Modeling Open and Closed Cyber-Physical Systems with Graph Automata and Boolean Networks

Predrag Tosic

Department of Computer Science, University of Idaho

ptosic@uidaho.edu

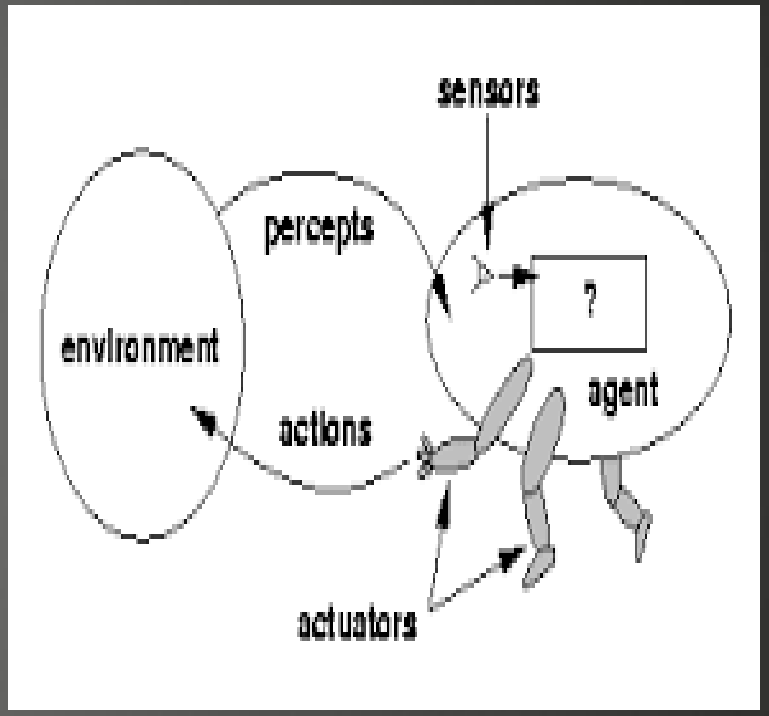
Sept. 11, 2017

Talk Outline

- Introduction: Agents and Their Environments
- Cyber-Physical and Multi-Agent Systems
- Boolean Networks, Graph & Cellular Automata
- **Open** & Closed Systems of Simple Reactive Agents
- Complexity Results for BNA models: Open vs. Closed
- Implications & interpretations of our results
- Conclusions

AI View of the World: An Agent Embedded in An Environment

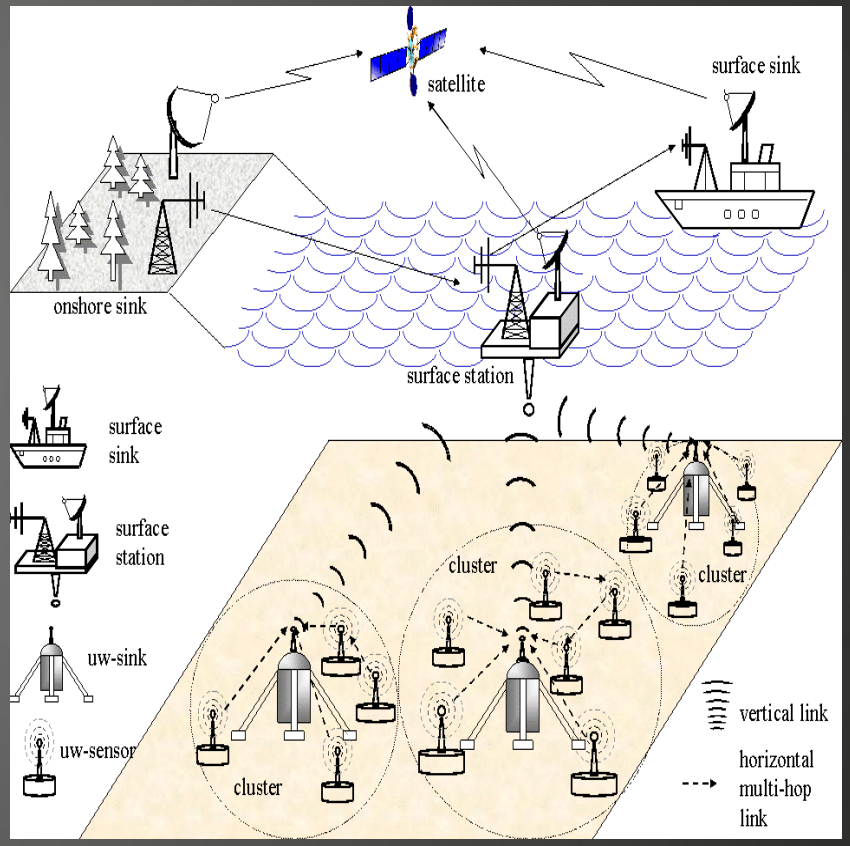
- **[AIMA]** by S. Russell and P. Norvig: the problem of AI is, given a problem and an environment, to design an "intelligent agent" that solves problem via dynamic interaction with its environment
  -- sensing
  -- decision making
  -- acting



