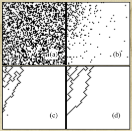
System Robustness

Principles of Complex Systems
CSYS/MATH 300, Spring, 2013 | #SpringPoCS2013

Prof. Peter Dodds
@peterdodds

Department of Mathematics & Statistics | Center for Complex Systems | Vermont Advanced Computing Center | University of Vermont

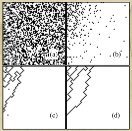


Outline

Robustness

- HOT theory
- Self-Organized Criticality
- COLD theory
- Network robustness

References



Outline

Robustness

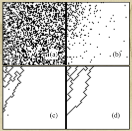
HOT theory

Self-Organized Criticality

COLD theory

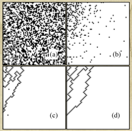
Network robustness

References



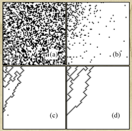
Robustness

- Many complex systems are prone to cascading catastrophic failure:
  - Blackouts
  - Disease outbreaks
  - Wildfires
  - Earthquakes
- But complex systems also show persistent robustness
- Robustness and Failure may be a power-law story...



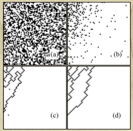
Robustness

- Many complex systems are prone to cascading catastrophic failure:
  - Blackouts
  - Disease outbreaks
  - Wildfires
  - Earthquakes
- But complex systems also show persistent robustness
- Robustness and Failure may be a power-law story...



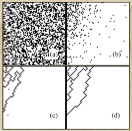
Robustness

- Many complex systems are prone to cascading catastrophic failure:
  - Blackouts
  - Disease outbreaks
  - Wildfires
  - Earthquakes
- But complex systems also show persistent robustness
- Robustness and Failure may be a power-law story...



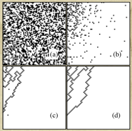
Robustness

- Many complex systems are prone to cascading catastrophic failure:
    - Blackouts
    - Disease outbreaks
    - Wildfires
    - Earthquakes
- But complex systems also show persistent robustness
- Robustness and Failure may be a power-law story...



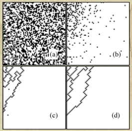
Robustness

- Many complex systems are prone to cascading catastrophic failure:
  - Blackouts
  - Disease outbreaks
  - Wildfires
  - Earthquakes
- But complex systems also show persistent robustness
- Robustness and Failure may be a power-law story...



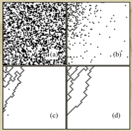
Robustness

- Many complex systems are prone to cascading catastrophic failure: exciting!!!
  - Blackouts
  - Disease outbreaks
  - Wildfires
  - Earthquakes
- But complex systems also show persistent robustness
- Robustness and Failure may be a power-law story...



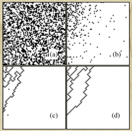
Robustness

- Many complex systems are prone to cascading catastrophic failure: exciting!!!
  - Blackouts
  - Disease outbreaks
  - Wildfires
  - Earthquakes
- But complex systems also show persistent robustness
- Robustness and Failure may be a power-law story...



Robustness

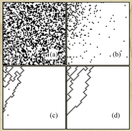
- Many complex systems are prone to cascading catastrophic failure: exciting!!!
  - Blackouts
  - Disease outbreaks
  - Wildfires
  - Earthquakes
- But complex systems also show persistent robustness (not as exciting but important...)
- Robustness and Failure may be a power-law story...



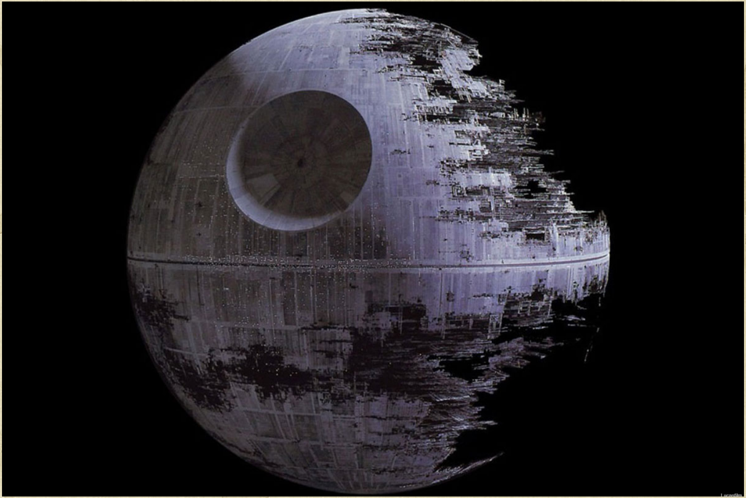
The UNIVERSITY of VERMONT

Robustness

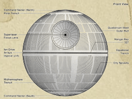
- Many complex systems are prone to cascading catastrophic failure: exciting!!!
  - Blackouts
  - Disease outbreaks
  - Wildfires
  - Earthquakes
- But complex systems also show persistent robustness (not as exciting but important...)
- Robustness and Failure may be a power-law story...



Our emblem of Robust-Yet-Fragile:

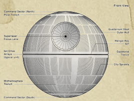


“That’s no moon ...”



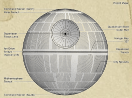
Robustness

- System robustness may result from
  1. Evolutionary processes
  2. Engineering/Design
- Idea: Explore systems optimized to perform under uncertain conditions.
- The handle:
  'Highly Optimized Tolerance' (HOT) [4, 5, 6, 10]
- The catchphrase: Robust yet Fragile
- The people: Jean Carlson and John Doyle (⊞)
- Great abstracts of the world #73: "There aren't any." [7]



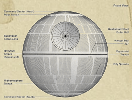
Robustness

- System robustness may result from
  1. Evolutionary processes
  2. Engineering/Design
- Idea: Explore systems optimized to perform under uncertain conditions.
- The handle:
  'Highly Optimized Tolerance' (HOT) [4, 5, 6, 10]
- The catchphrase: Robust yet Fragile
- The people: Jean Carlson and John Doyle (⊞)
- Great abstracts of the world #73: "There aren't any." [7]



Robustness

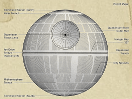
- System robustness may result from
  1. Evolutionary processes
  2. Engineering/Design
- Idea: Explore systems optimized to perform under uncertain conditions.
- The handle:
  'Highly Optimized Tolerance' (HOT) [4, 5, 6, 10]
- The catchphrase: Robust yet Fragile
- The people: Jean Carlson and John Doyle (⊞)
- Great abstracts of the world #73: "There aren't any." [7]



The UNIVERSITY of VERMONT

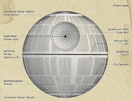
Robustness

- System robustness may result from
  1. Evolutionary processes
  2. Engineering/Design
- Idea: Explore systems optimized to perform under uncertain conditions.
- The handle:
  'Highly Optimized Tolerance' (HOT) [4, 5, 6, 10]
- The catchphrase: Robust yet Fragile
- The people: Jean Carlson and John Doyle (⊞)
- Great abstracts of the world #73: "There aren't any." [7]



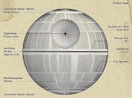
Robustness

- System robustness may result from
  1. Evolutionary processes
  2. Engineering/Design
- Idea: Explore systems optimized to perform under uncertain conditions.
- The handle:
  'Highly Optimized Tolerance' (HOT) [4, 5, 6, 10]
- The catchphrase: Robust yet Fragile
- The people: Jean Carlson and John Doyle (⊞)
- Great abstracts of the world #73: "There aren't any." [7]



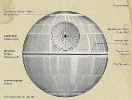
Robustness

- System robustness may result from
  1. Evolutionary processes
  2. Engineering/Design
- Idea: Explore systems optimized to perform under uncertain conditions.
- The handle:
  'Highly Optimized Tolerance' (HOT) [4, 5, 6, 10]
- The catchphrase: Robust yet Fragile
- The people: Jean Carlson and John Doyle (⊞)
- Great abstracts of the world #73: "There aren't any." [7]



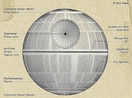
Robustness

- System robustness may result from
  1. Evolutionary processes
  2. Engineering/Design
- Idea: Explore systems optimized to perform under uncertain conditions.
- The handle:
  'Highly Optimized Tolerance' (HOT) [4, 5, 6, 10]
- The catchphrase: Robust yet Fragile
- The people: Jean Carlson and John Doyle (⊞)
- Great abstracts of the world #73: "There aren't any." [7]



Robustness

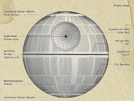
- System robustness may result from
  1. Evolutionary processes
  2. Engineering/Design
- Idea: Explore systems optimized to perform under uncertain conditions.
- The handle:
  'Highly Optimized Tolerance' (HOT) [4, 5, 6, 10]
- The catchphrase: Robust yet Fragile
- The people: Jean Carlson and John Doyle (⊞)
- Great abstracts of the world #73: "There aren't any." [7]



Robustness

Features of HOT systems: [5, 6]

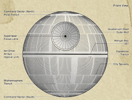
- High performance and robustness
- Designed/evolved to handle known stochastic environmental variability
- Fragile in the face of unpredicted environmental signals
- Highly specialized, low entropy configurations
- Power-law distributions appear (of course...)



