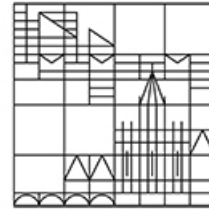


Fachbereich Informatik und
Informationswissenschaft

Universität
Konstanz



Lecture Notes Network Dynamics

taught in Winter terms 2010, 2011

by

Sven Kosub

April 27, 2012

Version v1.0

Contents

1	Case Study: Transportation Networks	1
1.1	The Braess Paradox	1
1.2	A Cell-based Model	4
1.3	Simulation by Synchronous Updates	5
1.4	Simulation by Sequential Updates	7
1.5	Update Order Dependencies	9
2	Networks	11
2.1	Networks in a Static Perspective	11
2.1.1	Population	11
2.1.2	Structure	11
2.1.3	Constraint	12
2.1.4	System	12
2.2	Networks in a Dynamical Perspective	13
2.2.1	Process	13
2.2.2	Trajectory	13
2.2.3	Global State Dynamic	14
2.2.4	Local State Dynamic	14
3	Ensembles	17
3.1	Structure Frameworks	18
3.1.1	Closure Properties	19
3.1.2	Forbidden Minors	22
3.1.3	Bounded Treewidth	24
3.1.4	Bounded Degree	24

3.2	Transition Frameworks	25
3.2.1	Closure Properties	25
3.2.2	Boolean Clones	27
3.2.3	Polymorphisms	31
3.3	Schedule Frameworks	32
3.4	More Ensembles	32
4	Phase Spaces	35
4.3	Similarity	35
4.3.1	Functional Equivalence	35
4.3.2	Black-Box Equivalence	42
4.3.3	Dynamical Equivalence	46
4.4	Features	47
4.4.1	Chaos	47
5	Simulation	53
5.1	Fixed Points	53
5.1.1	Boolean Ensembles	54
A	Mathematical Tools	65
A.1	Sets and relations	65
A.2	Graph theory	67
A.3	Algorithmics	69
	Bibliography	70

Case Study: Transportation Networks

1

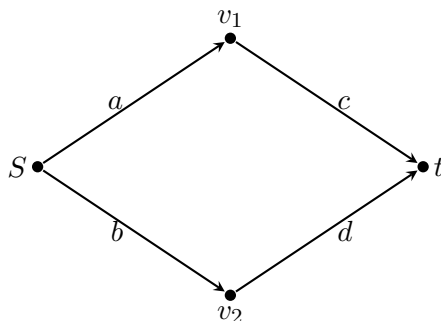
1.1 The Braess Paradox

Introduced in 1968 by Dietrich Braess. Exemplifies that a new best alternative for each individual can make the overall situation worse.

Definition 1.1 A (non-atomic) **congestion model** is a tuple $(A, F, (S_i)_{i \in A}, W)$ such that

- (i) $A = \{1, \dots, k\}$ is a finite, non-empty set of **traveller (agent) types**.
- (ii) F is a finite, non-empty set of **roads (facilities)**.
- (iii) $S_i \subseteq \mathcal{P}(F)$ is a non-empty set of **pathways (strategies)** for $i \in A$.
- (iv) for each $f \in F$, $w_f : [0, 1] \rightarrow \mathbb{R}_+$ is a **cost function**, i.e., if a fraction $x \in [0, 1]$ of travellers choose road $f \in F$ then each traveller costs $w_f(x)$.

Consider the following congestion model (traffic scenario):



- roads: $F = \{a, b, c, d\}$
- pathways: $S = \{\{a, c\}, \{b, d\}\}$
- cost functions:

$w_a(x) = x$	$w_b(x) = 1$
$w_c(x) = 1$	$w_d(x) = x$

Note that $A = \{1\}$, i.e., only one type of travellers.

A **traffic assignment (strategy distribution)** is a mapping

$$x : S_1 \cup \dots \cup S_k \rightarrow [0, 1] \text{ s.t. } \sum_{\substack{s \in S_i \\ \text{for some } i}} x(s) = 1.$$

Given a traffic assignment x :

- *congestion* $\mu_f(x)$ of $f \in F$ is defined by

$$\mu_f(x) =_{\text{def}} \sum_{i=1}^k \sum_{\substack{s \in S_i \\ f \in s}} x(s)$$

- *individual (private) cost* $c_s(x)$ of travellers of type $i \in A$ using pathway $s \in S_i$ is defined by

$$c_s(x) =_{\text{def}} \sum_{f \in s} w_f(\mu_f(x))$$

- *social cost* $C(x)$ is defined by

$$C(x) =_{\text{def}} \sum_{i=1}^k \sum_{s \in S_i} c_s(x) \cdot x(s)$$

In our traffic scenario, consider traffic assignment

$$x(\{a, c\}) = x(\{b, d\}) = 1/2$$

- congestion:

$$\mu_a(x) = \mu_b(x) = \mu_c(x) = \mu_d(x) = 1/2$$

- individual cost:

$$\begin{aligned} c_{\{a,c\}}(x) &= 1/2 + 1 = 3/2 \\ c_{\{b,d\}}(x) &= 1 + 1/2 = 3/2 \end{aligned}$$

- social cost:

$$C(x) = 3/2 \cdot 1/2 + 3/2 \cdot 1/2 = 3/2$$

- traffic assignment incurs minimum social cost.

(A traffic assignment x^* is called *social optimum* if $C(x^*) \leq C(x)$ for all traffic assignments x .)

Definition 1.2 Let (a, F, S, w) be a congestion model.

A traffic assignment x is called **Wardrop equilibrium** if, and only if for all $i \in A$ and for all $s, s' \in S_i$ such that $x(s) > 0$.

$$c_s(x) \leq c_{s'}(x)$$

Note that the following is equivalent:

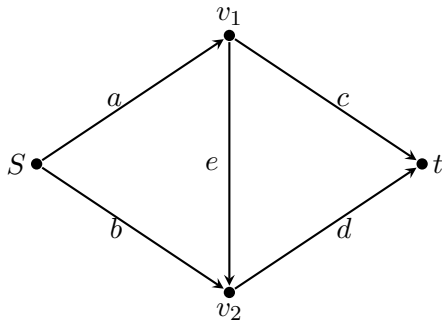
- (1) x is a Wardrop equilibrium
- (2) x is a (non-atomic) Nash equilibrium, i.e., for all $i \in A$, $\varepsilon > 0$ and for all $s, s' \in S_i$, it holds that

$$c_s(x) \leq c_{s'}(\tilde{x}), \text{ where for } s'' \in S_i$$

$$\tilde{x}(s'') = \begin{cases} x(s) - \varepsilon & s'' = s \\ x(s') + \varepsilon & s'' = s' \\ x(s'') & \text{otherwise} \end{cases}$$

In our scenario, $x(\{a, c\}) = x(\{b, d\}) = \frac{1}{2}$ is a Wardrop equilibrium.

Now, we introduce a new road between v_1, v_2 at cost 0!



