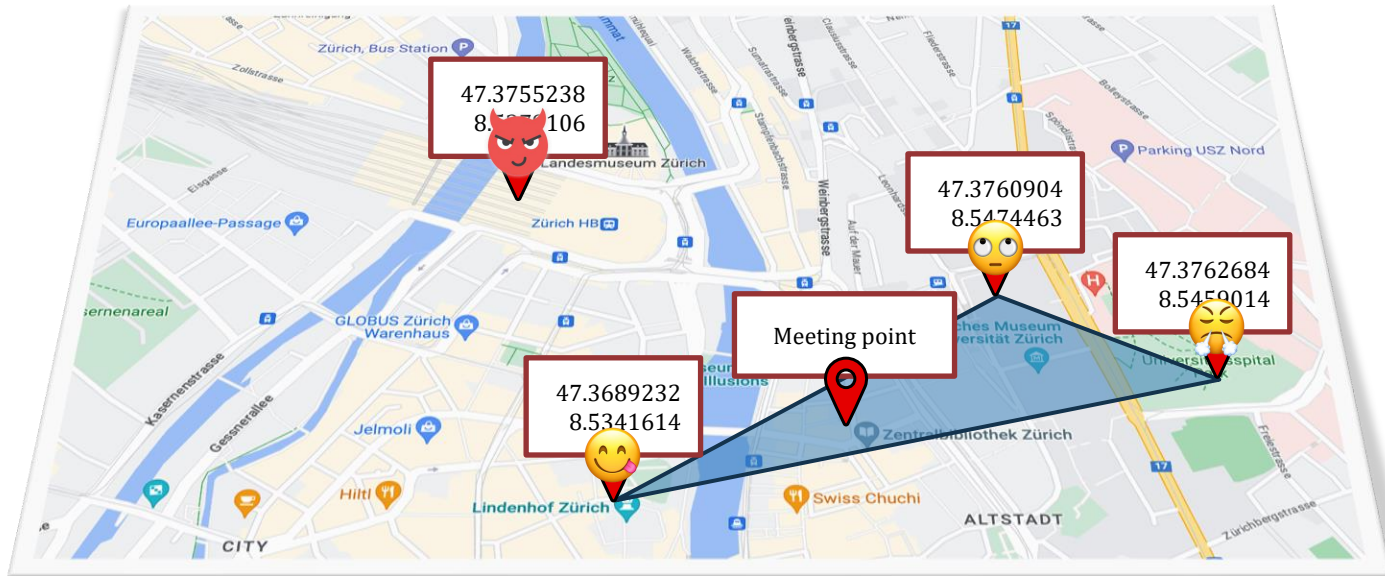
Convex Validity in \mathbb{R}^D



Convex Validity in \mathbb{R}^D

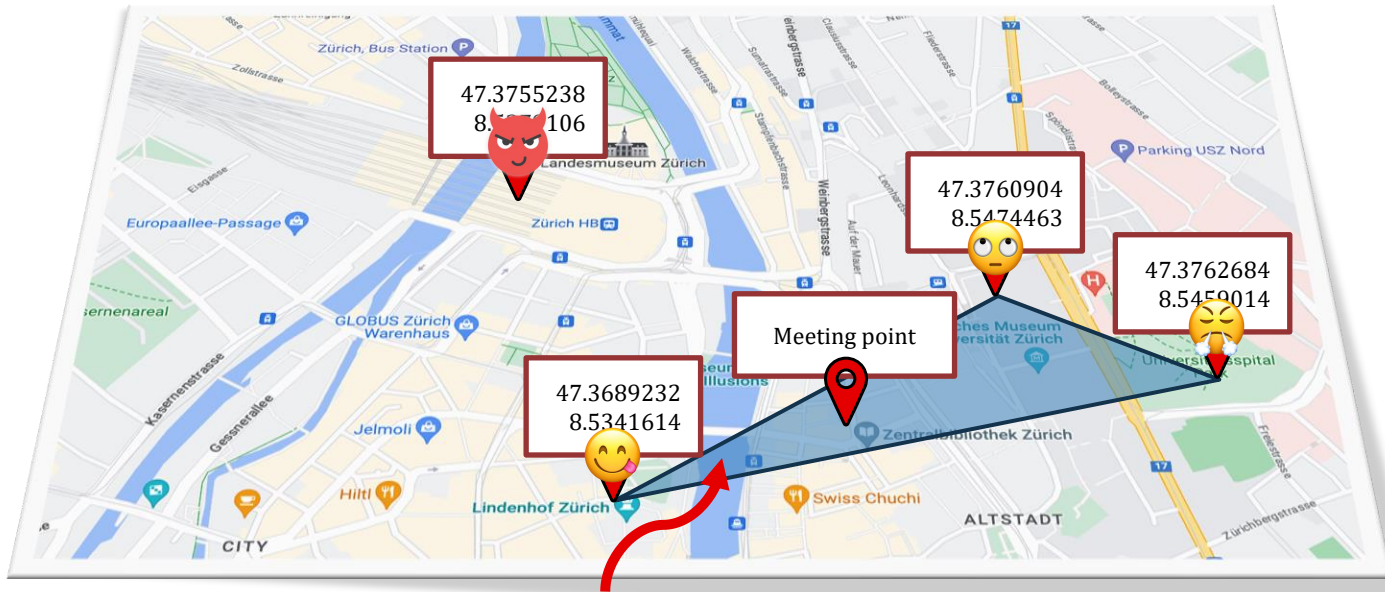


Convex Validity in \mathbb{R}^D



Q: Is there more to convexity?

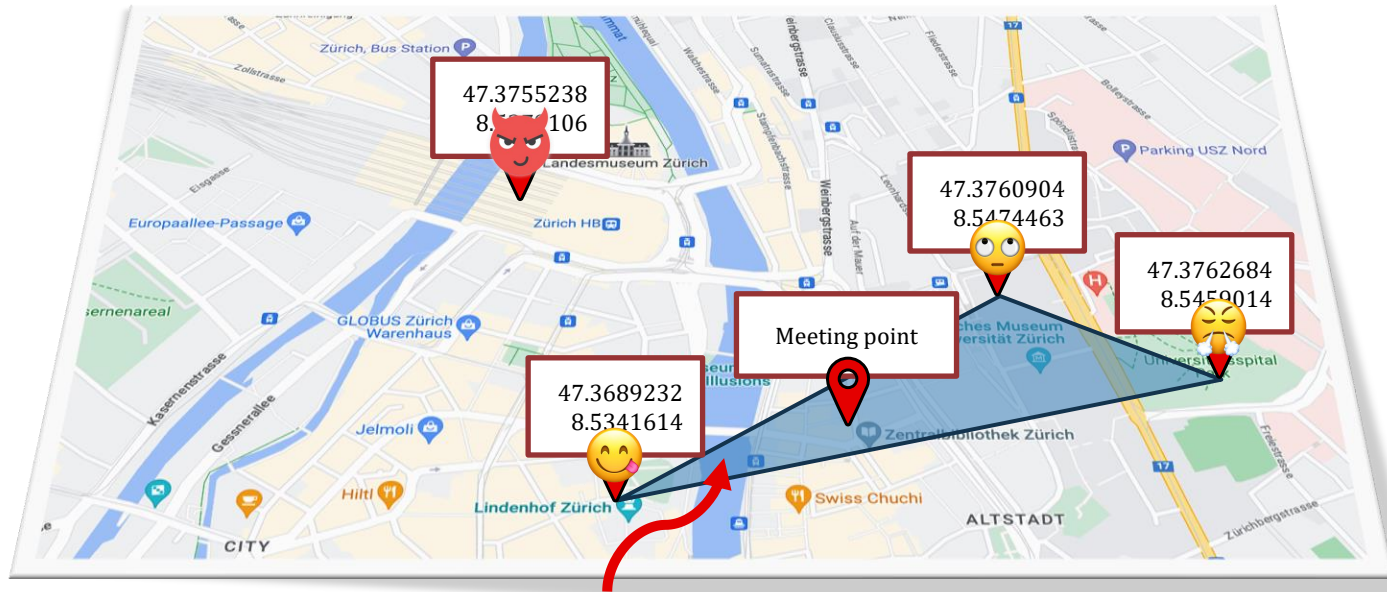
Convex Validity in \mathbb{R}^D



river

Q: Is there more to convexity? **YES!**

Convex Validity in \mathbb{R}^D

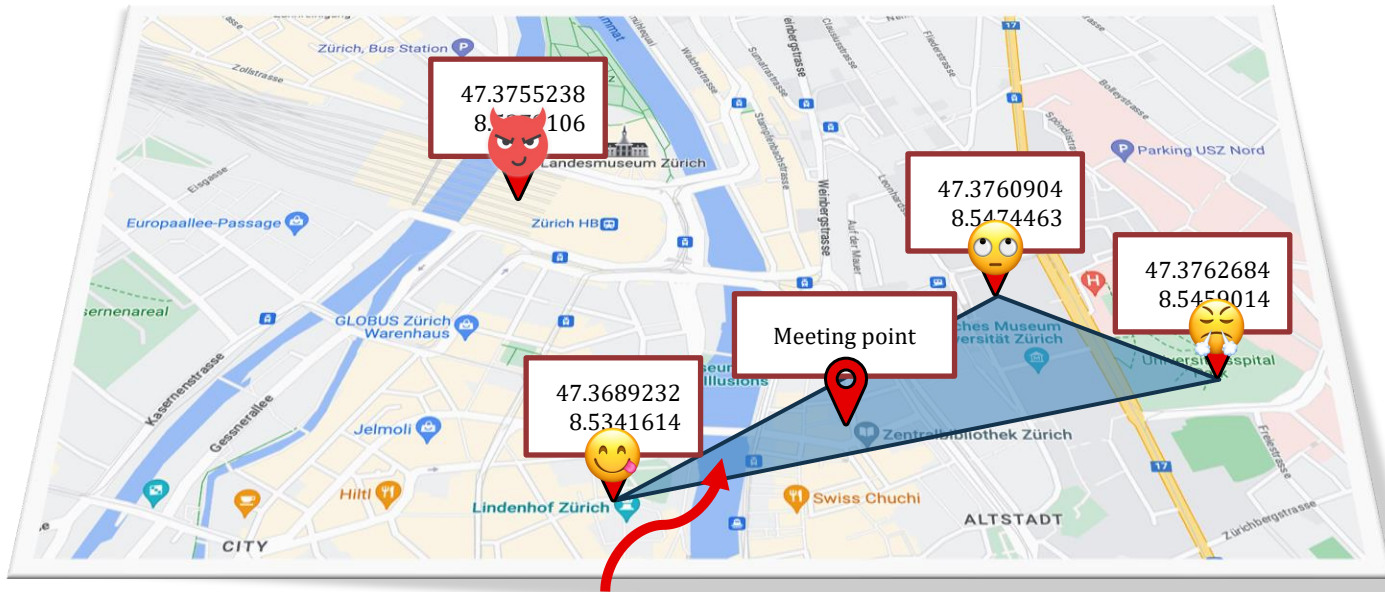


river

Q: Is there more to convexity? **YES!**

Graphs

Convex Validity in \mathbb{R}^D

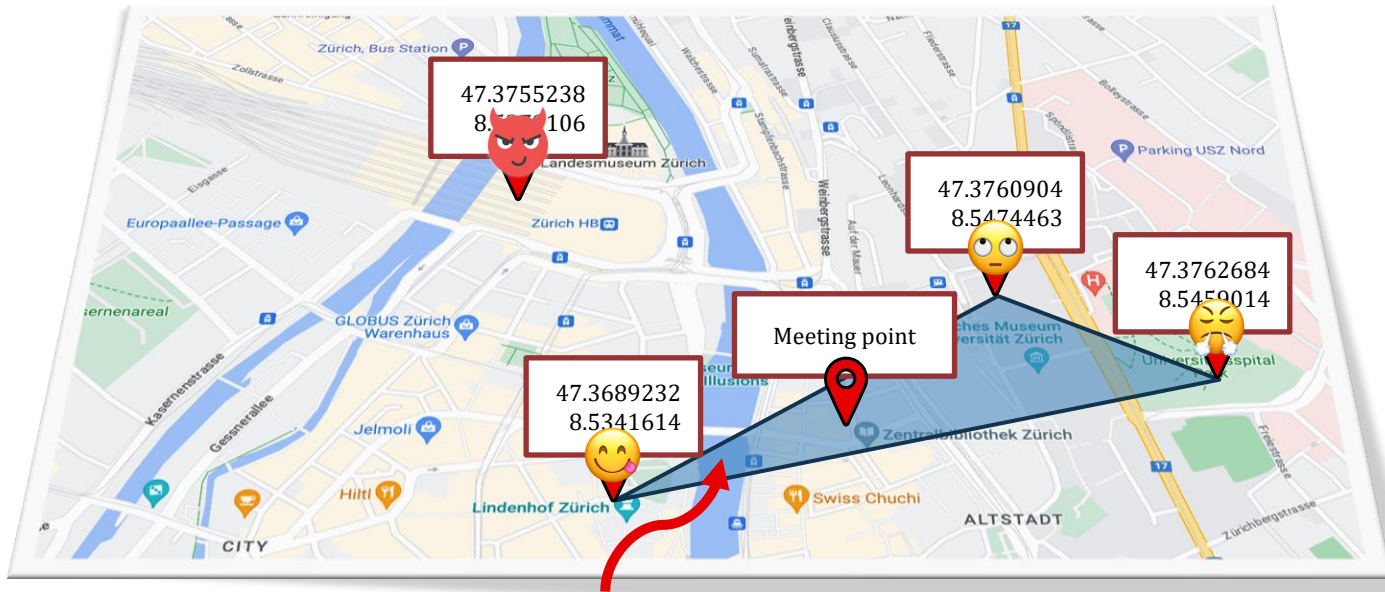


river

Q: Is there more to convexity? **YES!**

Graphs, lattices

Convex Validity in \mathbb{R}^D



river

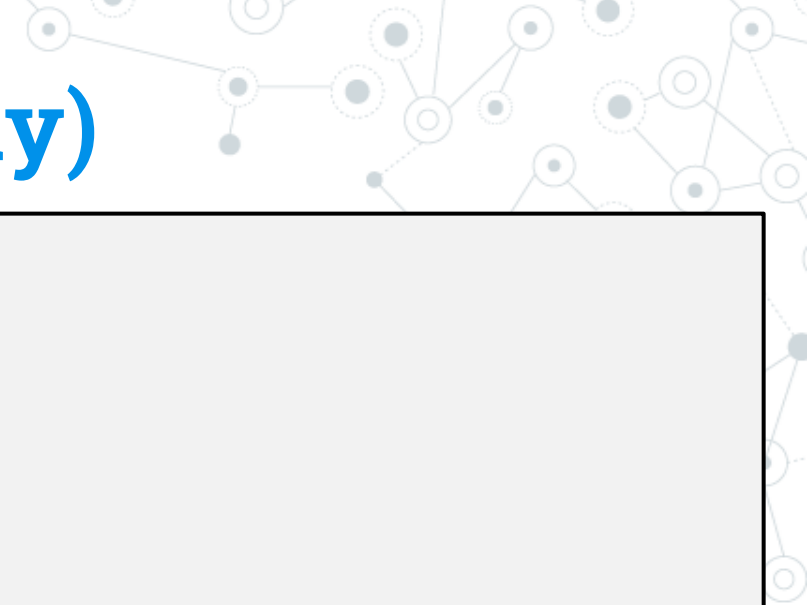
Q: Is there more to convexity? **YES!**

Graphs, lattices, etc.