Robustness

Features of HOT systems: [5, 6]

- High performance and robustness
- Designed/evolved to handle known stochastic environmental variability
- Fragile in the face of unpredicted environmental signals
- Highly specialized, low entropy configurations
- Power-law distributions appear (of course...)

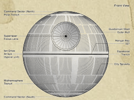


The UNIVERSITY of VERMONT

Robustness

Features of HOT systems: [5, 6]

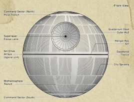
- High performance and robustness
- Designed/evolved to handle known stochastic environmental variability
- Fragile in the face of unpredicted environmental signals
- Highly specialized, low entropy configurations
- Power-law distributions appear (of course...)



Robustness

Features of HOT systems: [5, 6]

- High performance and robustness
- Designed/evolved to handle known stochastic environmental variability
- Fragile in the face of unpredicted environmental signals
- Highly specialized, low entropy configurations
- Power-law distributions appear (of course...)

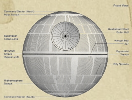


The UNIVERSITY of VERMONT

Robustness

Features of HOT systems: [5, 6]

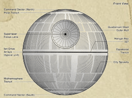
- High performance and robustness
- Designed/evolved to handle known stochastic environmental variability
- Fragile in the face of unpredicted environmental signals
- Highly specialized, low entropy configurations
- Power-law distributions appear (of course...)



Robustness

Features of HOT systems: [5, 6]

- High performance and robustness
- Designed/evolved to handle known stochastic environmental variability
- Fragile in the face of unpredicted environmental signals
- Highly specialized, low entropy configurations
- Power-law distributions appear (of course...)

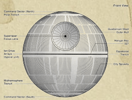


Robustness

HOT combines things we've seen:

- Variable transformation
- Constrained optimization

- Need power law transformation between variables: $(Y = X^{-\alpha})$
- Recall PLIPLO is bad...
- MIWO is good
- X has a characteristic size but Y does not

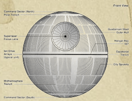


Robustness

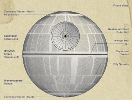
HOT combines things we've seen:

- Variable transformation
- Constrained optimization

- Need power law transformation between variables: $(Y = X^{-\alpha})$
- Recall PLIPLO is bad...
- MIWO is good
- X has a characteristic size but Y does not



Robustness



HOT combines things we've seen:

- Variable transformation
- Constrained optimization

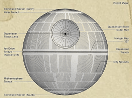
- Need power law transformation between variables: $(Y = X^{-\alpha})$
- Recall PLIPLO is bad...
- MIWO is good
- X has a characteristic size but Y does not

Robustness

HOT combines things we've seen:

- Variable transformation
- Constrained optimization

- Need power law transformation between variables: $(Y = X^{-\alpha})$
- Recall PLIPLO is bad...
- MIWO is good
- X has a characteristic size but Y does not

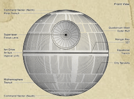


Robustness

HOT combines things we've seen:

- Variable transformation
- Constrained optimization

- Need power law transformation between variables: $(Y = X^{-\alpha})$
- Recall PLIPLO is bad...
- MIWO is good
- X has a characteristic size but Y does not

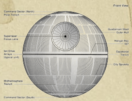


Robustness

HOT combines things we've seen:

- Variable transformation
- Constrained optimization

- Need power law transformation between variables: $(Y = X^{-\alpha})$
- Recall PLIPLO is bad...
- MIWO is good: Mild In, Wild Out
- X has a characteristic size but Y does not

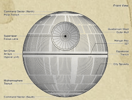


Robustness

HOT combines things we've seen:

- Variable transformation
- Constrained optimization

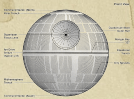
- Need power law transformation between variables: $(Y = X^{-\alpha})$
- Recall PLIPLO is bad...
- MIWO is good: Mild In, Wild Out
- X has a characteristic size but Y does not



Robustness

Forest fire example: [5]

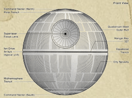
- Square $N \times N$ grid
- Sites contain a tree with probability ρ = density
- Sites are empty with probability $1 - \rho$
- Fires start at location (i, j) according to some distribution P_{ij}
- Fires spread from tree to tree (nearest neighbor only)
- Connected clusters of trees burn completely
- Empty sites block fire
- Best case scenario:
  Build firebreaks to maximize average # trees left intact given one spark



Robustness

Forest fire example: [5]

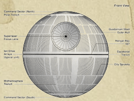
- Square $N \times N$ grid
- Sites contain a tree with probability ρ = density
- Sites are empty with probability $1 - \rho$
- Fires start at location (i, j) according to some distribution P_{ij}
- Fires spread from tree to tree (nearest neighbor only)
- Connected clusters of trees burn completely
- Empty sites block fire
- Best case scenario:
  Build firebreaks to maximize average # trees left intact given one spark



Robustness

Forest fire example: [5]

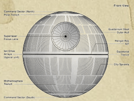
- Square $N \times N$ grid
- Sites contain a tree with probability ρ = density
- Sites are empty with probability $1 - \rho$
- Fires start at location (i, j) according to some distribution P_{ij}
- Fires spread from tree to tree (nearest neighbor only)
- Connected clusters of trees burn completely
- Empty sites block fire
- Best case scenario:
  Build firebreaks to maximize average # trees left intact given one spark



Robustness

Forest fire example: [5]

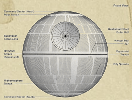
- Square $N \times N$ grid
- Sites contain a tree with probability ρ = density
- Sites are empty with probability $1 - \rho$
- Fires start at location (i, j) according to some distribution P_{ij}
- Fires spread from tree to tree (nearest neighbor only)
- Connected clusters of trees burn completely
- Empty sites block fire
- Best case scenario:
  Build firebreaks to maximize average # trees left intact given one spark



Robustness

Forest fire example: [5]

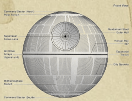
- Square $N \times N$ grid
- Sites contain a tree with probability ρ = density
- Sites are empty with probability $1 - \rho$
- Fires start at location (i, j) according to some distribution P_{ij}
- Fires spread from tree to tree (nearest neighbor only)
- Connected clusters of trees burn completely
- Empty sites block fire
- Best case scenario:
  Build firebreaks to maximize average # trees left intact given one spark



Robustness

Forest fire example: [5]

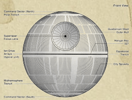
- Square $N \times N$ grid
- Sites contain a tree with probability ρ = density
- Sites are empty with probability $1 - \rho$
- Fires start at location (i, j) according to some distribution P_{ij}
- Fires spread from tree to tree (nearest neighbor only)
- Connected clusters of trees burn completely
- Empty sites block fire
- Best case scenario:
  Build firebreaks to maximize average # trees left intact given one spark



Robustness

Forest fire example: [5]

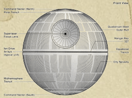
- Square $N \times N$ grid
- Sites contain a tree with probability ρ = density
- Sites are empty with probability $1 - \rho$
- Fires start at location (i, j) according to some distribution P_{ij}
- Fires spread from tree to tree (nearest neighbor only)
- Connected clusters of trees burn completely
- Empty sites block fire
- Best case scenario:
  Build firebreaks to maximize average # trees left intact given one spark



Robustness

Forest fire example: [5]

