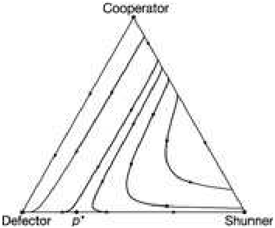
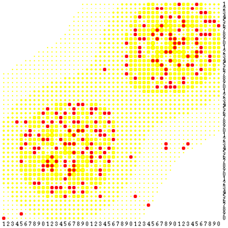
Addressing the validation problem for social simulations: the Adversarial Scheduling Approach

Author Gabriel Istrate, eAustria Research Institute, Timişoara, Romania. email: gabrielistrate@acm.org

Conference International Workshop on Challenges and Visions in the Social Sciences

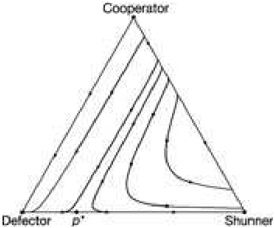
Two sides of the same coin

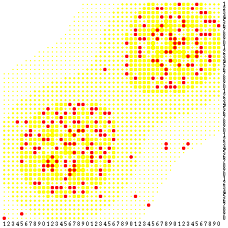




- Share family of similar interaction-based models (Blume and Durlauf)

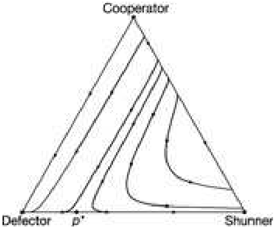
Two sides of the same coin

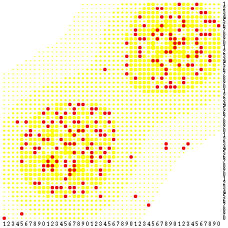




- Share family of similar interaction-based models (Blume and Durlauf)
- Agents located at the vertices of a graph.

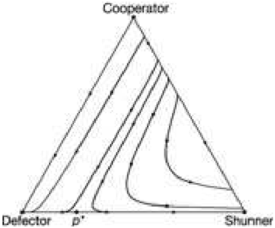
Two sides of the same coin

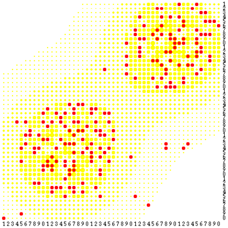




- Share family of similar interaction-based models (Blume and Durlauf)
- Agents located at the vertices of a graph.
- They have a state.

Two sides of the same coin





- Share family of similar interaction-based models (Blume and Durlauf)
- Agents located at the vertices of a graph.
- They have a state.
- They update the state based on social (network) interaction. Game-theoretic.

Interpreting results

- For a large class of models: results$\Rightarrow$ stylized facts $\Rightarrow$ **mechanisms**.

Interpreting results

- For a large class of models: results$\Rightarrow$ stylized facts $\Rightarrow$ **mechanisms**.
- Causal connections between features of the simulation process/mathematical result.

Interpreting results

- For a large class of models: results$\Rightarrow$ stylized facts $\Rightarrow$ **mechanisms**.
- Causal connections between features of the simulation process/mathematical result.
- Example from EGT: equilibrium selection.

Interpreting results

- For a large class of models: results$\Rightarrow$ stylized facts $\Rightarrow$ **mechanisms**.
- Causal connections between features of the simulation process/mathematical result.
- Example from EGT: equilibrium selection.
- Peyton-Young: adding continuous noise to best-response dynamics can select (risk dominant) equilibria.

Verification & Validation

- How can we make sure that conclusions do not crucially depend on particular assumptions of the model ?

Verification & Validation

- How can we make sure that conclusions do not crucially depend on particular assumptions of the model ?
- Not merely academic: TRANSIMS (multi-million $ program, U.S. DOT), epidemics (Epstein et al., EPISIMS).

Verification & Validation

- How can we make sure that conclusions do not crucially depend on particular assumptions of the model ?
- Not merely academic: TRANSIMS (multi-million $ program, U.S. DOT), epidemics (Epstein et al., EPISIMS).
- Factors: social network topology, interaction structure ("higher-order emergence"),

Verification & Validation

- How can we make sure that conclusions do not crucially depend on particular assumptions of the model ?
- Not merely academic: TRANSIMS (multi-million $ program, U.S. DOT), epidemics (Epstein et al., EPISIMS).
- Factors: social network topology, interaction structure ("higher-order emergence"),
- Scheduling: order in which agents update.