The Environment affects the Agent(s), and vice versa
-- we can control agent's behavior, but not the environment's

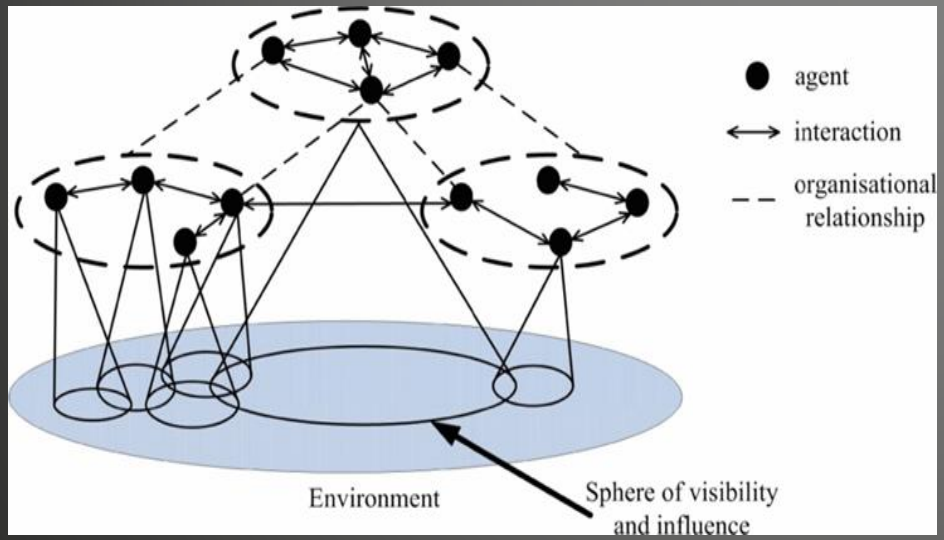
Cyber-Physical Systems

- Systems that include both physical and computational ("cyber") components
- Often both human and artificial decision-makers
- Complex interactions among sensing, communication, control and computation
- Decision need to be made, actions taken in real-time, under various constraints on resources and knowledge



Modeling Multi-Agent & Cyber-Physical Systems

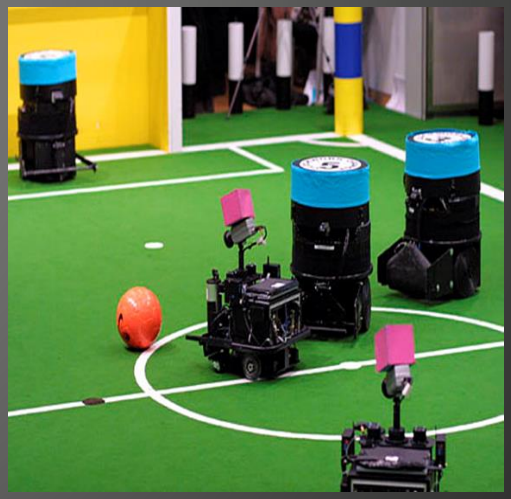
- Analysis and control get much harder, when multiple agents are acting in an environment
- Early coupled automata based models
- Theories and models of situated agents



- Focusing on interaction among simple, reactive agents and their (possibly stochastic) environment

