Verification: Architecture, networks, robustness, and complexity

John Doyle

John G Braun Professor

Control and Dynamical Systems

BioEngineering, Electrical Engineering

Caltech

"Architecture" in engineering, medicine, and social systems

Design (& understanding) shifting from

- Components to
- Platforms to
- Systems to
- Systems of systems to
- Architectures

Increasingly dominated by

- network fragilities
- cascading failures
- infectious hijacking
- "unintended consequences"

Why is this hard?

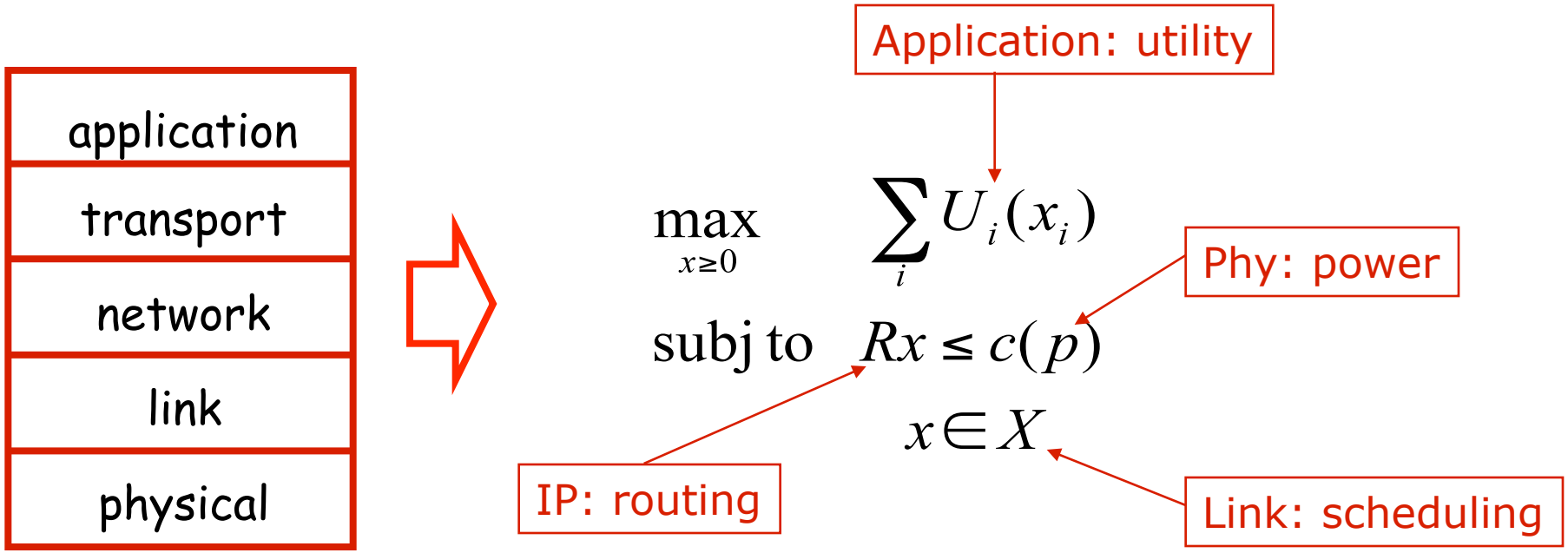
- P≠NP≠coNP
- Distributed, layered
- Dynamic, physical
- Adversarial

Why is their hope?

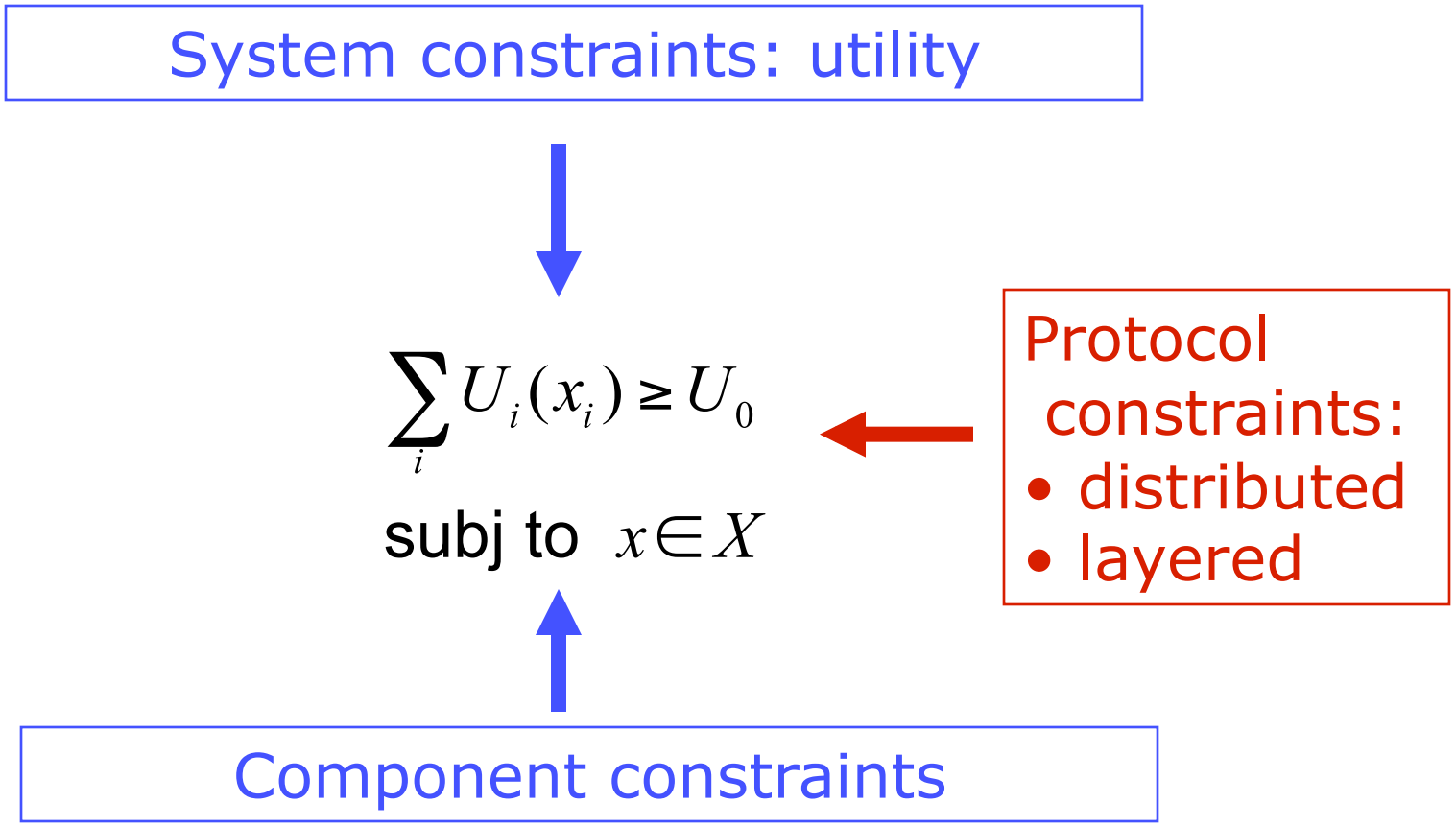
- Duality is unifying
- More "constrained" ⇒ more verifiable?

Architecture as dual decomposition?

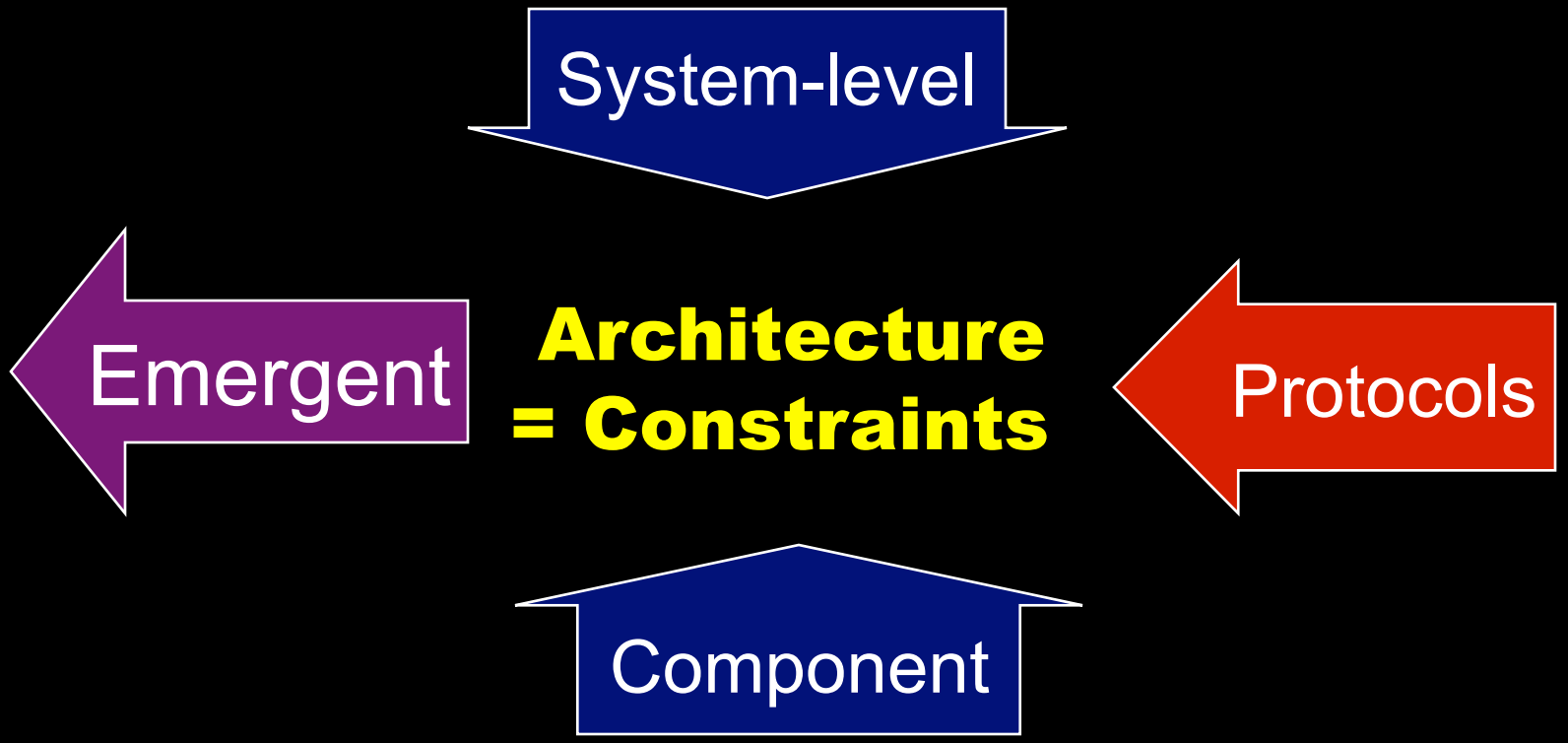
Existing theory



Architecture as dual decomposition



Emergent constraint:
Nontrivial consequence of other constraints



Verification =
{emergent∩bad=∅}

Why is this hard?