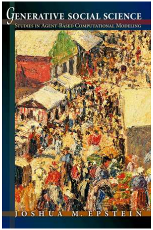
Verification & Validation

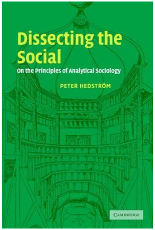
- How can we make sure that conclusions do not crucially depend on particular assumptions of the model ?
- Not merely academic: TRANSIMS (multi-million $ program, U.S. DOT), epidemics (Epstein et al., EPISIMS).
- Factors: social network topology, interaction structure ("higher-order emergence"),
- Scheduling: order in which agents update.
- Synchronous vs asynchronous (Huberman & Glance)

Verification & Validation

- How can we make sure that conclusions do not crucially depend on particular assumptions of the model ?
- Not merely academic: TRANSIMS (multi-million $ program, U.S. DOT), epidemics (Epstein et al., EPISIMS).
- Factors: social network topology, interaction structure ("higher-order emergence"),
- Scheduling: order in which agents update.
- Synchronous vs asynchronous (Huberman & Glance)
- Want something more principled.

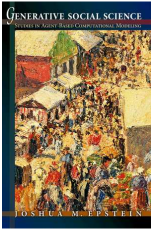
Context: Mechanism-based explanations

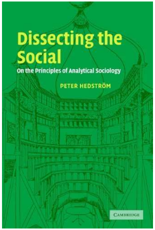




- Epstein: to explain a phenomenon P is to *grow* it.

Context: Mechanism-based explanations

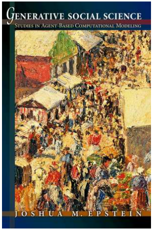


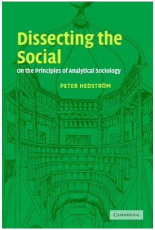


- Epstein: to explain a phenomenon *P* is to *grow* it.
- Analytical social science (Hedström): *mechanism-based explanations*.

...
eventually P.

Context: Mechanism-based explanations



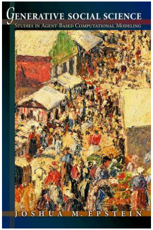


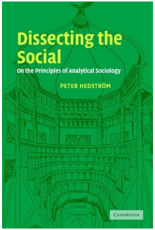
- Epstein: to explain a phenomenon P is to *grow* it.
- Analytical social science (Hedström): *mechanism-based explanations*.

...
eventually P.

- One of three types of explanations. Covering laws, statistical correlation laws. Mechanisms concatenate

Context: Mechanism-based explanations



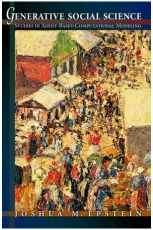


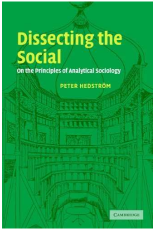
- Epstein: to explain a phenomenon P is to *grow* it.
- Analytical social science (Hedström): *mechanism-based explanations*.

...
eventually P.

- One of three types of explanations. Covering laws, statistical correlation laws. Mechanisms concatenate
- *What precisely is a mechanism ?* Can we automate identification of mechanisms in social simulations ?

Context: Mechanism-based explanations





- Epstein: to explain a phenomenon P is to *grow* it.
- Analytical social science (Hedström): *mechanism-based explanations*.

...
eventually P.

- One of three types of explanations. Covering laws, statistical correlation laws. Mechanisms concatenate
- *What precisely is a mechanism ?* Can we automate identification of mechanisms in social simulations ?
- Formal logic in social science: not that popular. Temporal logic (Elster), logic in organization theory (Péli, Hannan).

Adversarial Scheduling

- This talk: develop a game-theoretical model in a bottom-up fashion. Identify mechanism-like explanations for properties each model in the sequence.

Adversarial Scheduling

- This talk: develop a game-theoretical model in a bottom-up fashion. Identify mechanism-like explanations for properties each model in the sequence.
- Concentrate on one single factor: *update order*.