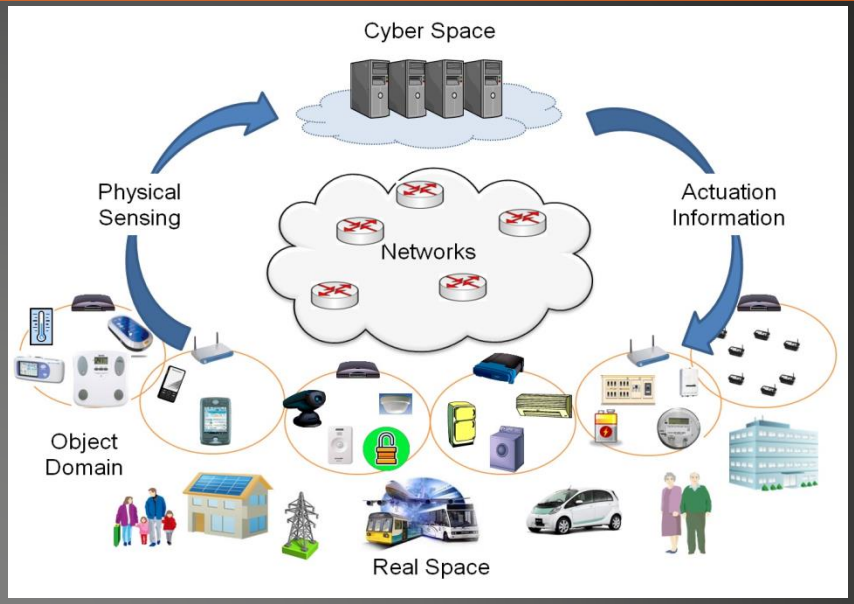
Examples

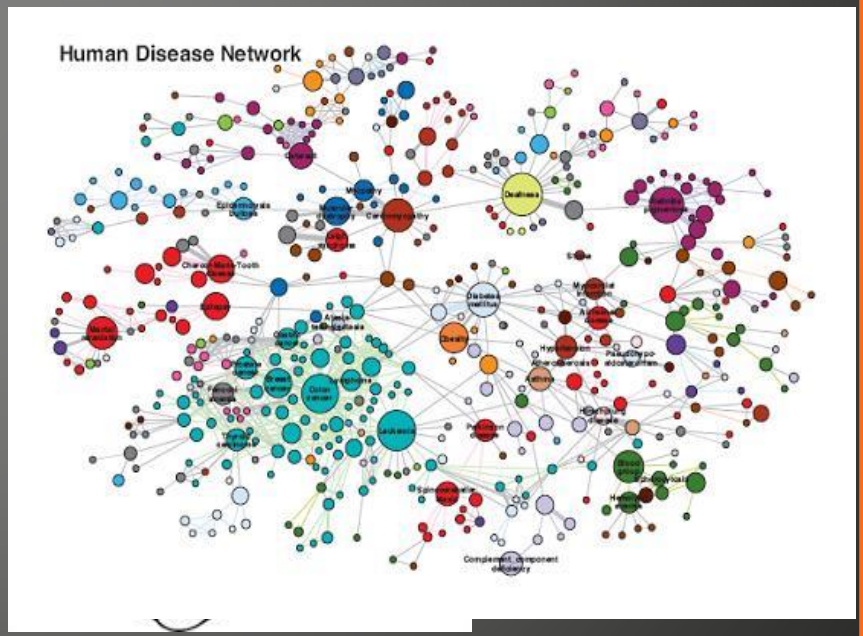
- Soccer robot teams (closed)
- Unmanned vehicles (military/surveillance, open)
- Traffic controller network (open)
- Epidemics propagation (depends)



ABM and Network Science

- Complex networks used in agent-based modeling (ABM) of complex Cyber-Physical Systems
- ABM of transportation systems, "smart grids", epidemics, …
- Discrete Network Models of emerging behavior, collective dynamics
- Computational and mathematical characterization of those networks' dynamics





Predicting Dynamics of Complex Networks with Simple Local Behaviors

- Network Science as the framework and formal foundation for massive-scale CPS, "Swarm Intelligence" & Multi-Agent Systems
- Methodology & tools from Mathematics, Physics, Complexity Science as well as Computer Science and Systems Engineering
  - discrete instead of continuous mathematics
  - agent-based modeling instead of PDE/ODE

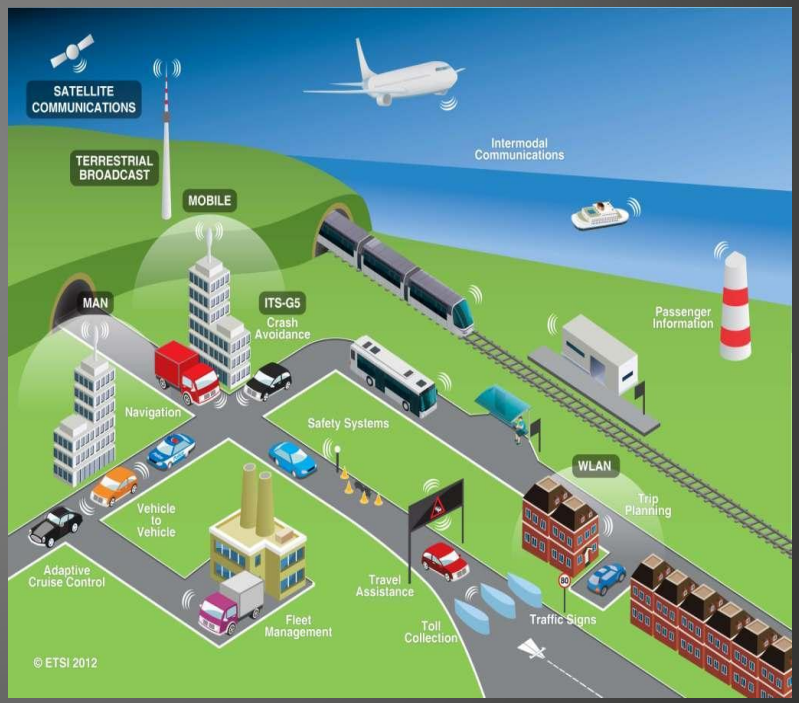
"Killer Apps": Next-Gen Distributed Large-Scale Cyber-Physical and Smart-n-Connected Systems made of variety of autonomous decision-making agents