- P≠NP≠coNP
- Distributed, layered
- Dynamic, physical
- Adversarial

Why is their hope?

- **Duality is unifying**
- More "constrained" ⇒ more verifiable?

Why is this hard?

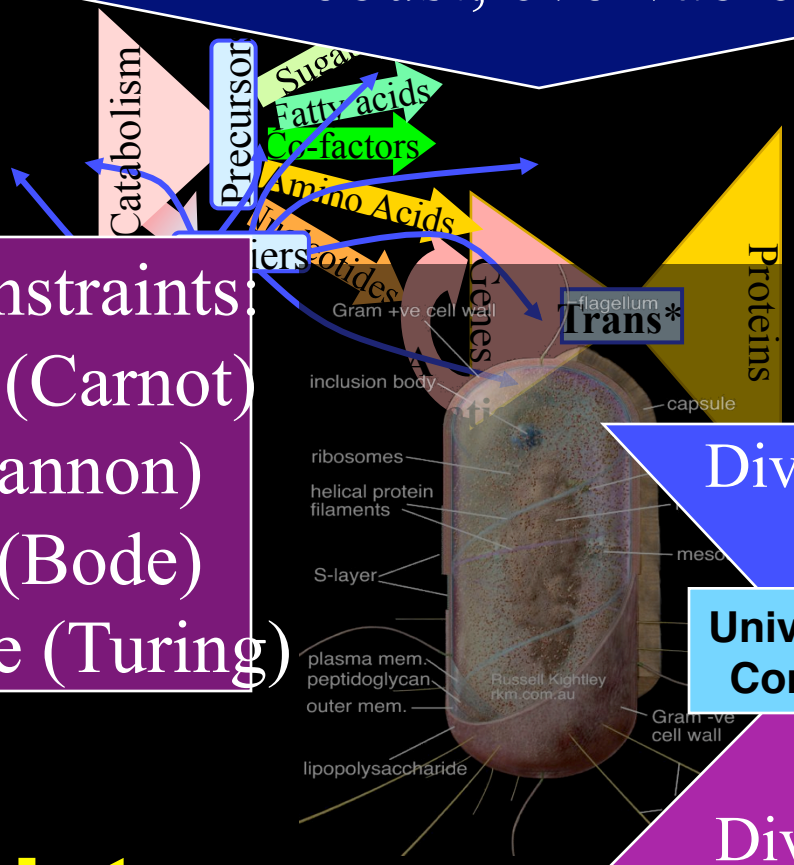
- P≠NP≠coNP
- Distributed, layered
- Dynamic, physical
- Adversarial

Why is their hope?

- Duality is unifying
- **More "constrained" ⇒ more verifiable?**

Systems requirements:
functional, efficient,
robust, evolvable

Hard constraints:
Thermo (Carnot)
Info (Shannon)
Control (Bode)
Compute (Turing)



Diverse

Universal Control

Diverse

Protocols

Constraints

Components and materials:
Energy, moieties

Emergent constraints: hard limits and tradeoffs

On systems and their components

- Thermodynamics (Carnot)
- Communications (Shannon)
- Control (Bode)
- Computation (Turing/Gödel)

Assume ***different*** architectures a priori.

No networks

Hard limits and tradeoffs

On systems and their components

- Thermodynamics (Carnot)
- Communications (Shannon)
- Control (Bode)
- Computation (Turing/Gödel)

} No dynamics or feedback

- Fragmented and incompatible
- Cannot be used as a basis for comparing architectures
- New unifications are encouraging

Hard limits and tradeoffs

On systems and their components

- Thermodynamics (Carnot)
- Communications (Shannon)
- Control (Bode)
- Computation (Turing/Gödel)

Robust/
fragile
is unifying
concept

- Include dynamics and feedback
- Extend to networks
- **New unifications are encouraging**

Hard tradeoffs

Conjecture: There are hard tradeoffs between

- Efficiency (energy, materials)
- Robustness

- So far: Hard tradeoffs in autocatalytic networks
- Conjecture: Robust, efficient, “modular” systems are more easily verified