Adversarial Scheduling

- This talk: develop a game-theoretical model in a bottom-up fashion. Identify mechanism-like explanations for properties each model in the sequence.
- Concentrate on one single factor: *update order*.
- Mathematical models: *controlled experiments* for causal identification.

Adversarial Scheduling

- This talk: develop a game-theoretical model in a bottom-up fashion. Identify mechanism-like explanations for properties each model in the sequence.
- Concentrate on one single factor: *update order*.
- Mathematical models: *controlled experiments* for causal identification.
- Bottom-up approach: start with basic features. Add (in a controlled manner) new features.

Adversarial Scheduling

- This talk: develop a game-theoretical model in a bottom-up fashion. Identify mechanism-like explanations for properties each model in the sequence.
- Concentrate on one single factor: *update order*.
- Mathematical models: *controlled experiments* for causal identification.
- Bottom-up approach: start with basic features. Add (in a controlled manner) new features.
- Strategy used in Sugarscape (Axtell-Epstein), Brownian agents (Schweitzer), varieties of emergence (Gilbert).

Adversarial Scheduling

- This talk: develop a game-theoretical model in a bottom-up fashion. Identify mechanism-like explanations for properties each model in the sequence.
- Concentrate on one single factor: *update order*.
- Mathematical models: *controlled experiments* for causal identification.
- Bottom-up approach: start with basic features. Add (in a controlled manner) new features.
- Strategy used in Sugarscape (Axtell-Epstein), Brownian agents (Schweitzer), varieties of emergence (Gilbert).
- Provides insight on ingredients of mechanism-based explanations.

Base model

- n uncoupled agents. Each makes a choice between two states, A and B. Each agent same utility, $u(A) > u(B)$.

Base model

- n uncoupled agents. Each makes a choice between two states, A and B. Each agent same utility, $u(A) > u(B)$.
- "Stylized" fact: system converges to "all A" state.

Base model

- n uncoupled agents. Each makes a choice between two states, A and B. Each agent same utility, $u(A) > u(B)$.
- "Stylized" fact: system converges to "all A" state.
- Stylized fact:

$\ldots$
fair(S) $\gg$ eventually $(\forall x)$ A(x)

Base model

- n uncoupled agents. Each makes a choice between two states, A and B. Each agent same utility, $u(A) > u(B)$.
- "Stylized" fact: system converges to "all A" state.
- Stylized fact:

...
fair(S) $\gg$ eventually $(\forall x)$ A(x)

- Left/right hand sides: properties of *processes*.

Base model

- n uncoupled agents. Each makes a choice between two states, A and B. Each agent same utility, $u(A) > u(B)$.
- "Stylized" fact: system converges to "all A" state.
- Stylized fact: $\dfrac{\ldots}{\text{fair(S)} \gg \text{eventually } (\forall x)\ \text{A(x)}}$
- Left/right hand sides: properties of *processes*.
- Second stylized fact: convergence time $n\log(n) + \theta(n)$. Coupon Collector Lemma.

Base model

- n uncoupled agents. Each makes a choice between two states, A and B. Each agent same utility, $u(A) > u(B)$.
- "Stylized" fact: system converges to "all A" state.
- Stylized fact: $\dfrac{\ldots}{\text{fair(S)} \gg \text{eventually } (\forall x)\ \text{A(x)}}$
- Left/right hand sides: properties of *processes*.
- Second stylized fact: convergence time $n\log(n) + \theta(n)$. Coupon Collector Lemma.
- Adversarial Scheduling: fair scheduler. One that touches all nodes.

Base model

- n uncoupled agents. Each makes a choice between two states, A and B. Each agent same utility, $u(A) > u(B)$.
- "Stylized" fact: system converges to "all A" state.
- Stylized fact: $\dfrac{\ldots}{\text{fair(S)} \gg \text{eventually } (\forall x)\ \text{A(x)}}$
- Left/right hand sides: properties of *processes*.
- Second stylized fact: convergence time $n\log(n) + \theta(n)$. Coupon Collector Lemma.
- Adversarial Scheduling: fair scheduler. One that touches all nodes.
- Second stylized fact: *covering law*. Mapping from one domain (balls and bins) to agents.

n balls $n\ln(n) + cn$ bins random(S)	$\Rightarrow$	n agents $n\ln(n) + cn$ time steps random(S)
Pr[nonempty]$\sim 1 - e^{-e^{-c}}$		Pr[state **A**]$\sim 1 - e^{-e^{-c}}$

Second version

- n uncoupled agents. Each makes a choice between two states, A and B.