…together with social networks and computational biology

A Classical Example of ABM and Applied Network Science: Modeling Traffic Systems

- Agent-Based Modeling (ABM) of a city traffic
- Discretize car speeds, road/street segments, etc.; finitely many actions
- Sensing and decision-making is local (e.g., car ahead of you, car coming from perpendicular street, stop light immediately ahead, etc.)

- Towards intelligent transportation systems:
  - smart/connected vehicles
  - synchronized & adaptable signaling
  - prediction & avoidance of traffic jams

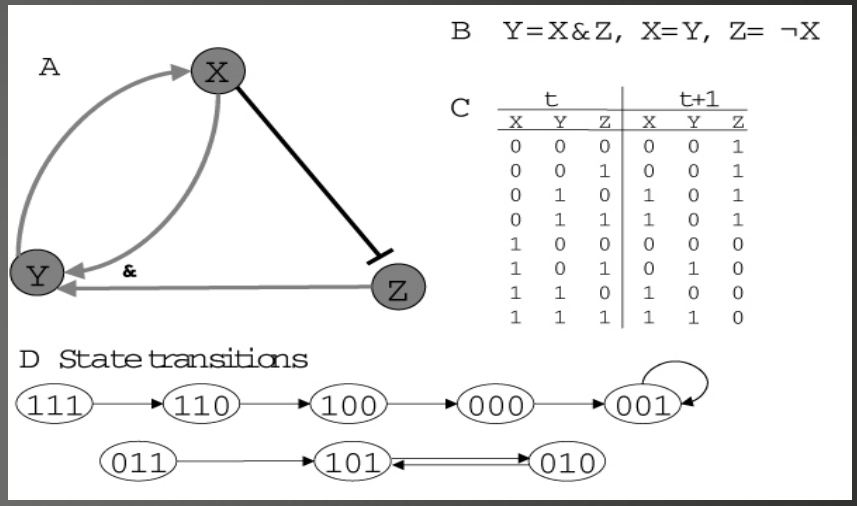


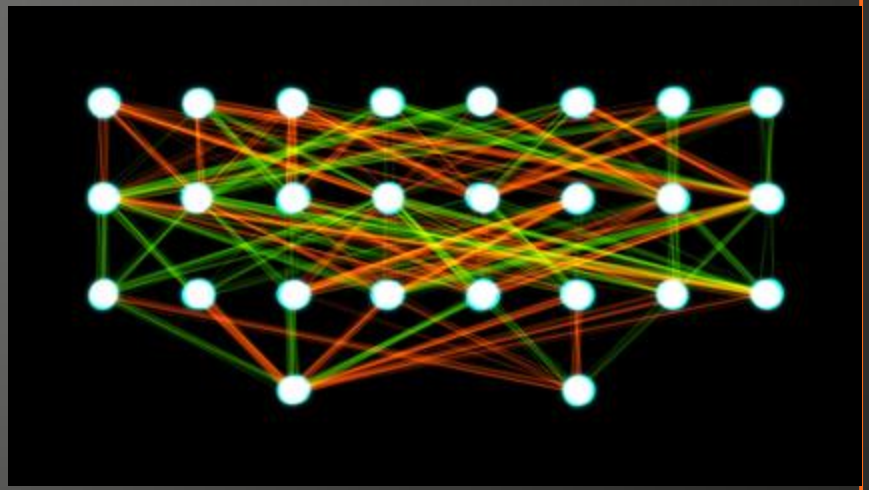
"System of systems"

modeling & simulation

Boolean Networks

- Restricted Communicating Finite State Machine (CFSM) models
- Binary-valued nodes (agents) interconnected together
- Each node has 2 states
- Nodes make up a FSM, updates state from time t to time t+1 according to some update rule
- Update rule specifies local behavior
- Graph structure (network topology) specifies interaction pattern among agents