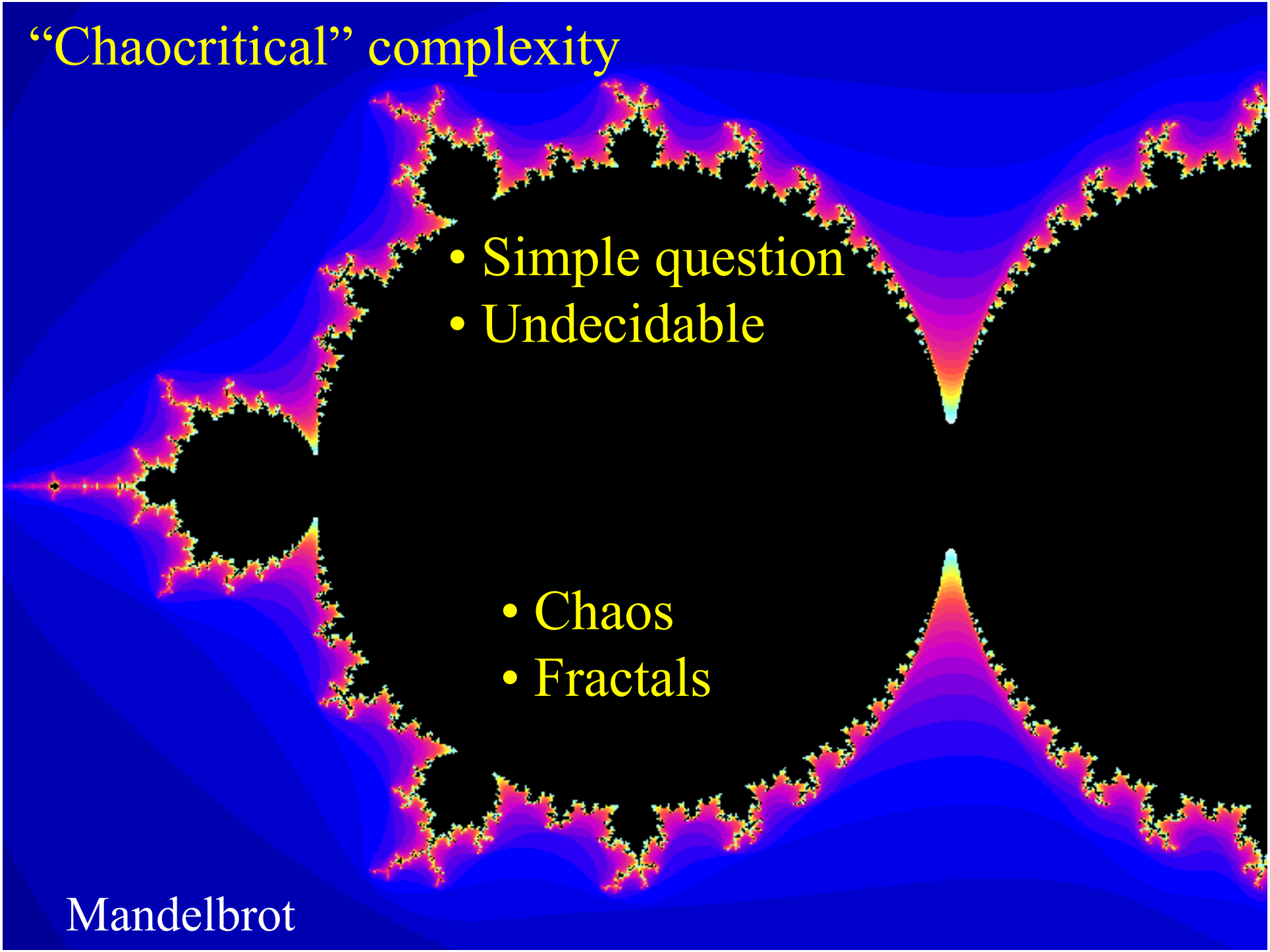
Hard limits and tradeoffs

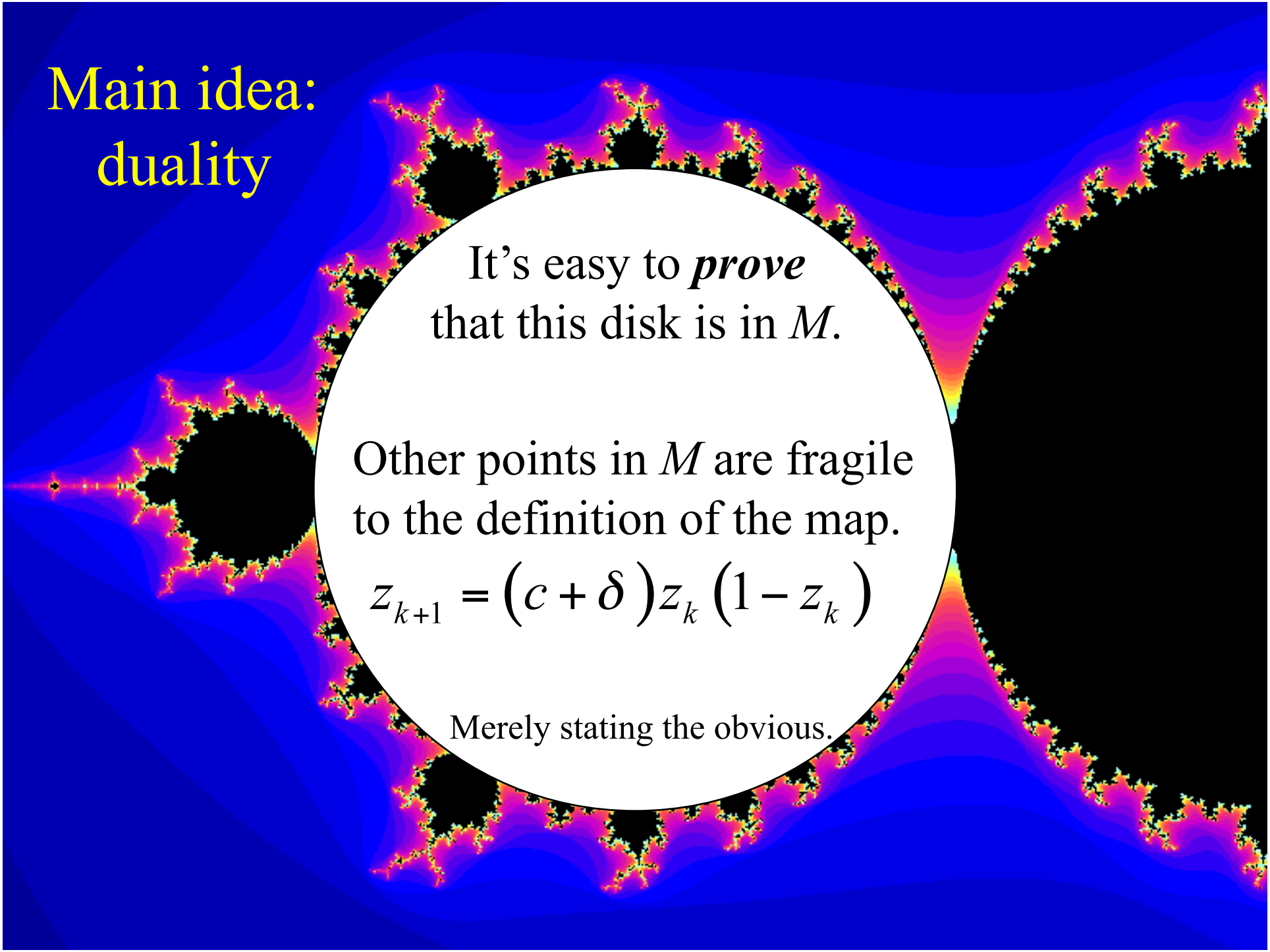
On systems and their components

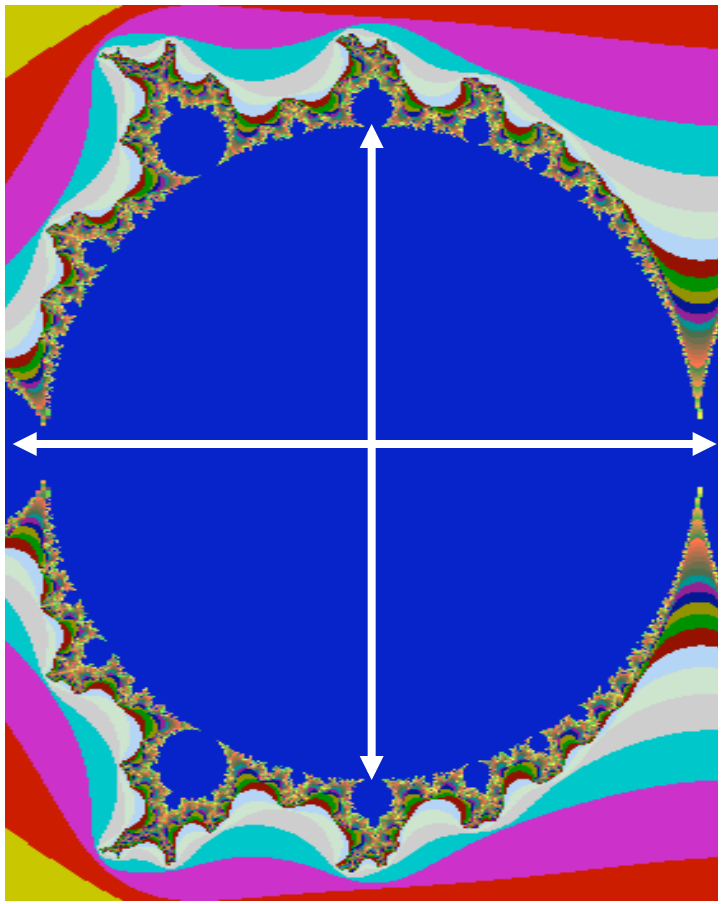
- Thermodynamics (Carnot)
- Communications (Shannon)
- Control (Bode)
- Computation (Turing/Gödel)

Robust/
fragile
is unifying
concept

- Include dynamics and feedback
- Extend to networks
- **New unifications are encouraging**


"Chaocritical" complexity
• Simple question
• Undecidable
• Chaos
• Fractals
Mandelbrot


Main idea:
duality
It's easy to ***prove***
that this disk is in *M*.
Other points in *M* are fragile
to the definition of the map.
$z_{k+1} = (c + \delta) z_k (1 - z_k)$
Merely stating the obvious.



c plane

Short proof

$$z_{k+1} = cz_k\left(1 - z_k\right)$$

$$V\left(z\right) = \left|z\right|^2$$

$$V\left(z_k\right) \geq V\left(z_{k+1}\right)$$

$$\Leftrightarrow \left|z_k\right|^2 - \left|cz_k\left(1 - z_k\right)\right|^2 \geq 0$$

$$\Leftrightarrow 1 \geq \left|c\left(1 - z_k\right)\right|$$

$V\left(z\right) = \left|z\right|^2$ decreases

$$\Leftarrow 1 \geq \left|c\right|\left(1 + \left|z\right|\right)$$

Sufficient condition

Proof method (general)

$$z_{k+1} = cz_k\left(1 - z_k\right)$$

1. Reduce (undecidable) problem in hybrid dynamical systems to
2. (NP-hard) problem in ***real*** semi-algebraic sets
3. Prove emptiness of algebraic problem using

$$V\left(z\right) = \left|z\right|^2$$

$$V\left(z_k\right) \geq V\left(z_{k+1}\right)$$

$$\Leftrightarrow \left|z_k\right|^2 - \left|cz_k\left(1 - z_k\right)\right|^2 \geq 0$$

$$\Leftrightarrow 1 \geq \left|c\left(1 - z_k\right)\right|$$

- Systematic (P) relaxations
- Positivstellinsatz (Psatz)
- Sum of Squares (SOS)

$$V\left(z\right) = \left|z\right|^2 \text{ decreases}$$

$$\Leftarrow 1 \geq \left|c\right|\left(1 + \left|z\right|\right)$$

CDS-SOSTOOLS

Proof method

1. Reduce (undecidable) problem in hybrid dynamical systems to
2. (NP-hard) problem in ***real*** semi-algebraic sets
3. Prove emptiness of algebraic problem using

- Systematic (P) relaxations
- Positivstellinsatz (Psatz)
- Sum of Squares (SOS)

CDS-SOSTOOLS

1. Reduce (undecidable) problem in discrete dynamical systems to
2. (NP-hard) problem in ***boolean*** algebra (3SAT)
3. Prove emptiness of algebraic problem using

- SAT solvers

Model Checking

Unification?

1. Reduce (undecidable) problem in ***hybrid*** dynamical systems to
2. (NP-hard) problem in ***mixed*** boolean and real semi-algebraic sets
3. Prove emptiness of algebraic problem using

- ?????