Second version

- n uncoupled agents. Each makes a choice between two states, A and B..
- Each agent chooses better state with fixed probability $1 - \epsilon$.

Second version

- n uncoupled agents. Each makes a choice between two states, A and B..
- Each agent chooses better state with fixed probability $1 - \epsilon$.
- Now: no longer fixed point A. Instead: "most" states A most of the time.

Second version

- n uncoupled agents. Each makes a choice between two states, A and B..
- Each agent chooses better state with fixed probability $1 - \epsilon$.
- Now: no longer fixed point A. Instead: "most" states A most of the time.
- What about adversarial scheduling ?

Second version

- n uncoupled agents. Each makes a choice between two states, A and B..
- Each agent chooses better state with fixed probability $1-\epsilon$.
- Now: no longer fixed point A. Instead: "most" states A most of the time.
- What about adversarial scheduling ?
- Is stylized fact

  $$\frac{\ldots}{\text{fair(S)} \gg \text{eventually } (\geq (1-\epsilon)\% x) \text{ A(x)}}$$ true ?
- If scheduler can act based on agents' state: can preclude state A !

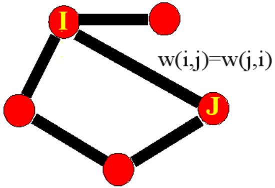
Second version

- n uncoupled agents. Each makes a choice between two states, A and B..
- Each agent chooses better state with fixed probability $1-\epsilon$.
- Now: no longer fixed point A. Instead: "most" states A most of the time.
- What about adversarial scheduling ?
- Is stylized fact

$\ldots$	
fair(S) $\gg$ eventually ($\geq (1-\epsilon)\% x$) A(x)	true ?

- If scheduler can act based on agents' state: can preclude state A !
- Scheduling effectively introduces coupling in the system.

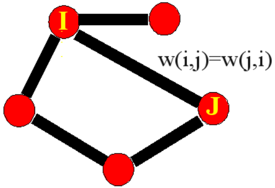
Third version: adding interaction



	A	B
A	(a,a)	(c,d)
B	(d,c)	(b,b)

- *A* is a *risk-dominant* equilibrium.

Third version: adding interaction



	A	B
A	(a,a)	(c,d)
B	(d,c)	(b,b)

- *A* is a *risk-dominant* equilibrium.
- $p^{\beta}(x_i \to z|\overline{x}) \sim e^{\beta \cdot \nu_i(z,\overline{x}_{-i})}$, where $\nu_i(z,\overline{x}_{-i})$, the payoff of the i'th agent should he play strategy z while the others' profile remains the same is given by $\nu_i(z,\overline{x}_{-i}) = \sum_{(i,j)\in E} w_{ij} m_{z,x_j}$.

Peyton-Young's result

Definition Consider a Markov process P^0 defined on a finite state space Ω. For each $\epsilon > 0$, define a Markov process P^ϵ on Ω. P^ϵ is a regular perturbed Markov process if all of the following conditions hold.

- P^ϵ is irreducible for every $\epsilon > 0$.

Peyton-Young's result

Definition Consider a Markov process P^0 defined on a finite state space Ω. For each $\epsilon > 0$, define a Markov process P^ϵ on Ω. P^ϵ is a regular perturbed Markov process if all of the following conditions hold.

- P^ϵ is irreducible for every $\epsilon > 0$.
- For every $x, y \in \Omega$, $\lim_{\epsilon > 0} P^\epsilon_{xy} = P^0_{xy}$.

Peyton-Young's result

Definition

Consider a Markov process P^0 defined on a finite state space Ω. For each $\epsilon > 0$, define a Markov process P^ϵ on Ω. P^ϵ is a regular perturbed Markov process if all of the following conditions hold.