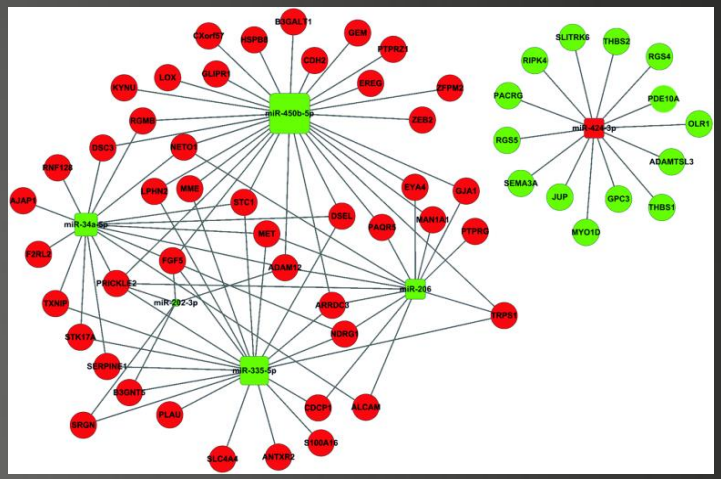




Biological & AI examples of BNs

Random Boolean Networks

- broad variety of phenomena in systems in biology, ecology, etc.

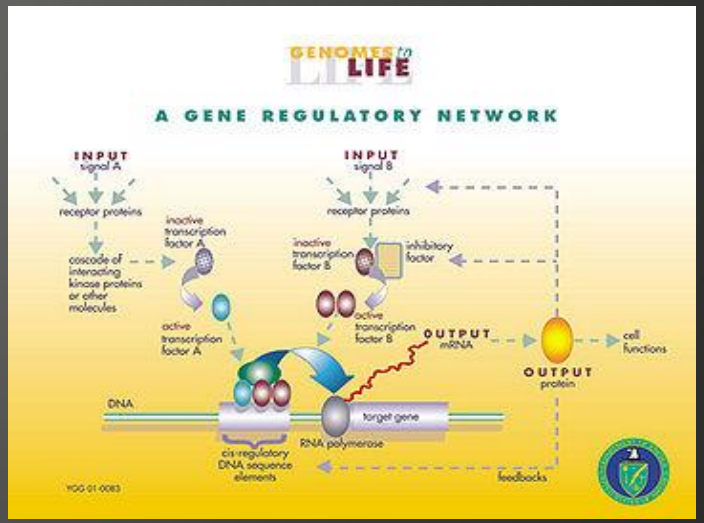


Gene Regulatory Networks

"GRN is a collection of regulators that interact with each other and with other substances in the cell to govern gene expression levels of mRNA and proteins"

Discrete Hopfield Networks

- approximating brains as associative memories storing discrete patterns



Formalizing Boolean Networks & Their Dynamics

- Network: directed or undirected G = (V, E)
- Node = agent; Edge = interaction/influence between two agents
- G usually assumed sparse: |V|=O(n)
- States are binary: Boolean networks
- Next state based on neighbors' (current) states

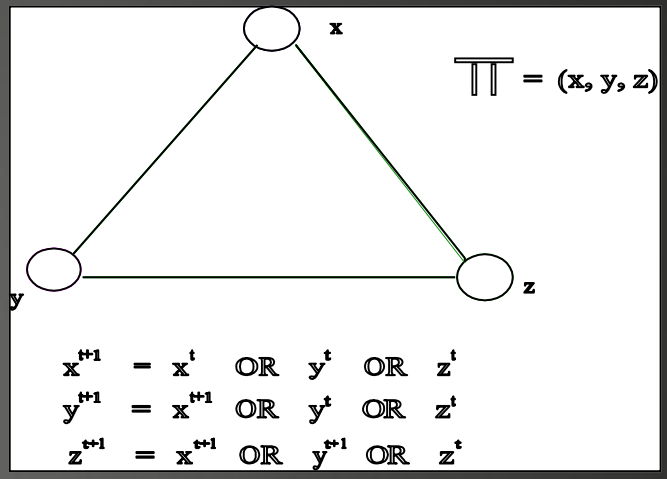
$$v_i^{t+1} \leftarrow f_i\left(v_{i1}^t, \ldots, v_{ik}^t\right)$$

- Special types of phase space configurations:
  - Gardens of Eden
  - (Temporal) Cycles
  - Transient
  - **Fixed Points**