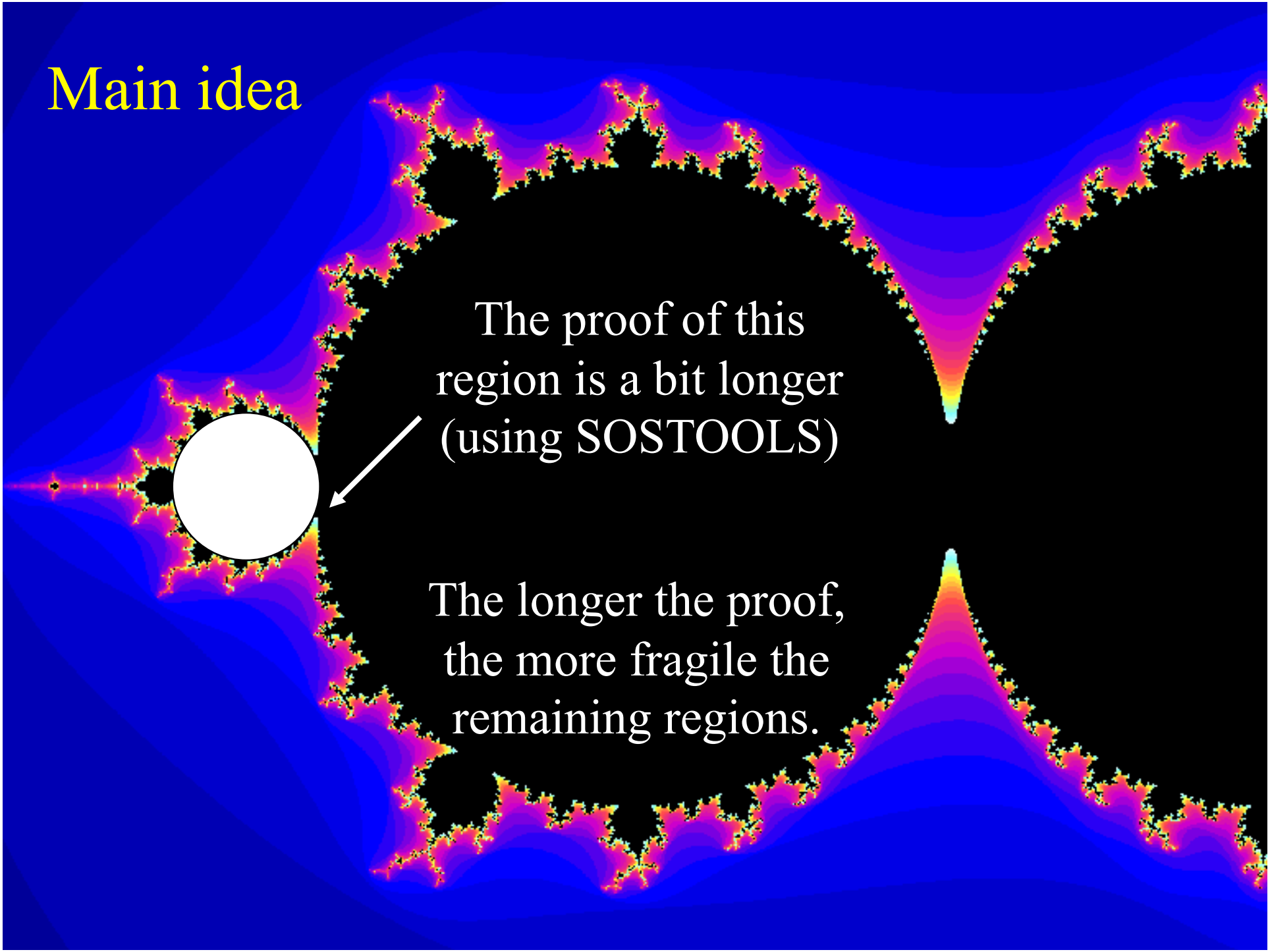
CDS-SOSTOOLS

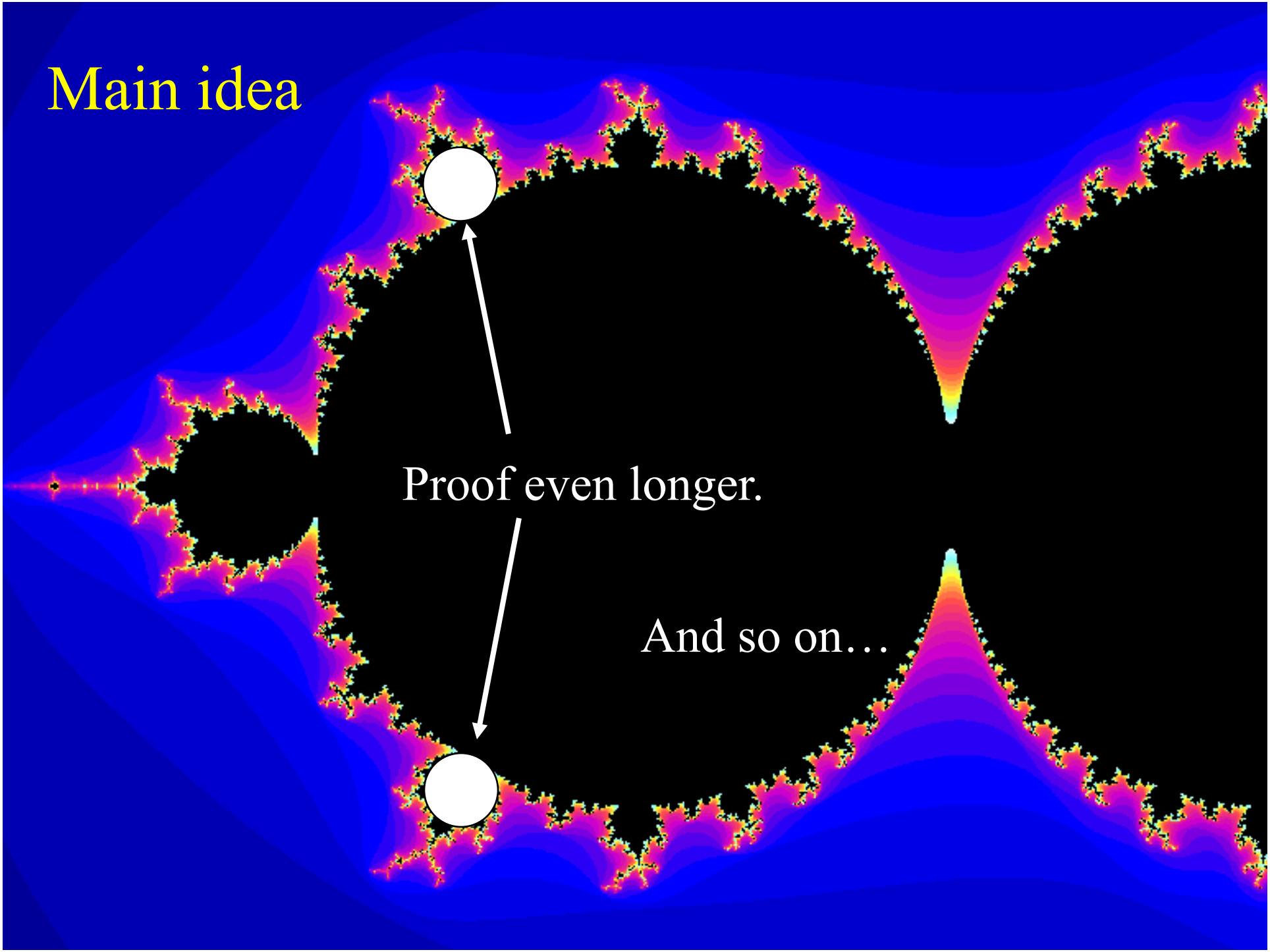
1. Reduce (undecidable) problem in discrete dynamical systems to
2. (NP-hard) problem in ***boolean*** algebra (3SAT)
3. Prove emptiness of algebraic problem using

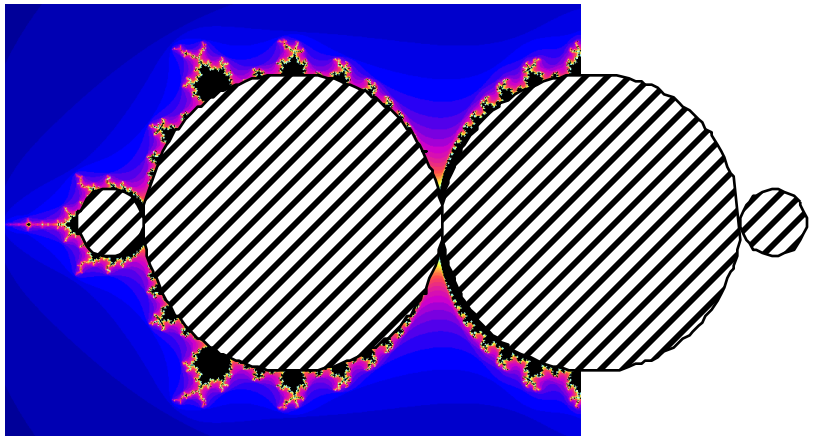
- SAT solvers

Model Checking

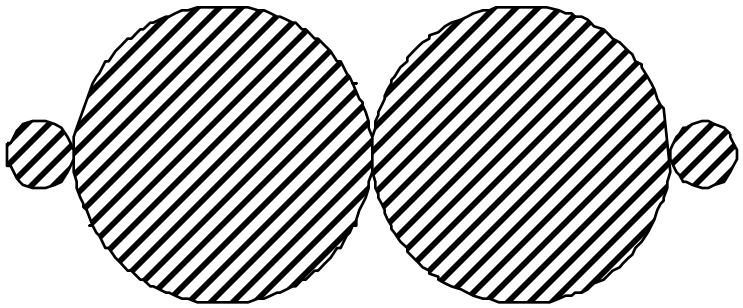
Main idea


The proof of this region is a bit longer (using SOSTOOLS)
The longer the proof, the more fragile the remaining regions.

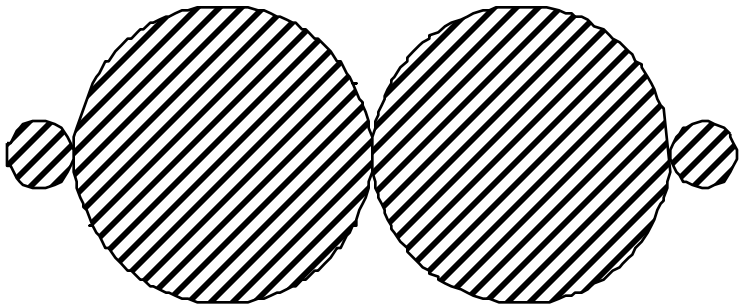

Main idea
Proof even longer.
And so on…



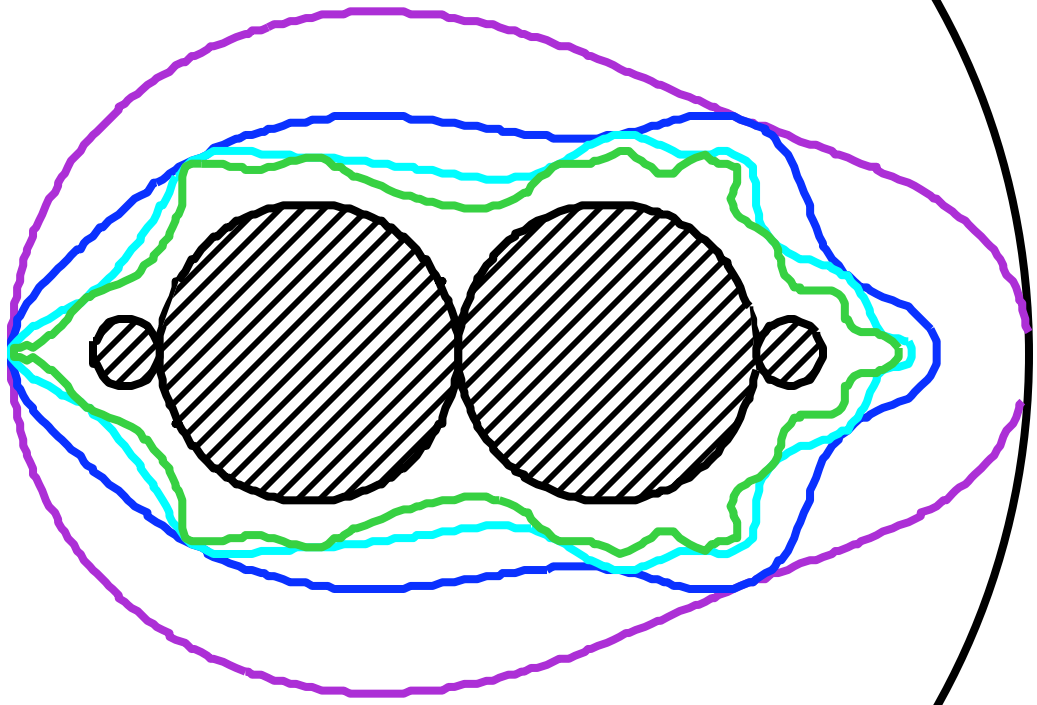
Easy to prove these points are in *Mset*.