- P^ϵ is irreducible for every $\epsilon > 0$.
- For every $x, y \in \Omega$, $\lim_{\epsilon > 0} P^\epsilon_{xy} = P^0_{xy}$.
- If $P_{xy} > 0$ then there exists $r(m) > 0$, the *resistance of transition $m = (x \rightarrow y)$*, such that as $\epsilon \rightarrow 0$, $P^\epsilon_{xy} = \Theta(\epsilon^{r(m)})$.

Let μ^ϵ be the (unique) stationary distribution of P^ϵ. A state S is *a stochastically stable strategy* if $\underline{\lim}_{\epsilon \rightarrow 0} \mu^\epsilon(S) > 0$.

- Peyton Young $\epsilon = e^\beta$.

Peyton-Young's result

Definition Consider a Markov process P^0 defined on a finite state space Ω. For each $\epsilon > 0$, define a Markov process P^ϵ on Ω. P^ϵ is a regular perturbed Markov process if all of the following conditions hold.

- P^ϵ is irreducible for every $\epsilon > 0$.
- For every $x, y \in \Omega$, $\lim_{\epsilon > 0} P^\epsilon_{xy} = P^0_{xy}$.
- If $P_{xy} > 0$ then there exists $r(m) > 0$, the *resistance of transition $m = (x \to y)$*, such that as $\epsilon \to 0$, $P^\epsilon_{xy} = \Theta(\epsilon^{r(m)})$.

Let μ^ϵ be the (unique) stationary distribution of P^ϵ. A state S is *a stochastically stable strategy* if $\underline{\lim}_{\epsilon \to 0} \mu^\epsilon(S) > 0$.

- Peyton Young $\epsilon = e^\beta$.
- Theorem (P.Y.) Under **random scheduling A** only stochastically stable state.

Peyton-Young under adversarial scheduling

- A random scheduler is nonadaptive: *does not use state information* in order to choose next agent to update.

Peyton-Young under adversarial scheduling

- A random scheduler is nonadaptive: *does not use state information* in order to choose next agent to update.
- Adaptive agents can preclude stability. Pick any agent. Schedule until it plays B. Go to other agent. Repeat.

Peyton-Young under adversarial scheduling

- A random scheduler is nonadaptive: *does not use state information* in order to choose next agent to update.
- Adaptive agents can preclude stability. Pick any agent. Schedule until it plays B. Go to other agent. Repeat.
- Basic non-adaptive agent: random walk on a given graph.

Peyton-Young under adversarial scheduling

- A random scheduler is nonadaptive: *does not use state information* in order to choose next agent to update.
- Adaptive agents can preclude stability. Pick any agent. Schedule until it plays B. Go to other agent. Repeat.
- Basic non-adaptive agent: random walk on a given graph.
- Generalizes random scheduler (random walk on the complete graph).

Peyton-Young under adversarial scheduling

- A random scheduler is nonadaptive: *does not use state information* in order to choose next agent to update.
- Adaptive agents can preclude stability. Pick any agent. Schedule until it plays B. Go to other agent. Repeat.
- Basic non-adaptive agent: random walk on a given graph.
- Generalizes random scheduler (random walk on the complete graph).
- Markov chain on state space $V^{\{A,B\}} \times V$.

Peyton-Young under adversarial scheduling

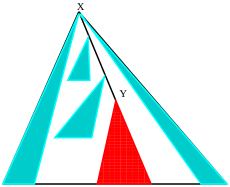
- A random scheduler is nonadaptive: *does not use state information* in order to choose next agent to update.
- Adaptive agents can preclude stability. Pick any agent. Schedule until it plays B. Go to other agent. Repeat.
- Basic non-adaptive agent: random walk on a given graph.
- Generalizes random scheduler (random walk on the complete graph).
- Markov chain on state space $V^{\{A,B\}} \times V$.
- **Theorem [GMR08]:** *Under RW scheduling the set $S_0 = \{(A,x) | x \in V\}$ is the set of stochastically stable states*.

Proof idea

Definition A *tree rooted at node j* is a set T of edges such that for any state $w \neq j$ there exists a unique (directed) path from w to j. The *resistance of a rooted tree T* is the sum of resistances of all edges in T.