- roads: $F = \{a, b, c, d, e\}$
- pathways:
 $S = \{\{a, c\}, \{a, e, d\}, \{b, d\}\}$
- cost functions:
 $c_a(x) = x$ $c_b(x) = 1$
 $c_c(x) = 1$ $c_d(x) = x$
 $c_e(x) = 0$

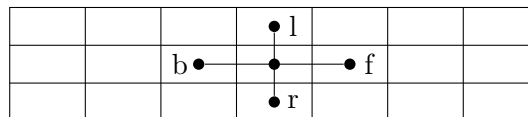
Wardrop equilibrium: $x(\{a, c\}) = x(\{b, d\}) = 0, x(\{a, e, d\}) = 1$
(any deviation to another pathway incurs cost 2)

Social cost: $C(x) = 1 \cdot 1 + 0 \cdot 1 + 1 \cdot 1 = 2$

1.2 A Cell-based Model

- large-scale computer simulation to analyze traffic flow of a resolution level of individual travellers, e.g., TRANSIMS of LANL → El Paso TRANSIMS Case Study
- Components:
 - (1) a population
 - (2) a location-based activity plan for each person
 - (3) a network description of all transportation pathways
- Information acquisition:
 - (1) extensive surveys
 - (2) extensive surveys
 - (3) complete description of roadway, walkways, public transportation etc.
- Simulation modules:
 - (1) Router: maps each activity plan to a path through network inclusive several modes of transportation
 - (2) Micro-simulator: executes travel plans in accordance with interdependent travel plans; typically on a 1-second time scale respecting driving rules, road signalling, fellow travellers, ...
- Simulation methodology:

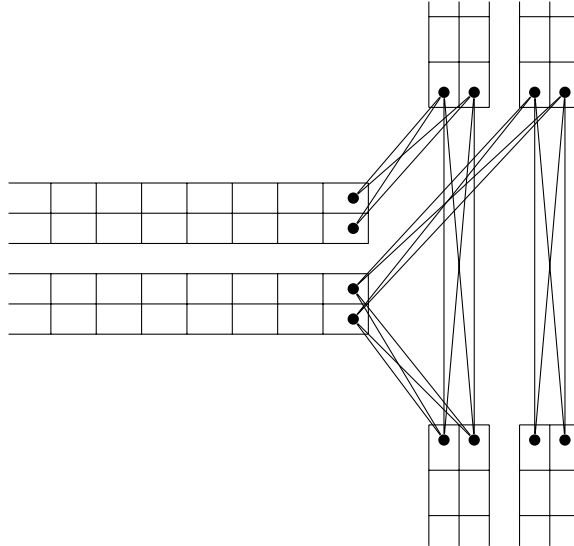
Compare travel times obtained by simulation with surveys; if too high, reroute a decreasing fraction of travellers and iterate simulation.
- Micro-simulator is defined as a local state dynamic:
 - (1) Road-network representation:
 - consists of nodes and links: nodes are, e.g., intersection, lane merging points; links connect nodes
 - each lane is discretized into cells
 - each cell has up to four neighbors (front, left, back, right):



cell i

- a cell can hold at most one vehicle

- link connectivity is specified, e.g., as



- (2) vehicle dynamics:

- discrete velocities: $0, 1, 2, 3, 4, 5, \dots$ measured in cells per update time step (e.g., cell length 7.5 m, i.e., 5 corresponds to $5 \cdot 7.5 \text{ m/s} = 37.5 \text{ m/s} = 135 \text{ km/h}$)

- (3) Update hierarchy:

- Φ_1 lane change decision
- Φ_2 lane change execution
- Φ_3 acceleration / deceleration
- Φ_4 movement
- overall computation is given by $\Phi_4 \circ \Phi_3 \circ \Phi_2 \circ \Phi_1$

1.3 Simulation by Synchronous Updates

We focus on velocity (Φ_3) / position (Φ_4) updates on a circuit (passing forbidden)

Formal specification is as follows:

- $X = \{0, 1, \dots, n - 1\}$ set of cells
- $D = \{\square, 0, 1, \dots, v_{\max}\}$ set of possible states for each cell
- $E = \text{Circ}_n = (X, \{(i, i+1 \bmod n) \mid i \in X\})$ is the network
- D^n is the set of possible configurations

Local state dynamic for acceleration (for $i \in X$):

$$f_i : D^n \rightarrow D : (x_0, \dots, x_{n-1}) \mapsto \begin{cases} \square & \text{if } x_i = \square \\ \min(x_{i+1}, v_{\max}, \Delta(i)) & \text{otherwise} \end{cases}$$

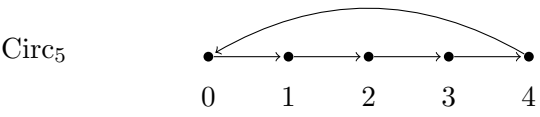
where $\Delta(i)$ is free space in front of cell i .

Local state dynamic for movement (for $i \in X$):

$$g_i : D^n \rightarrow D : (x_0, \dots, x_{n-1}) \mapsto \begin{cases} \square & \text{if } x_i > 0 \\ \delta(i) & \text{if } x_i = \square \text{ and } x_i - \delta(i) = \delta(i) \\ x_i & \text{otherwise} \end{cases}$$

where $\delta(i) - 1$ is free space behind cell i .

Example: $n = 5$, $v_{\max} = 1$



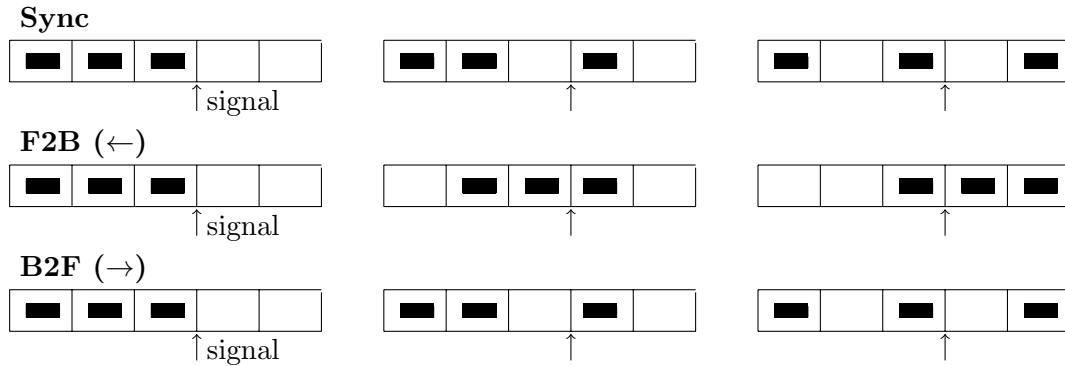
Time	States					Type
t=0	0	0	0	\square	\square	position
	0	0	1	\square	\square	velocity
t=1	0	0	\square	1	\square	position
	0	1	\square	1	\square	velocity
t=2	0	\square	1	\square	1	position
	1	\square	1	\square	0	velocity
t=3	\square	1	\square	1	0	position
	\square	1	\square	0	1	velocity
t=4	1	\square	1	0	\square	position
t=5	\square	1	0	\square	1	position
t=6	1	0	\square	1	\square	position
t=7	0	\square	1	\square	1	position
\vdots	 equal to $t=2$					

Amount of resources:

- $\mathcal{O}(n)$ state accesses (time) per update
- space is double of the size of simulation (i.e., $\mathcal{O}(n)$ additional space)

1.4 Simulation by Sequential Updates

Situation: vehicles waiting for green light



Observation:

- synchronous schedules show sequential semantics
- synchronous schedules not necessary
- (quasi-)sequential schedules save space

Front-to-back schedule:

In each update step, only $v_{\max} + 1$ cells are updated, i.e., a constant number

$$\alpha : t \mapsto \{n - t \bmod n, \dots, n - t - v_{\max} \bmod n\}$$

Simulation proceeds in phases:

- a phase is complete if each cell has been updated $v_{\max} + 1$ times
- a phase “corresponds” to one step in the synchronous case

Example: $n = 5$, $v_{\max} = 1$, front-to-back schedule

Time	States					Type
t=0	0	0	0	□	□	position
				↓	↓	
t=1	0	0	0	□	□	velocity
				↓	↓	
	0	0	0	□	□	position
t=2	0	0	1	□	□	velocity
				↓	↓	
	0	0	□	1	□	position
t=3	0	□	1	1	□	position
t=4	□	1	1	1	□	position
t=5	□	1	1	1	□	position
t=6	□	1	1	□	1	position
⋮						

Amount of resources:

- each update needs $v_{\max} + 1$ state accesses, i.e., $n \cdot (v_{\max} + 1)^2 = \mathcal{O}(n)$ time per phase
- each update needs $v_{\max} + 1$ additional variables, i.e., $v_{\max} + 1 = \mathcal{O}(1)$ additional space

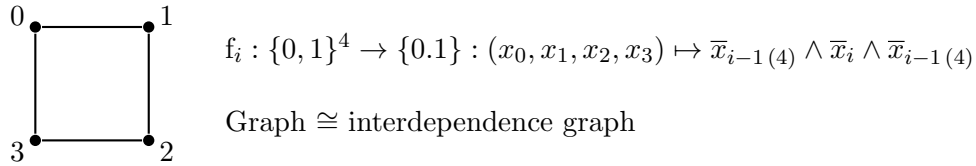
Fundamental question (for the theory of computer simulation)

Which (quasi-)sequential update schedules are “optimal” with respect to adequacy and cost?