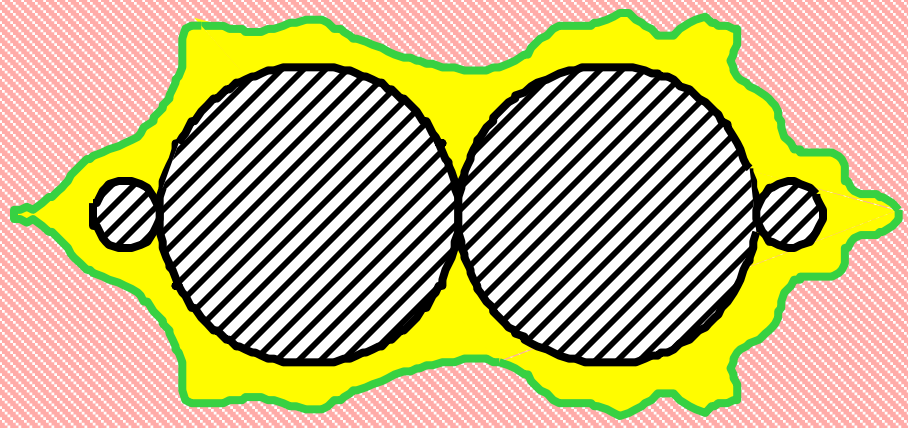
Easy to prove these points are not in *Mset*.



Proofs get harder.
(But all still “easy.”)

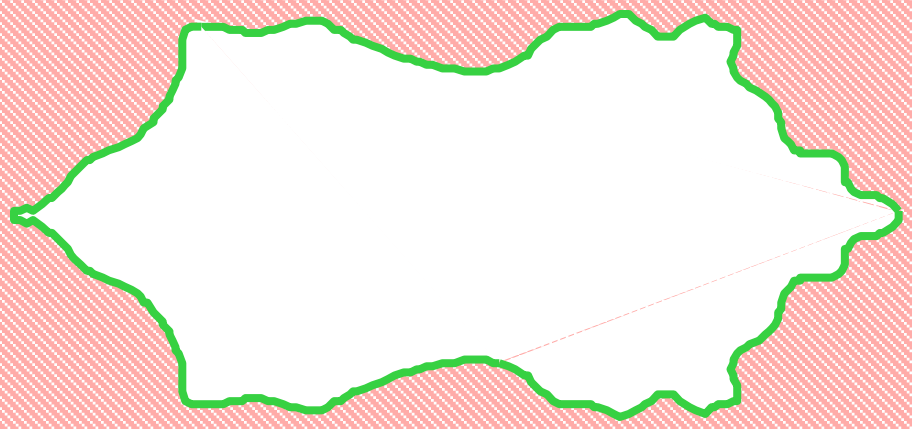


What’s left gets
more fragile.



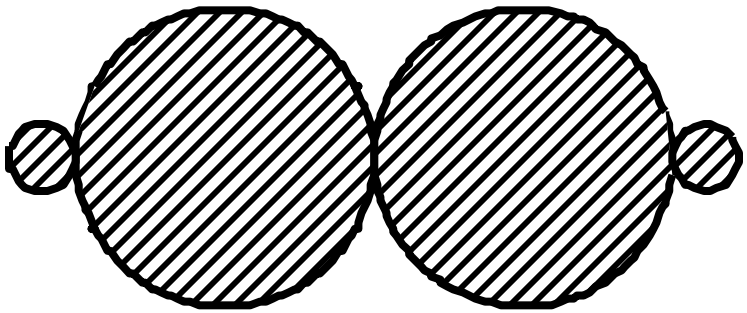
What's left gets
more fragile.

This is robustly and provably ***not*** in ***M.***



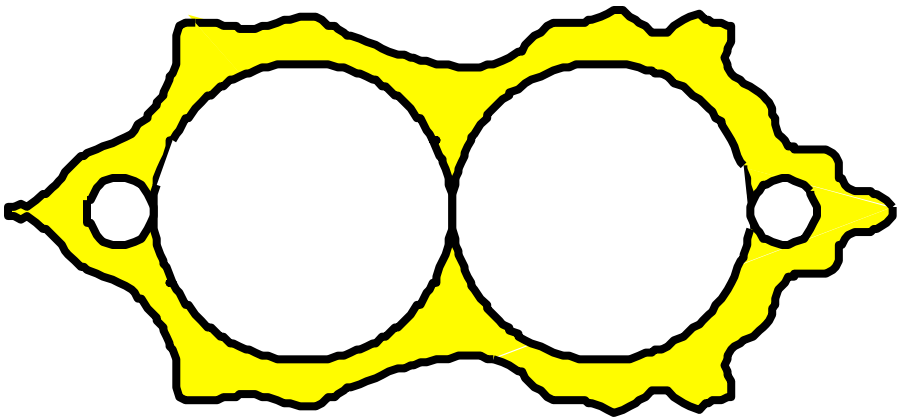
Using SOSTOOLS

This is robustly and provably in ***M.***

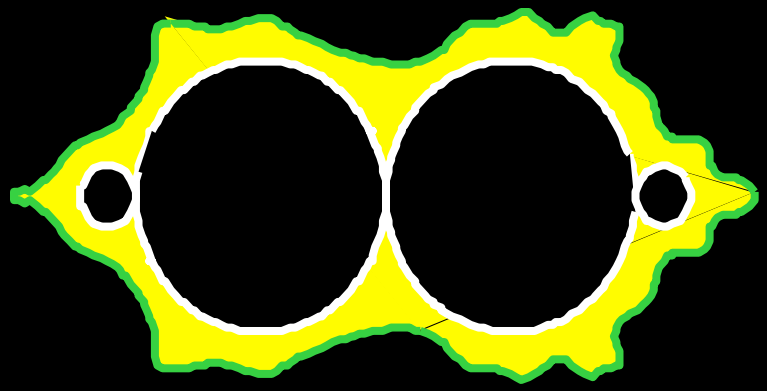


Also using SOSTOOLS

What's left is fragile.

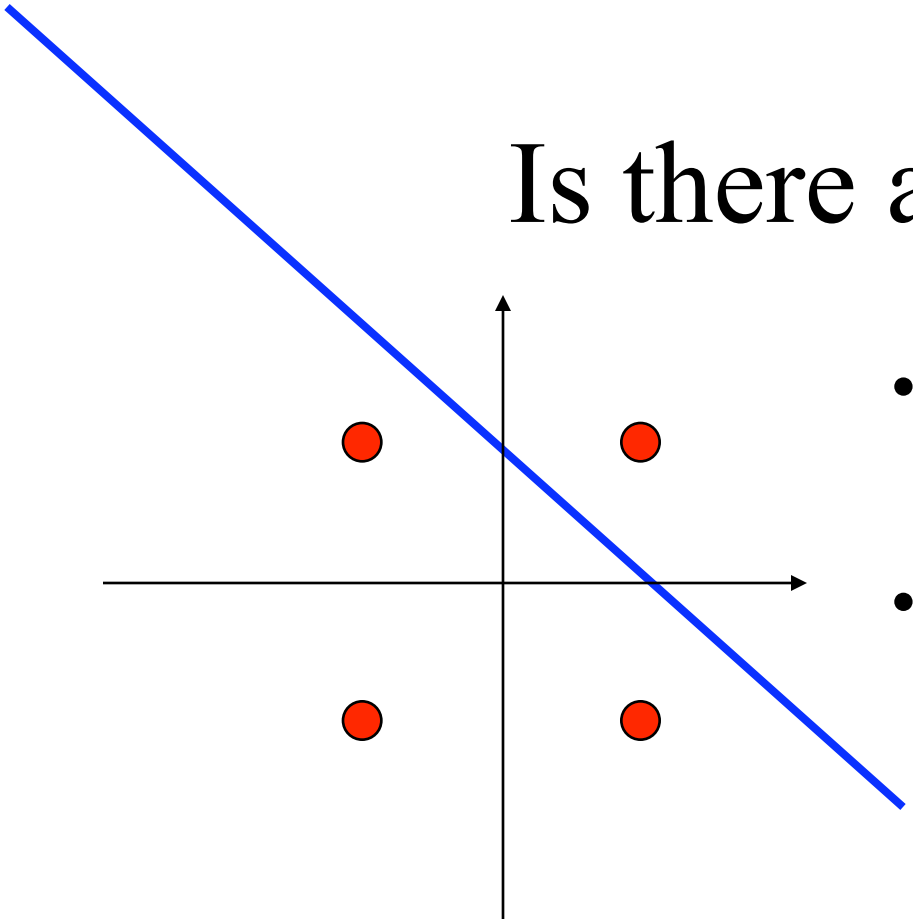


Conjecture: Proof complexity implies problem fragility holds in some greater generality



- Long proofs indicate a fragility, which is either
  - a true fragility (a useful answer) or
  - an artifact of the model (which must then be rectified)
- Brings back together two research areas that have been separated for decades:
  - Numerical analysis and ill-conditioning
  - Computational complexity (P, NP/coNP, undecidable)
- How general?