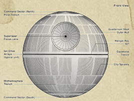
- Square $N \times N$ grid
- Sites contain a tree with probability ρ = density
- Sites are empty with probability $1 - \rho$
- Fires start at location (i, j) according to some distribution P_{ij}
- Fires spread from tree to tree (nearest neighbor only)
- Connected clusters of trees burn completely
- Empty sites block fire
- Best case scenario:
  Build firebreaks to maximize average # trees left intact given one spark



Robustness

Forest fire example: [5]

- Square $N \times N$ grid
- Sites contain a tree with probability ρ = density
- Sites are empty with probability $1 - \rho$
- Fires start at location (i, j) according to some distribution P_{ij}
- Fires spread from tree to tree (nearest neighbor only)
- Connected clusters of trees burn completely
- Empty sites block fire
- Best case scenario:
  Build firebreaks to maximize average # trees left intact given one spark



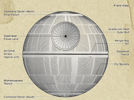
Robustness

Forest fire example: [5]

- Build a forest by adding one tree at a time
- Test D ways of adding one tree
- D = design parameter
- Average over P_{ij} = spark probability
- $D = 1$: random addition
- $D = N^2$: test all possibilities

Measure average area of forest left untouched

- $f(c)$ = distribution of fire sizes c (= cost)
- Yield = $Y = \rho - \langle c \rangle$



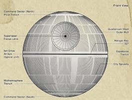
Robustness

Forest fire example: [5]

- Build a forest by adding one tree at a time
- Test D ways of adding one tree
- D = design parameter
- Average over P_{ij} = spark probability
- $D = 1$: random addition
- $D = N^2$: test all possibilities

Measure average area of forest left untouched

- $f(c)$ = distribution of fire sizes c (= cost)
- Yield = $Y = \rho - \langle c \rangle$



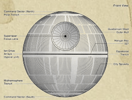
Robustness

Forest fire example: [5]

- Build a forest by adding one tree at a time
- Test D ways of adding one tree
- D = design parameter
- Average over P_{ij} = spark probability
- $D = 1$: random addition
- $D = N^2$: test all possibilities

Measure average area of forest left untouched

- $f(c)$ = distribution of fire sizes c (= cost)
- Yield = $Y = \rho - \langle c \rangle$



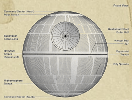
Robustness

Forest fire example: [5]

- Build a forest by adding one tree at a time
- Test D ways of adding one tree
- D = design parameter
- Average over P_{ij} = spark probability
- $D = 1$: random addition
- $D = N^2$: test all possibilities

Measure average area of forest left untouched

- $f(c)$ = distribution of fire sizes c (= cost)
- Yield = $Y = \rho - \langle c \rangle$



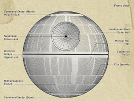
Robustness

Forest fire example: [5]

- Build a forest by adding one tree at a time
- Test D ways of adding one tree
- D = design parameter
- Average over P_{ij} = spark probability
- $D = 1$: random addition
- $D = N^2$: test all possibilities

Measure average area of forest left untouched

- $f(c)$ = distribution of fire sizes c (= cost)
- Yield = $Y = \rho - \langle c \rangle$



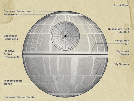
Robustness

Forest fire example: [5]

- Build a forest by adding one tree at a time
- Test D ways of adding one tree
- D = design parameter
- Average over P_{ij} = spark probability
- $D = 1$: random addition
- $D = N^2$: test all possibilities

Measure average area of forest left untouched

- $f(c)$ = distribution of fire sizes c (= cost)
- Yield = $Y = \rho - \langle c \rangle$



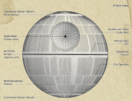
Robustness

Forest fire example: [5]

- Build a forest by adding one tree at a time
- Test D ways of adding one tree
- D = design parameter
- Average over P_{ij} = spark probability
- $D = 1$: random addition
- $D = N^2$: test all possibilities

Measure average area of forest left untouched

- $f(c)$ = distribution of fire sizes c (= cost)
- Yield = $Y = \rho - \langle c \rangle$



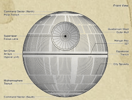
Robustness

Forest fire example: [5]

- Build a forest by adding one tree at a time
- Test D ways of adding one tree
- D = design parameter
- Average over P_{ij} = spark probability
- $D = 1$: random addition
- $D = N^2$: test all possibilities

Measure average area of forest left untouched

- $f(c)$ = distribution of fire sizes c (= cost)
- Yield = $Y = \rho - \langle c \rangle$



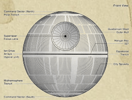
Robustness

Forest fire example: [5]

- Build a forest by adding one tree at a time
- Test D ways of adding one tree
- D = design parameter
- Average over P_{ij} = spark probability
- $D = 1$: random addition
- $D = N^2$: test all possibilities

Measure average area of forest left untouched

- $f(c)$ = distribution of fire sizes c (= cost)
- Yield = $Y = \rho - \langle c \rangle$



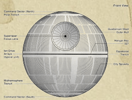
Robustness

Forest fire example: [5]

- Build a forest by adding one tree at a time
- Test D ways of adding one tree
- D = design parameter
- Average over P_{ij} = spark probability
- $D = 1$: random addition
- $D = N^2$: test all possibilities

Measure average area of forest left untouched

- $f(c)$ = distribution of fire sizes c (= cost)
- Yield = $Y = \rho - \langle c \rangle$



Robustness

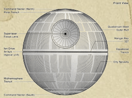
Specifics:

- $$P_{ij} = P_{i;a_x,b_x} P_{j;a_y,b_y}$$

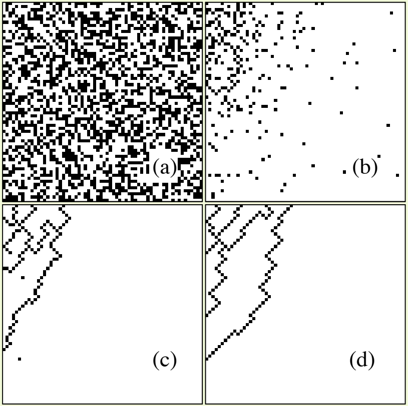
  where

  $$P_{i;a,b} \propto e^{-[(i+a)/b]^2}$$

- In the original work, $b_y > b_x$
- Distribution has more width in y direction.



HOT Forests



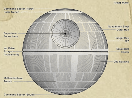
[5]

$N = 64$

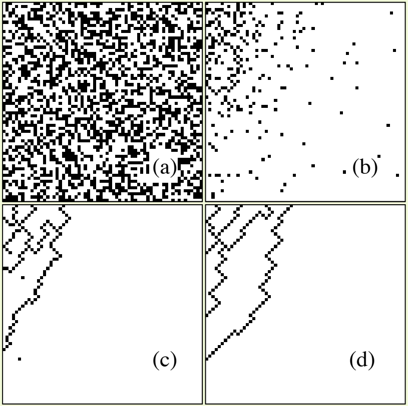
(a) $D = 1$
(b) $D = 2$
(c) $D = N$
(d) $D = N^2$

P_{ij} has a Gaussian decay

- Optimized forests do well on average
- But rare extreme events occur



HOT Forests



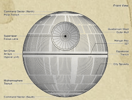
[5]

$N = 64$

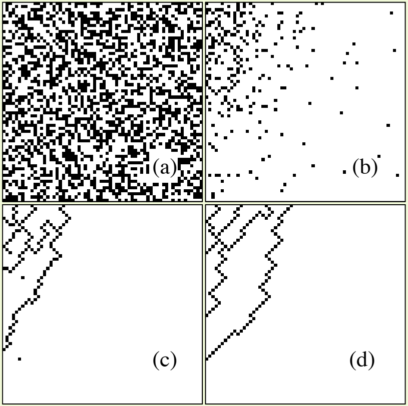
(a) $D = 1$
(b) $D = 2$
(c) $D = N$
(d) $D = N^2$

P_{ij} has a Gaussian decay

- Optimized forests do well on average
- But rare extreme events occur



HOT Forests



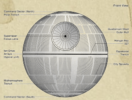
[5]