1.5 Update Order Dependencies

How many different simulation behaviors can be expected for different schedules?

We are given the following (simplified, idealized) simulation:

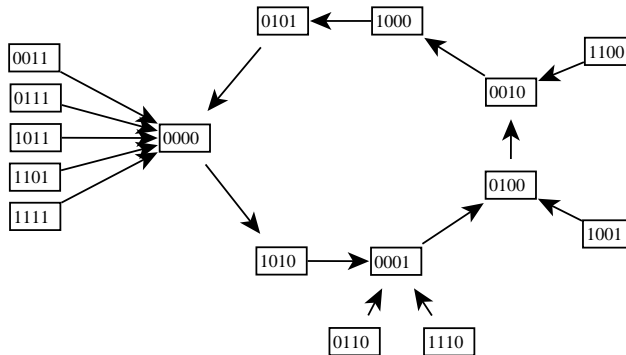


We only consider sequential updates represented by permutations $\pi : \{0, 1, 2, 3\} \rightarrow \{0, 1, 2, 3\}$, i.e., in time step $t \in \mathbb{N}_+$, actor $\pi(\text{mod}(t - 1, 4))$ updates in state.

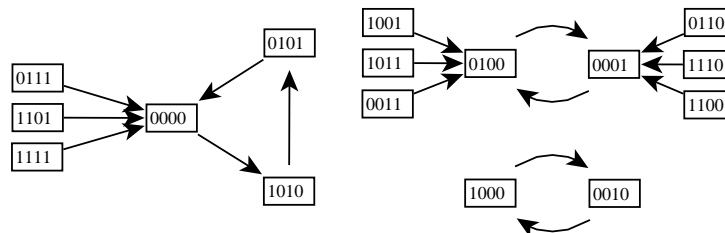
Examples:

- $\pi_1 = \begin{pmatrix} 0 & 1 & 2 & 3 \\ 0 & 1 & 3 & 2 \end{pmatrix}$, i.e., update sequence is 0, 1, 2, 3, 0, 1, 2, 3, ...

We consider phases consisting of 4 time steps:



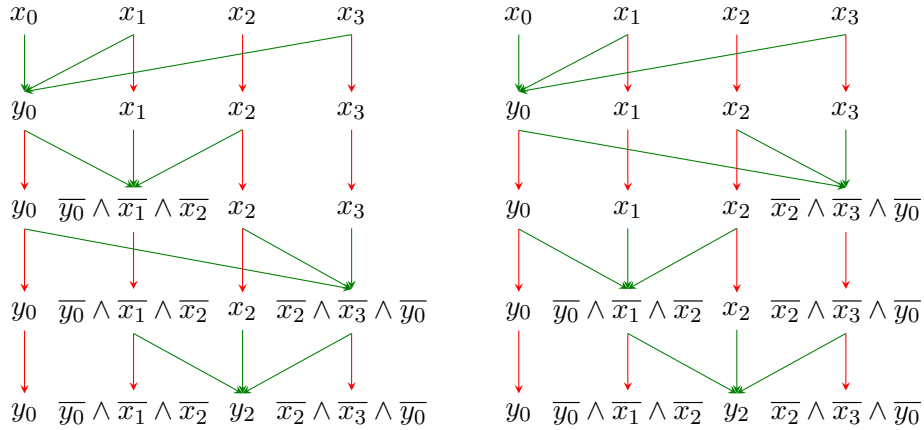
- $\pi_2 = \begin{pmatrix} 0 & 1 & 2 & 3 \\ 0 & 3 & 1 & 2 \end{pmatrix}$



(phase space = complete information on simulation behavior)

When do permutations induce the same phase space?

Example: $\pi_1 = \begin{pmatrix} 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \end{pmatrix}$, $\pi_2 = \begin{pmatrix} 0 & 1 & 2 & 3 \\ 0 & 2 & 1 & 3 \end{pmatrix}$



Observation: Given a graph $G = (V, E)$, two permutations $\pi, \pi' : V \rightarrow V$ induce the same phase space if there is a k such that $\{\pi(k), \pi(k+1)\} \notin E$ and $\pi(i) = \pi'(i)$ for $i \notin \{k, k+1\}$.

Example: For $\pi : \{0, 1, 2, 3\} \rightarrow \{0, 1, 2, 3\}$ and Circ_4 , this is true for:

- $(\underline{0}, \underline{2}, \underline{1}, \underline{3})$, $(2, 0, 1, 3)$, $(2, 0, 3, 1)$, $(0, 2, 3, 1)$
- $(\underline{1}, \underline{3}, \underline{0}, \underline{2})$, $(3, 1, 0, 2)$, $(3, 1, 2, 0)$, $(1, 3, 2, 0)$
- $(0, \underline{1}, \underline{3}, \underline{2})$, $(0, 3, 1, 2)$
- $(2, \underline{1}, \underline{3}, \underline{0})$, $(2, 3, 1, 0)$
- $(1, \underline{0}, \underline{2}, \underline{3})$, $(1, 2, 0, 3)$
- $(3, \underline{0}, \underline{2}, \underline{1})$, $(3, 2, 0, 1)$

That is, $\leq 24 - 16 + 6 = 14$ permutations do not possess the above property, i.e., there are at most 14 different sequential simulation behaviors.

2.1 Networks in a Static Perspective

2.1.1 Population

An *actor* x is any variable with values ranging in a set D called *attribute type*. If x takes on a value $z \in D$, then z is called *state* of x .

Definition 2.1

1. A **population** $X = \{x_1, \dots, x_n\}$ is a finite set of actors x_1, \dots, x_n with attribute types D_1, \dots, D_n .
2. A population $X = \{x_1, \dots, x_n\}$ with attribute types D_1, \dots, D_n is said to be **homogeneous** of attribute type D if $D = D_1 = \dots = D_n$.

In the following we restrict ourselves to homogeneous populations. Therefore we omit the word “homogeneous.”

Let $X = \{x_1, \dots, x_n\}$ be a population with attribute type D . A *configuration* (assignment, interpretation) is a mapping $I : X \rightarrow D$, i.e. a configuration I assigns a state $I(x) \in D$ to each actor $x \in X$. As an alternative notation we also refer to a tuple $(z_1, \dots, z_n) \in D^n$ such that $z_i = I(x_i)$ as a configuration.

2.1.2 Structure

The fundamental relation in network analysis is the *dyad*. A dyad relates two actors of a population. We use graph theory to describe dyadic structures of populations.

Definition 2.2 Let $X = \{x_1, \dots, x_n\}$ be a population of attribute type D .

1. A **structure** is a set $E \subseteq X \times X$.
2. The elements of $X \times X$ are called **dyads**.
3. The elements of a structure E are called **edges**.

Structures can be directed, undirected, or mixed. They are allowed to have *annotations* (weights) with certain attribute types. That is, a structure E may be equipped with a weight function $w : E \rightarrow A$ where A is the attribute type.