Is there a crash?

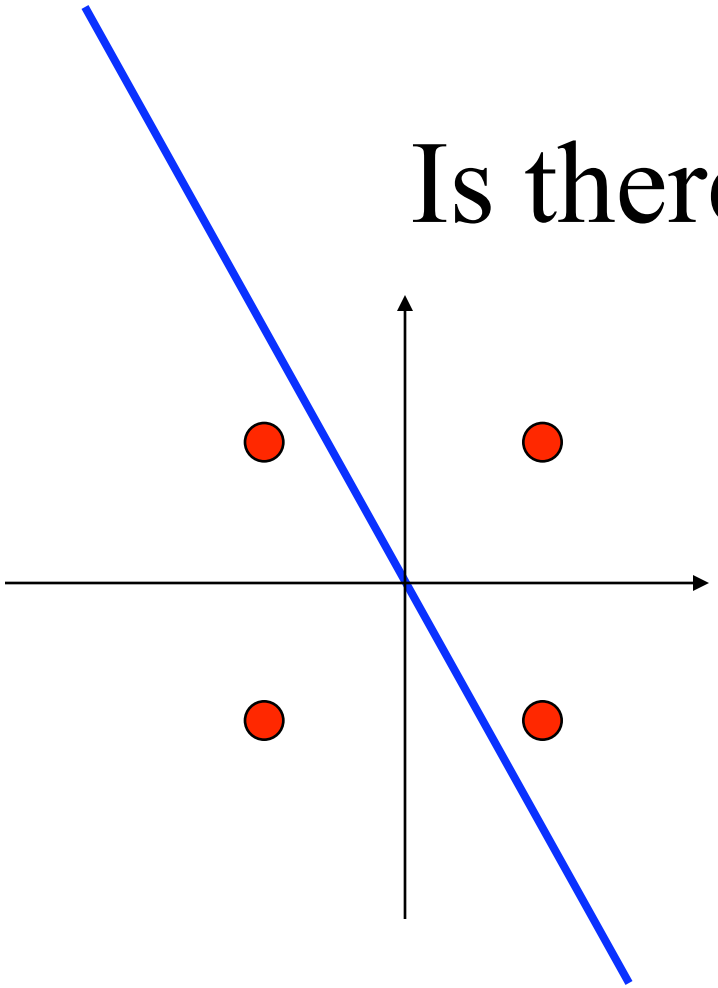


- Given: a line in the plane

  $a_0 + a_1x + a_2y = 0$
- Question: does it hit a corner of the square?

  $x^2=1$, $y^2=1$

Is there a crash?



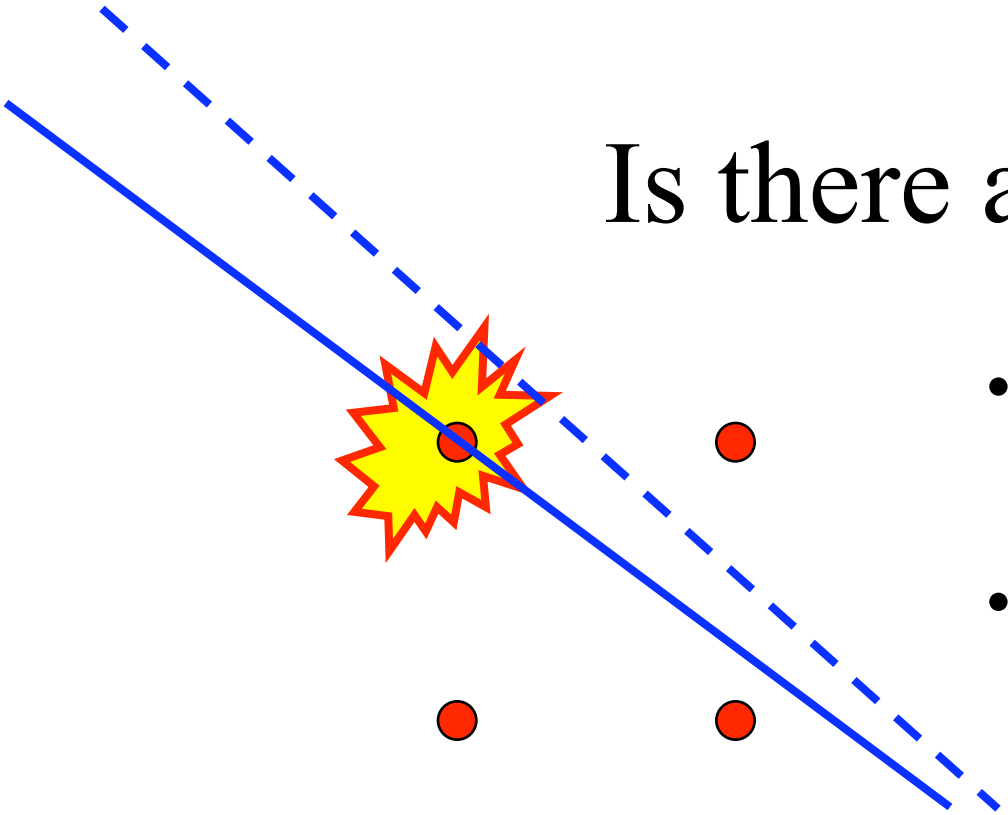
- Given: a line in the plane

  $a_0x + a_1y = 0$

- Question: does it hit a corner of the square?

  $x^2=1$, $y^2=1$

Is there a crash?

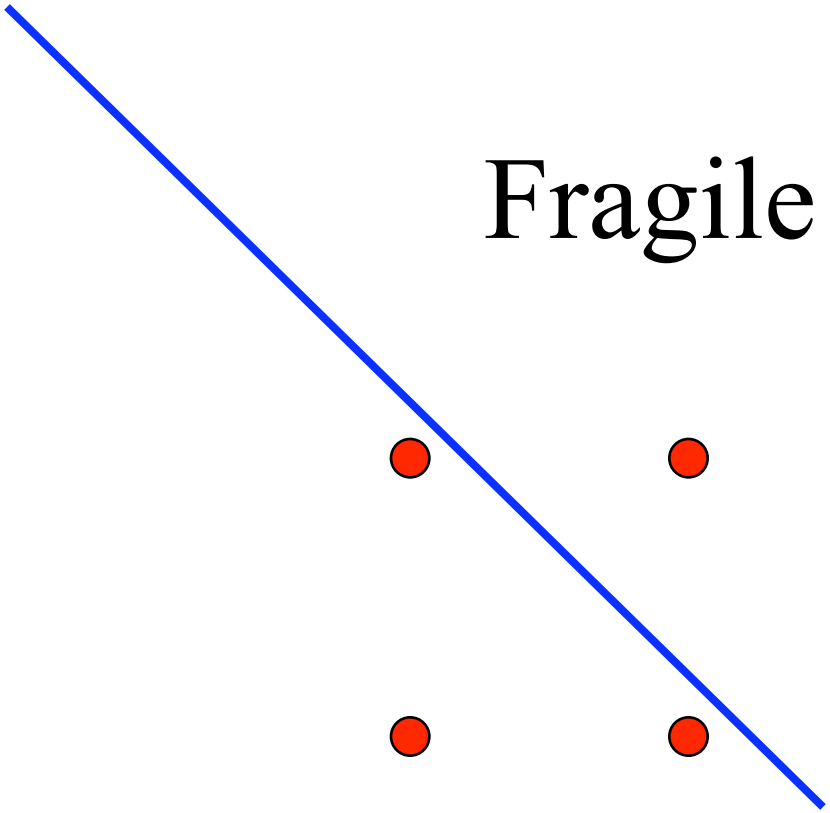


- Given: a line in the plane

  $a_0 + a_1x + a_2y = 0$
- Question: does it hit a corner of the square?

  $x^2=1,\ y^2=1$
- Crash = hits a corner

Fragile = near miss

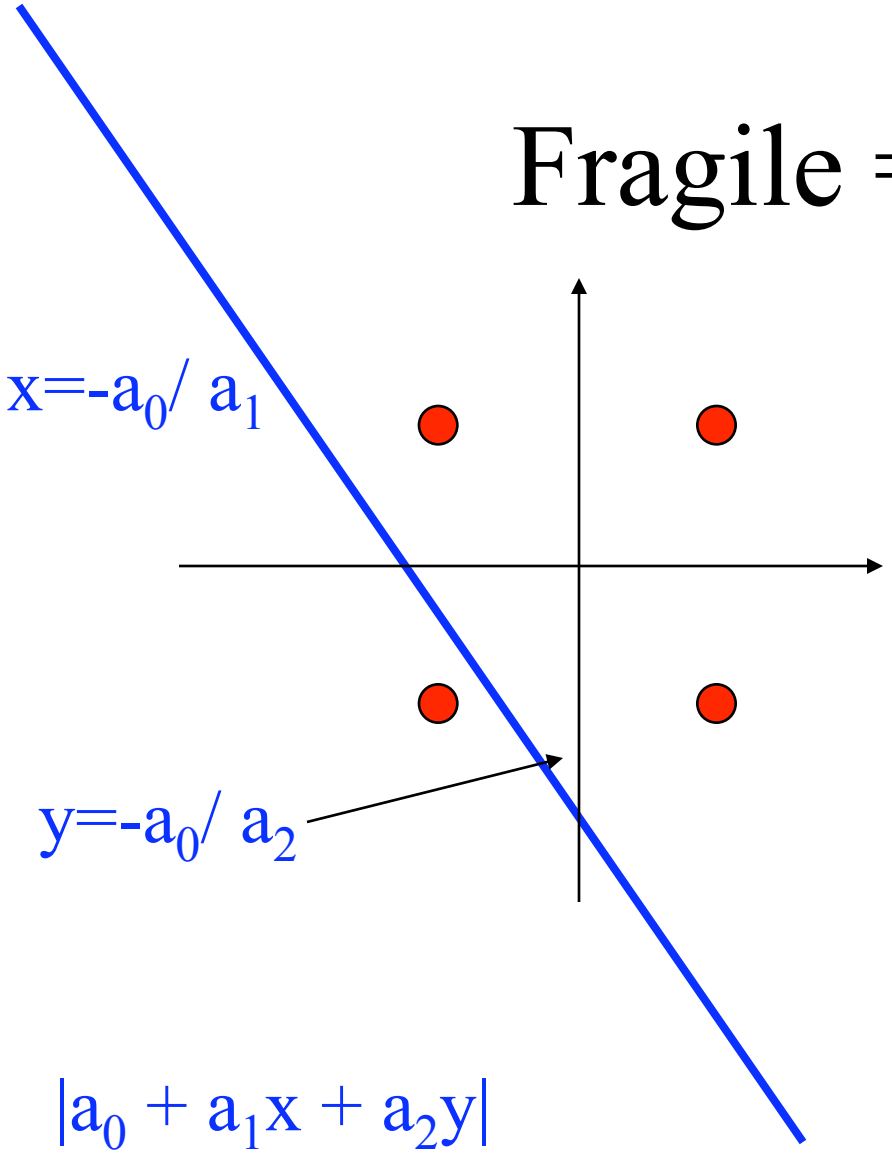


- Given: a line in the plane

  $a_0 + a_1x + a_2y = 0$
- Question: does it hit a corner of the square?