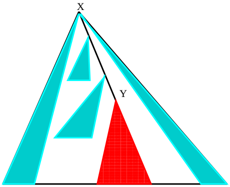
Proposition *Let P^ϵ be a regular perturbed Markov process, and for each $\epsilon > 0$ let μ^ϵ be the unique stationary distribution of P^ϵ. Then $\lim_{\epsilon \to 0} \mu^\epsilon = \mu^0$ exists, and μ^0 is a stationary distribution of P^0. The stochastically stable states are precisely those states z such that there exists a tree rooted at z of minimal resistance (among all rooted trees).*

Proof idea (II)



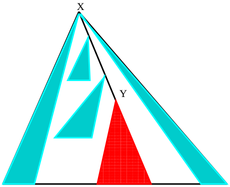
- Use Peyton-Young's criterion for stochastic stability.

Proof idea (II)



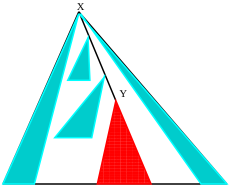
- Use Peyton-Young's criterion for stochastic stability.
- Choose $Y \in S_0$, $X \notin S_0$ and a tree T of minimal potential rooted at X. Transform T into a tree of smaller potential rooted at Y.

Proof idea (II)



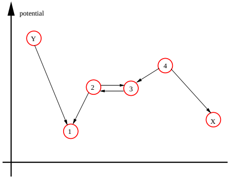
- Use Peyton-Young's criterion for stochastic stability.
- Choose $Y \in S_0$, $X \notin S_0$ and a tree T of minimal potential rooted at X. Transform T into a tree of smaller potential rooted at Y.
- "Reverse" path from X to Y. Transform subtrees of T (four cases).

Proof idea (II)



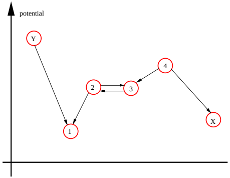
- Use Peyton-Young's criterion for stochastic stability.
- Choose $Y \in S_0$, $X \notin S_0$ and a tree T of minimal potential rooted at X. Transform T into a tree of smaller potential rooted at Y.
- "Reverse" path from X to Y. Transform subtrees of T (four cases).

Proof idea (III)



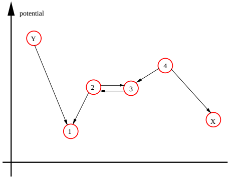
- Crucial: **potential game**. There is a "potential function" on $V^{\{A,B\}}$ whose variations measure resistance.

Proof idea (III)



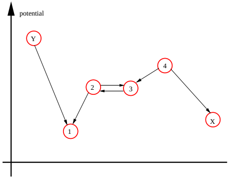
- Crucial: **potential game**. There is a "potential function" on $V^{\{A,B\}}$ whose variations measure resistance.
- Positive resistance move I: state does not change, changing state would be better. Positive resistance move II: state changes, but not optimal move.

Proof idea (III)



- Crucial: **potential game**. There is a "potential function" on $V^{\{A,B\}}$ whose variations measure resistance.
- Positive resistance move I: state does not change, changing state would be better. Positive resistance move II: state changes, but not optimal move.
- State **A**: highest potential. *Potential difference between path and its "reverse" = difference between endpoints.*

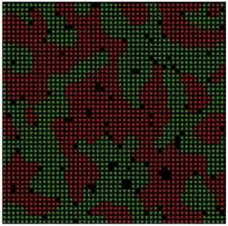
Proof idea (III)

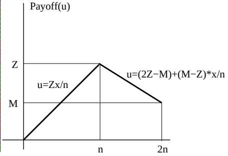


- Crucial: **potential game**. There is a "potential function" on $V^{\{A,B\}}$ whose variations measure resistance.
- Positive resistance move I: state does not change, changing state would be better. Positive resistance move II: state changes, but not optimal move.
- State **A**: highest potential. *Potential difference between path and its "reverse" = difference between endpoints.*
-

Potential game & . . .
fair(S) & nonadaptive(S) $\gg$ "$(\forall x)$ A(x) only stable state"

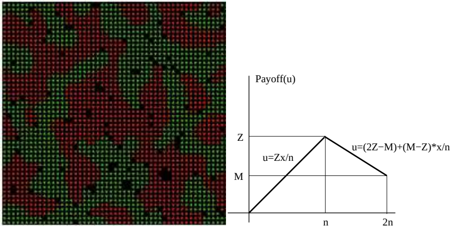
Application: Schelling's Segregation Model





- Similar idea: *Schelling's Segregation Model.* Peyton-Young (1-D), Zhang, Pollicott& Weiss (2-D).