Convexity (abstractly)



Convexity (abstractly)



Convexity (abstractly)

A *convexity space* C on a ground set X

Convexity (abstractly)

A *convexity space* C on a ground set X specifies which subsets of X are convex s.t.:

Convexity (abstractly)

A *convexity space* C on a ground set X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.

Convexity (abstractly)

A *convexity space* C on a ground set X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.
- Intersections of convex sets are convex.

Convexity (abstractly)

A *convexity space* C on a *ground set* X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.
- Intersections of convex sets are convex.

The *convex hull* of $S \subseteq X$ is the intersection of all convex sets containing S .

Convexity (abstractly)

A *convexity space* C on a *ground set* X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.
- Intersections of convex sets are convex.

The *convex hull* of $S \subseteq X$ is the intersection of all convex sets containing S .

Convex Validity: output y is in the convex hull of *honest* inputs.

Convexity (abstractly)

A *convexity space* C on a *ground set* X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.
- Intersections of convex sets are convex.

The *convex hull* of $S \subseteq X$ is the intersection of all convex sets containing S .

Convex Validity: output y is in the convex hull of *honest* inputs.

Examples:

Convexity (abstractly)

A *convexity space* C on a ground set X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.
- Intersections of convex sets are convex.

The *convex hull* of $S \subseteq X$ is the intersection of all convex sets containing S .

Convex Validity: output y is in the convex hull of **honest** inputs.

Examples:

- \mathbb{R}^D with **straight-line convexity**

Convexity (abstractly)

A *convexity space* C on a ground set X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.
- Intersections of convex sets are convex.

The *convex hull* of $S \subseteq X$ is the intersection of all convex sets containing S .

Convex Validity: output y is in the convex hull of **honest** inputs.

Examples:

- \mathbb{R}^D with **straight-line convexity**
- Vertices of a graph G with **monophonic/geodesic convexity**

Convexity (abstractly)

A *convexity space* C on a ground set X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.
- Intersections of convex sets are convex.

The *convex hull* of $S \subseteq X$ is the intersection of all convex sets containing S .

Convex Validity: output y is in the convex hull of **honest** inputs.

Examples:

- \mathbb{R}^D with **straight-line convexity**
- Vertices of a graph G with **monophonic/geodesic convexity**
- Elements of a lattice L with **algebraic convexity**

Convexity (abstractly)

A *convexity space* C on a ground set X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.
- Intersections of convex sets are convex.

The *convex hull* of $S \subseteq X$ is the intersection of all convex sets containing S .

Convex Validity: output y is in the convex hull of **honest** inputs.

Examples:

- \mathbb{R}^D with **straight-line convexity**
- Vertices of a graph G with **monophonic/geodesic convexity**
- Elements of a lattice L with **algebraic convexity**
- Any set X and **all sets are convex**

Convexity (abstractly)

A *convexity space* C on a ground set X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.
- Intersections of convex sets are convex.

The *convex hull* of $S \subseteq X$ is the intersection of all convex sets containing S .

Convex Validity: output y is in the convex hull of **honest** inputs.

Examples:

- \mathbb{R}^D with **straight-line convexity**
- Vertices of a graph G with **monophonic/geodesic convexity**
- Elements of a lattice L with **algebraic convexity**
- Any set X and **all sets are convex** \rightarrow **Honest-Input Validity**

Convexity (abstractly)

A *convexity space* C on a ground set X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.
- Intersections of convex sets are convex.

The *convex hull* of $S \subseteq X$ is the intersection of all convex sets containing S .

Convex Validity: output y is in the convex hull of **honest** inputs.

Examples:

- \mathbb{R}^D with **straight-line convexity**
- Vertices of a graph G with **monophonic/geodesic convexity**
- Elements of a lattice L with **algebraic convexity**
- Any set X and **all sets are convex** \rightarrow **Honest-Input Validity**
- Any set X and \emptyset , X and singletons are convex

Convexity (abstractly)

A *convexity space* C on a ground set X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.
- Intersections of convex sets are convex.

The *convex hull* of $S \subseteq X$ is the intersection of all convex sets containing S .

Convex Validity: output y is in the convex hull of **honest** inputs.

Examples:

- \mathbb{R}^D with **straight-line convexity**
- Vertices of a graph G with **monophonic/geodesic convexity**
- Elements of a lattice L with **algebraic convexity**
- Any set X and **all sets are convex** \rightarrow **Honest-Input Validity**
- Any set X and \emptyset , X and singletons are convex \rightarrow **Strong Validity**

Convexity (abstractly)

A *convexity space* C on a ground set X specifies which subsets of X are convex s.t.:

- \emptyset and X are convex.
- Intersections of convex sets are convex.

The *convex hull* of $S \subseteq X$ is the intersection of all convex sets containing S .

Convex Validity: output y is in the convex hull of **honest** inputs.

Examples:

- \mathbb{R}^D with **straight-line convexity** \rightarrow **Honest-Range Validity** ($D = 1$)
- **Vertices of a graph G** with **monophonic/geodesic convexity**
- **Elements of a lattice L** with **algebraic convexity**
- Any set X and **all sets are convex** \rightarrow **Honest-Input Validity**
- Any set X and \emptyset , X and singletons are convex \rightarrow **Strong Validity**



Q: Given C ,





Q: Given C , find largest t s.t. we can achieve Agreement and Convex Validity for $\leq t$ corruptions?



A faint background network graph with nodes and edges is visible in the top right and bottom left corners of the slide.

Q: Given C , find largest t s.t. we can achieve Agreement and Convex Validity for $\leq t$ corruptions?

Convex Consensus (CC)

A background network graph with nodes and edges, some nodes highlighted in grey and others in white.

Q: Given C , find largest t s.t. we can achieve Agreement and Convex Validity for $\leq t$ corruptions?

Convex Consensus (CC)

Pb.: How can we even specify t ?



2.

A geometric invariant

The Helly number



Helly's Theorem



Helly's Theorem

Helly's Theorem:



Helly's Theorem

Helly's Theorem:

Given family of convex sets in \mathbb{R}^D (with **straight-line convexity**):



Helly's Theorem

Helly's Theorem:

Given family of convex sets in \mathbb{R}^D (with **straight-line convexity**):

Every **$D + 1$** intersect \Rightarrow **all** of them intersect.

Helly's Theorem



Helly's Theorem:

Given family of convex sets in \mathbb{R}^D (with **straight-line convexity**):
Every $D + 1$ intersect \Rightarrow **all** of them intersect.

Example:



Helly's Theorem

Helly's Theorem:

Given family of convex sets in \mathbb{R}^D (with **straight-line convexity**):
Every $D + 1$ intersect \Rightarrow **all** of them intersect.

Example:

100 disks in \mathbb{R}^2

Helly's Theorem




Helly's Theorem:

Given family of convex sets in \mathbb{R}^D (with **straight-line convexity**):

Every $D + 1$ intersect \Rightarrow **all** of them intersect.

Example:

100 disks in \mathbb{R}^2 : every **3** intersect



Helly's Theorem

Helly's Theorem:

Given family of convex sets in \mathbb{R}^D (with **straight-line convexity**):

Every **$D + 1$** intersect \Rightarrow **all** of them intersect.

Example:

100 disks in \mathbb{R}^2 : every **3** intersect \Rightarrow **all** 100 intersect.

Helly's Theorem

Helly's Theorem:

Given family of convex sets in \mathbb{R}^D (with **straight-line convexity**):

Every **$D + 1$** intersect \Rightarrow **all** of them intersect.

Example:

100 disks in \mathbb{R}^2 : every **3** intersect \Rightarrow **all** 100 intersect.

~~(every **2** intersect \Rightarrow **all** 100 intersect)~~

Helly's Theorem

Helly's Theorem:

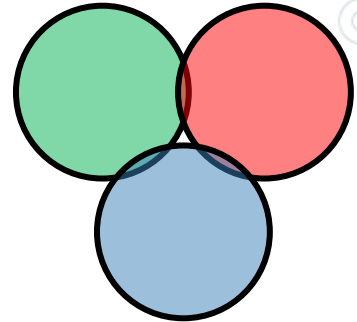
Given family of convex sets in \mathbb{R}^D (with **straight-line convexity**):

Every **$D + 1$** intersect \Rightarrow **all** of them intersect.

Example:

100 disks in \mathbb{R}^2 : every **3** intersect \Rightarrow **all** 100 intersect.

~~(every **2** intersect \Rightarrow **all** 100 intersect)~~



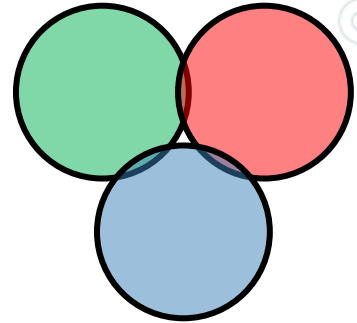
Helly's Theorem

Helly's Theorem:

Given family of convex sets in \mathbb{R}^D (with **straight-line convexity**):
Every $D + 1$ intersect \Rightarrow **all** of them intersect.

Example:

100 disks in \mathbb{R}^2 : every **3** intersect \Rightarrow **all** 100 intersect.
(every ~~**2**~~ intersect \Rightarrow ~~**all**~~ 100 intersect)



Helly's Theorem (abstract):

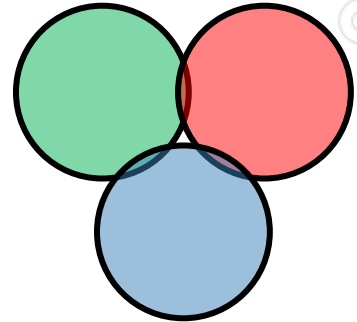
Helly's Theorem

Helly's Theorem:

Given family of convex sets in \mathbb{R}^D (with **straight-line convexity**):
Every $D + 1$ intersect \Rightarrow **all** of them intersect.

Example:

100 disks in \mathbb{R}^2 : every **3** intersect \Rightarrow **all** 100 intersect.
(every ~~**2**~~ intersect \Rightarrow ~~**all**~~ 100 intersect)



Helly's Theorem (abstract):

Given family of convex sets in a **convexity space** C :
Every ω intersect \Rightarrow **all** of them intersect.

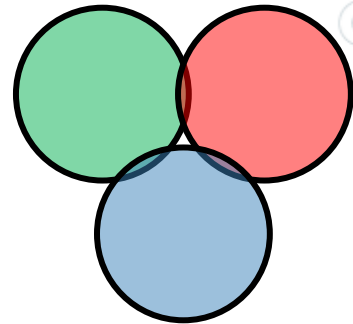
Helly's Theorem

Helly's Theorem:

Given family of convex sets in \mathbb{R}^D (with **straight-line convexity**):
Every $D + 1$ intersect \Rightarrow **all** of them intersect.

Example:

100 disks in \mathbb{R}^2 : every **3** intersect \Rightarrow **all** 100 intersect.
(every ~~**2**~~ intersect \Rightarrow ~~**all**~~ 100 intersect)



Helly's Theorem (abstract):

Given family of convex sets in a **convexity space C** :
Every ω intersect \Rightarrow **all** of them intersect.

Helly number of C

A background network graph with nodes and edges, some nodes highlighted in grey and others in white.

Q: Given C , find largest t s.t. we can achieve **Agreement and Convex Validity** for $\leq t$ corruptions?

Convex Consensus (CC)

Pb.: How can we even specify t ?

A network graph background consisting of various nodes (circles) and edges (lines) in shades of gray and blue, scattered across the slide.

Q: Given C , find largest t s.t. we can achieve Agreement and Convex Validity for $\leq t$ corruptions?

Convex Consensus (CC)

Pb.: How can we even specify t ? ω

Q: Given C , find largest t s.t. we can achieve Agreement and Convex Validity for $\leq t$ corruptions?

Convex Consensus (CC)

A: depends on *network model*.

Pb.: How can we even specify t ? ω

A decorative network diagram in the top-left corner, consisting of various sized nodes (some solid grey, some hollow white) connected by thin grey lines, forming a complex web structure.

3.

(A)synchrony

And the best-of-both-worlds

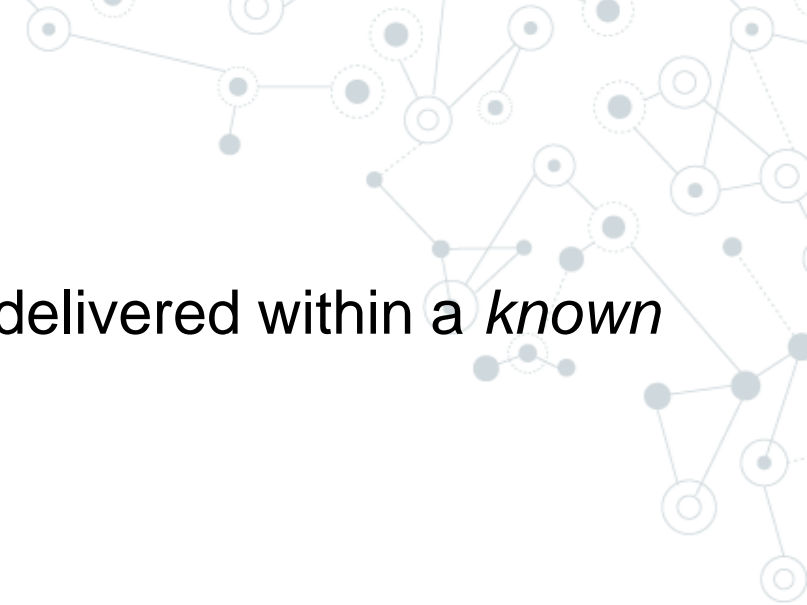
A decorative network diagram in the bottom-right corner, similar to the one in the top-left, with nodes and connecting lines forming a network pattern.

