Robustness

Carlos Gershenson

IIMAS & C3, UNAM

http://turing.iimas.unam.mx/~cgg/teach/Pamplona

Contenido

Coping with Complexity

- **Adaptation** (learning, evolution; feedback control). The system changes to cope with the change.
- **Anticipation**(cognition; feedforward control). The system predicts a change to cope with, and adjusts accordingly. This is a special case of adaptation, where the system does not require to experience a situation before responding to it.
- **Robustness** (buffering). A system is robust if it continues to function in the face of perturbations (Wagner, 2007).

Robustness

- Present in many biological systems at multiple scales (from molecules to ecosystems).
- Not so much in engineered systems...
- Understand better phenomenon to build robust systems.
- More general than stability...
- How would you relate it to Ashby's concept of a machine?
- Perturbations can be functional, structural, material, etc...

Questions about Robustness (Jen, 2005 + my spoon)

- What are its origins? How does it evolve?
- What are the organizational principles (redundancy, modularity, neutrality, hierarchy, degeneracy, spatial structure, diversification) that characterize robust entities? How are these related? What is their cost?
- What is the effect of robustness on evolvability, adaptability, and fitness?
- When is robustness desirable and when not?

Robustness Below Gene Level

- Neutral landscapes are characterized by large domains where changes in sequence have no effect on phenotype.
- RNA folding:
  - Neutral networks percolate the entire genotype space (there are always neutral point mutations)
  - All phenotypes are present around a relatively small neighbourhood of a given sequence.
- Protein folding:
  - Proteins modular and robust
  - similar to RNA...
  - universal principles? (we'll see next class!)

Robustness Above Gene Level

- Development
- Metabolic pathways
- Metabolic networks

Robustness Beyond Organism

- Ecology (resilience, stability)
    - Food webs
    - Keystone species
    - Homeostasis
    - Weak links (Csermely, 2006).

Robustness in Artificial Systems

- WWW, TCP/IP
- P2P systems
- *Ad hoc* networks
- Robust social systems???
- Can there be a robust economy???

Robustness in Networks

- Scale-free networks are robust to random attacks but fragile to directed attacks to hubs.
- How can you improve robustness?
- Link all hubs?
- Duplicate hubs?
- Shift or hide hubs?

Robustness in RBNs (Gershenson, Kauffman & Shmulevich, 2006)

- Random Boolean networks (RBNs), being general models, can be used to explore theories of how evolution can take place in rugged landscapes; or even change the landscapes.
- What happens when redundant nodes are added?
- Redundancy consists on having more than one copy of an element type.
    - Diploidy.
    - Growth in GRN seems to be via duplication + mutation.
- Redundancy is a way of “smoothening” fitness landscapes.
- Too much redundancy could reduce the rate of adaptation of an evolutionary process.

Kauffman's Conjecture (2000, p. 195)

1-bit mutations to a minimal program will change drastically the output of the program, making it indistinguishable from noise.

- The conjecture points to the necessity of having some redundancy to allow smooth transitions as a program changes in an evolutionary search.
- How to obtain a minimal program?
- How to measure redundancy or compressibility (Kolmogorov, Chaitin, Solomonoff)?
- Let's better compare more and less redundant networks.

Redundant Links

- Redundant links are fictitious links, i.e. functionality does not change.
- Check sensitivity to initial conditions (similar states converge or diverge?)
- Check damage spreading: 1 bit mutations of lookup tables, then compare overlap of the state space transitions using normalised Hamming distances (Equation 1).
- Result: no difference, fictitious links do not affect stability of networks.

$$H(A,B) = \frac{1}{n}\sum_{i}^{n} |a_i - b_i| \tag{1}$$

Redundant Nodes

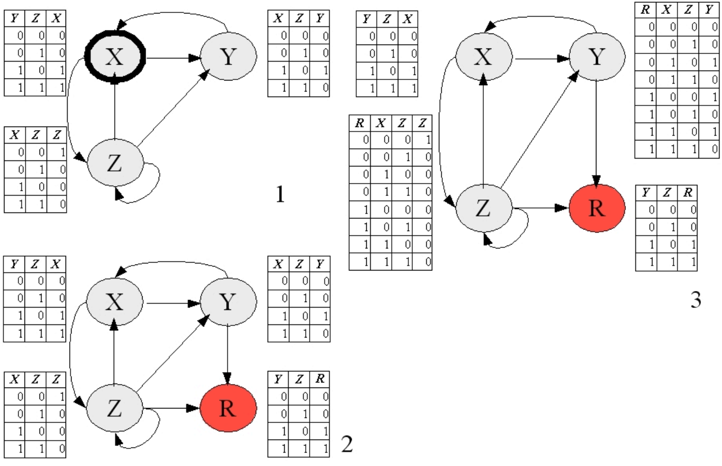
1. Select randomly a node X to be “duplicated”.
2. Add a new node R to the network ($N^{new} = N^{old} + 1$), with the same inputs and lookup table as X (i.e. $k_R = k_X$, $f_R = f_X$), and outputs to the same nodes of which X is input:

$$k_i^{new} = k_i^{old} \cup k_{i_R} \quad if\ \exists k_{i_X}, \forall i \tag{2}$$