2.1.3 Constraint

Constraints limit the set of possible configurations of populations.

Definition 2.3 Let $X = \{x_1, \dots, x_n\}$ be a population of attribute type D . A **constraint** on X is any relation $R \subseteq D^n$.

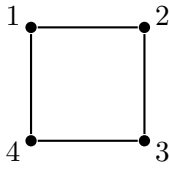
A configuration $I : X \rightarrow D$ is said to be **admissible** with respect to constraint R if and only if $(I(x_1), \dots, I(x_n)) \in R$.

Example: We discuss constraints for some populations and structures.

Let $X = \{1, 2, 3, 4\}$ be a population of type $D = \{0, 1\}$. Let $E = \text{Circ}_4$ be a cycle containing the four actors. Define R to be the following constraint:

$$R =_{\text{def}} \{ (0, 0, 0, 0), (0, 1, 0, 1), (1, 0, 1, 0), (1, 1, 1, 1) \}$$

Then, constraint R corresponds to the set of solution of the following set of equations:

	$x_1 = x_4 \oplus x_1 \oplus x_2$ $x_2 = x_1 \oplus x_2 \oplus x_3$ $x_3 = x_2 \oplus x_3 \oplus x_4$ $x_4 = x_3 \oplus x_4 \oplus x_1$
-------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------

Here, \oplus denotes XOR. Clearly, x_1 does not directly depend on x_3 and x_2 does not directly depend on x_4 . All other dependencies between the variable are represented by an edge in the structure. Therefore, we call E the interdependence structure for R .

2.1.4 System

Definition 2.4 Let D be an attribute type.

A (homogeneous) **system** S is a triple (X, E, R) such that X is a population of attribute type D , E is a structure on X , and R is a constraint on X .

Example:

The triple $(\{1, 2, 3, 4\}, \text{Circ}_4, \{(0, 0, 0, 0), (0, 1, 0, 1), (1, 0, 1, 0), (1, 1, 1, 1)\})$ as in the example above is a system.

A **state** of S is any admissible configuration $I : X \rightarrow D$.

2.2 Networks in a Dynamical Perspective

2.2.1 Process

Let $[a, b] =_{\text{def}} \mathbb{N} \cap \{x \mid a \leq x \leq b\}$ for $a, b \in \mathbb{N}$.

We allow b to be ∞ and define $[a, \infty] =_{\text{def}} \mathbb{N} \setminus \{0, \dots, a-1\}$.

Definition 2.5 Let $h \in \mathbb{N} \cup \{\infty\}$.

A **process** P is any finite or infinite sequence $(S_i)_{i \in [0, h]}$ of systems S_i ; h is called **observational horizon** of process P .

We also denote a process by $P = (X_i, E_i, R_i)_{i \in [0, h]}$ when referring to the components of the systems.

Basically, we can identify three (non-excluding) important subtypes of processes:

- A process of type $(X_i, \emptyset, \emptyset)_{i \in [0, h]}$ is called a **population process**.
 - A process of type $(X_i, E_i, \emptyset)_{i \in [0, h]}$ is called a **structure process**.
 - A process of type $(X, E, R_i)_{i \in [0, h]}$ is called a **state process**.
- $\left. \begin{array}{l} \text{population process} \\ \text{structure process} \end{array} \right\} \text{network modelling}$
 $\text{state process} \longrightarrow \text{network dynamics}$

In this course the focus is solely on *state processes*.

2.2.2 Trajectory

Systems running through a process can take several paths of state changes depending, e.g., on initial or environmental conditions. Such paths are called trajectories.

Definition 2.6 Let $P = (X, E, R_i)_{i \in [0, h]}$ be any state process with a population of size $\|X\| = n$.

A **trajectory** is a finite or infinite sequence $(I_j)_{j \in [0, h]}$ of admissible configurations, i.e., $(I_j(x_1), \dots, I_j(x_n)) \in R_j$ for all $j \in [0, h]$.

The set of possible trajectories of a process $(X, E, R_i)_{i \in [0, h]}$ is $\prod_{i=0}^h R_i$ and can thus be, in the case $h = \infty$, uncountable in general, even for a binary attribute type.

2.2.3 Global State Dynamic

A global (deterministic) state dynamic is a mechanism for selecting trajectories of a state process.

Definition 2.7 Let $P = (X, E, R_i)_{i \in [0, h]}$ be a state process of attribute type D and a population of size n .

1. A **global state dynamic** is a mapping $\mathbf{F} : D^n \times [1, h] \rightarrow D^n$.
2. A global state dynamic \mathbf{F} is said to be **compatible with** P if $\mathbf{F}(R_0, t) \subseteq R_t$ for all $t \in [1, h]$.

Example:

Let $P_1 = (X, E, R_i^1)_{i \in [0, h]}$ and $P_2 = (X, E, R_i^2)_{i \in [0, h]}$ be state processes of type $D = \{0, 1\}$, population $X = \{1, 2, 3\}$, structure $E = K^3$, $h = \infty$, and constraints $R_i^1 = D^3$, $R_i^2 = \{(z_1, z_2, z_3) \in D^3 \mid 1 \leq z_1 + z_2 + z_3 \leq 2\}$. Consider the following global state dynamics:

$$\begin{aligned} \mathbf{F}_1 &: (z_1, z_2, z_3, t) \mapsto (1 - z_1, 1 - z_2, 1 - z_3) \\ \mathbf{F}_2 &: (z_1, z_2, z_3, t) \mapsto \begin{cases} (1 - z_1, z_2, z_3) & \text{if } t - 1 \equiv 0 \quad (3) \\ (z_1, 1 - z_2, z_3) & \text{if } t - 1 \equiv 1 \quad (3) \\ (z_1, z_2, 1 - z_3) & \text{if } t - 1 \equiv 2 \quad (3) \end{cases} \end{aligned}$$

Then,

- $\mathbf{F}_1, \mathbf{F}_2$ are global state dynamics.
- \mathbf{F}_1 is compatible with P_1 and P_2 .
- \mathbf{F}_2 is compatible with P_1 ,
but incompatible with P_2 ($\mathbf{F}_2(1, 0, 0, 1) = (0, 0, 0) \notin R_1^2$).

2.2.4 Local State Dynamic

Definition 2.8 Let $P_1 = (X, E, R_i)_{i \in [0, h]}$ be a state process of attribute type D and population size n .

- (1) A **local transition** on X is a mapping $f : D^n \rightarrow D$
- (2) An **update schedule** on X is a mapping $\alpha : [1, h] \rightarrow \mathcal{P}(X)$
- (3) A **local state dynamic** on X is a pair (F, α)
such that $F = \{f_1, \dots, f_n\}$ is a set of local transitions on X , where f_i is associated with actor $x_i \in X$, and α is an update schedule on X .

Example: Consider again P_1 and the following LSD:

$F = \{f_1, f_2, f_3\}$; for $i \in \{1, 2, 3\}$, $z_1, z_2, z_3 \in \{0, 1\}$ define:

$$f_i : \{z_1, z_2, z_3\} \mapsto \begin{cases} z_i & \text{if } z_1 + z_2 + z_3 = 1 \\ 1 - z_i & \text{otherwise} \end{cases}$$

$$\alpha_1 : \mathbb{N}_+ \rightarrow \mathcal{P}(X) : t \mapsto \{1, 2, 3\} \quad \text{synchronous update schedule}$$

$$\alpha_2 : \mathbb{N}_+ \rightarrow \mathcal{P}(X) : t \mapsto \begin{cases} \{2\} & \text{if } t-1 \equiv 0 \quad (3) \\ \{1\} & \text{if } t-1 \equiv 1 \quad (3) \\ \{3\} & \text{if } t-1 \equiv 2 \quad (3) \end{cases} \quad \text{sequential update schedule}$$

(As a sequence: 2 1 3 2 1 3 2 1 3 ...)