Application: Schelling's Segregation Model



- Similar idea: *Schelling's Segregation Model.* Peyton-Young (1-D), Zhang, Pollicott& Weiss (2-D).
- Scheduler: **(Markovian contagion)** To each pair of vertices e we associate a probability distribution D_e on $V \times V$. We then choose the next scheduled pair according to the following process: If t_i is the pair scheduled at stage i, we chose t_{i+1}, the next scheduled pair, by sampling from D_{t_i}.

Application: Schelling's Segregation Model (II)

- Agents' utility function: $u_i(\cdot) = rw(\cdot) + \epsilon$, where r is a positive constant, and $w(x)$ is defined as the difference between the number of neighbours of x having the same color and the number of neighbors of x having the opposite color. Further assumptiion: same constant r, but possibly different constants ϵ.
- Update:

$$Pr[\text{switch}] = \frac{e^{\beta[u_1(\cdot|\text{switch})+u_2(\cdot|\text{switch})]}}{e^{\beta[u_1(\cdot|\text{switch})+u_2(\cdot|\text{switch})]} + e^{\beta[u_1(\cdot|\text{not switch})+u_2(\cdot|\text{not switch})]}},$$

Application: Schelling's Segregation Model (II)

- Agents' utility function: $u_i(\cdot) = rw(\cdot) + \epsilon$, where r is a positive constant, and $w(x)$ is defined as the difference between the number of neighbours of x having the same color and the number of neighbors of x having the opposite color. Further assumptiion: same constant r, but possibly different constants ϵ.
- Update:

$$Pr[\text{switch}] = \frac{e^{\beta[u_1(\cdot|\text{switch})+u_2(\cdot|\text{switch})]}}{e^{\beta[u_1(\cdot|\text{switch})+u_2(\cdot|\text{switch})]} + e^{\beta[u_1(\cdot|\text{not switch})+u_2(\cdot|\text{not switch})]}},$$

- Several details specific to Schelling's SM. Structure of stochastically stable states.

Back to Potential Games and Peyton-Young

- Second P-Y: how convergence time relates to network structure.

Back to Potential Games and Peyton-Young

- Second P-Y: how convergence time relates to network structure.
- Given a graph G, a nonempty subset S of vertices and a real number $0 \leq r \leq 1/2$ we say that *S is r-close-knit* if

$$\forall S' \subseteq S, S' \neq \emptyset, \quad \frac{e(S', S)}{\sum_{i \in S'} deg(i)} \geq r,$$

where $e(S', S)$ is the number of edges with one endpoint in S' and the other in S, and $deg(i)$ is the degree of vertex i. A graph G is *(r, k)-close-knit* if every vertex is part of a r-close-knit set S, with $|S| = k$.

Back to Potential Games and Peyton-Young

- Second P-Y: how convergence time relates to network structure.
- Given a graph G, a nonempty subset S of vertices and a real number $0 \leq r \leq 1/2$ we say that *S is r-close-knit* if

$$\forall S' \subseteq S, S' \neq \emptyset, \quad \frac{e(S', S)}{\sum_{i \in S'} deg(i)} \geq r,$$

  where $e(S', S)$ is the number of edges with one endpoint in S' and the other in S, and $deg(i)$ is the degree of vertex i. A graph G is *(r, k)-close-knit* if every vertex is part of a r-close-knit set S, with $|S| = k$.
- Does not extend to Markovian contagion: *line graph* L_{2n+1} on $2n+1$ nodes labeled $-n, \ldots, -1, 0, 1 \ldots n$. Random walk from the origin. Convergence time $\theta(n^2)$.