$N = 64$

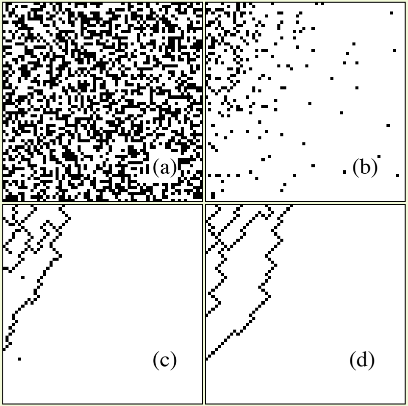
(a) $D = 1$
(b) $D = 2$
(c) $D = N$
(d) $D = N^2$

P_{ij} has a Gaussian decay

- Optimized forests do well on average
- But rare extreme events occur



HOT Forests



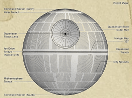
[5]

$N = 64$

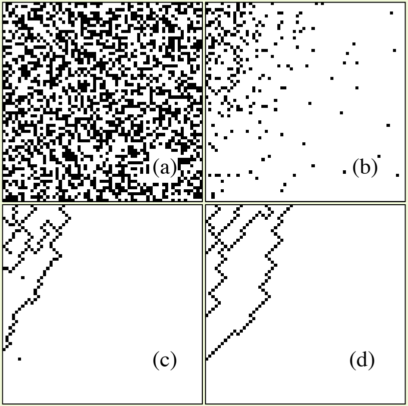
(a) $D = 1$
(b) $D = 2$
(c) $D = N$
(d) $D = N^2$

P_{ij} has a Gaussian decay

- Optimized forests do well on average (robustness)
- But rare extreme events occur



HOT Forests



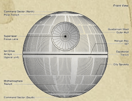
[5]

$N = 64$

(a) $D = 1$
(b) $D = 2$
(c) $D = N$
(d) $D = N^2$

P_{ij} has a Gaussian decay

- Optimized forests do well on average (robustness)
- But rare extreme events occur (fragility)



HOT Forests

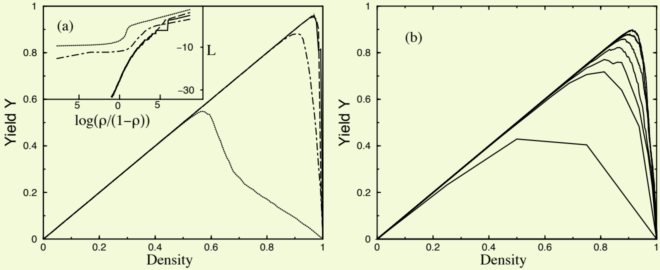
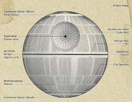


FIG. 2. Yield vs density $Y(\rho)$: (a) for design parameters $D = 1$ (dotted curve), 2 (dot-dashed), N (long dashed), and N^2 (solid) with $N = 64$, and (b) for $D = 2$ and $N = 2, 2^2, \ldots, 2^7$ running from the bottom to top curve. The results have been averaged over 100 runs. The inset to (a) illustrates corresponding loss functions $L = \log[\langle f \rangle / (1 - \langle f \rangle)]$, on a scale which more clearly differentiates between the curves.

[5]



HOT Forests:

- Y = 'the average density of trees left unburned in a configuration after a single spark hits.' [5]

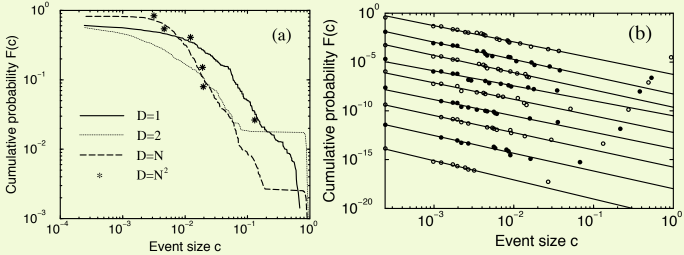
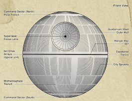


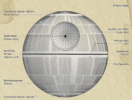
FIG. 3. Cumulative distributions of events $F(c)$: (a) at peak yield for $D = 1, 2, N$, and N^2 with $N = 64$, and (b) for $D = N^2$, and $N = 64$ at equal density increments of 0.1, ranging at $\rho = 0.1$ (bottom curve) to $\rho = 0.9$ (top curve).



Random Forests

$D = 1$: Random forests = Percolation [11]

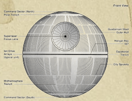
- Randomly add trees
- Below critical density ρ_c, no fires take off
- Above critical density ρ_c, percolating cluster of trees burns
- Only at ρ_c, the critical density, is there a power-law distribution of tree cluster sizes
- Forest is random and featureless



Random Forests

$D = 1$: Random forests = Percolation [11]

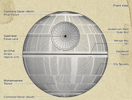
- Randomly add trees
- Below critical density ρ_c, no fires take off
- Above critical density ρ_c, percolating cluster of trees burns
- Only at ρ_c, the critical density, is there a power-law distribution of tree cluster sizes
- Forest is random and featureless



Random Forests

$D = 1$: Random forests = Percolation [11]

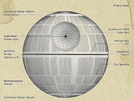
- Randomly add trees
- Below critical density ρ_c, no fires take off
- Above critical density ρ_c, percolating cluster of trees burns
- Only at ρ_c, the critical density, is there a power-law distribution of tree cluster sizes
- Forest is random and featureless



Random Forests

$D = 1$: Random forests = Percolation [11]

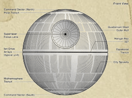
- Randomly add trees
- Below critical density ρ_c, no fires take off
- Above critical density ρ_c, percolating cluster of trees burns
- Only at ρ_c, the critical density, is there a power-law distribution of tree cluster sizes
- Forest is random and featureless



Random Forests

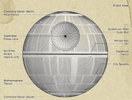
$D = 1$: Random forests = Percolation [11]

- Randomly add trees
- Below critical density ρ_c, no fires take off
- Above critical density ρ_c, percolating cluster of trees burns
- Only at ρ_c, the critical density, is there a power-law distribution of tree cluster sizes
- Forest is random and featureless



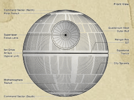
HOT forests nutshell:

- Highly structured
- Power law distribution of tree cluster sizes for $\rho > \rho_c$
- No specialness of ρ_c
- Forest states are tolerant
- Uncertainty is okay if well characterized
- If P_{ij} is characterized poorly, failure becomes highly likely



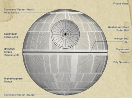
HOT forests nutshell:

- Highly structured
- Power law distribution of tree cluster sizes for $\rho > \rho_c$
- No specialness of ρ_c
- Forest states are tolerant
- Uncertainty is okay if well characterized
- If P_{ij} is characterized poorly, failure becomes highly likely



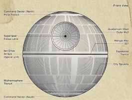
HOT forests nutshell:

- Highly structured
- Power law distribution of tree cluster sizes for $\rho > \rho_c$
- No specialness of ρ_c
- Forest states are tolerant
- Uncertainty is okay if well characterized
- If P_{ij} is characterized poorly, failure becomes highly likely



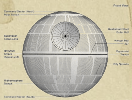
HOT forests nutshell:

- Highly structured
- Power law distribution of tree cluster sizes for $\rho > \rho_c$
- No specialness of ρ_c
- Forest states are tolerant
- Uncertainty is okay if well characterized
- If P_{ij} is characterized poorly, failure becomes highly likely



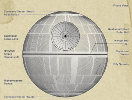
HOT forests nutshell:

- Highly structured
- Power law distribution of tree cluster sizes for $\rho > \rho_c$
- No specialness of ρ_c
- Forest states are tolerant
- Uncertainty is okay if well characterized
- If P_{ij} is characterized poorly, failure becomes highly likely



HOT forests nutshell:

- Highly structured
- Power law distribution of tree cluster sizes for $\rho > \rho_c$
- No specialness of ρ_c
- Forest states are tolerant
- Uncertainty is okay if well characterized
- If P_{ij} is characterized poorly, failure becomes highly likely



HOT forests—Real data:

"Complexity and Robustness," Carlson & Dolye [6]

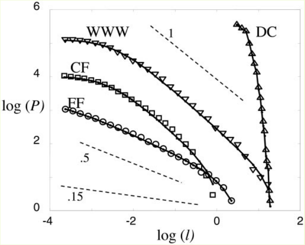
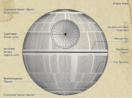


Fig. 1. Log–log (base 10) comparison of DC, WWW, CF, and FF data (symbols) with PLR models (solid lines) (for $\beta = 0, 0.9, 0.9, 1.85$, or $\alpha = 1/\beta = \infty, 1.1, 1.1, 0.054$, respectively) and the SOC FF model ($\alpha = 0.15$, dashed). Reference lines of $\alpha = 0.5$, 1 (dashed) are included. The cumulative distributions of frequencies $\mathcal{P}(l \geq l_i)$ vs. l_i describe the areas burned in the largest 4,284 fires from 1986 to 1995 on all of the U.S. Fish and Wildlife Service Lands (FF) (17), the >10,000 largest California brushfires from 1878 to 1999 (CF) (18), 130,000 web file transfers at Boston University during 1994 and 1995 (WWW) (19), and code words from DC. The size units [1,000 km^2 (FF and CF), megabytes (WWW), and bytes (DC)] and the logarithmic decimation of the data are chosen for visualization.