An LSD induces a GSD.

Definition 2.9 Let $P = (X, E, R_i)_{i \in [0, h]}$ be a state process of attribute type D and population size n . Let (F, α) be a local state dynamic on X .

- (1) For each actor $x_i \in X$ and for each subset $U \subseteq X$ of actors, **activity function** $\varphi_i[U]$ is defined for configuration $\vec{z} = (z_1, \dots, z_n) \in D^n$ by

$$\varphi_i[U](\vec{z}) =_{\text{def}} \begin{cases} f_i(z_1, \dots, z_n) & \text{if } x_i \in U \\ z_i & \text{if } x_i \notin U \end{cases}$$

- (2) For each set $U \subseteq X$, the **global transition (function)** $\mathbf{F}_F[U] : D^n \rightarrow D^n$ is defined for configuration $\vec{z} = (z_1, \dots, z_n)$ by

$$\mathbf{F}_F[U](\vec{z}) =_{\text{def}} (\varphi_1[U](\vec{z}), \dots, \varphi_n[U](\vec{z}))$$

- (3) The global state dynamic $\mathbf{F}_{(F, \alpha)} : D^n \times [1, h] \rightarrow D^n$ **induced by the local state dynamic** (F, α) is defined for $t \in [1, h]$ by

$$\mathbf{F}_{(F, \alpha)}(\cdot, t) =_{\text{def}} \left(\prod_{k=1}^t \mathbf{F}_F[\alpha(k)] \right) (\cdot),$$

i.e., by the composition of global transitions specified by the update schedule.

Note that $f \cdot g$ is the function defined by $f \cdot g : x \mapsto g(f(x))$.

The following shall elucidate that in detail:

$$\begin{aligned} \left(\prod_{k=1}^3 \mathbf{F}_F[\alpha(k)] \right) (\vec{z}) &= \mathbf{F}_F[\alpha(3)] \cdot \mathbf{F}_F[\alpha(2)] \cdot \mathbf{F}_F[\alpha(1)](\vec{z}) \\ &= \mathbf{F}_F[\alpha(3)] \left(\mathbf{F}_F[\alpha(2)] \left(\mathbf{F}_F[\alpha(1)](\vec{z}) \right) \right) \end{aligned}$$

Example continued: Let $U_1 = \{1, 2\}$, $U_2 = \{1, 2, 3\}$. Then it holds:

$$\begin{array}{lll} \varphi_1[U_1] = f_1 & \varphi_2[U_1] = f_2 & \varphi_3[U_1] = \text{id}_3 \\ \varphi_1[U_2] = f_1 & \varphi_2[U_2] = f_2 & \varphi_3[U_2] = f_3 \end{array}$$

The global transition function is as follows:

$$\mathbf{F}_F[U_1](z_1, z_2, z_3) = (f_1(z_1, z_2, z_3), f_2(z_1, z_2, z_3), z_3), \text{ e.g.,}$$

$$\mathbf{F}_F[U_1](1, 1, 1) = (0, 0, 1) \quad \mathbf{F}_F[U_1](1, 0, 1) = (0, 1, 1) \quad \mathbf{F}_F[U_1](0, 0, 1) = (0, 0, 1)$$

$$\mathbf{F}_F[U_2](1, 1, 1) = (0, 0, 0) \quad \mathbf{F}_F[U_2](0, 0, 0) = (1, 1, 1)$$

The GSD induced by (F, α_1) and (F, α_2) are as follows:

$\vec{z} \backslash t \rightarrow a_1(t)$	1	2	3	4	5
	{1, 2, 3}	{1, 2, 3}	{1, 2, 3}	{1, 2, 3}	{1, 2, 3}
000	111	000	111	000	111
001	001	001	001	001	001
010	010	010	010	010	010
011	100	100	100	100	100
100	100	100	100	100	100
101	010	010	010	010	010
110	001	001	001	001	001
111	000	111	000	111	000

$\vec{z} \backslash t \rightarrow a_2(t)$	1	2	3	4	5
	2	1	3	2	1
000	010	010	010	010	010
001	001	001	001	001	001
010	010	010	010	010	010
011	001	001	001	001	001
100	100	100	100	100	100
101	111	011	010	010	010
110	100	100	100	100	100
111	101	001	001	001	001

That is,

- $\mathbf{F}_{(F, \alpha_2)}(\vec{z}, t) = \mathbf{F}_{(F, \alpha_2)}(\vec{z}, 3)$ for $t \geq 3$
- each trajectory of (\mathbf{F}, α_2) reaches a fixed point
- $\mathbf{F}_{(F, \alpha_1)}$ generates oscillations
- each permutation on $\{1, 2, 3\}$ has the same effect as permutation $(2, 1, 3)$

Let $P = (X, E, R)_{i \in [0, \infty]}$ be a state process of attribute type D and population size $n, R = D^n$.

Let (F, α) be an LSD on X such that the induced GSD is compatible with P (i.e., $\alpha : \mathbb{N}_+ \rightarrow \mathcal{P}(X)$).

Then, (X, E) is the *interdependence graph* of F if, and only if for all $f_i \in F$,

$$N_E(x_i) = \{x_j \mid x_i \text{ depends on } x_j \text{ in } f_i\} \setminus \{x_i\}$$

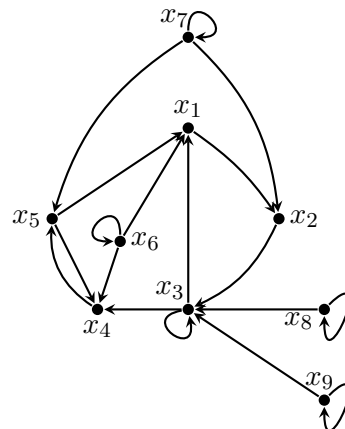
where,

- $N_E(x_i) = \{x_j \mid \{x_i, x_j\} \in E\}$ is the neighborhood of x_i
- x_i depends on x_j in $f_i \iff_{\text{def}} x_j$ is not fictive in f_i ;
a variable x_j is fictive in f_i if, and only if

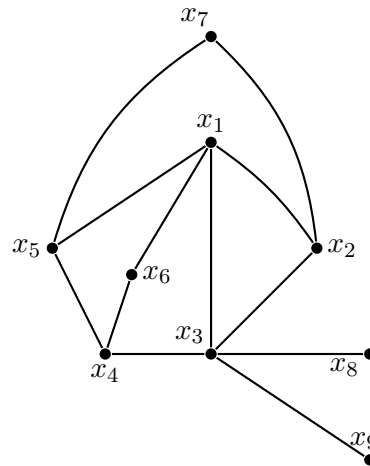
$$\begin{aligned} &\text{for all } z_1, \dots, z_{j-1}, z_{j+1}, \dots, z_n \in D, z_j, z'_j \in D, \\ &f_i(z_1, \dots, z_{j-1}, z_j, z_{j+1}, \dots, z_n) = f_i(z_1, \dots, z_{j-1}, z'_j, z_{j+1}, \dots, z_n) \end{aligned}$$

Example: Tryptophan operon, $D = \{0, 1\}$

$$\begin{aligned} f_1 &: (x_1, \dots, x_9) \mapsto x_3 \wedge \overline{x_5} \wedge x_6 \\ f_2 &: (x_1, \dots, x_9) \mapsto x_1 \wedge x_7 \\ f_3 &: (x_1, \dots, x_9) \mapsto x_2 \wedge \overline{x_3} \wedge x_8 \wedge x_9 \\ f_4 &: (x_1, \dots, x_9) \mapsto \overline{x_3} \wedge \overline{x_5} \wedge x_6 \\ f_5 &: (x_1, \dots, x_9) \mapsto x_4 \wedge x_7 \\ f_i &: (x_1, \dots, x_9) \mapsto x_i \quad \text{for } i \in \{6, \dots, 9\} \end{aligned}$$



interdependence graph:



Let \mathcal{F} be a class of functions (over domain D).