Network Models



Network Models

Synchronous model: messages get delivered within a *known* amount of time Δ .



Network Models

A decorative network diagram in the top right corner of the slide. It consists of several interconnected nodes and edges. The nodes are represented by circles of varying sizes and colors, including light blue, dark blue, and grey. Some nodes are solid, while others are hollow. The edges are thin lines connecting the nodes, some solid and some dashed. The overall structure is a complex, interconnected graph.

Synchronous model: messages get delivered within a *known* amount of time Δ .

Problem: small deviations from Δ break the protocol.

A decorative network diagram in the bottom left corner of the slide. It consists of several interconnected nodes and edges. The nodes are represented by circles of varying sizes and colors, including light blue, dark blue, and grey. Some nodes are solid, while others are hollow. The edges are thin lines connecting the nodes, some solid and some dashed. The overall structure is a complex, interconnected graph.

Network Models

A decorative network diagram in the top right corner of the slide. It consists of several nodes, represented by circles of varying sizes and colors (some solid grey, some hollow white), connected by thin grey lines. The nodes are arranged in a somewhat circular pattern, with some lines extending outwards.

Synchronous model: messages get delivered within a *known* amount of time Δ .

Problem: small deviations from Δ break the protocol.

Asynchronous model: messages get delivered *eventually*.

A decorative network diagram in the bottom left corner of the slide. It consists of several nodes, represented by circles of varying sizes and colors (some solid grey, some hollow white), connected by thin grey lines. The nodes are arranged in a somewhat circular pattern, with some lines extending outwards.

Network Models

A decorative network diagram in the top right corner of the slide. It consists of several nodes, represented by circles of varying sizes and colors (some solid grey, some hollow white), connected by thin grey lines. The nodes are arranged in a somewhat circular pattern, with some lines extending outwards.

Synchronous model: messages get delivered within a *known* amount of time Δ .

Problem: small deviations from Δ break the protocol.

Asynchronous model: messages get delivered *eventually*.

Problem: requires lower t .

A decorative network diagram in the bottom left corner of the slide. It consists of several nodes, represented by circles of varying sizes and colors (some solid grey, some hollow white), connected by thin grey lines. The nodes are arranged in a somewhat circular pattern, with some lines extending outwards.

Network Models

A decorative network diagram in the top right corner, featuring a complex web of interconnected nodes and edges. The nodes are represented by circles of varying sizes and colors (grey, white, blue), and the edges are thin lines connecting them.

Synchronous model: messages get delivered within a *known* amount of time Δ .

Problem: small deviations from Δ break the protocol.

Asynchronous model: messages get delivered *eventually*.

Problem: requires lower t .

A tradeoff? Can we somehow get the best of both worlds?



Network Models

A decorative network diagram in the top right corner of the slide. It consists of several nodes, represented by circles of varying sizes and colors (some solid grey, some hollow white), connected by thin grey lines. The nodes are arranged in a somewhat circular pattern, with some lines extending outwards.

Synchronous model: messages get delivered within a *known* amount of time Δ .

Problem: small deviations from Δ break the protocol.

Asynchronous model: messages get delivered *eventually*.

Problem: requires lower t .

A tradeoff? Can we somehow get the best of both worlds?

Network-agnostic model:

A decorative network diagram in the bottom left corner of the slide. It consists of several nodes, represented by circles of varying sizes and colors (some solid grey, some hollow white), connected by thin grey lines. The nodes are arranged in a somewhat circular pattern, with some lines extending outwards.

Network Models



Synchronous model: messages get delivered within a *known* amount of time Δ .

Problem: small deviations from Δ break the protocol.

Asynchronous model: messages get delivered *eventually*.

Problem: requires lower t .

A tradeoff? Can we somehow get the best of both worlds?

Network-agnostic model: Given $t_a \leq t_s$:



Network Models



Synchronous model: messages get delivered within a *known* amount of time Δ .

Problem: small deviations from Δ break the protocol.

Asynchronous model: messages get delivered *eventually*.

Problem: requires lower t .

A tradeoff? Can we somehow get the best of both worlds?

Network-agnostic model: Given $t_a \leq t_s$:

- Tolerate t_s corruptions if the network is synchronous.
- 

Network Models

Synchronous model: messages get delivered within a *known* amount of time Δ .

Problem: small deviations from Δ break the protocol.

Asynchronous model: messages get delivered *eventually*.

Problem: requires lower t .

A tradeoff? Can we somehow get the best of both worlds?

Network-agnostic model: Given $t_a \leq t_s$:

- Tolerate t_s corruptions if the network is synchronous.
- Tolerate t_a corruptions if the network is asynchronous.

Network Models

Synchronous model: messages get delivered within a *known* amount of time Δ .

Problem: small deviations from Δ break the protocol.

Asynchronous model: messages get delivered *eventually*.

Problem: requires lower t .

A tradeoff? Can we somehow get the best of both worlds?

Network-agnostic model: Given $t_a \leq t_s$:

- Tolerate t_s corruptions if the network is synchronous.
- Tolerate t_a corruptions if the network is asynchronous.
- **The protocol does not know which one is the case!**

Network Models

Synchronous model: messages get delivered within a *known* amount of time Δ .

Problem: small deviations from Δ break the protocol.

Asynchronous model: messages get delivered *eventually*.

Problem: requires lower t .

A tradeoff? Can we somehow get the best of both worlds?

Network-agnostic model: Given $t_a \leq t_s$:

- Tolerate t_s corruptions if the network is synchronous.
- Tolerate t_a corruptions if the network is asynchronous.
- **The protocol does not know which one is the case!**
- **(Question here:** what pairs (t_s, t_a) are possible?)



4.

Results

Tight results for all 3 models!



CC Solvable Iff



CC Solvable Iff

Synchronous model:



CC Solvable Iff

Synchronous model: $t < \frac{n}{\omega}$



CC Solvable Iff

Synchronous model: $t < \frac{n}{\omega}$

Asynchronous model:



CC Solvable Iff

Synchronous model: $t < \frac{n}{\omega}$

Asynchronous model: $t < \frac{n}{\omega+1}$



CC Solvable Iff

Synchronous model: $t < \frac{n}{\omega}$

Asynchronous model: $t < \frac{n}{\omega+1}$

Network-agnostic model:



CC Solvable Iff

Synchronous model: $t < \frac{n}{\omega}$

Asynchronous model: $t < \frac{n}{\omega+1}$

Network-agnostic model: $\omega t_s < n$

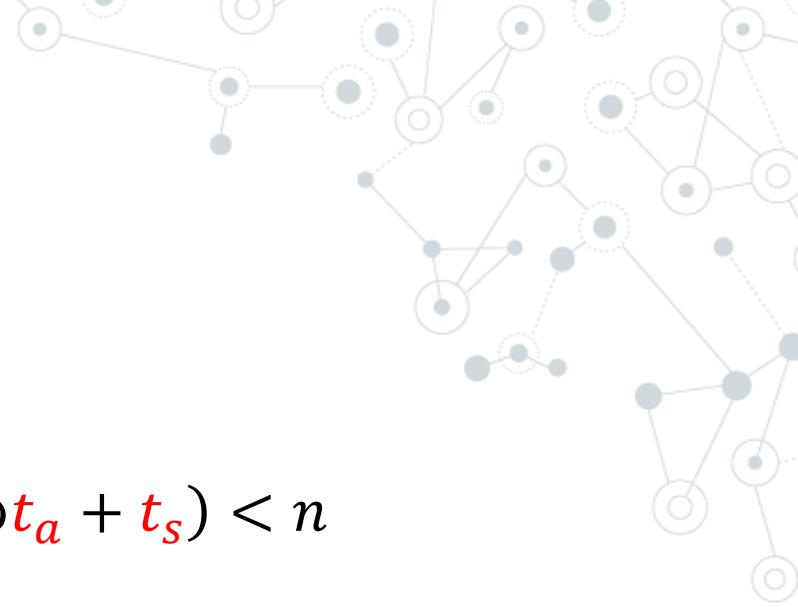


CC Solvable Iff

Synchronous model: $t < \frac{n}{\omega}$

Asynchronous model: $t < \frac{n}{\omega+1}$

Network-agnostic model: $\max(\omega t_s, \omega t_a + t_s) < n$



CC Solvable Iff

Synchronous model: $t < \frac{n}{\omega}$

Asynchronous model: $t < \frac{n}{\omega+1}$

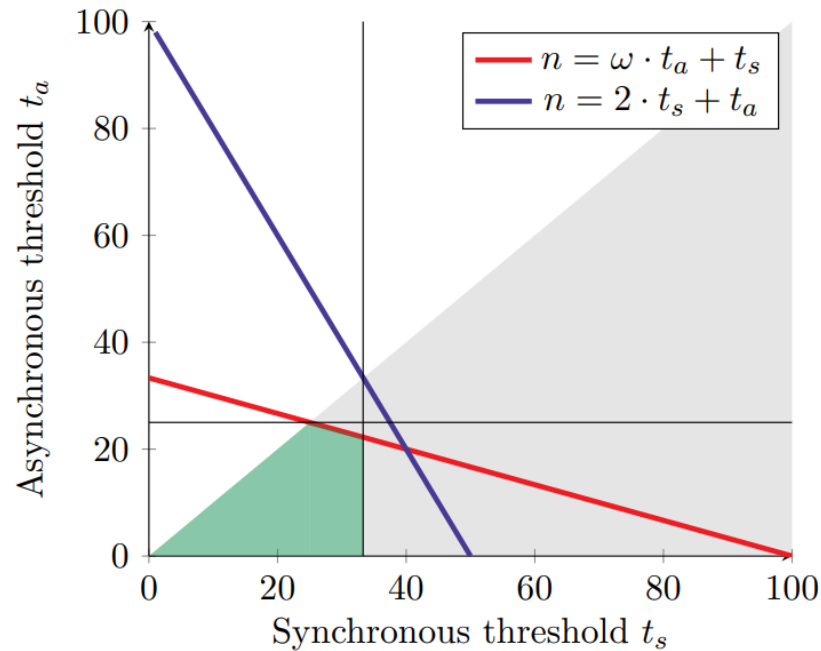
Network-agnostic model: $\max(\omega t_s, \omega t_a + t_s, 2t_s + t_a) < n$

CC Solvable Iff

Synchronous model: $t < \frac{n}{\omega}$

Asynchronous model: $t < \frac{n}{\omega+1}$

Network-agnostic model: $\max(\omega t_s, \omega t_a + t_s, 2t_s + t_a) < n$



(b) $\omega = 3$

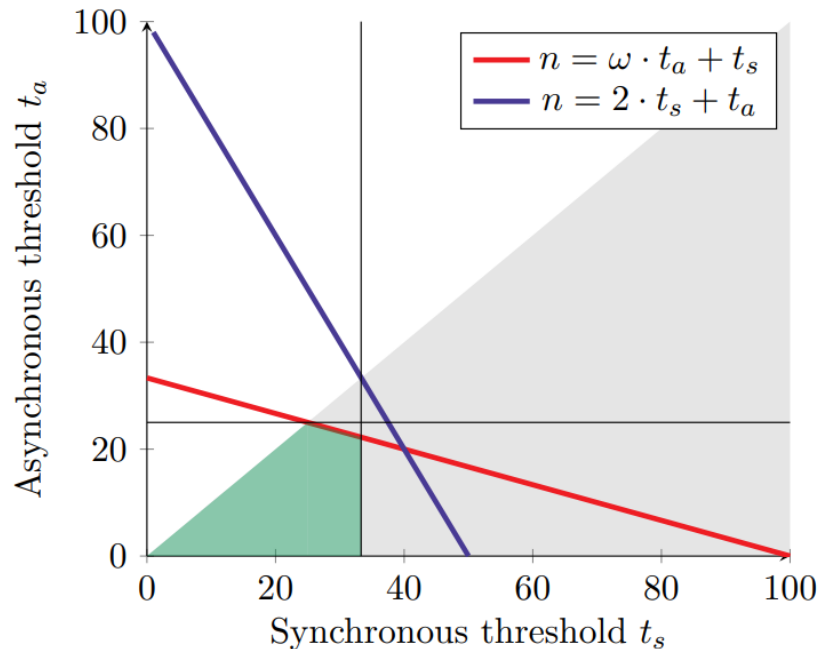
CC Solvable Iff

Synchronous model: $t < \frac{n}{\omega}$ Imp Pos

Asynchronous model: $t < \frac{n}{\omega+1}$ Imp Pos

Network-agnostic model: $\max(\omega t_s, \omega t_a + t_s, 2t_s + t_a) < n$ Pos

Imp1 Imp2 Imp3



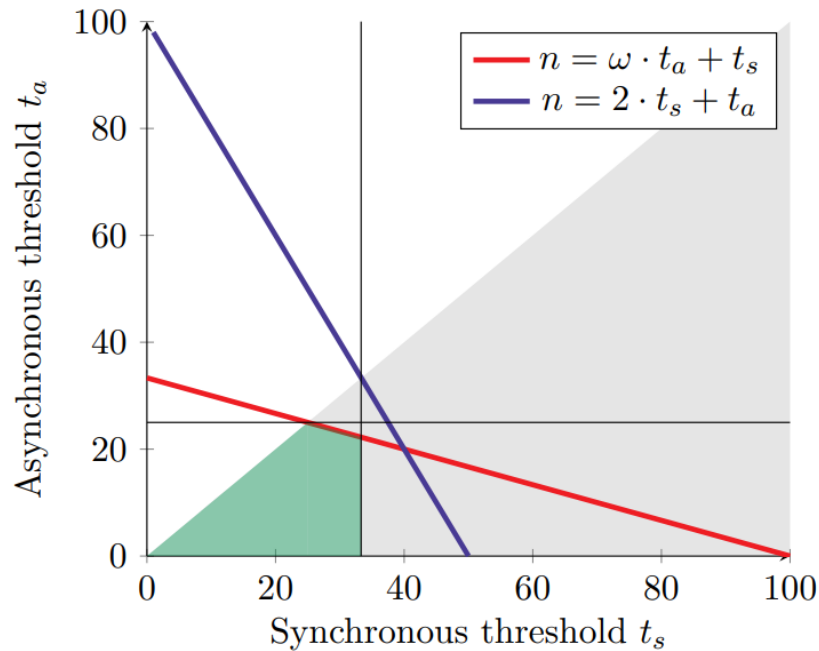
CC Solvable Iff

Synchronous model: $t < \frac{n}{\omega}$ Imp Pos

Asynchronous model: $t < \frac{n}{\omega+1}$ Imp Pos

Network-agnostic model: $\max(\omega t_s, \omega t_a + t_s, 2t_s + t_a) < n$ Pos

Imp1
Imp2
Imp3



(b) $\omega = 3$

$t = \frac{n}{\omega}$ is impossible



$t = \frac{n}{\omega}$ is impossible

Example for \mathbb{R}^2 :



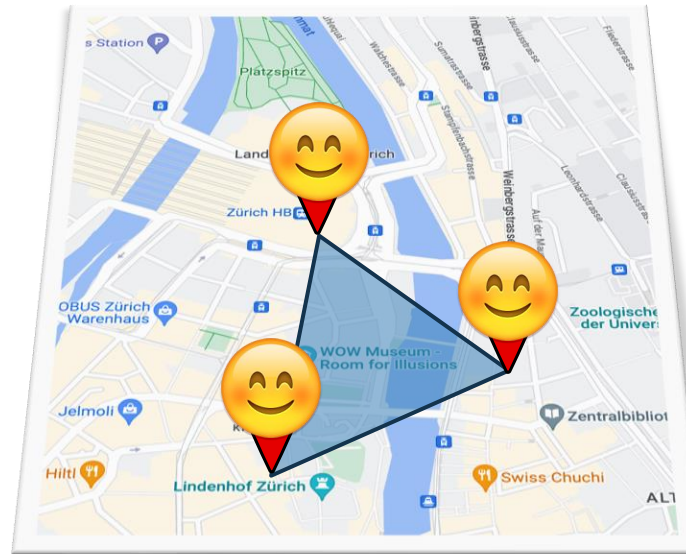
$t = \frac{n}{\omega}$ is impossible

Example for \mathbb{R}^2 : $t = \frac{n}{3}$ is impossible.