Discrete Network Dynamics:
Configuration Spaces

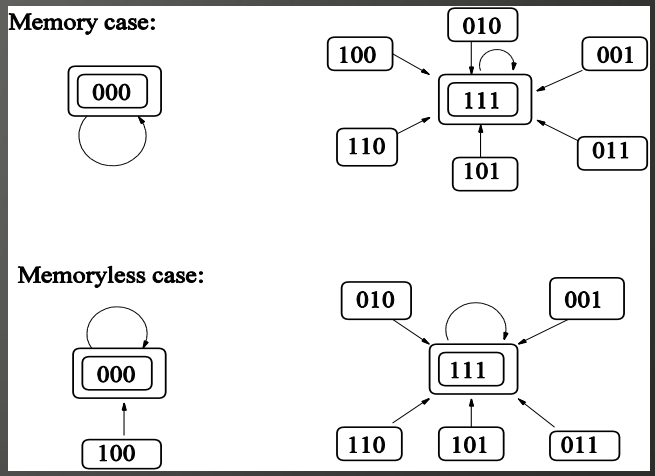
Configuration space (or *phase space*) of a *discrete* dynamical system (CA or BNA) is a directed graph where:

- Vertices are *configurations*
- Directed edges are *transitions* from one configuration to another



Three fundamental types of configurations (for deterministic CA / Graph Automata / BNA):

- **FP**s and **C**ycle **C**onfigurations are *recurrent*
- **TC**s (transient configurations) are not



Some Classes of Problems About BNA Configuration Space Properties

Problems about forward (esp. asymptotic) dynamics

- Reachability of stable states, cycle states, particular subset of states
- Speed of convergence to a stable state, cyclical/periodic states, …

Problems about backward dynamics

- Existence / number of predecessors; size/depth of "basin of attraction"
- Is dynamics reversible / is global map a bijection on the set of configurations?

Existence of various configuration types

- Does a given BNA have a "fixed point"? Temporal cycles? Long transients?
- Does a given configuration have a predecessor? A big basin of attraction?

Enumeration of particular types of configurations

- How many FPs? How many temporal cycles? (i.e., how many possible dynamics?)
- Given an arbitrary or specific state, how many predecessors / ancestors?

Exploring Phase Spaces: Objectives

- Existence of stable configurations
- Counting stable configurations
- Finding reachable configurations
- Length of transient chains / how long until convergence
- Invertible dynamics and Gardens of Eden
- …

Foundational Work on CA, GA, BNA Models and Their Dynamics

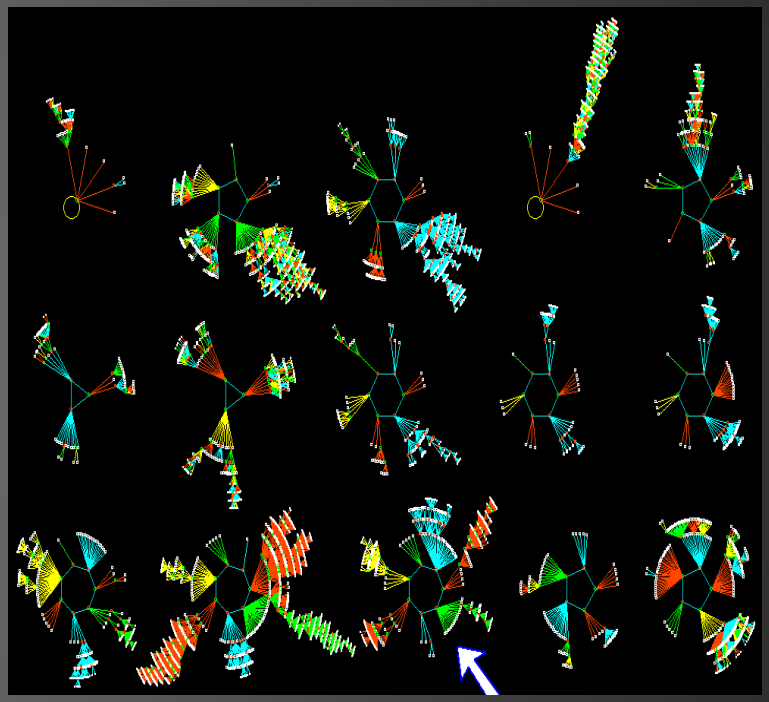
- Computational complexity aspects of CA
  - work by K. Sutner, J. Kari, B. Duran, etc.
- Complexity of fundamental dynamic properties of DHNs
  - e.g., work by Orponen and Floreen (1980's and 1990's)