- PLR = probability-loss-resource.
- Minimize cost subject to resource (barrier) constraints:
  $C = \sum_i p_i l_i$
  given
  $l_i = f(r_i)$ and $\sum r_i \leq R$.



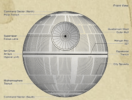
HOT theory:

The abstract story, using figurative forest fires:

- Given some measure of failure size y_i and correlated resource size x_i. with relationship $y_i = x_i^{-\alpha}$, $i = 1, \ldots, N_{\text{sites}}$.
- Design system to minimize $\langle y \rangle$ subject to a constraint on the x_i.
- Minimize cost:

$$C = \sum_{i=1}^{N_{\text{sites}}} Pr(y_i) y_i$$

Subject to $\sum_{i=1}^{N_{\text{sites}}} x_i = \text{constant}$.



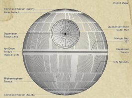
HOT theory:

The abstract story, using figurative forest fires:

- Given some measure of failure size y_i and correlated resource size x_i. with relationship $y_i = x_i^{-\alpha}$, $i = 1, \ldots, N_{\text{sites}}$.
- Design system to minimize $\langle y \rangle$ subject to a constraint on the x_i.
- Minimize cost:

$$C = \sum_{i=1}^{N_{\text{sites}}} Pr(y_i) y_i$$

Subject to $\sum_{i=1}^{N_{\text{sites}}} x_i = \text{constant}$.



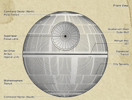
HOT theory:

The abstract story, using figurative forest fires:

- Given some measure of failure size y_i and correlated resource size x_i. with relationship $y_i = x_i^{-\alpha}$, $i = 1, \ldots, N_{\text{sites}}$.
- Design system to minimize $\langle y \rangle$ subject to a constraint on the x_i.
- Minimize cost:

$$C = \sum_{i=1}^{N_{\text{sites}}} Pr(y_i) y_i$$

Subject to $\sum_{i=1}^{N_{\text{sites}}} x_i = \text{constant}$.



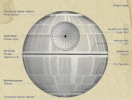
HOT theory:

The abstract story, using figurative forest fires:

- Given some measure of failure size y_i and correlated resource size x_i. with relationship $y_i = x_i^{-\alpha}$, $i = 1, \ldots, N_{\text{sites}}$.
- Design system to minimize $\langle y \rangle$ subject to a constraint on the x_i.
- Minimize cost:

$$C = \sum_{i=1}^{N_{\text{sites}}} Pr(y_i) y_i$$

Subject to $\sum_{i=1}^{N_{\text{sites}}} x_i = \text{constant}$.



1. Cost: Expected size of fire:

$$C_{\text{fire}} \propto \sum_{i=1}^{N_{\text{sites}}} (p_i a_i) a_i = \sum_{i=1}^{N_{\text{sites}}} p_i a_i^2$$

- a_i = area of ith site's region
- p_i = avg. prob. of fire at site in ith site's region

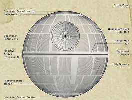
2. Constraint: building and maintaining firewalls

$$C_{\text{firewalls}} \propto \sum_{i=1}^{N_{\text{sites}}} a_i^{1/2} a_i^{-1}$$

- We are assuming isometry.
- In d dimensions, 1/2 is replaced by $(d-1)/d$

3. Insert question from assignment 5 (⊞) to find:

$$\boxed{p_i \propto a_i^{-\gamma} = a_i^{-(2+1/d)}}.$$



1. Cost: Expected size of fire:

$$C_{\text{fire}} \propto \sum_{i=1}^{N_{\text{sites}}} (p_i a_i) a_i = \sum_{i=1}^{N_{\text{sites}}} p_i a_i^2$$

- a_i = area of ith site's region
- p_i = avg. prob. of fire at site in ith site's region

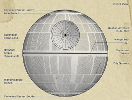
2. Constraint: building and maintaining firewalls

$$C_{\text{firewalls}} \propto \sum_{i=1}^{N_{\text{sites}}} a_i^{1/2} a_i^{-1}$$

- We are assuming isometry.
- In d dimensions, 1/2 is replaced by $(d-1)/d$

3. Insert question from assignment 5 (⊞) to find:

$$\boxed{p_i \propto a_i^{-\gamma} = a_i^{-(2+1/d)}.}$$



The UNIVERSITY of VERMONT

1. Cost: Expected size of fire:

$$C_{\text{fire}} \propto \sum_{i=1}^{N_{\text{sites}}} (p_i a_i) a_i = \sum_{i=1}^{N_{\text{sites}}} p_i a_i^2$$

- a_i = area of ith site's region
- p_i = avg. prob. of fire at site in ith site's region

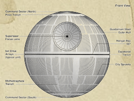
2. Constraint: building and maintaining firewalls

$$C_{\text{firewalls}} \propto \sum_{i=1}^{N_{\text{sites}}} a_i^{1/2} a_i^{-1}$$

- We are assuming isometry.
- In d dimensions, 1/2 is replaced by $(d-1)/d$

3. Insert question from assignment 5 (⊞) to find:

$$p_i \propto a_i^{-\gamma} = a_i^{-(2+1/d)}.$$



1. Cost: Expected size of fire:

$$C_{\text{fire}} \propto \sum_{i=1}^{N_{\text{sites}}} (p_i a_i) a_i = \sum_{i=1}^{N_{\text{sites}}} p_i a_i^2$$

- a_i = area of ith site's region
- p_i = avg. prob. of fire at site in ith site's region

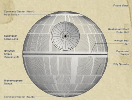
2. Constraint: building and maintaining firewalls

$$C_{\text{firewalls}} \propto \sum_{i=1}^{N_{\text{sites}}} a_i^{1/2} a_i^{-1}$$

- We are assuming isometry.
- In d dimensions, 1/2 is replaced by $(d-1)/d$

3. Insert question from assignment 5 (⊞) to find:

$$p_i \propto a_i^{-\gamma} = a_i^{-(2+1/d)}.$$



1. Cost: Expected size of fire:

$$C_{\text{fire}} \propto \sum_{i=1}^{N_{\text{sites}}} (p_i a_i) a_i = \sum_{i=1}^{N_{\text{sites}}} p_i a_i^2$$

- a_i = area of ith site's region
- p_i = avg. prob. of fire at site in ith site's region

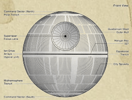
2. Constraint: building and maintaining firewalls

$$C_{\text{firewalls}} \propto \sum_{i=1}^{N_{\text{sites}}} a_i^{1/2} a_i^{-1}$$

- We are assuming isometry.
- In d dimensions, 1/2 is replaced by $(d-1)/d$

3. Insert question from assignment 5 (⊞) to find:

$$p_i \propto a_i^{-\gamma} = a_i^{-(2+1/d)}.$$



1. Cost: Expected size of fire:

$$C_{\text{fire}} \propto \sum_{i=1}^{N_{\text{sites}}} (p_i a_i) a_i = \sum_{i=1}^{N_{\text{sites}}} p_i a_i^2$$

- a_i = area of ith site's region
- p_i = avg. prob. of fire at site in ith site's region

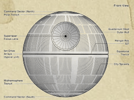
2. Constraint: building and maintaining firewalls

$$C_{\text{firewalls}} \propto \sum_{i=1}^{N_{\text{sites}}} a_i^{1/2} a_i^{-1}$$

- We are assuming isometry.
- In d dimensions, 1/2 is replaced by $(d-1)/d$

3. Insert question from assignment 5 (⊞) to find:

$$p_i \propto a_i^{-\gamma} = a_i^{-(2+1/d)}.$$



1. Cost: Expected size of fire:

$$C_{\text{fire}} \propto \sum_{i=1}^{N_{\text{sites}}} (p_i a_i) a_i = \sum_{i=1}^{N_{\text{sites}}} p_i a_i^2$$

- a_i = area of ith site's region
- p_i = avg. prob. of fire at site in ith site's region

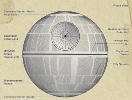
2. Constraint: building and maintaining firewalls

$$C_{\text{firewalls}} \propto \sum_{i=1}^{N_{\text{sites}}} a_i^{1/2} a_i^{-1}$$

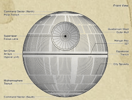
- We are assuming isometry.
- In d dimensions, 1/2 is replaced by $(d-1)/d$

3. Insert question from assignment 5 (⊞) to find:

$$\boxed{p_i \propto a_i^{-\gamma} = a_i^{-(2+1/d)}.}$$



Outline

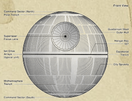


Avalanches of Sand and Rice...