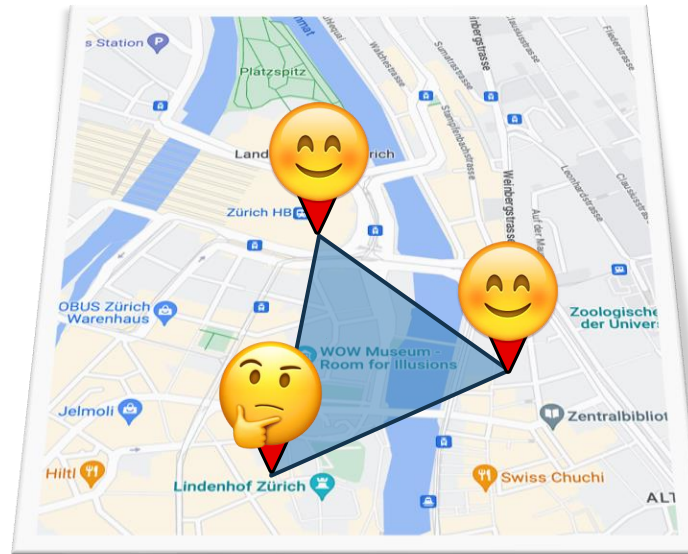
$t = \frac{n}{\omega}$ is impossible

Example for \mathbb{R}^2 : $t = \frac{n}{3}$ is impossible.



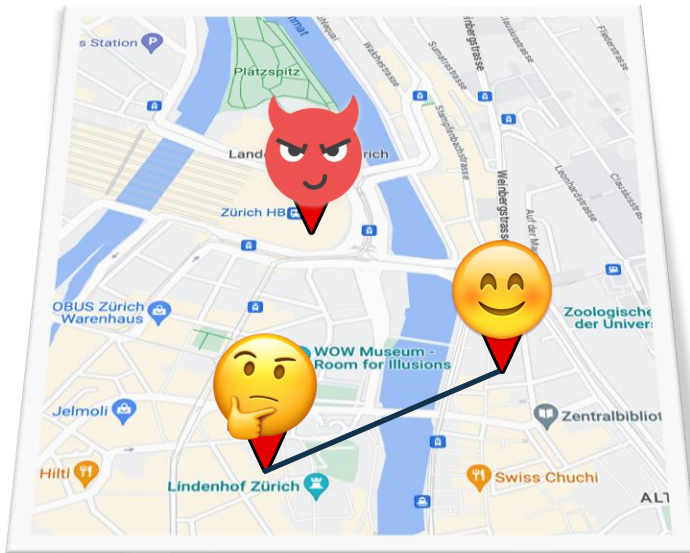
$t = \frac{n}{\omega}$ is impossible

Example for \mathbb{R}^2 : $t = \frac{n}{3}$ is impossible.



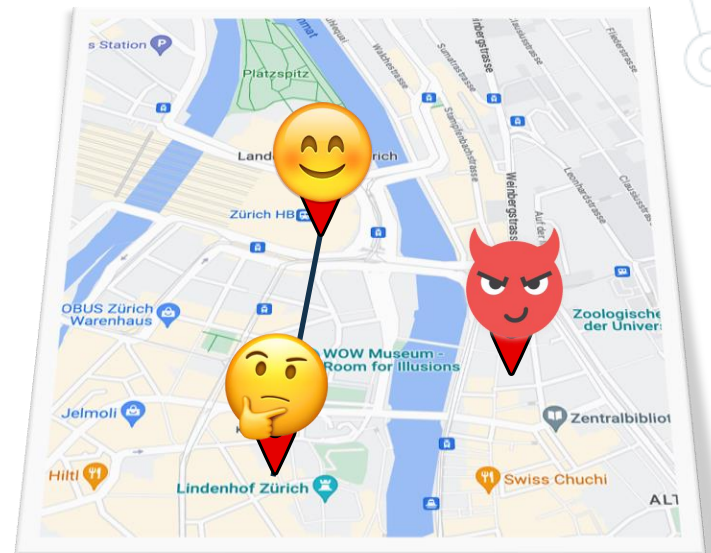
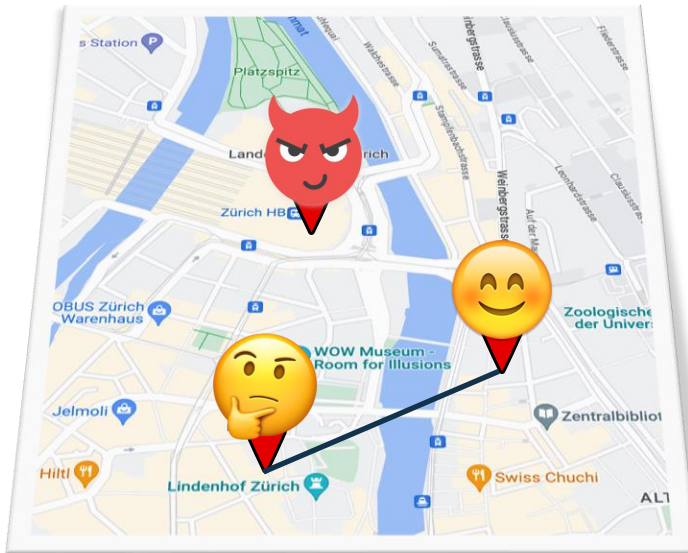
$$t = \frac{n}{\omega} \text{ is impossible}$$

Example for \mathbb{R}^2 : $t = \frac{n}{3}$ is impossible.



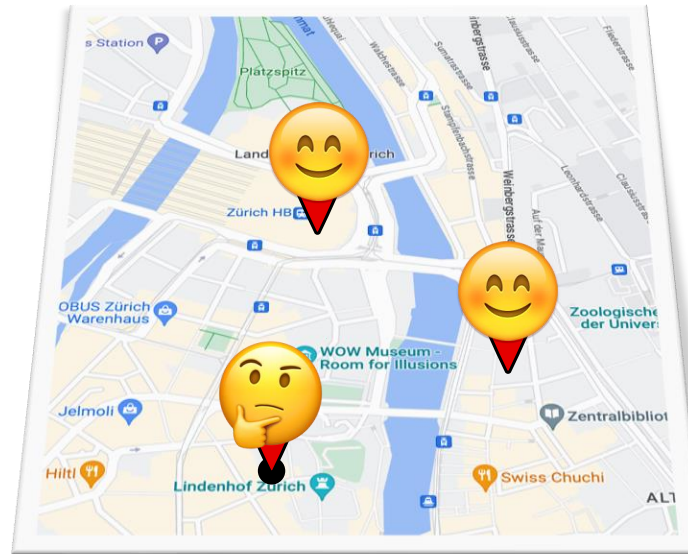
$t = \frac{n}{\omega}$ is impossible

Example for \mathbb{R}^2 : $t = \frac{n}{3}$ is impossible.



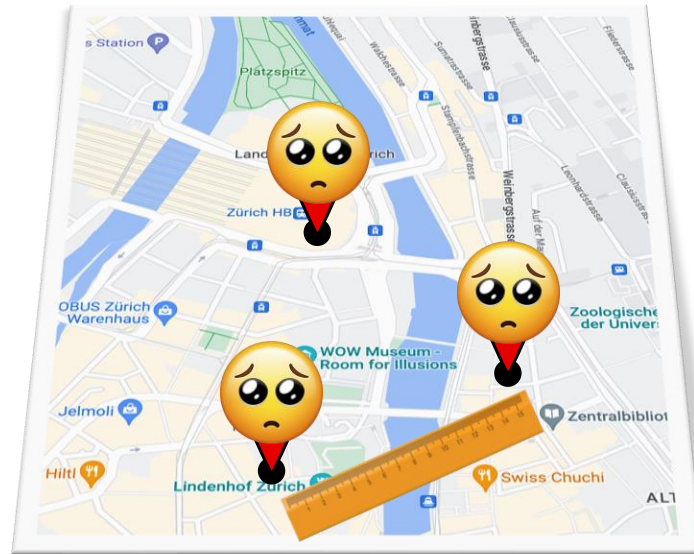
$t = \frac{n}{\omega}$ is impossible

Example for \mathbb{R}^2 : $t = \frac{n}{3}$ is impossible.



$t = \frac{n}{\omega}$ is impossible

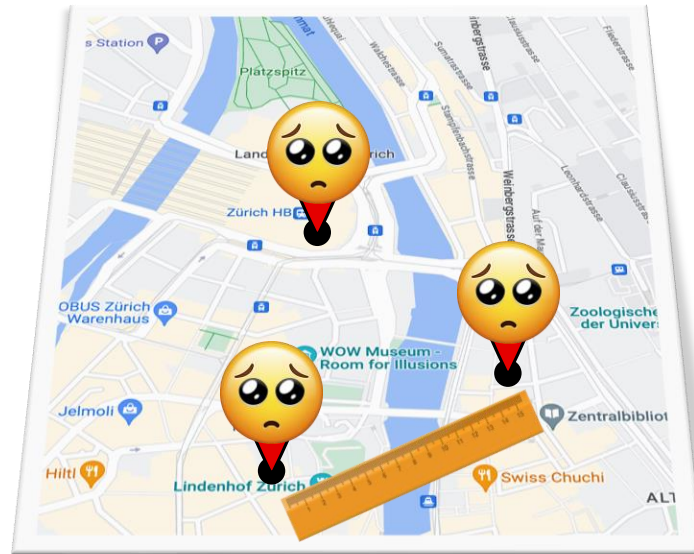
Example for \mathbb{R}^2 : $t = \frac{n}{3}$ is impossible.



General *convexity spaces* C :

$t = \frac{n}{\omega}$ is impossible

Example for \mathbb{R}^2 : $t = \frac{n}{3}$ is impossible.

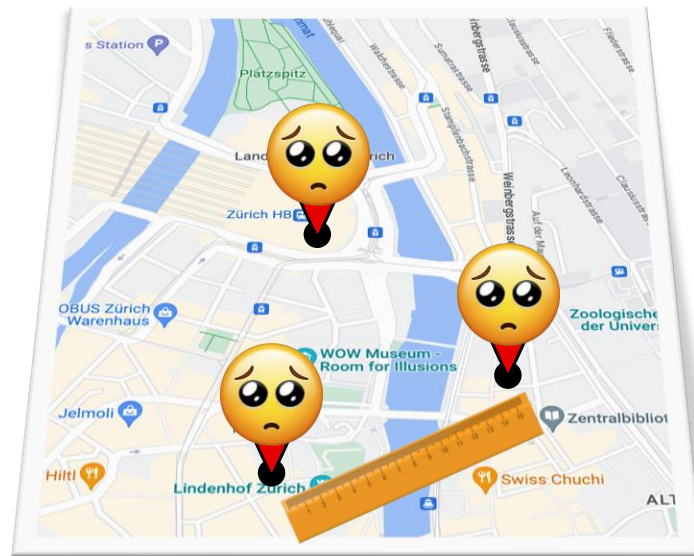


General *convexity spaces* C :

- ω parties, 1 corrupted.