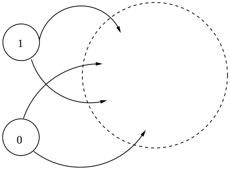
Back to Potential Games and Peyton-Young

- Second P-Y: how convergence time relates to network structure.
- Given a graph G, a nonempty subset S of vertices and a real number $0 \leq r \leq 1/2$ we say that *S is r-close-knit* if

$$\forall S' \subseteq S, S' \neq \emptyset, \quad \frac{e(S', S)}{\sum_{i \in S'} deg(i)} \geq r,$$

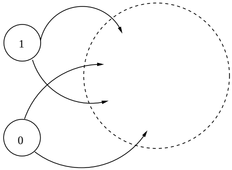
where $e(S', S)$ is the number of edges with one endpoint in S' and the other in S, and $deg(i)$ is the degree of vertex i. A graph G is *(r, k)-close-knit* if every vertex is part of a r-close-knit set S, with $|S| = k$.
- Does not extend to Markovian contagion: *line graph* L_{2n+1} on $2n + 1$ nodes labeled $-n, \ldots, -1, 0, 1 \ldots n$. Random walk from the origin. Convergence time $\theta(n^2)$.
- Generalization: parameter called *Matthews bound* for Markov chains. Upper bounds on convergence time.

Extension: work in progress



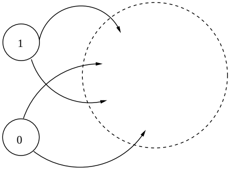
- Markovian contagion: only one agent able to update.

Extension: work in progress



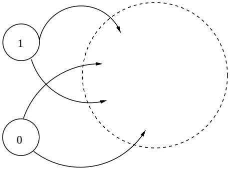
- Markovian contagion: only one agent able to update.
- Extension: *influence model* (Asavathiratham, 2000).

Extension: work in progress



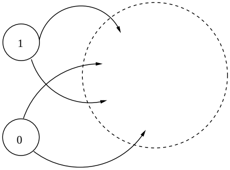
- Markovian contagion: only one agent able to update.
- Extension: *influence model* (Asavathiratham, 2000).
- "Evil-rain" model (binary influence).

Extension: work in progress



- Markovian contagion: only one agent able to update.
- Extension: *influence model* (Asavathiratham, 2000).
- "Evil-rain" model (binary influence).
- Two sites inject 0/1 in the system. Ones perform random walk until deleted by injected 0.

Extension: work in progress



- Markovian contagion: only one agent able to update.
- Extension: *influence model* (Asavathiratham, 2000).
- "Evil-rain" model (binary influence).
- Two sites inject 0/1 in the system. Ones perform random walk until deleted by injected 0.
- Markovian model: $V^{\{A,B\}\times\{0,1\}}$.
- **Stochastic stability $\equiv$ "all A" states.**

Can this be done in Social Simulations ?

- I believe: to a limited extent yes.

Can this be done in Social Simulations ?

- I believe: to a limited extent yes.
- Similar to *model checking* paradigm in hardware verification (e.g. Clarke, Grumberg, Peled).
- Needed: specification language for social simulation.

Can this be done in Social Simulations ?

- I believe: to a limited extent yes.
- Similar to *model checking* paradigm in hardware verification (e.g. Clarke, Grumberg, Peled).
- Needed: specification language for social simulation.
- Abstract State Machines (Gläesser). Computational Criminology.

Can this be done in Social Simulations ?

- I believe: to a limited extent yes.
- Similar to *model checking* paradigm in hardware verification (e.g. Clarke, Grumberg, Peled).
- Needed: specification language for social simulation.
- Abstract State Machines (Gläesser). Computational Criminology.
- Ingredient: contexts as first-class objects. Situation Theory (Devlin)

Can this be done in Social Simulations ?

- I believe: to a limited extent yes.
- Similar to *model checking* paradigm in hardware verification (e.g. Clarke, Grumberg, Peled).
- Needed: specification language for social simulation.
- Abstract State Machines (Gläesser). Computational Criminology.
- Ingredient: contexts as first-class objects. Situation Theory (Devlin)
- Express (some form of) causality.

Can this be done in Social Simulations ?

- I believe: to a limited extent yes.
- Similar to *model checking* paradigm in hardware verification (e.g. Clarke, Grumberg, Peled).
- Needed: specification language for social simulation.
- Abstract State Machines (Gläesser). Computational Criminology.
- Ingredient: contexts as first-class objects. Situation Theory (Devlin)
- Express (some form of) causality.
- Ingredient: processes as first-class objects. *Executable* specification to make model-checking/monitoring tractable.