SOC theory

SOC = Self-Organized Criticality

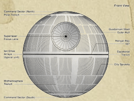
- Idea: natural dissipative systems exist at ‘critical states’;
- Analogy: Ising model with temperature somehow self-tuning;
- Power-law distributions of sizes and frequencies arise ‘for free’;
- Introduced in 1987 by Bak, Tang, and Weisenfeld [3, ?, 8]:
  “Self-organized criticality - an explanation of 1/f noise” (PRL, 1987);
- Problem: Critical state is a very specific point;
- Self-tuning not always possible;
- Much criticism and arguing...



SOC theory

SOC = Self-Organized Criticality

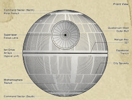
- Idea: natural dissipative systems exist at 'critical states';
- Analogy: Ising model with temperature somehow self-tuning;
- Power-law distributions of sizes and frequencies arise 'for free';
- Introduced in 1987 by Bak, Tang, and Weisenfeld [3, ?, 8]:
  "Self-organized criticality - an explanation of 1/f noise" (PRL, 1987);
- Problem: Critical state is a very specific point;
- Self-tuning not always possible;
- Much criticism and arguing...



SOC theory

SOC = Self-Organized Criticality

- Idea: natural dissipative systems exist at 'critical states';
- Analogy: Ising model with temperature somehow self-tuning;
- Power-law distributions of sizes and frequencies arise 'for free';
- Introduced in 1987 by Bak, Tang, and Weisenfeld [3, ?, 8]:
  "Self-organized criticality - an explanation of 1/f noise" (PRL, 1987);
- Problem: Critical state is a very specific point;
- Self-tuning not always possible;
- Much criticism and arguing...

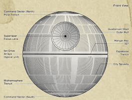


The University of Vermont

SOC theory

SOC = Self-Organized Criticality

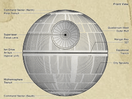
- Idea: natural dissipative systems exist at ‘critical states’;
- Analogy: Ising model with temperature somehow self-tuning;
- Power-law distributions of sizes and frequencies arise ‘for free’;
- Introduced in 1987 by Bak, Tang, and Weisenfeld [3, ?, 8]:
  “Self-organized criticality - an explanation of 1/f noise” (PRL, 1987);
- Problem: Critical state is a very specific point;
- Self-tuning not always possible;
- Much criticism and arguing...



SOC theory

SOC = Self-Organized Criticality

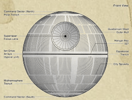
- Idea: natural dissipative systems exist at 'critical states';
- Analogy: Ising model with temperature somehow self-tuning;
- Power-law distributions of sizes and frequencies arise 'for free';
- Introduced in 1987 by Bak, Tang, and Weisenfeld [3, ?, 8]:
  "Self-organized criticality - an explanation of 1/f noise" (PRL, 1987);
- Problem: Critical state is a very specific point;
- Self-tuning not always possible;
- Much criticism and arguing...



SOC theory

SOC = Self-Organized Criticality

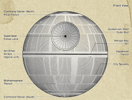
- Idea: natural dissipative systems exist at ‘critical states’;
- Analogy: Ising model with temperature somehow self-tuning;
- Power-law distributions of sizes and frequencies arise ‘for free’;
- Introduced in 1987 by Bak, Tang, and Weisenfeld [3, **?**, 8]:
  “Self-organized criticality - an explanation of 1/f noise” (PRL, 1987);
- Problem: Critical state is a very specific point;
- Self-tuning not always possible;
- Much criticism and arguing...



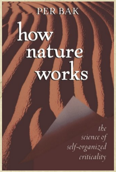
SOC theory

SOC = Self-Organized Criticality

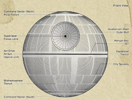
- Idea: natural dissipative systems exist at ‘critical states’;
- Analogy: Ising model with temperature somehow self-tuning;
- Power-law distributions of sizes and frequencies arise ‘for free’;
- Introduced in 1987 by Bak, Tang, and Weisenfeld [3, ?, 8]:
  “Self-organized criticality - an explanation of 1/f noise” (PRL, 1987);
- Problem: Critical state is a very specific point;
- Self-tuning not always possible;
- Much criticism and arguing...



Per Bak's Magnum Opus:



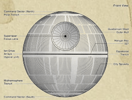
"How Nature Works: the Science of Self-Organized Criticality" (⊞) by Per Bak (1997). [2]



Robustness

HOT versus SOC

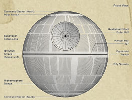
- Both produce power laws
- Optimization versus self-tuning
- HOT systems viable over a wide range of high densities
- SOC systems have one special density
- HOT systems produce specialized structures
- SOC systems produce generic structures



Robustness

HOT versus SOC

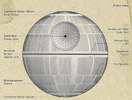
- Both produce power laws
- Optimization versus self-tuning
- HOT systems viable over a wide range of high densities
- SOC systems have one special density
- HOT systems produce specialized structures
- SOC systems produce generic structures



Robustness

HOT versus SOC

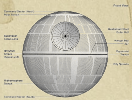
- Both produce power laws
- Optimization versus self-tuning
- HOT systems viable over a wide range of high densities
- SOC systems have one special density
- HOT systems produce specialized structures
- SOC systems produce generic structures



Robustness

HOT versus SOC

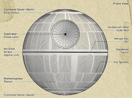
- Both produce power laws
- Optimization versus self-tuning
- HOT systems viable over a wide range of high densities
- SOC systems have one special density
- HOT systems produce specialized structures
- SOC systems produce generic structures



Robustness

HOT versus SOC

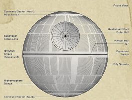
- Both produce power laws
- Optimization versus self-tuning
- HOT systems viable over a wide range of high densities
- SOC systems have one special density
- HOT systems produce specialized structures
- SOC systems produce generic structures



Robustness

HOT versus SOC

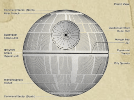
- Both produce power laws
- Optimization versus self-tuning
- HOT systems viable over a wide range of high densities
- SOC systems have one special density
- HOT systems produce specialized structures
- SOC systems produce generic structures



HOT theory—Summary of designed tolerance [6]

Table 1. Characteristics of SOC, HOT, and data

	Property	SOC	HOT and Data
1	Internal configuration	Generic, homogeneous, self-similar	Structured, heterogeneous, self-dissimilar
2	Robustness	Generic	Robust, yet fragile
3	Density and yield	Low	High
4	Max event size	Infinitesimal	Large
5	Large event shape	Fractal	Compact
6	Mechanism for power laws	Critical internal fluctuations	Robust performance
7	Exponent α	Small	Large
8	α vs. dimension d	$\alpha \approx (d-1)/10$	$\alpha \approx 1/d$
9	DDOFs	Small (1)	Large (∞)
10	Increase model resolution	No change	New structures, new sensitivities
11	Response to forcing	Homogeneous	Variable



To read: ‘Complexity and Robustness’ [6]

Colloquium

Complexity and robustness

J. M. Carlson*† and John Doyle‡

*Department of Physics, University of California, Santa Barbara, CA 93106; and ‡Control and Dynamical Systems, California Institute of Technology, Pasadena, CA 91125

Highly optimized tolerance (HOT) was recently introduced as a conceptual framework to study fundamental aspects of complexity. HOT is motivated primarily by systems from biology and engineering and emphasizes, (*i*) highly structured, nongeneric, self-dissimilar internal configurations, and (*ii*) robust yet fragile external behavior. HOT claims these are the most important features of complexity and not accidents of evolution or artifices of engineering design but are inevitably intertwined and mutually reinforcing. In the spirit of this collection, our paper contrasts HOT with alternative perspectives on complexity, drawing on real-world examples and also model systems, particularly those from self-organized criticality.