$$t = \frac{n}{\omega} \text{ is impossible}$$

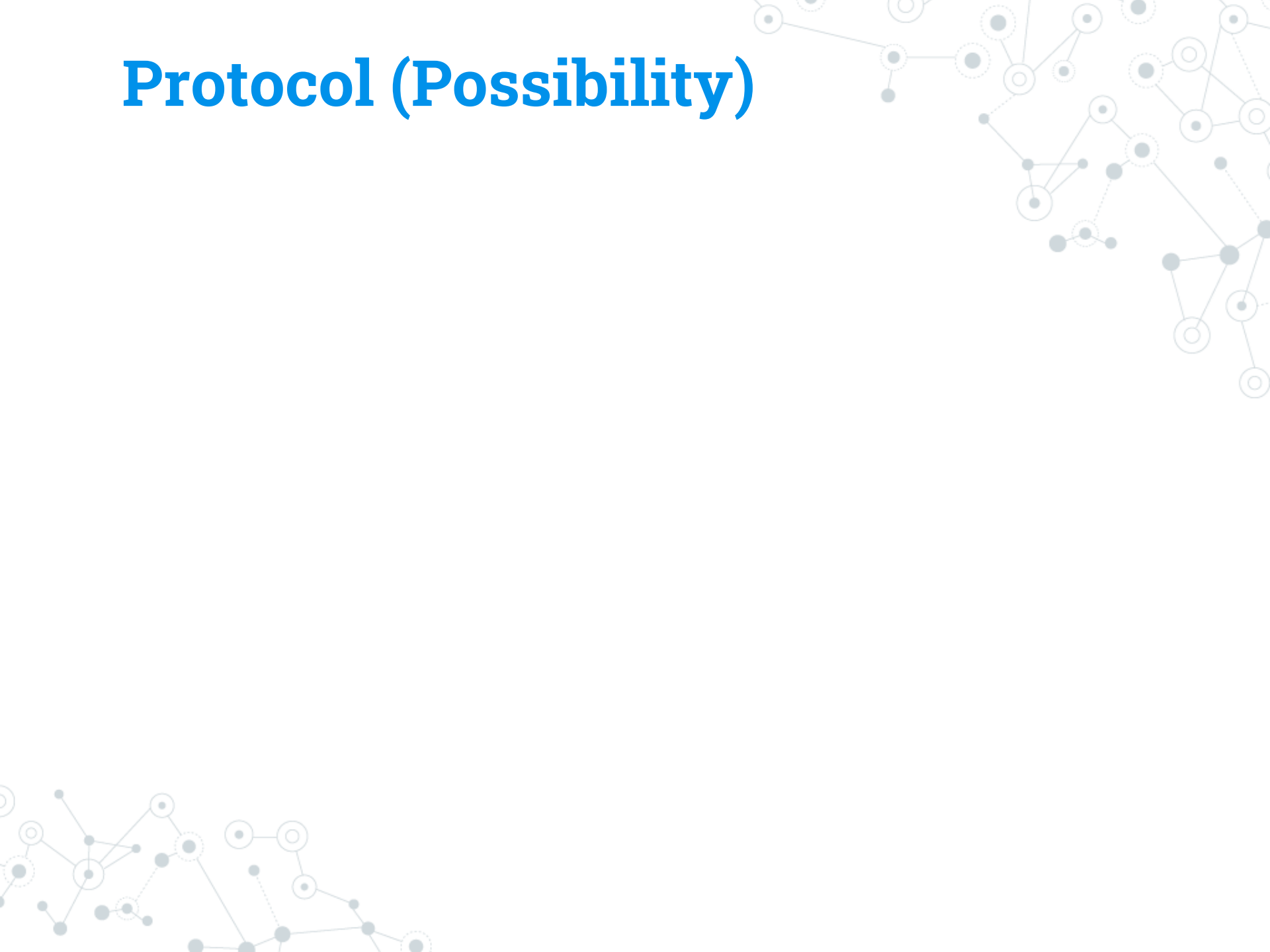
Example for \mathbb{R}^2 : $t = \frac{n}{3}$ is impossible.



General *convexity spaces* C :

- ω parties, 1 corrupted.
- *Bad* instance guaranteed by def. of ω .
- Previously known: \mathbb{R}^D and *convex geometries*.

Protocol (Possibility)



Protocol (Possibility)

A decorative network diagram in the top right corner, consisting of various nodes (some solid grey, some hollow white) connected by thin grey lines, forming a complex web-like structure.

A) Parties *distribute* inputs to get a *common view* on (x_1, \dots, x_n) :

A decorative network diagram in the bottom left corner, similar to the one in the top right, with nodes and connecting lines.

Protocol (Possibility)

A) Parties *distribute* inputs to get a *common view* on (x_1, \dots, x_n) :

(some values *unknown*: $x_i = \perp$)

Protocol (Possibility)

A) Parties *distribute* inputs to get a *common view* on (x_1, \dots, x_n) :

(some values *unknown*: $x_i = \perp$)

1. Known honest values are correct;

Protocol (Possibility)

A) Parties *distribute* inputs to get a *common view* on (x_1, \dots, x_n) :

(some values *unknown*: $x_i = \perp$)

1. Known honest values are correct;
2. $\geq n - t_s$ values known;

Protocol (Possibility)

A) Parties *distribute* inputs to get a *common view* on (x_1, \dots, x_n) :

(some values *unknown*: $x_i = \perp$)

1. Known honest values are correct;
2. $\geq n - t_s$ values known;
3. Synchronous network \Rightarrow all honest values known.

Protocol (Possibility)

A) Parties *distribute* inputs to get a *common view* on (x_1, \dots, x_n) :

(some values *unknown*: $x_i = \perp$)

1. Known honest values are correct;
2. $\geq n - t_s$ values known;
3. Synchronous network \Rightarrow all honest values known.

“Core-Set Agreement” for $2t_s + t_a < n$

Protocol (Possibility)

A) Parties *distribute* inputs to get a *common view* on (x_1, \dots, x_n) :

(some values *unknown*: $x_i = \perp$)

1. Known honest values are correct;
2. $\geq n - t_s$ values known;
3. Synchronous network \Rightarrow all honest values known.

“Core-Set Agreement” for $2 t_s + t_a < n$ (main tech. novelty)

Protocol (Possibility)

A) Parties *distribute* inputs to get a *common view* on (x_1, \dots, x_n) :

(some values *unknown*: $x_i = \perp$)

1. Known honest values are correct;
2. $\geq n - t_s$ values known;
3. Synchronous network \Rightarrow all honest values known.

“Core-Set Agreement” for $2t_s + t_a < n$ (main tech. novelty)

B) Parties *locally* and *deterministically* compute a **valid** output

Protocol (Possibility)

A) Parties *distribute* inputs to get a *common view* on (x_1, \dots, x_n) :

(some values *unknown*: $x_i = \perp$)

1. Known honest values are correct;
2. $\geq n - t_s$ values known;
3. Synchronous network \Rightarrow all honest values known.

“Core-Set Agreement” for $2t_s + t_a < n$ (main tech. novelty)

B) Parties *locally* and *deterministically* compute a **valid** output by taking the “safe area”

Protocol (Possibility)

A) Parties *distribute* inputs to get a *common view* on (x_1, \dots, x_n) :

(some values *unknown*: $x_i = \perp$)

1. Known honest values are correct;
2. $\geq n - t_s$ values known;
3. Synchronous network \Rightarrow all honest values known.

“Core-Set Agreement” for $2t_s + t_a < n$ (**main tech. novelty**)

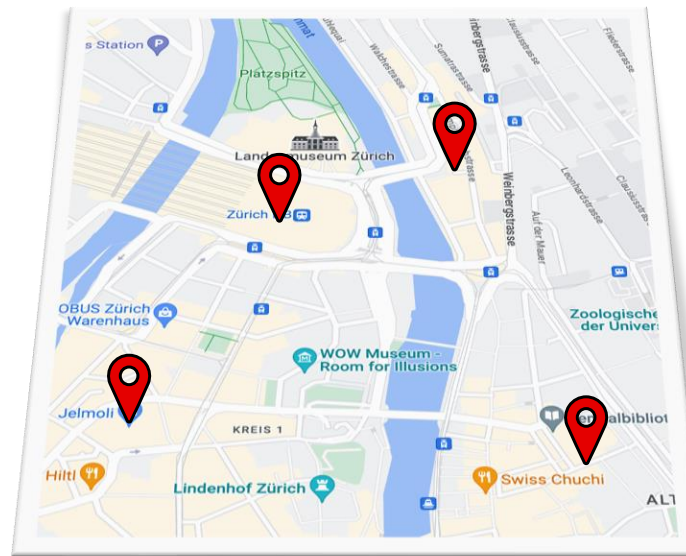
B) Parties *locally* and *deterministically* compute a **valid** output by taking the “*safe area*” (**Agreement** is for free).

Safe Area



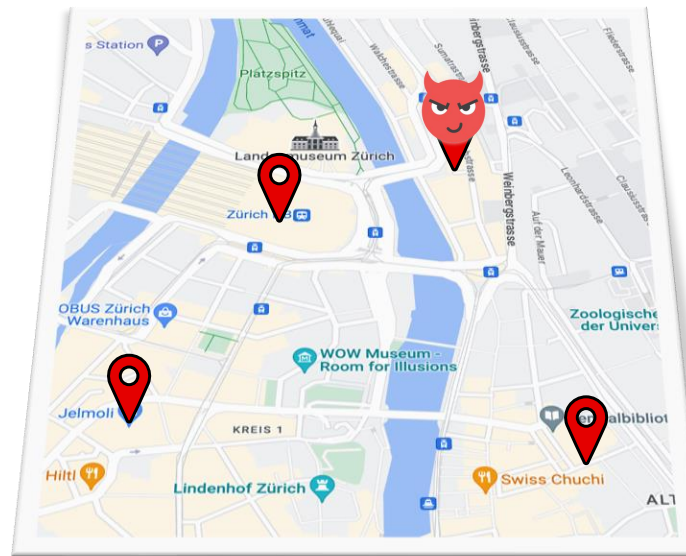
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



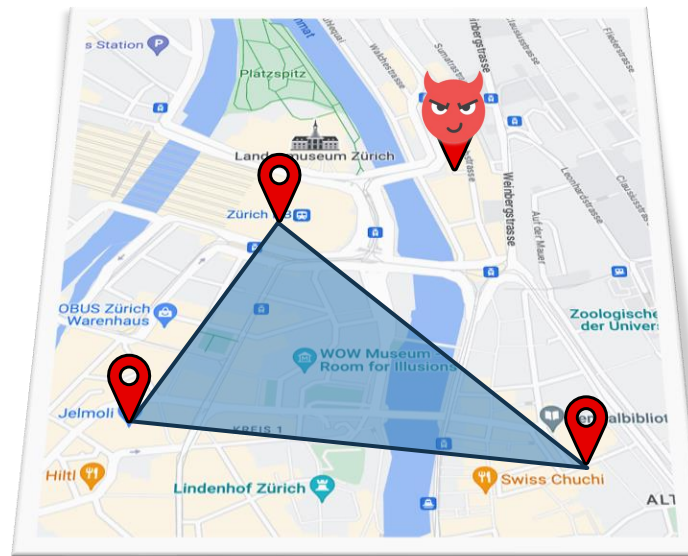
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



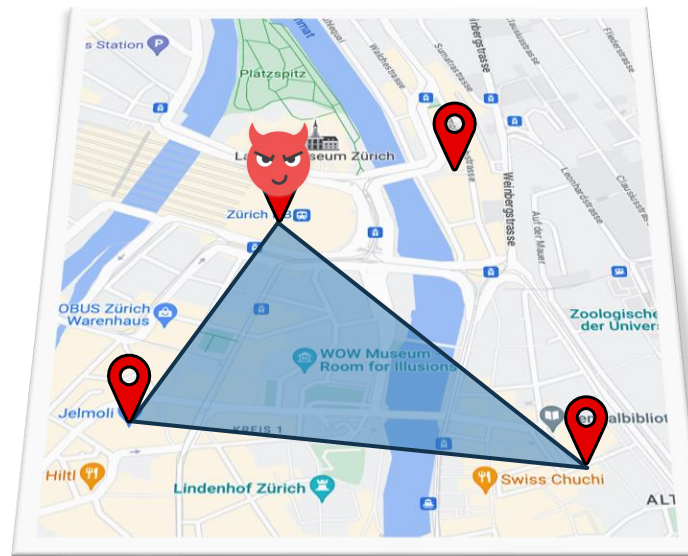
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



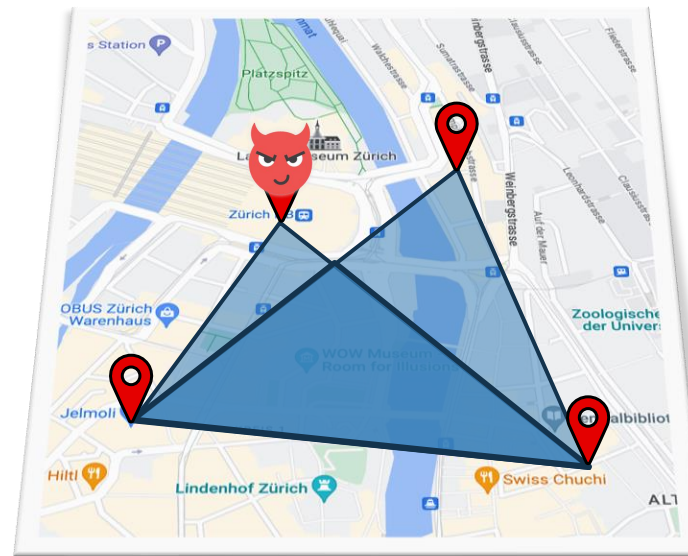
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



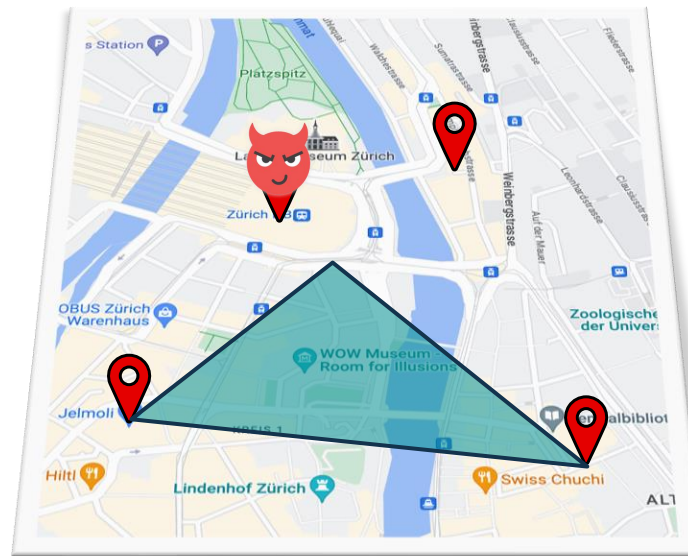
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



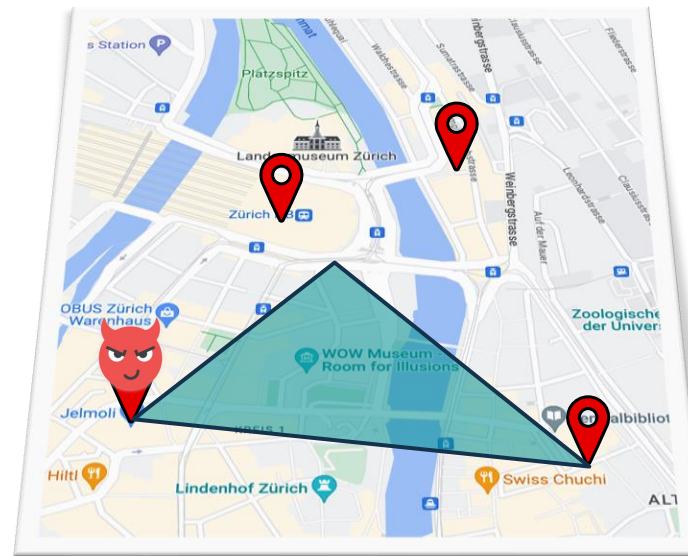
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