  $x^2=1$, $y^2=1$
- Crash = hits a corner
- Fragile = near miss

Fragile = near miss



$|a_0 + a_1x + a_2y|$

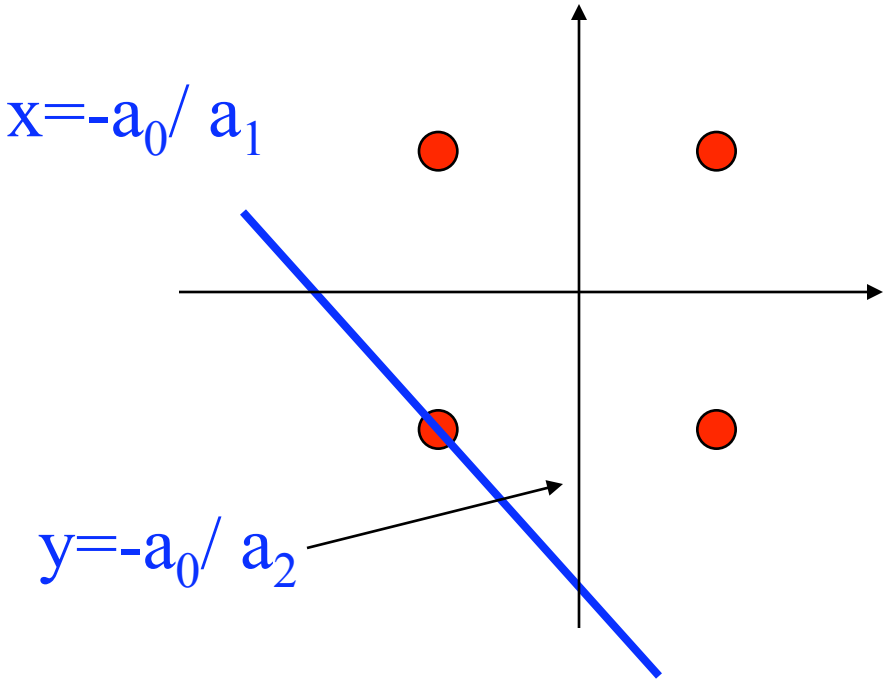
$(-a_0 - a_1x)/\ a_2=y$

- Given: a line in the plane

  $a_0 + a_1x + a_2y = 0$
- Question: does it hit a corner of the square?

  $x^2=1,\ y^2=1$
- Crash = hits a corner
- Fragile = near miss

(1/2,1/4,1/4)



- Given: a line in the plane

  $a_0 + a_1x + a_2y = 0$
- Question: does it hit a corner of the square?

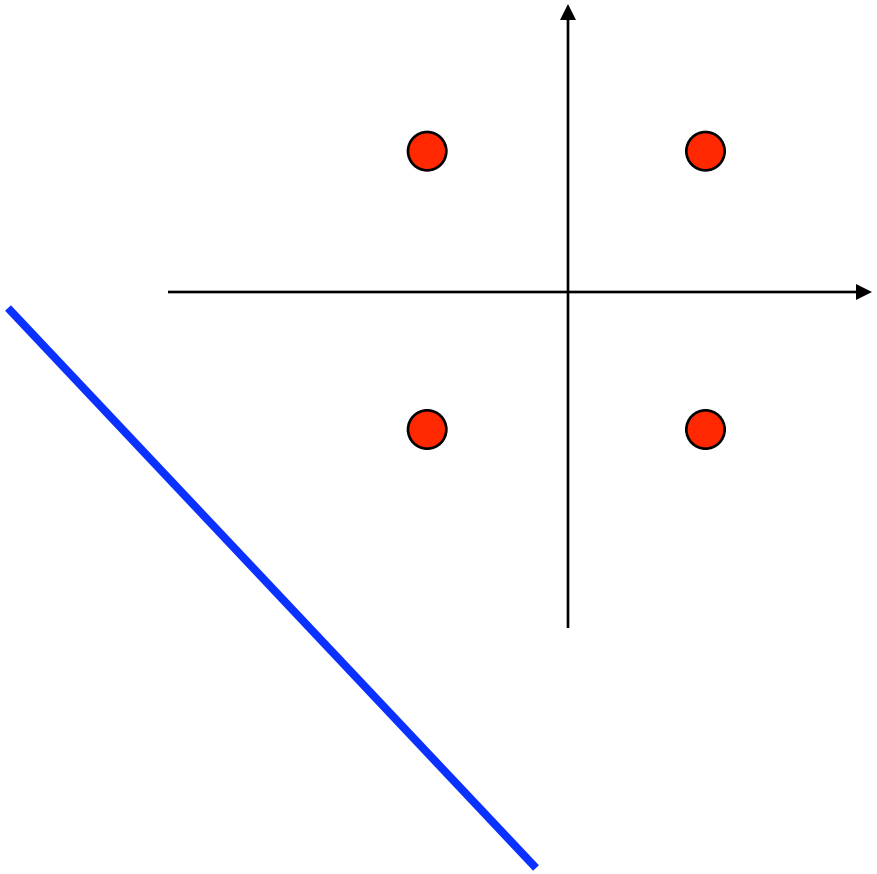
  $x^2=1,\ y^2=1$
- Crash = hits a corner
- Fragile = near miss

$|a_0 + a_1x + a_2y|$

$(-a_0 - a_1x)/a_2=y$

$x=-a_0/\ a_1$

(2/3,1/6,1/6)



- Given: a line in the plane

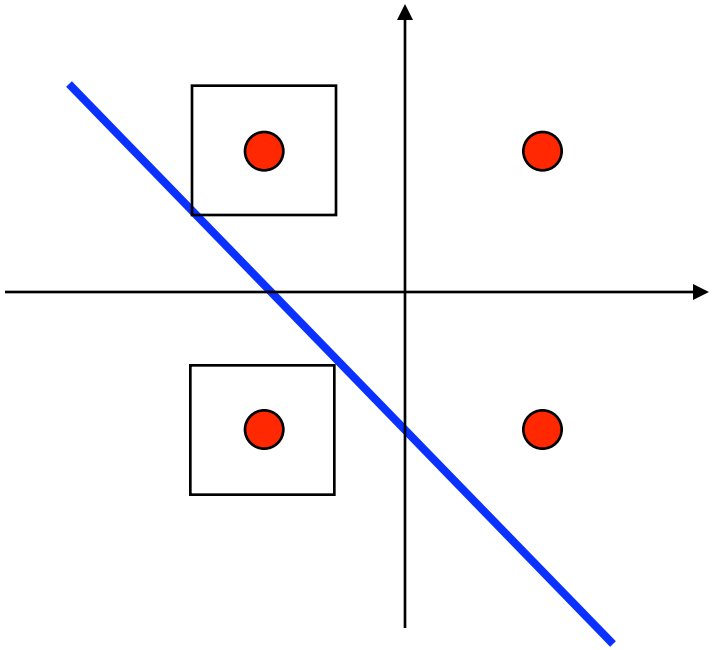
  $a_0 + a_1x + a_2y = 0$
- Question: does it hit a corner of the square?

  $x^2=1,\ y^2=1$
- Crash = hits a corner
- Fragile = near miss

$y=-a_0/\ a_2$

$x=-a_0/\ a_1$

(1/3,1/3,1/3)



$$R_3 = \left\{ \min \delta \;\middle|\; \min_{\|x_i|-1|\le\delta} \left| \sum a_i x_i \right| = 0 \right\}$$

- Given: a line in the plane

  $a_0 + a_1 x + a_2 y = 0$
- Question: does it hit a corner of the square?

  $x^2=1,\ y^2=1$
- Crash = hits a corner
- Fragile = near miss

$y=-a_0/\ a_2$

The “simplest” hard problem

NPP
(Number
partitioning
problem)

Given $a_1 \geq a_2 \geq \cdots \geq a_n \geq 0$

Compute

$$\min_{x_i^2=1} \left| \sum a_i x_i \right|$$

$$= \min_{x_i=\pm 1} \left| \sum a_i x_i \right|$$

$$= \min_{\pm} \left| a_1 \pm a_2 \pm \cdots \pm a_n \right|$$

A “classic” NP complete problem

Karmakar – Karp heuristics:

$$a_1 \geq a_2 \geq \ldots \geq a_{n-1} \geq a_n$$

If

$$a_1 \geq a_2 + \ldots + a_{n-1} + a_n$$

then the optimal solution is

$$a_1 - a_2 - \ldots - a_{n-1} - a_n$$

Can also be derived using SOS/SDP.