A vision shared by most researchers in complex systems is that certain intrinsic, perhaps even universal, features capture fundamental aspects of complexity in a manner that transcends specific domains. It is in identifying these features that sharp differences arise. In disciplines such as biology, engineering, sociology, economics, and ecology, individual complex systems are necessarily the objects of study, but there often appears to be little common ground between their models, abstractions, and methods. Highly optimized tolerance (HOT) (1–6) is one recent attempt, in a long history of efforts, to develop a general framework for studying complexity. The HOT view is motivated by examples from biology and engineering. Theoretically, it builds on mathematics and abstractions from control, communications, and computing. In this paper, we retain the motivating examples but avoid theories and mathematics that may be unfamiliar to a nonengineering audience. Instead, we aim to make contact with the models, concepts, and abstractions that have been loosely collected under the rubric of a “new science of complexity” (NSOC) (7) or “complex adaptive systems” (CAS), and particularly the concept of self-organized criticality (SOC) (8, 9). SOC is only one element of NSOC/CAS but is a useful representative, because it has a well-developed theory and broad range of claimed applications.

In Table 1, we contrast HOT’s emphasis on design and rare configurations with the perspective provided by NSOC/CAS/SOC, which emphasizes structural complexity as “emerging between order and disorder,” (*i*) at a bifurcation or phase transition in an interconnection of components that is (*ii*) otherwise largely random. Advocates of NSOC/CAS/SOC are inspired by critical phenomena, fractals, self-similarity, pattern formation, and self-organization in statistical physics, and bifurcations and deterministic chaos from dynamical systems. Motivating examples vary from equilibrium statistical mechanics of interacting spins on a lattice to the spontaneous formation of spatial patterns in systems far from equilibrium. This approach suggests a unity from apparently wildly different examples, because details of component behavior and their interconnection are seen as largely irrelevant to system-wide behavior.

Table 1 shows that SOC and HOT predict not just different but exactly opposite features of complex systems. HOT suggests that random interconnections of components say little about the complexity of real systems, that the details can matter enormously, and that generic (e.g., low codimension) bifurcations and phase transitions play a peripheral role. In principle, Table 1 could have a separate column for Data, by which we mean the observable features of real systems. Because HOT and Data turn out to be identical for these features, we can collapse the table as shown. This is a strong claim, and the remainder of this paper is devoted to justifying it in as much detail as space permits.

Table 1. Characteristics of SOC, HOT, and data

	Property	SOC	HOT and Data
1	Internal configuration	Generic, homogeneous, self-similar	Structured, heterogeneous, self-dissimilar
2	Robustness	Generic	Robust, yet fragile
3	Density and yield	Low	High
4	Max event size	Infinitesimal	Large
5	Large event shape	Fractal	Compact
6	Mechanism for power laws	Critical internal fluctuations	Robust performance
7	Exponent α	Small	Large
8	α vs. dimension d	$\alpha \approx (d - 1)/10$	$\alpha \approx 1/d$
9	DDOFs	Small (1)	Large (∞)
10	Increase model resolution	No change	New structures, new sensitivities
11	Response to forcing	Homogeneous	Variable

What Do We Mean By Complexity?

To motivate the theoretical discussion of complex systems, we briefly discuss concrete and hopefully reasonably familiar examples and begin to fill in the “Data” part of Table 1. We start with biological cells and their modern technological counterparts such as very large-scale integrated central processing unit (CPU) chips. Each is a complex system, composed of many components, but is also itself a component in a larger system of organs or laptop or desktop personal computers or embedded in control systems of vehicles such as automobiles or commercial jet aircraft like the Boeing 777. These are again components of the even larger networks that make up organisms and ecosystems, computer networks, and air and ground transportation. Although extremely complex, these systems have available reasonably complete descriptions and thus make good starting examples. Engineering systems are obviously better understood than biological systems, but the gap is closing. Engineers now build systems of almost bewildering levels of complexity, and biologists are beginning to move beyond the components to characterizing the networks they create. Although no one person understands in complete detail how all these systems work, there is now a rich variety of accessible introductory material in each area that gives additional details well beyond what is discussed here. In each of the following paragraphs, we consider a critical question about complexity and the answers that these example systems suggest.

This paper results from the Arthur M. Sackler Colloquium of the National Academy of Sciences, “Self-Organized Complexity in the Physical, Biological, and Social Sciences,” held March 23–24, 2001, at the Arnold and Mabel Beckman Center of the National Academies of Science and Engineering in Irvine, CA.

Abbreviations: NSOC, new science of complexity; CAS, complex adaptive systems; SOC, self-organized criticality; HOT, highly optimized tolerance; CPU, central processing unit; DC, data compression; DDOF, design degree of freedom; CF, California brushfires; FF, U.S. Fish and Wildlife Service land fires; PLR, probability-loss-resource; WWW, World Wide Web.

†To whom reprint requests should be addressed. E-mail: carlson@physics.ucsb.edu.

www.pnas.org/cgi/doi/10.1073/pnas.012582499

What Distinguishes the Internal Configurations of Systems as Complex? It is not the mere number of component parts. Any macroscopic material has a huge number of molecules. It is the extreme heterogeneity of the parts and their organization into intricate and highly structured networks, with hierarchies and multiple scales (Table 1.1). (Some researchers have suggested that “complicated” be used to describe this feature.) Even bacterial cells have thousands of genes, most coding for proteins that form elaborate regulatory networks. A modern CPU has millions of transistors and millions of supporting circuit elements; many computers have billions of transistors, and the Internet will soon have billions of nodes. The 777 is fully “fly-by-wire,” with 150,000 different subsystems, many of them quite complex, including roughly 1,000 CPUs that operate and automate all vehicle functions. Even automobiles have dozens of CPUs performing a variety of control functions. If self-similarity describes multiscale systems with similar structure at different scales, then these systems could be described as highly self-dissimilar, that is, extremely different at different scales and levels of abstraction. Just the design and manufacture of the 777 involved a global software and computing infrastructure with roughly 10,000 work stations, terabytes of data, and a one billion dollar price tag.

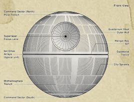
What Does This Complexity Achieve? In each example, it is possible to build similar systems with orders of magnitude fewer components and much less internal complexity. The simplest bacteria have hundreds of genes. Much simpler CPUs, computers, networks, jets, and cars can be and have been built. What is lost in these simpler systems is not their basic functionality but their robustness. By robustness, we mean the maintenance of some desired system characteristics despite fluctuations in the behavior of its component parts or its environment. Although we can loosely speak of robustness without reference to particular systems characteristics, or particular component or environmental uncertainties, this can often be misleading, as we will see. All of our motivating examples illustrate this tradeoff between robustness and internal simplicity. Although it has become a cliche that greater complexity creates unreliability, the actual story is more complicated.

What Robustness Would Be Lost in Simpler Systems? Simple bacteria with several hundred genes, like mycoplasma, require carefully controlled environments, whereas *Escherichia coli*, with almost 10 times the number of genes, can survive in highly fluctuating environments. Large internetworks do not change the basic capabilities of computers but instead improve their responsiveness to variations in a user’s needs and failures of individual computers. A jet with many fewer components and no very large-scale integrated chips or CPUs could be built with the same speed and payload as a 777, but it would be much less robust to component variations, failures, or fluctuations such as payload size and distribution or atmospheric conditions. Whereas older automobiles were simpler, new vehicles have elaborate control systems for air bags, ride control, antilock braking, antiskid turning, cruise control, satellite navigation, emergency notification, cabin temperature regulation, and automatic tuning of radios. At the same size and efficiency, they are safer, more robust, and require less maintenance. Thus robustness drives internal complexity and is the most striking feature of these complex systems.