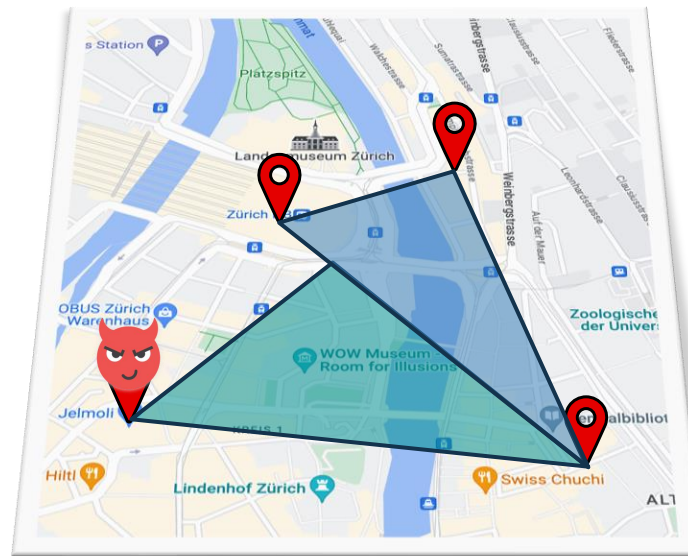
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



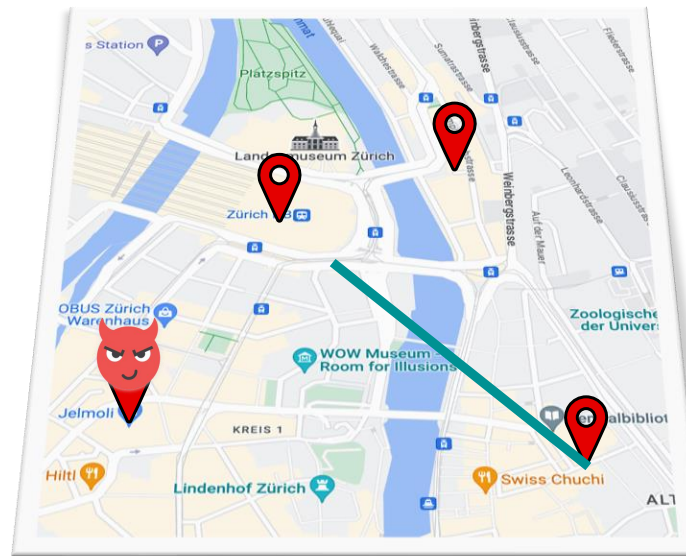
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



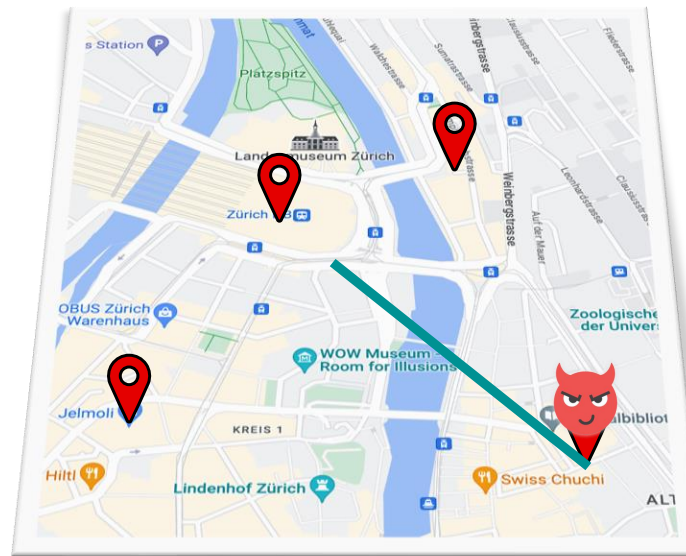
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



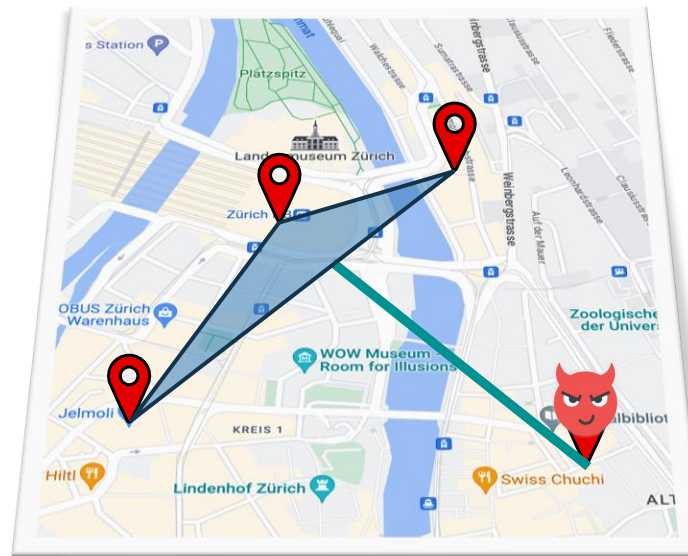
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



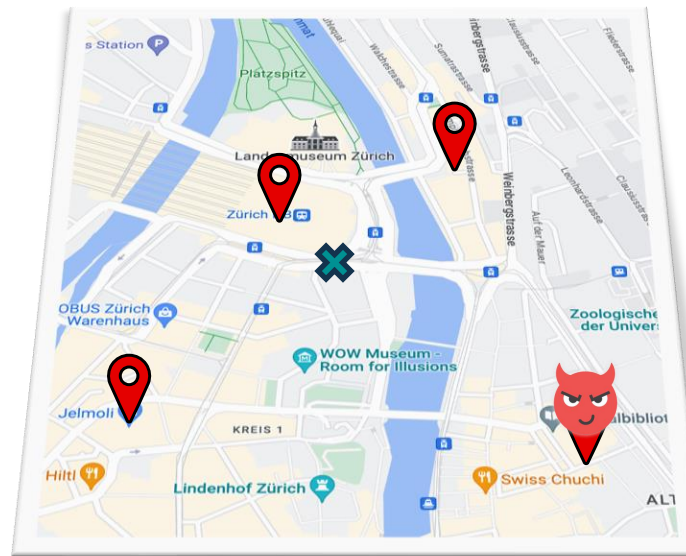
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



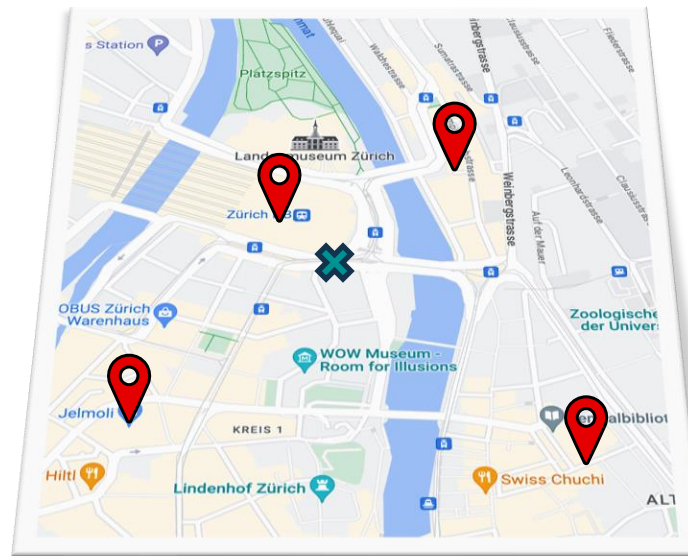
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



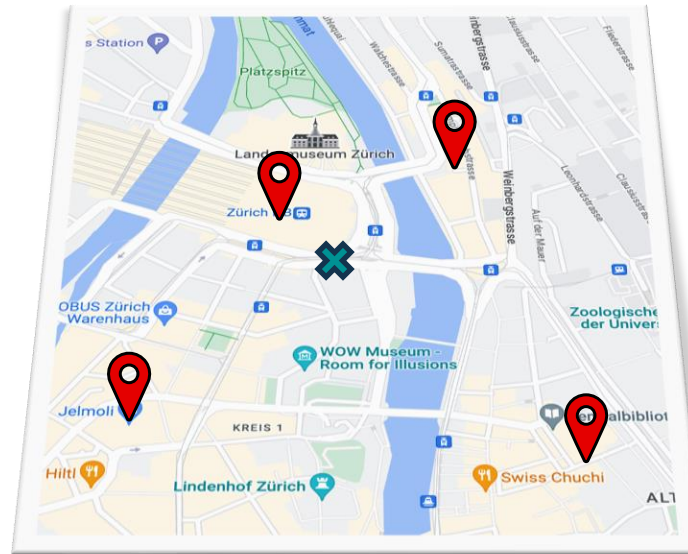
Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?

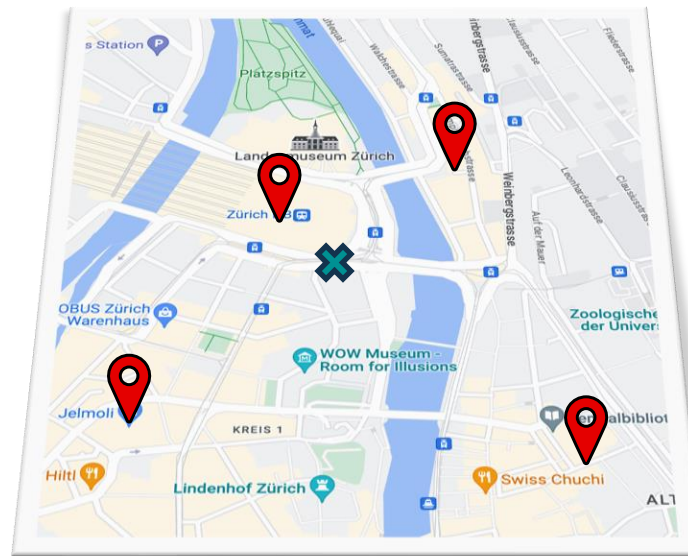


Generally: a values in the common view, $\leq b$ corrupted

Safe area: intersection of convex hulls of subsets of size $a - b$

Safe Area

e.g., 4 values in the common view, ≤ 1 corrupted, what to do?



Generally: a values in the common view, $\leq b$ corrupted

Safe area: intersection of convex hulls of subsets of size $a - b$

Any point in safe area is **valid**, select one *deterministically*.

Safe Area (cont'd)



Safe Area (cont'd)

Who are *a* and *b*?



Safe Area (cont'd)

Who are a and b ?

The common view has $\geq n - t_s$ values,



Safe Area (cont'd)

Who are a and b ?

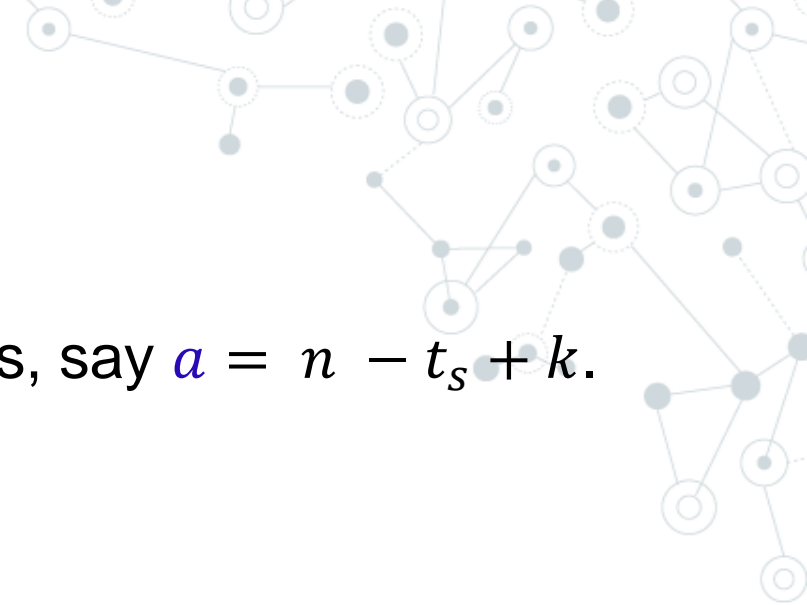
The common view has $\geq n - t_s$ values, say $a = n - t_s + k$.

Safe Area (cont'd)

Who are a and b ?

The common view has $\geq n - t_s$ values, say $a = n - t_s + k$.

The network is synchronous?



Safe Area (cont'd)

Who are a and b ?

The common view has $\geq n - t_s$ values, say $a = n - t_s + k$.

The network is synchronous?

\Rightarrow at most k of these values are corrupted.

Safe Area (cont'd)

Who are a and b ?

The common view has $\geq n - t_s$ values, say $a = n - t_s + k$.

The network is synchronous?

\Rightarrow at most k of these values are corrupted.

The network is asynchronous?

Safe Area (cont'd)

Who are a and b ?

The common view has $\geq n - t_s$ values, say $a = n - t_s + k$.

The network is synchronous?

\Rightarrow at most k of these values are corrupted.

The network is asynchronous?

\Rightarrow at most t_a of these values are corrupted.

Safe Area (cont'd)

Who are a and b ?

The common view has $\geq n - t_s$ values, say $a = n - t_s + k$.

The network is synchronous?

\Rightarrow at most k of these values are corrupted.

The network is asynchronous?

\Rightarrow at most t_a of these values are corrupted.

We don't know which?

Safe Area (cont'd)

Who are a and b ?

The common view has $\geq n - t_s$ values, say $a = n - t_s + k$.

The network is synchronous?

\Rightarrow at most k of these values are corrupted.

The network is asynchronous?

\Rightarrow at most t_a of these values are corrupted.

We don't know which?

$\Rightarrow b = \max(k, t_a)$

Safe Area (cont'd)

Who are a and b ?

The common view has $\geq n - t_s$ values, say $a = n - t_s + k$.

The network is synchronous?

\Rightarrow at most k of these values are corrupted.

The network is asynchronous?

\Rightarrow at most t_a of these values are corrupted.

We don't know which?

$\Rightarrow b = \max(k, t_a)$

Q: Why is the safe area non-empty?