- ***One-way CA*** (1-way CA on Cayley graphs model by Zs. Roka)
- ***Graph Automata*** of B. Martin and C. Nichitiu, E. Remila
- *Sequential node updates*
  - R. Laubenbacher, B. Pareigis (theory of simulation in social sciences)
  - B. Huberman, N. Glance (work on *dynamics in games*, early 2000's)

- *Foundational work on* ***SDS*** **and SyDS** in the context of developing theory of large-scale computer simulations of complex infrastructures (e.g., *TRANSIMS*)
  - C. Barrett et al. [1999 – 2003]
  - C. Reidys, H. Mortveit [2000 - 03]
  - Our early work w/ M. Marathe, H. Hunt 2000 - 01,
  - Then PhD dissertation (2003 – 2006), then independently (2010 -)

Some Quick Remarks on Methodology: Typical / "Average" vs. Worst-Case Behavior

- Most common approach to investigating collective dynamics, emerging behavior is via *extensive computer simulations*
- Simulations provide insights into possible / typical / "expected" behavior of a complex system or network
- For lower bounds on complexity (worst-case behavior), theoretical tools are more suitable



- Two approaches are complementary, both are needed and useful

Some Fundamental Results

Theorem 1: Starting from an arbitrary initial configuration, determining if deterministtic dynamics will eventually end up in a FP or a temporal cycle is PSPACE-complete

Theorem 2: Determining if an arbitrary BN or GA has a FP is NP-complete. Determining if there a cyclic configuration is NP-hard (Question: why not obvious that it's NP-complete?)

Theorem 3: Determining if an arbitrary BN or GA has (at least one) Transient or Garden of Eden configuration is NP-complete.

Theorem 4: If local update rules are monotone functions then there existence of (at least one) FP is guaranteed

Computational Complexity of Counting

The problems of *computing the permanent* and *counting (perfect) matchings* in bipartite graphs [L. Valiant '79]

- **Class #P** : those counting problems accepted by *poly-time bounded* ***Nondeterministic Turing Machine*** s.t. the # of accepting computations equals the # of problem's solutions

- **#P-complete problems:** the hardest in **#P**
- **NP**-complete decision problems (seem to always) have their counting versions **#P-complete**

  … but so do several problems (whose decision versions are in) **P**

  (e.g., bipartite matchings, 2CNF SAT, MON-2CNF SAT)

Theoretical Computer Science 201: How to Prove #P-Completeness

- We need efficient (polynomial-time) reductions that *preserve the # of solutions*
- ***Strongly parsimonious Reductions*** : poly- time transformations that *exactly* preserve the # of solutions: $\#\Lambda(f(I)) = \#\Lambda(I)$
- ***Weakly Parsimonious Reductions*** : poly-time transformations **f** that allow the # of solutions of $\Lambda(I)$ to be efficiently recovered from $\#\Lambda(f(I))$

Note: weakly parsimonious reductions are in general easier to design than the strongly parsimonious ones

… yet suffice for establishing **#P**-completeness

Counting Fixed Points (FPs)

Theorem 5:

- (i) Determining #FP exactly for an arbitrary DHN or other BN is **#P-complete** in general
- (ii) Approximately counting #FP is NP-hard in general - the underlying graph can be restricted to planar and/or bipartite, the graph can also be made *uniformly sparse*

Theorem 6: Determining #FP exactly for an DHN / BN with monotone node update rules is **#P**-complete

- Not surprising, given a broad variety of **Monotone CNF formulae** for which counting satisfying assignments is **#P-complete**
- - hardness still holds for sparse monotone CNF formulae

Counting FPs under monotone linear threshold update rules

Theorem 7 : Counting **#FP** of *Monotone* BN / GA is **#P**-complete even when all of the following conditions *simultaneously* hold:

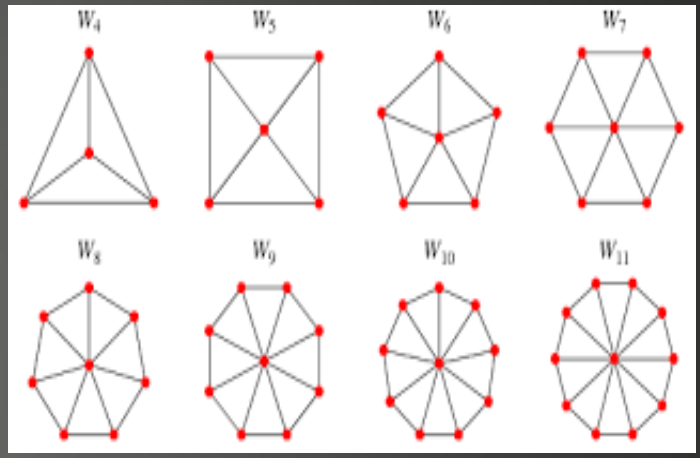
- monotone update rules are *linear threshold functions* with $w_{ij} > 0$
- *only two different integer weights are used*
- each node has *at most* (alternatively, *exactly)* 3 neighbors