Let \mathcal{G} be a class of (undirected) graphs.

Let \mathcal{S} be a class of schedules.

An $(\mathcal{F}, \mathcal{G}, \mathcal{S})$ *ensemble* is the set of all LSD (F, α) such that

- $F \leq \mathcal{F}$
- $\alpha \in \mathcal{S}$
- $(X, E) \in \mathcal{G}$ where (X, E) is the interdependence graph of F

Example:

Synchronous tryptophan operon belongs to the (BF, PLANAR, SYNC) ensemble.

3.1 Structure Frameworks

We are interested in graph classes which are closed under the following operations:

- isomorphisms
- vertex deletion
- edge deletion
- edge contraction

3.1.1 Closure Properties

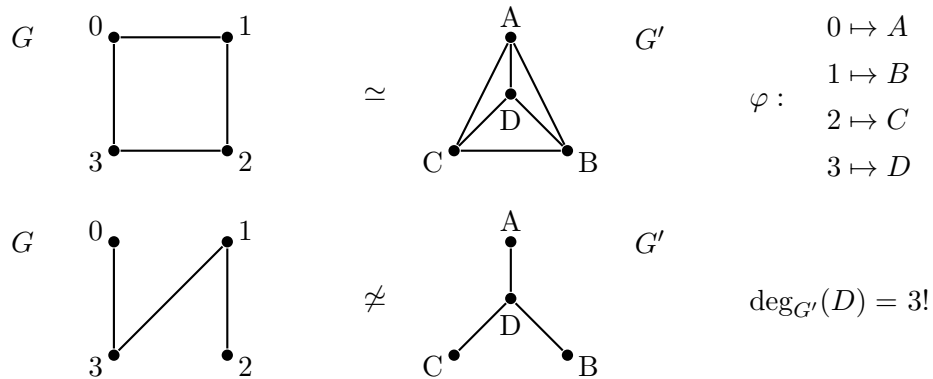
Let $G = (V, E)$ be an undirected graph.

1. Isomorphisms:

Let $G' = (V', E')$ be another undirected graph. Then,

$$G \simeq G' \iff_{\text{def}} \begin{array}{l} \text{there is a bijective mapping } \varphi : V \rightarrow V' \\ \text{such that for all } u, v \in V, \\ \{u, v\} \in E \iff \{\varphi(u), \varphi(x)\} \in E' \end{array}$$

Example:



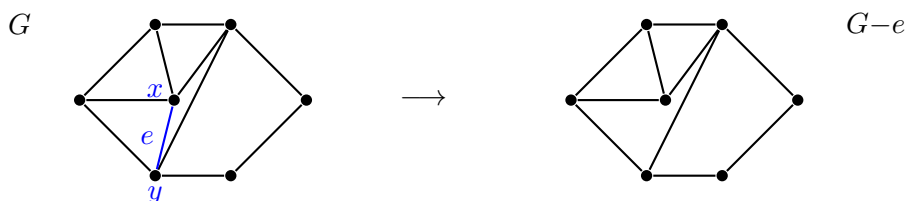
Two graphs are isomorphic if, and only if they can be drawn by the same picture.

2. Edge deletion:

Let $e \in E$ be an edge. Then, the graph $G - e$ obtained by deleting edge e in G is the graph $G' = (V', E')$ such that

$$\begin{array}{l} V' =_{\text{def}} V \\ E' =_{\text{def}} E \setminus \{e\} \end{array}$$

Example:



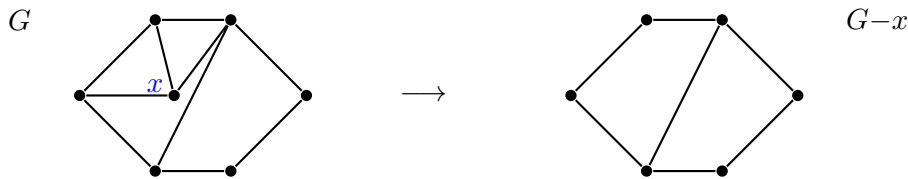
3. Vertex deletion:

Let $v \in V$ be a vertex. Then, the graph $G-v$ obtained by deleting vertex v in G is the graph $G' = (V', E')$ such that

$$V' =_{\text{def}} V \setminus \{v\}$$

$$E' =_{\text{def}} E \setminus \{e \mid e = \{u, v\} \in E \text{ for some } u \in V\}$$

Example:



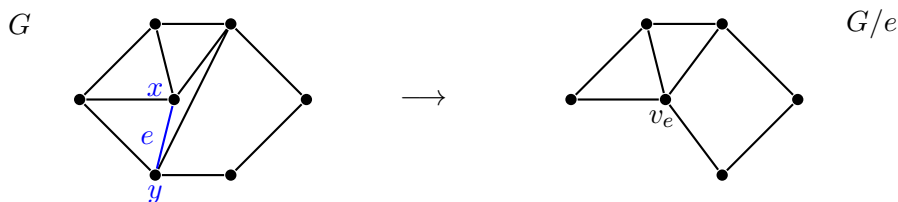
4. Edge contraction:

Let $e = \{x, y\} \in E$ be an edge. Then, the graph G/e obtained by contracting edge e in G is the graph $G' = (V', E')$ such that

$$V' =_{\text{def}} (V \setminus \{x, y\}) \cup \{v_e\} \text{ where } v_e \notin V \cup E$$

$$E' =_{\text{def}} \{ \{v, w\} \mid \{v, w\} \in E, \{v, w\} \cap \{x, y\} = \emptyset \} \\ \cup \{ \{v_e, w\} \mid \{x, w\} \in E \mid \{e\} \text{ or } \{y, w\} \in E \mid \{e\} \}$$

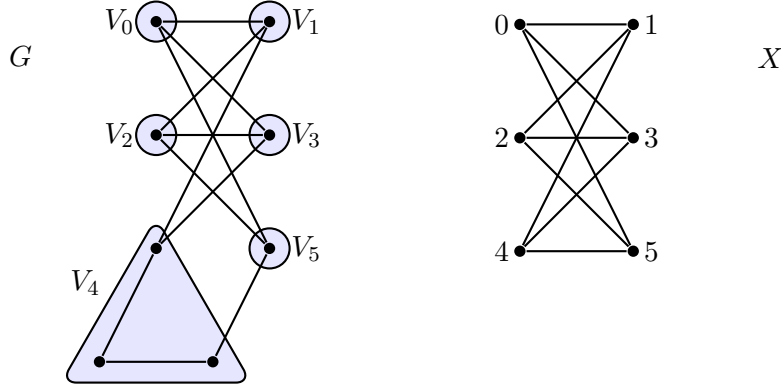
Example:



Let X be any graph and let $\{V_x \mid x \in V(X)\}$ be partition of V into (non-empty) connected subgraphs such that for all $x, y \in V(X)$

$$\{x, y\} \in E(X) \Leftrightarrow \text{there is an edge in } G \text{ connecting some vertex} \\ \text{from } V_x \text{ and some vertex from } V_y.$$

Example:



Then, we say that $G = MX$ (more precisely, $G \in MX$). A set V_x is called *branch set*.

Proposition 3.1 *Let G, X be graphs. Then,*

$$G = MX \Leftrightarrow \text{there exist graphs } G_0, \dots, G_r \text{ and edges } e_i \in E(G_i) \text{ such that } G_0 = G, G_r \simeq X, \text{ and for all } 0 \leq i < r, G_{i+1} = G_i/e_i.$$

Proof: Induction on $d = \|V(G)\| - \|V(X)\| \geq 0$.