Safe Area (cont'd)

Who are a and b ?

The common view has $\geq n - t_s$ values, say $a = n - t_s + k$.

The network is synchronous?

\Rightarrow at most k of these values are corrupted.

The network is asynchronous?

\Rightarrow at most t_a of these values are corrupted.

We don't know which?

$\Rightarrow b = \max(k, t_a)$

Q: Why is the safe area non-empty?

A: We intersect **many convex sets**,

Safe Area (cont'd)

Who are a and b ?

The common view has $\geq n - t_s$ values, say $a = n - t_s + k$.

The network is synchronous?

\Rightarrow at most k of these values are corrupted.

The network is asynchronous?

\Rightarrow at most t_a of these values are corrupted.

We don't know which?

$\Rightarrow b = \max(k, t_a)$

Q: Why is the safe area non-empty?

A: We intersect **many convex sets**, but it suffices to show any ω intersect;

Safe Area (cont'd)

Who are a and b ?

The common view has $\geq n - t_s$ values, say $a = n - t_s + k$.

The network is synchronous?

\Rightarrow at most k of these values are corrupted.

The network is asynchronous?

\Rightarrow at most t_a of these values are corrupted.

We don't know which?

$\Rightarrow b = \max(k, t_a)$

Q: Why is the safe area non-empty?

A: We intersect **many convex sets**, but it suffices to show any ω intersect; Pigeonhole Principle;

Safe Area (cont'd)

Who are a and b ?

The common view has $\geq n - t_s$ values, say $a = n - t_s + k$.

The network is synchronous?

\Rightarrow at most k of these values are corrupted.

The network is asynchronous?

\Rightarrow at most t_a of these values are corrupted.

We don't know which?

$\Rightarrow b = \max(k, t_a)$

Q: Why is the safe area non-empty?

A: We intersect **many convex sets**, but it suffices to show any ω intersect; Pigeonhole Principle; $\max(\omega t_s, \omega t_a + t_s) < n$