What Is the Price Paid for These Highly Structured Internal Configurations and the Resulting Robustness? Although there is the expense of additional components, this is usually more than made up for by increased efficiency, manufacturability, evolvability of the system, and the ability to use sloppier and hence cheaper components. It is far more serious that these systems can be catastrophically disabled by cascading failures initiated by tiny perturbations. They are “robust, yet fragile,” that is, robust to what is common or anticipated but potentially fragile to what is rare or unanticipated and also to flaws in design, manufacturing, or maintenance (Table 1.2). Because robustness is achieved by very specific internal structures, when any of these systems is disassembled, there is very little latitude in reassembly if a working system is expected. Although large variations or even failures in components can be tolerated if they are designed for through redundancy and feedback regulation, what is rarely tolerated, because it is rarely a design requirement, is nontrivial rearrangements of the interconnection of internal parts. The fraction of all possible amino acid sequences or complementary metal oxide semiconductor circuits that yield functioning proteins or chips is vanishingly small. Portions of macromolecular networks as well as whole cells of advanced organisms can function *in vitro*, but we do not yet know how to reassemble them into fully functional cells and organisms. In contrast, when arbitrary interconnection is a specific design requirement, such as in routers in an internet protocol network, then this can be robustly designed for but with some added expense in resources.

How Does “Robust, Yet Fragile” Manifest Itself in the Example Systems? Biological organisms are highly robust to uncertainty in their environments and component parts yet can be catastrophically disabled by tiny perturbations to genes or the presence of microscopic pathogens or trace amounts of toxins that disrupt structural elements or regulatory control networks. The 777 is robust to large-scale atmospheric disturbances, variations in cargo loads and fuels, turbulent boundary layers, and inhomogeneities and aging of materials, but could be catastrophically disabled by microscopic alterations in a handful of very large-scale integrated chips or by software failures. (Such a vulnerability is completely absent from a hypothetical simpler vehicle.) This scenario fortunately is vanishingly unlikely but illustrates the issue that this complexity can amplify small perturbations, and the design engineer must ensure that such perturbations are extremely rare. The 777 is merely a component in a large, highly efficient, and inexpensive air traffic network, but also one that can have huge cascading delays. Processor chips are similarly robust to large variations in the analog behavior of their CMOS circuit elements and can perform a literally “universal” array of computations but can fail completely if an element is removed or the circuit rearranged. Processor, memory, and other chips can be organized into highly fault-tolerant computers and networks, creating platforms for complex software systems with their own hierarchies of components. These software systems can perform a broad range of functions primarily limited only by the programmer’s imagination but can crash from a single line of faulty code.

How Does NSOC/CAS Differ from HOT with Respect to the Complexity of the Example Systems? As a specific, if somewhat whimsical, example, note that a 777 is sufficiently automated that it can fly without pilots, so we could quite fairly describe the mechanism by which it can transport people and material through the air



Outline

Robustness

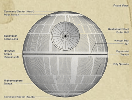
HOT theory

Self-Organized Criticality

COLD theory

Network robustness

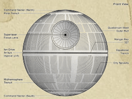
References



COLD forests

Avoidance of large-scale failures

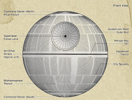
- Constrained Optimization with Limited Deviations [9]
- Weight cost of larges losses more strongly
- Increases average cluster size of burned trees...
- ... but reduces chances of catastrophe
- Power law distribution of fire sizes is truncated



COLD forests

Avoidance of large-scale failures

- Constrained Optimization with Limited Deviations [9]
- Weight cost of larges losses more strongly
- Increases average cluster size of burned trees...
- ... but reduces chances of catastrophe
- Power law distribution of fire sizes is truncated

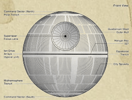


The UNIVERSITY of VERMONT

COLD forests

Avoidance of large-scale failures

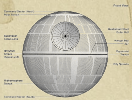
- Constrained Optimization with Limited Deviations [9]
- Weight cost of larges losses more strongly
- Increases average cluster size of burned trees...
- ... but reduces chances of catastrophe
- Power law distribution of fire sizes is truncated



COLD forests

Avoidance of large-scale failures

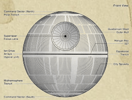
- Constrained Optimization with Limited Deviations [9]
- Weight cost of larges losses more strongly
- Increases average cluster size of burned trees...
- ... but reduces chances of catastrophe
- Power law distribution of fire sizes is truncated



COLD forests

Avoidance of large-scale failures

- Constrained Optimization with Limited Deviations [9]
- Weight cost of larges losses more strongly
- Increases average cluster size of burned trees...
- ... but reduces chances of catastrophe
- Power law distribution of fire sizes is truncated



Cutoffs

Observed:

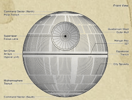
- Power law distributions often have an exponential cutoff

$$P(x) \sim x^{-\gamma} e^{-x/x_c}$$

where x_c is the approximate cutoff scale.

- May be Weibull distributions:

$$P(x) \sim x^{-\gamma} e^{-ax^{-\gamma+1}}$$



Cutoffs

Observed:

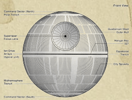
- Power law distributions often have an exponential cutoff

$$P(x) \sim x^{-\gamma} e^{-x/x_c}$$

  where x_c is the approximate cutoff scale.

- May be Weibull distributions:

$$P(x) \sim x^{-\gamma} e^{-ax^{-\gamma+1}}$$



Outline

Robustness

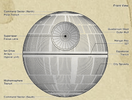
HOT theory

Self-Organized Criticality

COLD theory

Network robustness

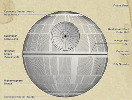
References



Robustness

We'll return to this later on:

- network robustness.
- Albert et al., Nature, 2000:
  "Error and attack tolerance of complex networks" [1]
- General contagion processes acting on complex networks. [13, 12]
- Similar robust-yet-fragile stories ...



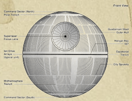
References I

[1] R. Albert, H. Jeong, and A.-L. Barabási.
Error and attack tolerance of complex networks.
Nature, 406:378–382, 2000. pdf (⊞)

[2] P. Bak.
How Nature Works: the Science of Self-Organized Criticality.
Springer-Verlag, New York, 1997.

[3] P. Bak, C. Tang, and K. Wiesenfeld.
Self-organized criticality - an explanation of 1/f noise.
Phys. Rev. Lett., 59(4):381–384, 1987. pdf (⊞)

[4] J. M. Carlson and J. Doyle.
Highly optimized tolerance: A mechanism for power laws in designed systems.
Phys. Rev. E, 60(2):1412–1427, 1999. pdf (⊞)

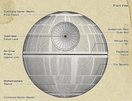


References II

[5] J. M. Carlson and J. Doyle.
Highly optimized tolerance: Robustness and design in complex systems.
Phys. Rev. Lett., 84(11):2529–2532, 2000. pdf (⊞)

[6] J. M. Carlson and J. Doyle.
Complexity and robustness.
Proc. Natl. Acad. Sci., 99:2538–2545, 2002. pdf (⊞)

[7] J. Doyle.
Guaranteed margins for LQG regulators.
IEEE Transactions on Automatic Control, 23:756–757, 1978. pdf (⊞)



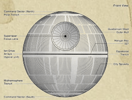
References III

[8] H. J. Jensen.
Self-Organized Criticality: Emergent Complex Behavior in Physical and Biological Systems.
Cambridge Lecture Notes in Physics. Cambridge University Press, Cambridge, UK, 1998.

[9] M. E. J. Newman, M. Girvan, and J. D. Farmer.
Optimal design, robustness, and risk aversion.
Phys. Rev. Lett., 89:028301, 2002.

[10] D. Sornette.
Critical Phenomena in Natural Sciences.
Springer-Verlag, Berlin, 1st edition, 2003.

[11] D. Stauffer and A. Aharony.
Introduction to Percolation Theory.
Taylor & Francis, Washington, D.C., Second edition, 1992.



References IV

[12] D. J. Watts and P. S. Dodds.
Influentials, networks, and public opinion formation.
Journal of Consumer Research, 34:441–458, 2007.
pdf (⊞)

[13] D. J. Watts, P. S. Dodds, and M. E. J. Newman.
Identity and search in social networks.
Science, 296:1302–1305, 2002. pdf (⊞)