General Implications:

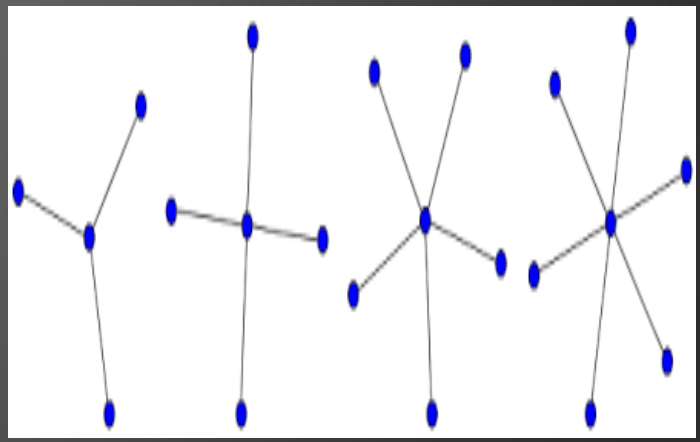
- Determining exact capacity of an associative memory cannot be done for non-trivial sizes (implications for comput. neuroscience)
- No short-cut to step-by-step simulation (implications for agent-based simulation in general)

Complexity of Determining #FPs for BN Defined on Star / Wheel Graphs

Theorem 8: Counting all FPs of a BN / DHN defined over a star graph or a wheel graph is **#P**-complete (in worst-case)



Theorem 8': Determining the exact size of the basic of attraction of an FP of a BN, DHN or other GA defined on a star or a wheel is **#P**-complete



Understanding Discrete Dynamics

- Given a current configuration of a BN or GA in a simple deterministic environment, determining in how many ways could that configuration have been reached is intractable
- Determining if stable/fixed or cyclic/oscillatory behavior will eventually be reached is PSPACE-complete
- Determining if here are FPs, unreachable configurations, transient configurations is NP-hard
  -- If behavior can be modeled with a binary function then these problems are still NP-complete in worst case
- Counting FPs, TCs, GEs, # predecessors is #P-complete

Discussion: Closed vs. Open Systems

Previous results were all for CLOSED systems: each node is an agent with well-defined, deterministic local behavior
What do those results imply for **OPEN systems**?

- If the central node captures the environment, then this environment's impact on peripheral nodes (i.e., agents) will be in general at least as complex as impact of a peer agent (of specified, known behavior)

Some immediate consequences:

- Enumerating all asymptotic / stable configurations of a collection of agents embedded in an environment is intractable, even if agents "cannot see each other", only influence each other via the environment
- This holds true even if the environment acts as a simple "switch" (only two possible states) and fully deterministic!

More Consequences for Open Systems

Theorem (#FP for Open Systems) : Enumerating all stable configurations of an agent ensemble where agents can influence each other only indirectly, through a deterministic "binary-valued" environment, is in general computationally intractable

Other implications:

- **Theorem 8'' (Open)**: Let a collection of reactive agents be interconnected into a ring, and embedded in a deterministic, "binary-valued" environment. Enumerating all stable config's of such agent ensemble is computationally intractable
- **Theorem 5'' (Open)**: Given a current configuration of an agent ensemble embedded in a simple deterministic environment, determining in how many ways could that configuration have been reached is in general computationally intractable

Conclusions

We study dynamics of Boolean Networks by formally investigating their configuration spaces

Among various configuration space properties, we focused on enumeration problems (how many fixed point configurations, predecessors, possible dynamics?)

These enumeration problems are provably intractable in BNA modeling closed multi-agent systems (no "environment")

Those enumeration problems are at least as hard in open systems setting – even when interaction patterns & environment behavior are fully deterministic and simple

Future Work

- Identify practical cases/examples of real-world networks of MAS/CPS with provable gap between open and closed systems
- Quantifying how complexity of "the environment" affects tractability of understanding dynamics of open systems

- Approximate counting: in which instances is approximating #FP and other #'s of interest computationally tractable?
- Simulations starting from a small set of initial states / GEs
- Higher probability for one state in some or all FSMs (capturing natural "bias" in a given application domain)
- Small *n* allows brute force search of stable configurations
- Focus on sparse graphs with structure (e.g., small-world; social/assortative networks; etc.)