- base of induction: Let $d = 0$. That is, $\|V(G)\| = \|V(X)\|$.
 - \Rightarrow Suppose $G = MX$. That is, $\|V_x\| = 1$ for all $x \in V(X)$, and $G \simeq X$. Choose $r = 0$. Then, $G_0 = G \simeq X$.
 - \Leftarrow Suppose G_0, \dots, G_r , edges $e_i \in E(G_i)$ such that $G_0 = G, G_r \simeq X, G_{i+1} = G_i/e_i$. Since $\|V(G_i/e_i)\| = \|V(G_i)\| - 1$, we have $r = 0$. So, $G_0 = G \simeq X$. Hence, $G = MX$.
- induction step: Let $d > 0$. Let G, X be graphs such that $\|V(G)\| = \|V(X)\|$
 - \Rightarrow Suppose $G = MX$. Let $x \in V(X)$ be a vertex such that $\|V_x\| \geq 2$. Then, there are $u, v \in V_x$ such that $e = \{u, v\} \in E(G)$. Consider G/e . It holds, that $G/e = MX$.
By induction hypothesis, there are G_0, \dots, G_r , edges $e_i \in E(G_i)$ such that $G_0 = G/e, G_r \simeq X, G_{i+1} = G_i/e_i$. Set $G'_0 = G, G'_i = G_{i-1}$ for $i > 0, e_0 = e$.
 - \Leftarrow Suppose G_0, \dots, G_r , edges $e_i \in E(G_i)$ s.t. $G_0 = G, G_r \simeq X, G_{i+1} = G_i/e_i$.
By induction hypothesis, $G/e_0 = G_1 = MX$. Assume $v_{e_0} \in V_x$ for $x \in V(X)$. Then, replace v_{e_0} by u, v such that $e_0 = \{u, v\}$.
Hence, $G = MX$. ■

A graph X is said to be a *minor* of a graph Y (i.e., $X \preceq Y$) if and only if there is a $G \subseteq Y$ such that $G = MX$.

A class \mathcal{G} of graphs is *closed under taking minors* if and only if for all graphs $G, G' : G \in \mathcal{G} \wedge G' \preceq G \Rightarrow G' \in \mathcal{G}$.

Proposition 3.2 *The minor relation \preceq is a partial order.*

Proof:

- (1.) Reflexivity: $X \preceq X$
- (2.) Transitivity: $X \preceq Y, Y \preceq Z, X \preceq Z$
- (3.) Antisymmetry: Suppose: $X \preceq Y, Y \preceq X$.
Thus, $\|V(X)\| = \|V(Y)\|$. Hence, $X = MY$ and $X \simeq Y$. ■

3.1.2 Forbidden Minors

Let X be any graph. We define

$$\text{Forb}_{\preceq}(X) =_{\text{def}} \{G \mid G \not\preceq X\}$$

For any set \mathcal{X} of graphs, we define

$$\text{Forb}_{\preceq}(\mathcal{X}) =_{\text{def}} \bigcap_{X \in \mathcal{X}} \text{Forb}_{\preceq}(X)$$

\mathcal{X} is called the set of *forbidden minors* (or *obstruction set*).

Examples:

- $\text{Forb}_{\preceq}(K^3)$ is the class of all forests
- $\text{Forb}_{\preceq}(K^2 \oplus K^2)$ is the class of all graphs where all degrees are incident:
 - stars $K_{1,n}$
 - triangles K^3 $\left. \vphantom{\begin{matrix} - \text{stars } K_{1,n} \\ - \text{triangles } K^3 \end{matrix}} \right\} \oplus \text{isolated vertices}$
- $\text{Forb}_{\preceq}(K^3, K^2 \oplus K^2)$ is the class of all graphs having a vertex cover of size 1 [Cattell & Dinveen 1994]. For $G = (V, E)$, a subset $U \subseteq V$ is a vertex cover of G if, and only if $\{u, v\} \cap U \neq \emptyset$ for all edges $\{u, v\} \in E$.
- $\text{Forb}_{\preceq}(K_{3,3}, K^5)$ is the class of all planar graphs [Kuratowski 1930, Wagner 1937].

- $\text{Forb}_{\preceq}(K^4)$ is the class of all series-parallel graphs.
- $\text{Forb}_{\preceq}(K_{2,3}, K^4)$ is the class of all outerplanar graphs.

Proposition 3.3 *For each set \mathcal{X} , $\text{Forb}_{\preceq}(\mathcal{X})$ is closed under taking minors.*

Proof: Let G, G' be graphs such that $G \in \text{Forb}_{\preceq}(\mathcal{X})$ and $G' \preceq G$. Assume $G' \notin \text{Forb}_{\preceq}(\mathcal{X})$. That is, there is an $X \in \mathcal{X}$ such that $X \preceq G'$. Thus, $X \preceq G$ by transitivity of \preceq . Hence, $G \notin \text{Forb}_{\preceq}(\mathcal{X})$. \downarrow Hence, $G' \in \text{Forb}_{\preceq}(\mathcal{X})$ ■

Proposition 3.4 *Let \mathcal{G} be a class of graphs closed under taking minors. Then, there is a set \mathcal{X} such that $\mathcal{G} = \text{Forb}_{\preceq}(\mathcal{X})$.*

Proof: Define $\mathcal{X} =_{\text{def}} \bar{\mathcal{G}}$. We have to show $\mathcal{G} = \text{Forb}_{\preceq}(\bar{\mathcal{G}})$.

- \supseteq Let $G \in \text{Forb}_{\preceq}(\bar{\mathcal{G}})$. Assume that $G \notin \mathcal{G}$, i.e., $G \in \bar{\mathcal{G}}$. Since $G \notin \text{Forb}_{\preceq}(\mathcal{G})$, we obtain $G \notin \text{Forb}_{\preceq}(\bar{\mathcal{G}})$. \downarrow Hence, $G \in \mathcal{G}$.
- \subseteq Let $G \in \mathcal{G}$. Assume that $G \notin \text{Forb}_{\preceq}(\bar{\mathcal{G}})$, i.e., there is an $H \in \bar{\mathcal{G}}$ such that $H \preceq G$. Since \mathcal{G} is closed under taking minor, $H \in \mathcal{G}$. \downarrow Hence, $G \in \text{Forb}_{\preceq}(\bar{\mathcal{G}})$. ■

Theorem 3.5 (Robertson & Seymour 1986-2004) *Let \mathcal{G} be any class of graphs. Then, \mathcal{G} is closed under taking minors \iff there exists a finite set $\{X_1, \dots, X_r\}$ of graphs such that $\mathcal{G} = \text{Forb}_{\preceq}(X_1, \dots, X_r)$.*

Example: (Warning!)

- class of graphs having vertex cover of size 6 need 260 forbidden minors.
- class of graphs having pathwidth 3 need ~60,000,000 forbidden minors.

Proposition 3.6 *If $X \preceq Y$ then $\text{Forb}_{\preceq}(X) \subseteq \text{Forb}_{\preceq}(Y)$.*

Proof: Let $G \in \text{Forb}_{\preceq}(X)$. Assume $G \notin \text{Forb}_{\preceq}(Y)$, i.e., $Y \preceq G$. Since $X \preceq Y$, $X \preceq G$ by transitivity of \preceq . Thus, $G \notin \text{Forb}_{\preceq}(X)$. \downarrow Hence, $G \in \text{Forb}_{\preceq}(Y)$ ■

3.1.3 Bounded Treewidth

Let $G = (V, E)$ be a graph, let T be a tree, and let $\mathcal{V} = \{V_t\}_{t \in V(T)}$ be a family of vertex sets $V_t \subseteq V(G)$.