3. Double the lookup tables of the nodes of which X is input with the following criterion: When $R = 0$, copy the old lookup table. When $R = 1$, and $X = 0$, copy the same values for all combinations when $X = 1$ and $R = 0$. Copy again the same values to the combinations where $X = 1$ and $R = 1$. In other words, make an inclusive OR function in which X OR R should be one, to obtain the old outputs when only X was one

$$\left.\begin{array}{l}
f_i^{new}(\sigma_{i_1}, \cdots, \sigma_{i_X} = 0, \sigma_{i_R} = 0 \cdots, \sigma_{i_{k_i}}) = f_i^{old}(\sigma_{i_1}, \cdots, \sigma_{i_X} = 0, \cdots, \sigma_{i_{k_i}}) \\
f_i^{new}(\sigma_{i_1}, \cdots, \sigma_{i_X} = 0, \sigma_{i_R} = 1 \cdots, \sigma_{i_{k_i}}) = f_i^{old}(\sigma_{i_1}, \cdots, \sigma_{i_X} = 1, \cdots, \sigma_{i_{k_i}}) \\
f_i^{new}(\sigma_{i_1}, \cdots, \sigma_{i_X} = 1, \sigma_{i_R} = 0 \cdots, \sigma_{i_{k_i}}) = f_i^{old}(\sigma_{i_1}, \cdots, \sigma_{i_X} = 1, \cdots, \sigma_{i_{k_i}}) \\
f_i^{new}(\sigma_{i_1}, \cdots, \sigma_{i_X} = 1, \sigma_{i_R} = 1 \cdots, \sigma_{i_{k_i}}) = f_i^{old}(\sigma_{i_1}, \cdots, \sigma_{i_X} = 1, \cdots, \sigma_{i_{k_i}})
\end{array}\right\} \quad if\ \exists k_{i_X}, \forall$$

Example RBN $N = 3$, $K = 2$



1

Y	Z	X
0	0	0
0	1	0
1	0	1
1	1	1

X	Z	Y
0	0	0
0	1	0
1	0	1
1	1	0

X	Z	Z
0	0	1
0	1	0
1	0	0
1	1	0

2

Y	Z	X
0	0	0
0	1	0
1	0	1
1	1	1

X	Z	Y
0	0	0
0	1	0
1	0	1
1	1	0

X	Z	Z
0	0	1
0	1	0
1	0	0
1	1	0

Y	Z	R
0	0	0
0	1	0
1	0	1
1	1	1

3

Y	Z	X
0	0	0
0	1	0
1	0	1
1	1	1

R	X	Z	Z
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

R	X	Z	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Y	Z	R
0	0	0
0	1	0
1	0	1
1	1	1

Simulation Experiments

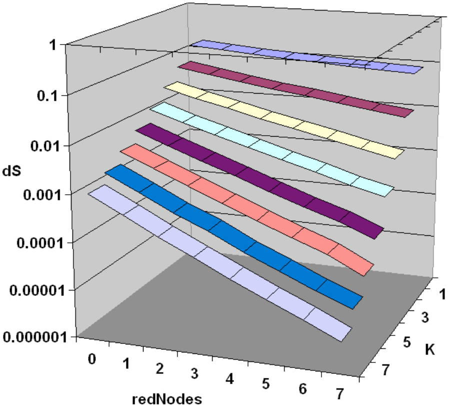
- Implemented in RBNLab `http://rbn.sourceforge.net`
- Measure the overlaps of state space transitions of 1-bit mutant nets with “original” ones as red nodes are added.
- The mutations consist in flipping a random bit from the lookup table of a randomly selected node.
- We used normalised Hamming distances (1) to measure the difference dS between state spaces:

$$dS = \frac{1}{2^n} \sum_{i}^{2^n} H(A_i^{t+1}, M_i^{t+1}) \tag{3}$$

Effect of Mutations

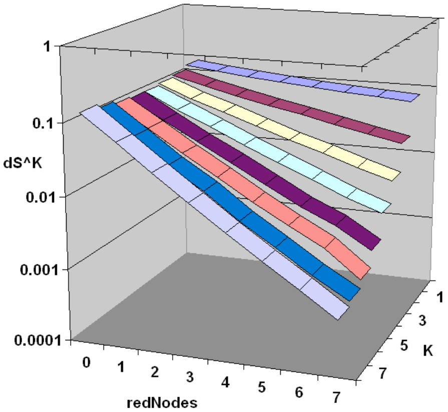
- If $dS = 0$, there is no difference between state spaces, and thus the mutation had no effect.
- A higher dS reflects a greater effect of the mutation in the state space.
- There is no correlation of state spaces, i.e. a mutation is maximally catastrophic, when $dS \simeq 0.5$
- Disadvantage: restricted to small networks (state space doubles for each node added).

Results $N = 7$, $p = 0.5$



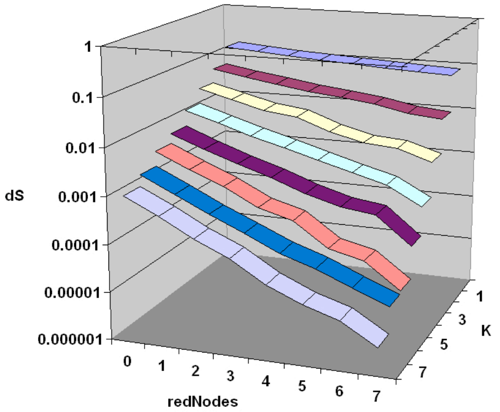
Note that the *dS* for RBNs without red nodes decreases as *K* increases. This is because the lookup tables are doubled each time *K* is incremented. Thus, a one-bit mutation will have less effect on a network with higher connectivity.

Results $N = 7$, $p = 0.5$, dS^K



1-bit mutation makes $dS^K \simeq 0.1$ for networks without red nodes. Adding red nodes increases the network robustness. The effect of red nodes is more evident for higher K values, where the network dynamics are more chaotic.

Results $N = 10$, $p = 0.5$



A network with 7 white + 3 red nodes has a lower dS than a network with ten white nodes, especially for high values of K.

More experiments

- Similar results with scale-free topologies.
- Similar results with different updating schemes (deterministic asynchronous, non-deterministic asynchronous, etc.)

Discussion

- Mutations in redundant nodes do not propagate through the network, whereas mutations in fictitious inputs do.
- "Useful" vs. "useless" redundancy.
- Useful redundancy *smoothens* rough fitness landscapes.
- Useful redundancy increases neutrality.
- Too much redundancy can make evolutionary search slow.
- No need to add redundancy in smooth landscapes.

On Kauffman's Conjecture

- A random mutation on a "minimal" RBN should produce a $dS \simeq 0.5$.
- But you cannot prove that a RBN is minimal (independently of a fixed universal model of computation).
- "Less compressible" RBNs, i.e. with no red nodes, have always a higher dS.
- As we add more red nodes, the RBN becomes more compressible, and dS decreases, showing an increase in robustness.
- The compressibility is directly proportional to the robustness of a network to random mutations.
- Positive evidence in favour of conjecture.

Redundancy vs. Degeneracy

- Evidence showed that robustness in molecular and genetic networks is due to degeneracy, not redundancy (Wagner, 2007).
- Maybe different types of robustness with different mechanisms?

Conclusions (you tell me)

- What is redundancy?
- Why is it useful?
- What are examples of robust systems?
- How can we achieve robustness in systems?
- What is the effect of redundant nodes in RBNs?

Bibliografía

- Csermely, P. (2006). *Weak Links: Stabilizers of Complex Systems from Proteins to Social Networks*. Springer.
- Gershenson, C., Kauffman, S. A., and Shmulevich, I. (2006). The role of redundancy in the robustness of random Boolean networks. In Rocha, L. M., Yaeger, L. S., Bedau, M. A., Floreano, D., Goldstone, R. L., and Vespignani, A., editors, *Artificial Life X, Proceedings of the Tenth International Conference on the Simulation and Synthesis of Living Systems*, pages 35–42. MIT Press. `http://arxiv.org/abs/nlin.AO/0511018`
- Jen, E. (Ed.) (2005). *Robust Design: A Repertoire of Biological, Ecological, and Engineering Case Studies* . Oxford University Press. `http://tinyurl.com/rahcly`
- Wagner, A. (2007). *Robustness and Evolvability in Living Systems*. Princeton University Press.