Toy problem: $R = \min_{x_i^2=1} \left| \sum a_i x_i \right|$

Assume $\sum a_i = 1$

In general: $R = \min_{x \in X} M(x)$

M = model of system

X = uncertainty set

R = robustness

Central computational problems in biology and advanced technologies can be written this way and are formally "hard" (NP/coNP hard or undecidable)

Various levels of paranoia

$$R = \min_{x_i^2 = 1} \left| \sum a_i x_i \right|$$

Explicitly modeled uncertainty

$$R_2 = \left\{ \min \sum |a_i - b_i| \;\middle|\; \min_{x_i^2 = 1} \left| \sum b_i x_i \right| = 0 \right\}$$

$$R_3 = \left\{ \min \delta \;\middle|\; \min_{\left||x_i| - 1\right| \le \delta} \left| \sum a_i x_i \right| = 0 \right\}$$

Uncertainty in parameters / model

Theorem: $R = R_2 = R_3$

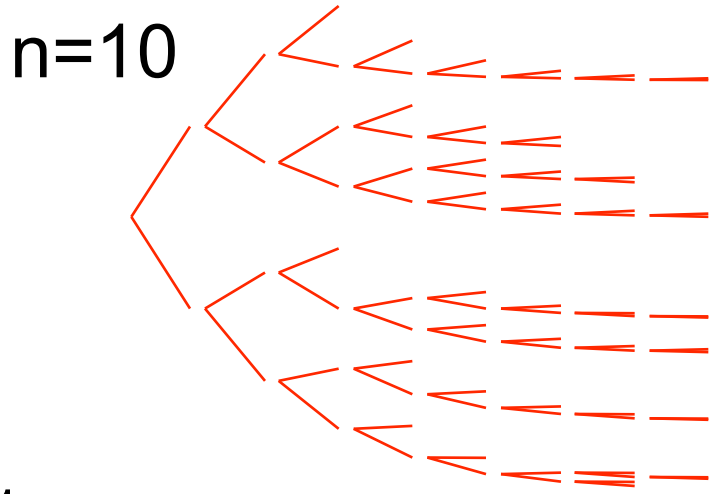
Features of NPP

1. All notions of robustness are equivalent
2. Robust problems are easy

- Both cannot hold in general
- Rather give up 1 than 2

This is the “punchline”

Robust problems are easy

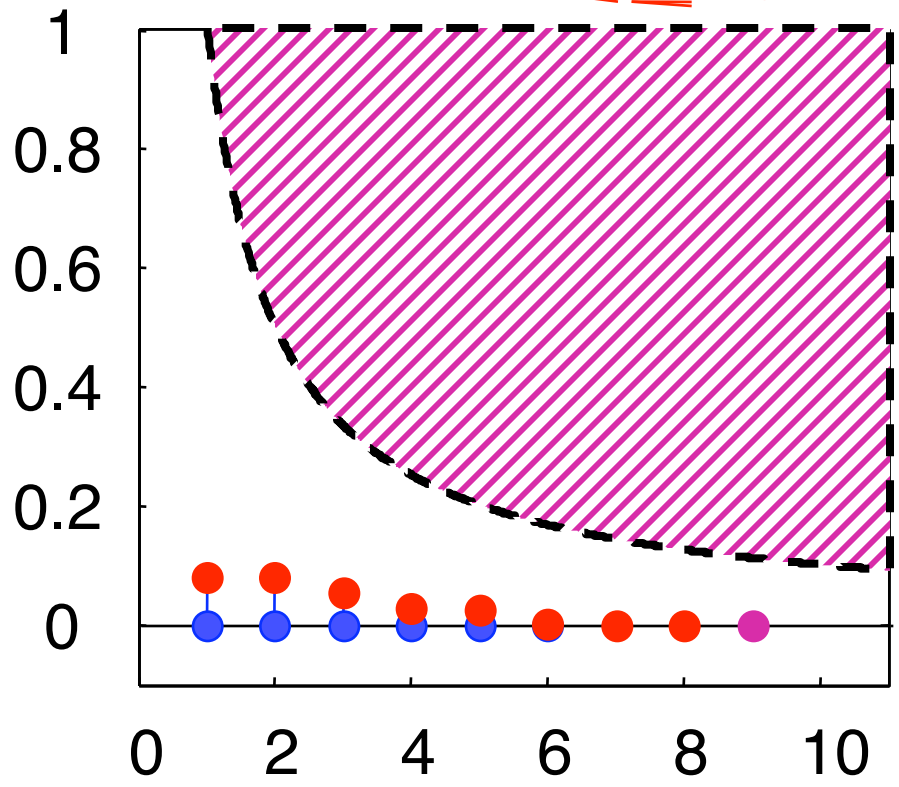


Theorem: $L \le \frac{1}{R}$

Robustness

$$R = \min_{x_i^2=1} \left| \sum a_i x_i \right|$$

$$L \le \frac{1}{R} = "Fragility"$$



"Complexity" L = tree depth (length)

Let's compare

NPP

$$\left.\begin{array}{l} L = \text{ tree depth} \\ \# = \text{operation count} \end{array}\right\} \Rightarrow \quad \# \le n 2^{L}$$

$$L \le \frac{1}{R} \quad \Rightarrow \quad \# \le O\left(n 2^{\frac{1}{R}} \right)$$

$$\begin{array}{l} \text{Linear} \\ \text{Program} \end{array} \Rightarrow \quad \# \le O\left(n^{2} \log\left(\frac{1}{R} \right) \right)$$

Why is NPP "harder"?

Random NPP instances (e.g. physics)

$$\min_{x_i^2=1} \left| \sum a_i x_i \right| \approx \frac{2^{-n}}{\sqrt{n}}$$

This is true "almost surely," but has proof length of $\# = O\left(2^n\right)$

Also holds in the worst case (e.g. CS).

Robust NPP instances are easy!

$$\# \leq O\left(n 2^{\frac{1}{R}} \right)$$

R big.

Rethinking "complexity"

Random & worst case

$$\# = O\left(2^n\right)$$

Ill-conditioning is less "fundamental"?

$$\text{Linear Program} \Rightarrow \quad \# \le O\left(n^2 \log\left(\frac{1}{R}\right)\right)$$

Maybe not.

$$\# \le O\left(n 2^{\frac{1}{R}}\right)$$

Fragility Approx-imation	Hard	2-SAT & Lin Eq (mod 2)	3-SAT
	Easy	Lattices/LP	NPP
		Easy	Hard
	Decision Problem		

By Andy Lamperski

Landscape(by Andy Lamperski)

Fragility Approx-imation	Hard	2-SAT	3-SAT
	Easy	Lattices/LP	NPP
		Easy	Hard
	Decision Problem		

- If a decision problem is hard, and robust instances are hard to distinguish from fragile instances, no algorithm for the decision problem runs quickly on the robust instances.
- Approximating fragility can be easy even if the decision problem is hard, and vice versa.

Number Partitioning

- Given non-negative integers $a_1 \geq a_2 \geq \ldots \geq a_n \geq 0$ with $\Sigma_i a_i = a$
- Decision problem: Are there x_i in {-1,1} such that $\Sigma_i a_i x_i = 0$
- NP-Complete
- Robustness: $R=(\min \{\Sigma_i a_i x_i : x_i^2 = 1\})$
- Approximating Robustness: Given any $\varepsilon>0$, there is a $\text{poly}(1/\varepsilon, n \log(a_1))$ algorithm returning R' such that $R \leq R' \leq (1+\varepsilon)R + \varepsilon a$

3-SAT

$$(x \vee y \vee \neg z) \wedge (\neg y \vee z) \wedge z$$

- Literal: a Boolean variable or its negation
- Clause: A disjunction of literals
- Decision problem: Given a conjunction of clauses with at most three literals, decide if there is an assignment to the Boolean variables that satisfies the formula
- NP-Complete

3-SAT

$$x \wedge \neg x \wedge \neg y \wedge \neg z \wedge (y \vee z)$$

- Robustness: #(of literals flipped to make the formula satisfiable)

  The formula above has robustness 2
- Fragility: #(of clauses)-robustness
- Computing the fragility corresponds to solving MAX-3-SAT

3-SAT

- Fix any $\varepsilon > 0$. Given a formula φ with m clauses, it is NP hard to distinguish between the following two cases:
  - φ satisfiable
  - Robustness $\geq (\frac{1}{8} - \varepsilon)m$
- Implies robustness and fragility are hard to approximate
- (Unless P=NP, no algorithm deciding 3-SAT can have polynomial running time when restricted to formulas with robustness at least $(\frac{1}{8}-\varepsilon)m$.)

2-SAT

- Fix any $\varepsilon > 0$. Given a formula φ with m clauses, it is NP hard to distinguish between the following two cases:
  - Robustness $\leq (1/12+\varepsilon)m$
  - Robustness $\geq (1/8-\varepsilon)m$
- Implies robustness and fragility are hard to approximate

Linear Equations (mod 2)

- Robustness: #(of equations perturbed to make the system feasible)

  e.g. $x_1+x_2=0$, $x_1+x_2=1$, $x_3=0$, $x_3=1$

  has robustness 2

- Calculating fragility corresponds to finding a vector solving the most equations (MAX-LIN-Mod-2)

Linear Equations (mod 2)

- Fix any $\varepsilon > 0$. Given a system $Ax=b$ with m equations, it is NP hard to distinguish between the following two cases:
  - Robustness $\geq (\frac{1}{2}-\varepsilon)m$
  - Robustness $\leq \varepsilon m$
- Implies robustness and fragility are hard to approximate
- Unless P=NP, no polynomial time algorithm can distinguish between very robust formulas and very fragile formulas.

Landscape

Fragility Approx-imation	Hard	2-SAT	3-SAT
	Easy	Lattices/LP	NPP
		Easy	Hard
	Decision Problem		

- If a decision problem is hard, and robust instances are hard to distinguish from fragile instances, no algorithm for the decision problem runs quickly on the robust instances.
- Approximating fragility can be easy even if the decision problem is hard, and vice versa.

Features in general

1. All notions of robustness are equivalent
2. Robust problems are easy

- Both cannot hold in general
- Rather give up 1 than 2

- **Need "physical" notions of robustness**