5. Outlook

And shameless self-advertising







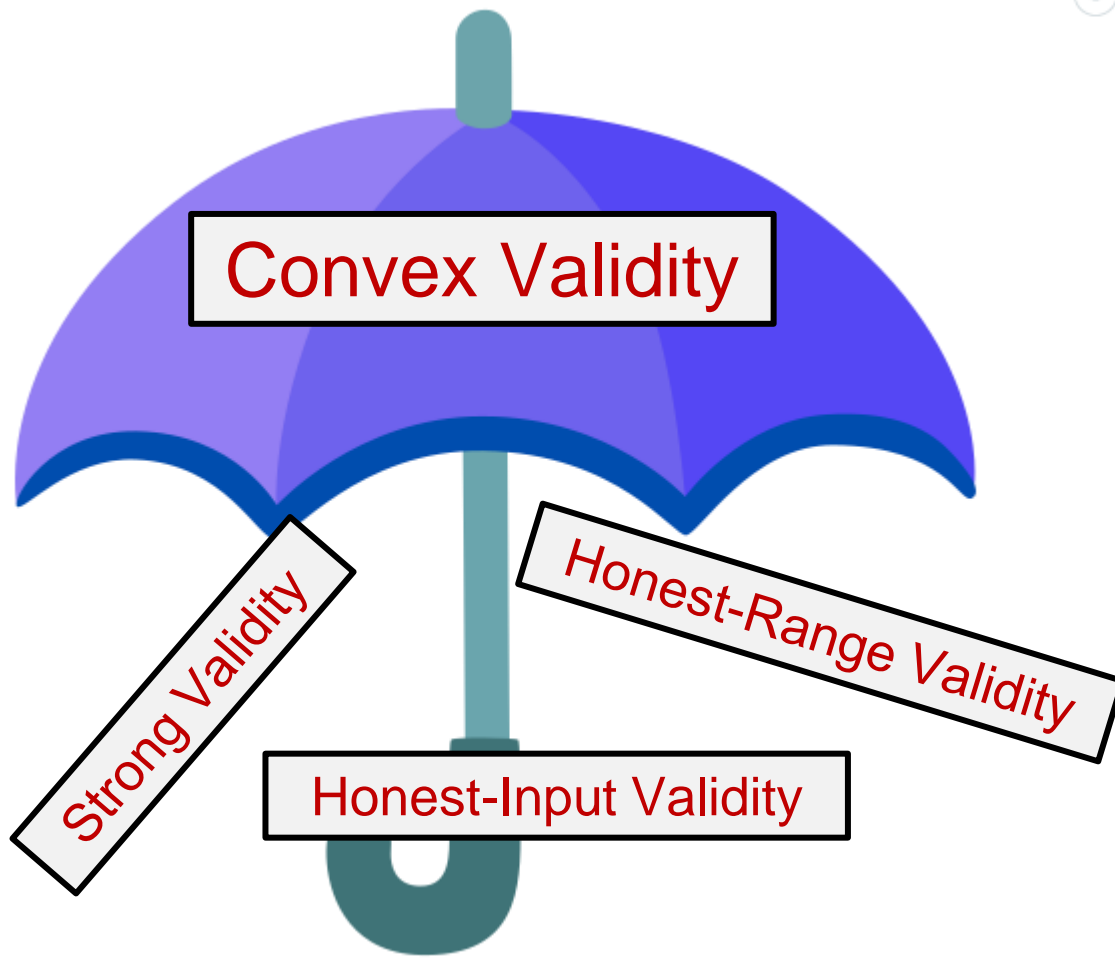
Convex Validity

Strong Validity

Honest-Range Validity

Honest-Input Validity





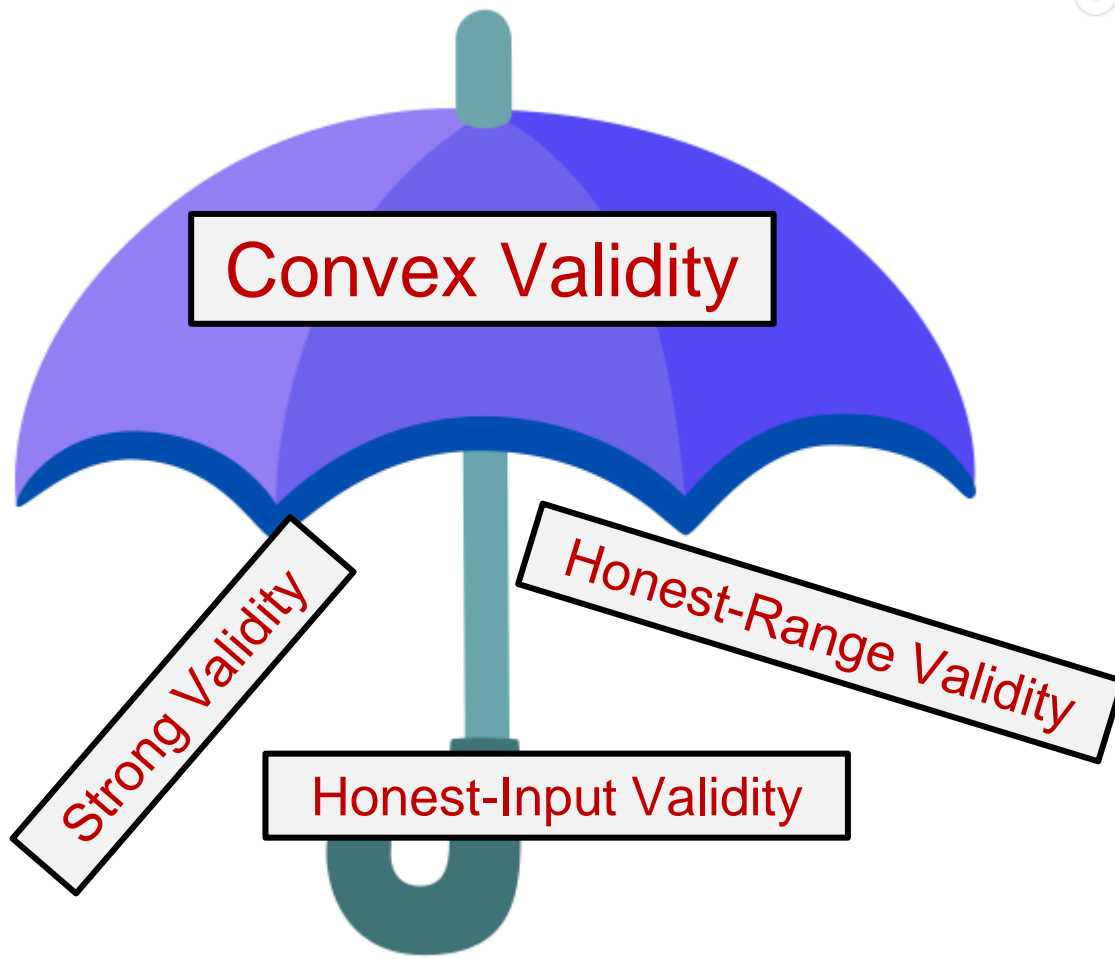
Convex Validity

Strong Validity

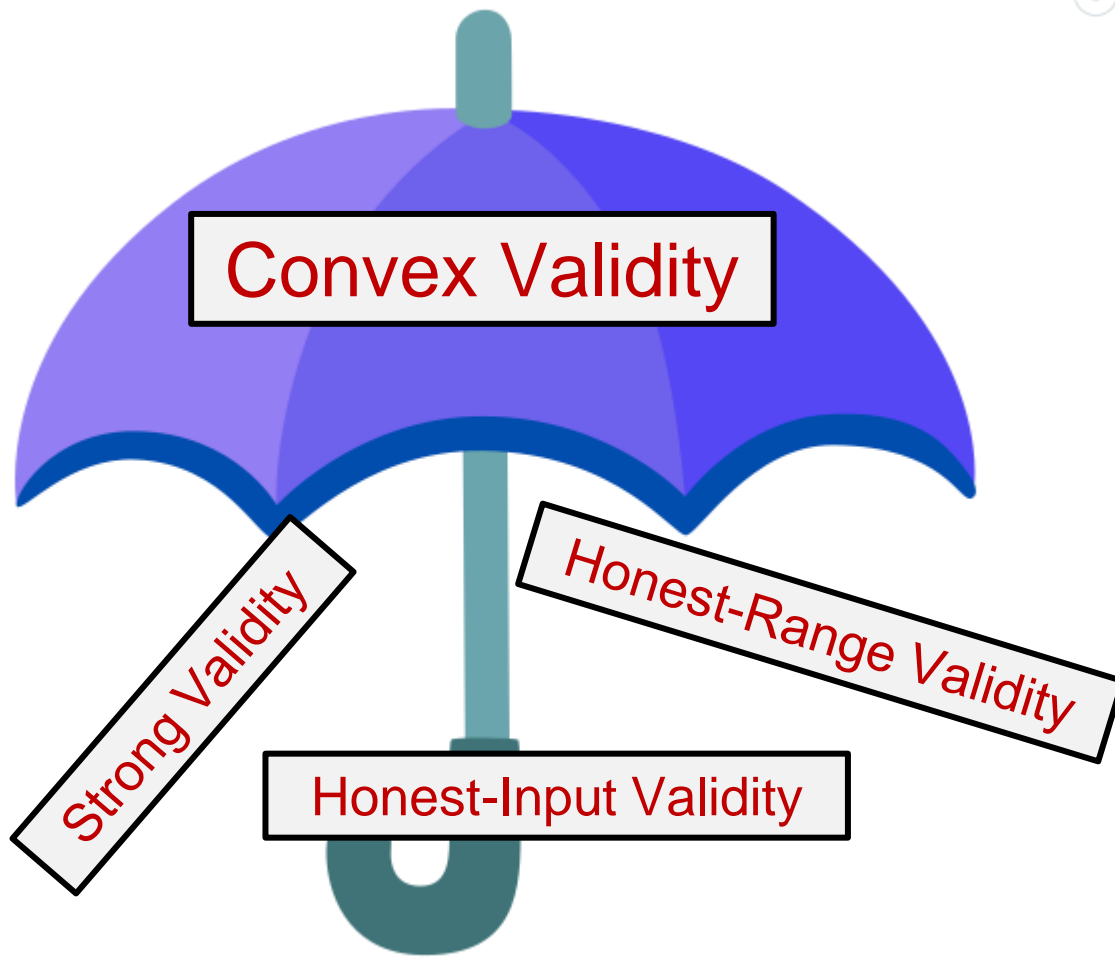
Honest-Range Validity

Honest-Input Validity

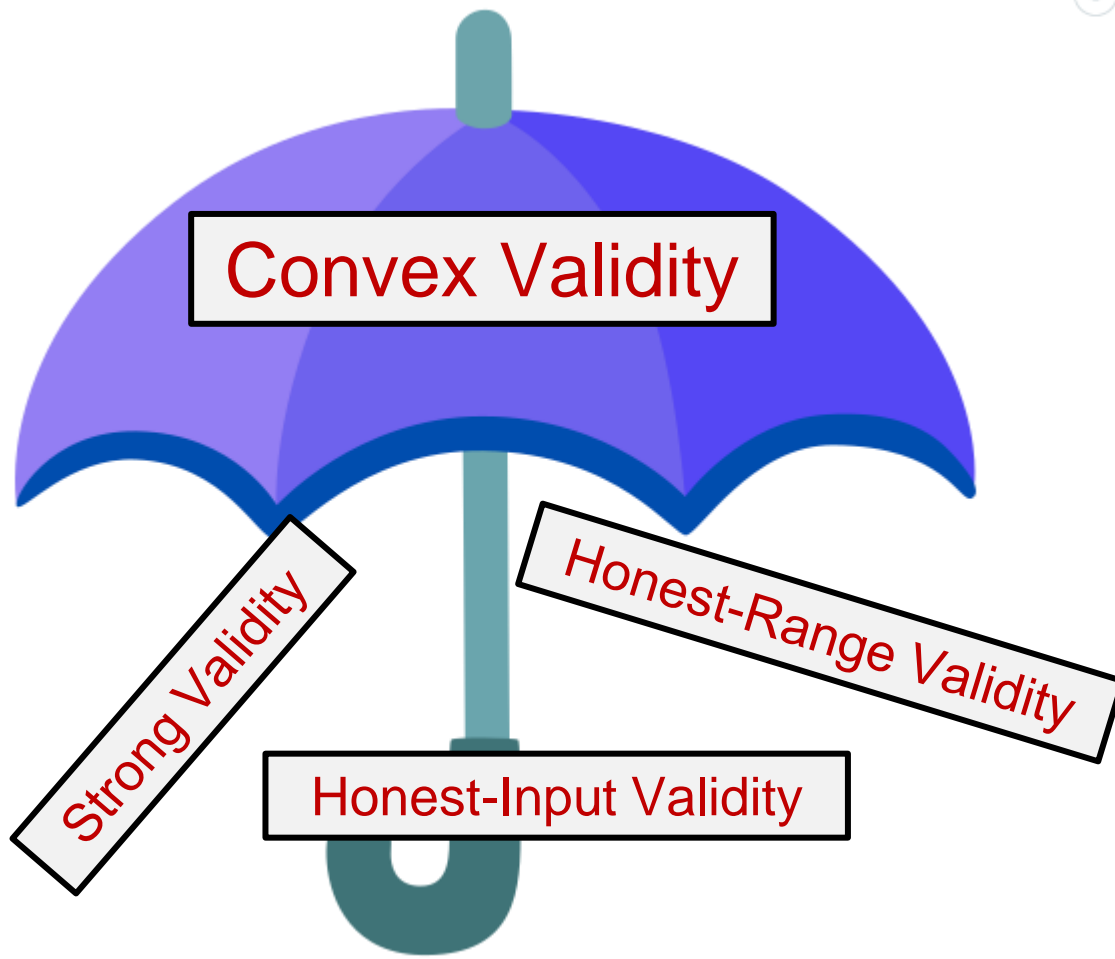




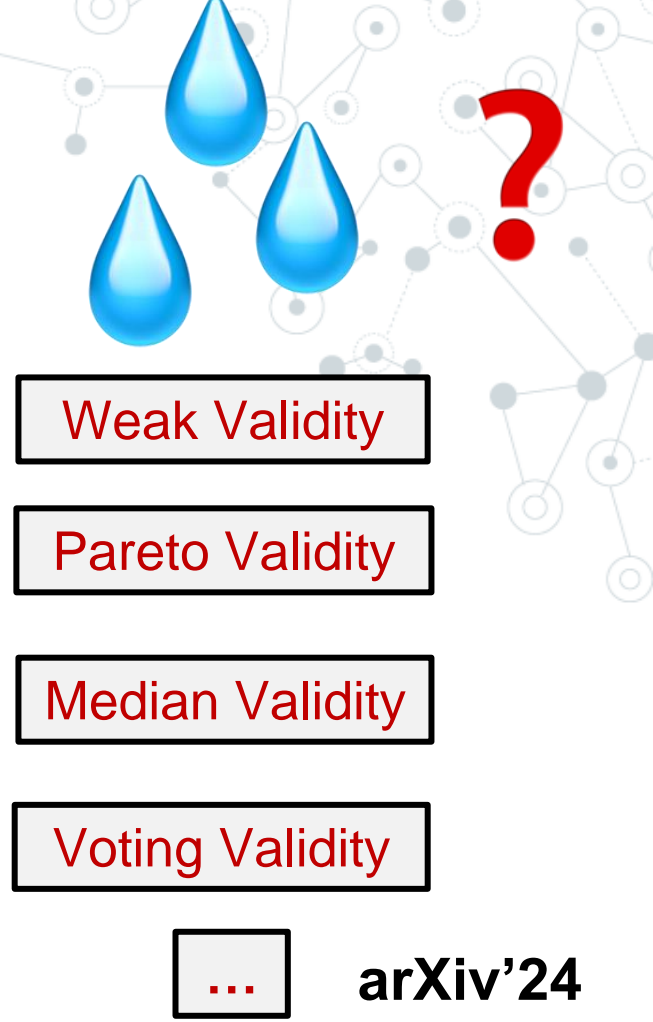
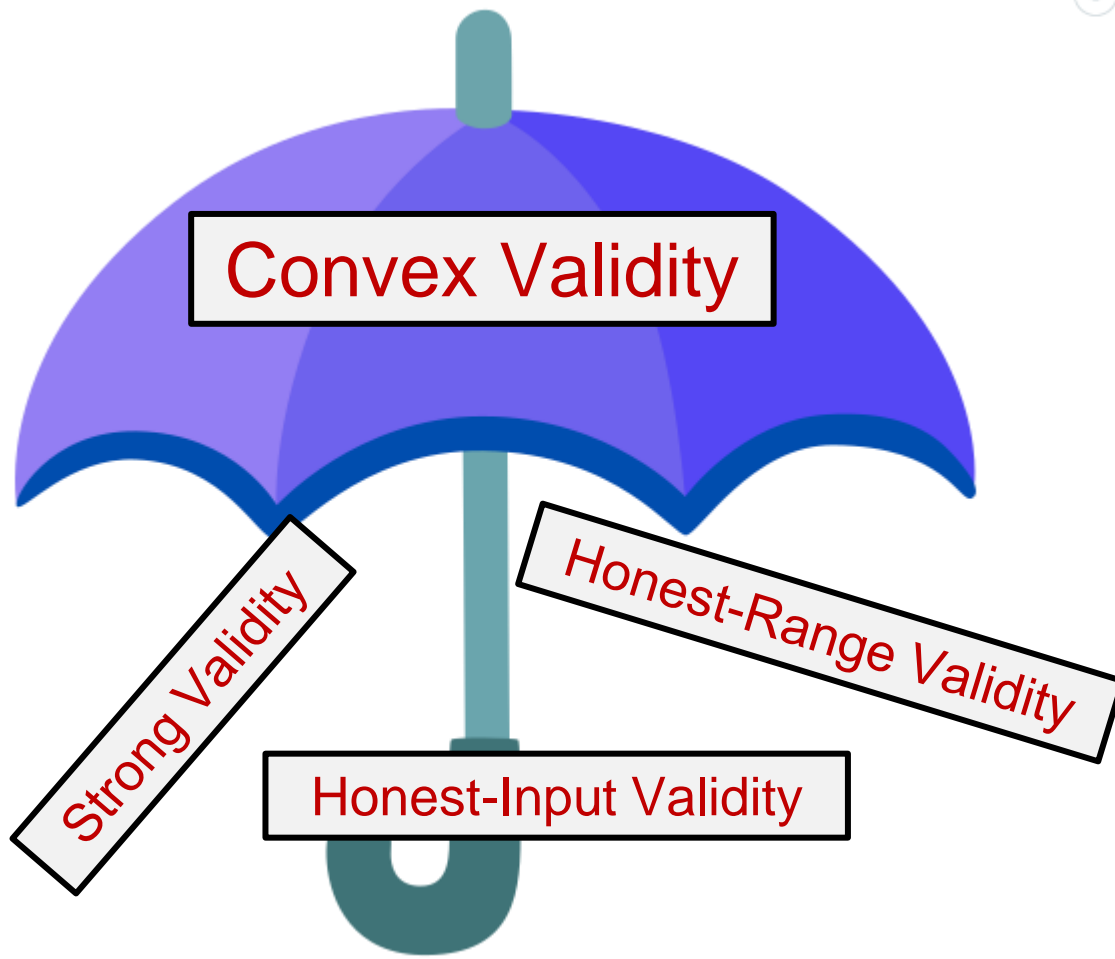
Weak Validity

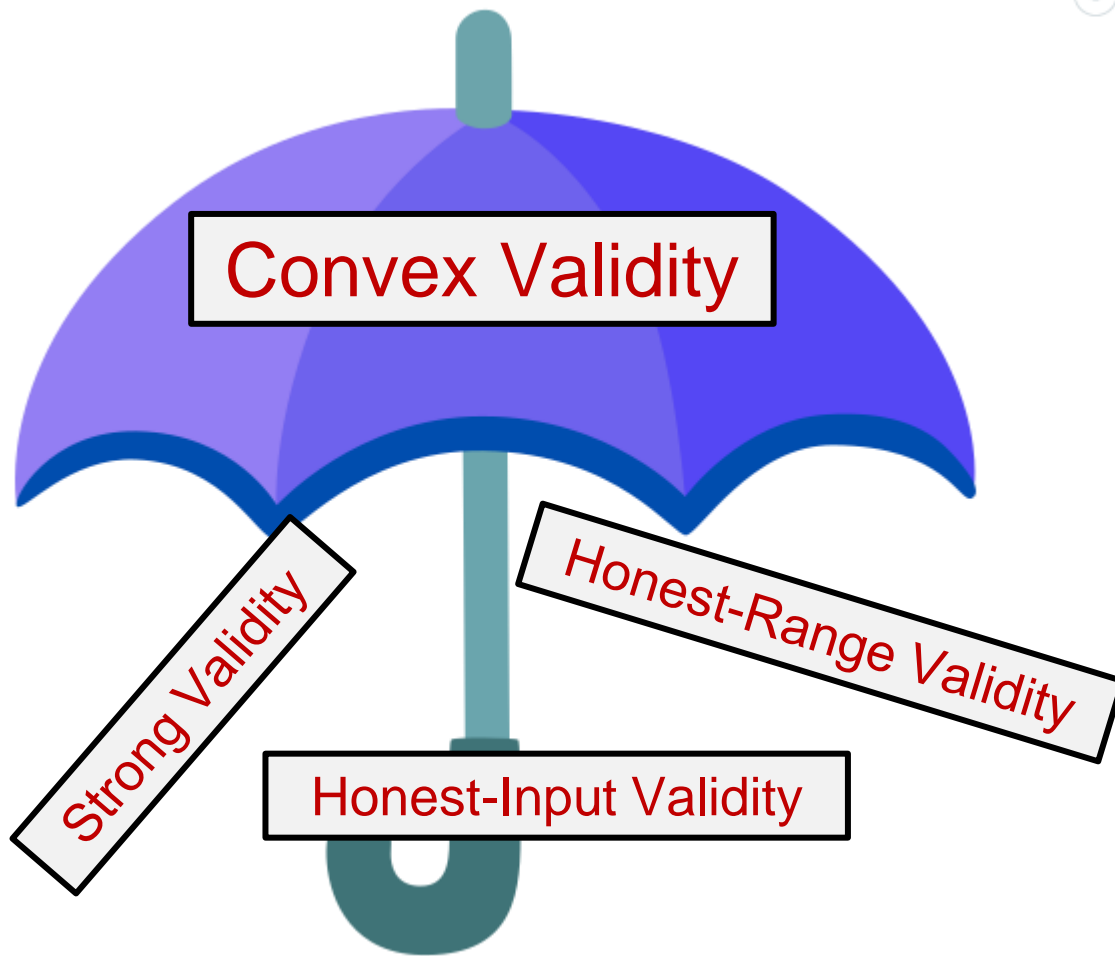


- Weak Validity
- Pareto Validity
- Median Validity
- Voting Validity
- ...



- Weak Validity
- Pareto Validity
- Median Validity
- Voting Validity
- ...

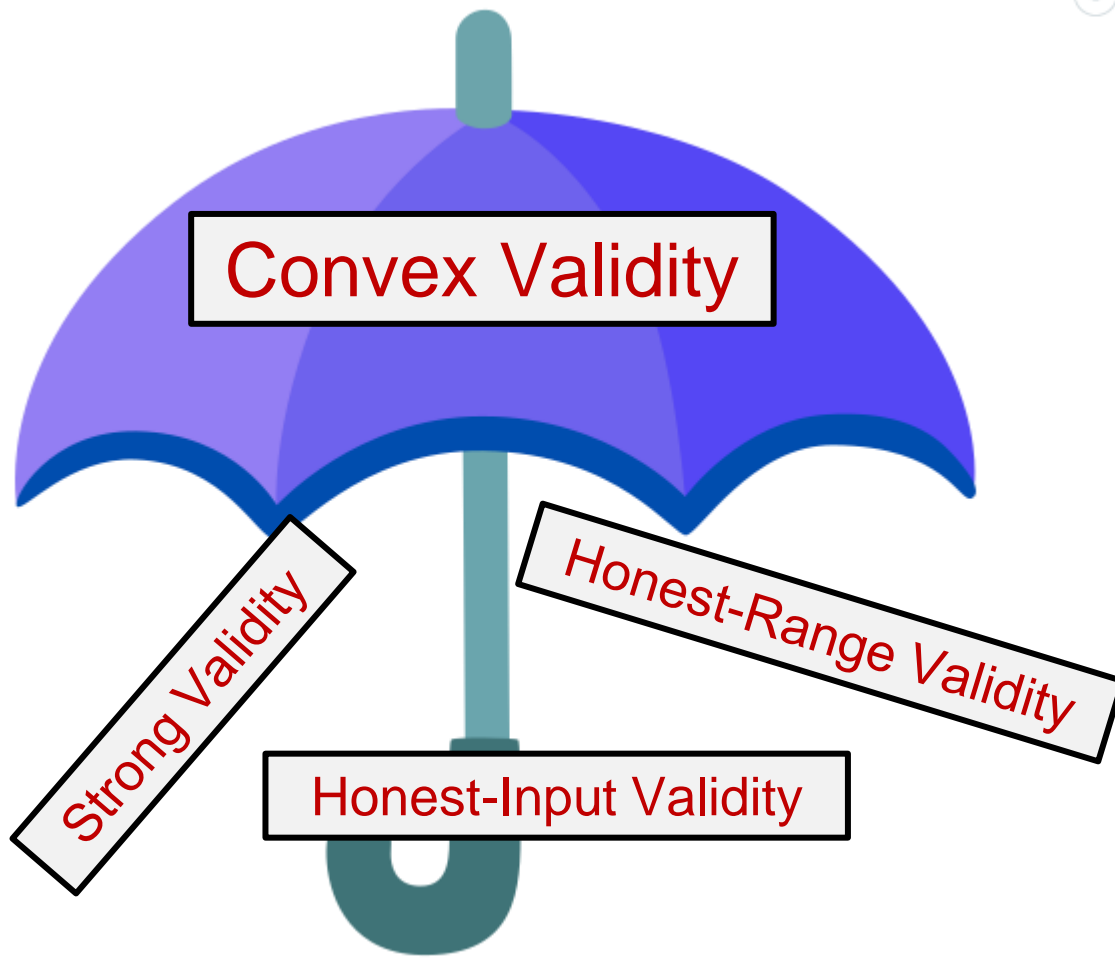




- Weak Validity
- Pareto Validity
- Median Validity
- Voting Validity

... arXiv'24

Approximate Agreement?



- Weak Validity
- Pareto Validity
- Median Validity
- Voting Validity
- ...

arXiv'24

Approximate Agreement?
Other adversaries?

Hope you enjoyed!



Hope you enjoyed!