(T, \mathcal{V}) is a *tree decomposition* of G if, and only if:

- (1) $V = \bigcup_{t \in V(T)} V_t$.
- (2) for all $e \in E$ there is a $t \in V(T)$ such that $e \subseteq V_t$.
- (3) for all $t_1, t_2, t_3 \in V(T)$ such that t_2 lies on a path from t_1 to t_3 in T , it holds that $V_{t_1} \cap V_{t_3} \subseteq V_{t_2}$.

The *width* of a tree decomposition (T, \mathcal{V}) is defined to be

$$\max\{||V_t|| - 1 \mid t \in V(T)\}.$$

Treewidth $\text{tw}(G)$ of a graph G is defined to be the minimum width of a tree decomposition of G .

Exmample:

Given $G = (V, E)$ and tree T consisting of a vertex. Then, $(T, \{V\})$ is a tree decomposition of G ; width is $n-1$ (for $||V|| = n$).

A class \mathcal{G} of graphs has *bounded treewidth* $\iff_{\text{def}} (\exists k \in \mathbb{N}) (\forall G \in \mathcal{G}) [tw(G) \leq k]$.

Theorem 3.7 (Robertson & Seymour 1986) *Let X be a graph. Then, X is planar $\iff \text{Forb}_{\preceq}(X)$ has bounded treewidth.*

3.1.4 Bounded Degree

A class \mathcal{G} of graphs has *bounded degree* $\iff_{\text{def}} (\exists k \in \mathbb{N}) (\forall G \in \mathcal{G}) [\Delta(G) \leq k]$.

Note that classes of bounded degree need not be closed under taking minors.

Proposition 3.8 *Let X be a graph.*

Then, X has a vertex cover of size one $\iff \text{Forb}_{\preceq}(X)$ has bounded degree.

Proof:

\Rightarrow Suppose X has vertex cover of size one. Then, X consists of some star $K_{1,k}$ and some isolated vertices u_1, \dots, u_r .

Assume $\text{Forb}_{\preceq}(X)$ does not have bounded degree. Thus, there is $G \in \text{Forb}_{\preceq}(X)$ such that $\Delta(G) \geq k + r$, i.e., G contains a subgraph $K_{1,k+r}$.

Hence, $X \preceq K_{1,k+r} \preceq G$. \nmid Therefore, $\text{Forb}_{\preceq}(X)$ has bounded degree.

⇐ Suppose X does not have any vertex cover of size one. That is, X contains a K^3 or a $K^2 \oplus K^2$ as subgraphs.

Consider two classes:

- (i) Let X contain a K^3 , i.e., $K^3 \preceq X$. As $\text{Forb}_{\preceq}(K^3)$ does not have bounded degree (it contains all trees), $\text{Forb}_{\preceq}(X) \supseteq \text{Forb}_{\preceq}(K^3)$ does not have bounded degree.
- (ii) Let X contain a $K^2 \oplus K^2$, i.e., $K^2 \oplus K^2 \preceq X$. As $\text{Forb}_{\preceq}(K^2 \oplus K^2)$ contain for all $k \in \mathbb{N}$ the star $K_{1,k}$, $\text{Forb}_{\preceq}(X) \supseteq \text{Forb}_{\preceq}(K^2 \oplus K^2)$ contains all stars as well. Hence, $\text{Forb}_{\preceq}(X)$ does not have bounded degree. ■

3.2 Transition Frameworks

We are interested in function classes closed under composition.

3.2.1 Closure Properties

Let D be an attribute type.

An n -ary *D-function* f is a mapping $f : D^n \rightarrow D$.

Let \mathcal{F} be any set of D -functions.

- (1) \mathcal{F} is closed under *introduction of fictive variables* if and only if for all $n \in \mathbb{N}$ and all n -ary $f \in \mathcal{F}$, the function

$$\varphi : D^{n+1} \rightarrow D : (x_1, \dots, x_n, x_{n+1}) \mapsto f(x_1, \dots, x_n)$$

belongs to \mathcal{F} .

- (2) \mathcal{F} is closed under *permutations of variables* if and only if for all $n \in \mathbb{N}$ and all n -ary $f \in \mathcal{F}$, and permutation $\pi \in \mathcal{S}$, the function

$$\varphi : D^n \rightarrow D : (x_1, \dots, x_n) \mapsto f(x_{\pi^{-1}(1)}, \dots, x_{\pi^{-1}(n)})$$

belongs to \mathcal{F} .

- (3) \mathcal{F} is closed under *identification of variables* if and only if for all $n \in \mathbb{N}$ and all n -ary $f \in \mathcal{F}$, the function

$$\varphi : D^{n-1} \rightarrow D : (x_1, \dots, x_{n-1}) \mapsto f(x_1, \dots, x_{n-1}, x_{n-1})$$

belongs to \mathcal{F} .

- (4) \mathcal{F} is closed under *substitution* if and only if for all $n, m \in \mathbb{N}$, n -ary $f \in \mathcal{F}$, m -ary $g \in \mathcal{F}$, the function

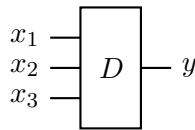
$$\begin{aligned} \varphi : D^{n+m-1} \rightarrow D : (x_1, \dots, x_{n-1}, x_n, \dots, x_{n+m-1}) \\ \mapsto f(x_1, \dots, x_{n-1}, g(x_n, \dots, x_{n+m-1})) \end{aligned}$$

belongs to \mathcal{F} .

For a class \mathcal{F} , define $[\mathcal{F}]$ to be the minimal class containing $\mathcal{F} \cup \{id\}$ which is closed under all operations above; $[\mathcal{F}]$ is called a *clone*.

Example: Suppose we are given a boolean gate (of fan-in 3)

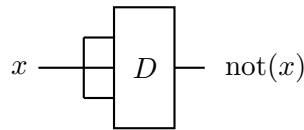
$$D(x_1, x_2, x_3) =_{\text{def}} (x_1 \wedge \bar{x}_2) \vee (x_1 \wedge \bar{x}_3) \vee (\bar{x}_2 \wedge \bar{x}_3)$$



Which functions can be expressed using D and constant 1?

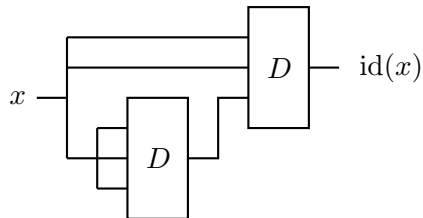
- not: $x \mapsto \bar{x}$

$$D(x, x, x) \equiv (x \wedge \bar{x}) \vee (x \wedge \bar{x}) \vee (\bar{x} \wedge \bar{x}) \equiv \bar{x}$$



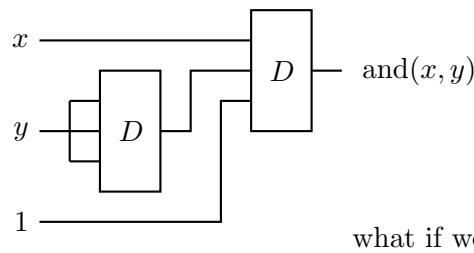
- id: $x \mapsto x$

$$D(x, x, \text{not}(x)) \equiv (x \wedge \bar{x}) \vee (x \wedge x) \vee (\bar{x} \wedge x) \equiv x$$



- and: $(x, y) \mapsto x \wedge y$ (using 1!)

$$D(x, \text{not}(y), 1) \equiv (x \wedge y) \vee (x \wedge 0) \vee (y \wedge 0) \equiv x \wedge y$$



3.2.2 Boolean Clones

Let BF denote the class of all $\{0, 1\}$ -functions (boolean functions).

The 0-ary boolean functions are:

- $c_0 =_{\text{def}} 0$ denoted in formulas by 0
- $c_1 =_{\text{def}} 1$ denoted in formulas by 1

The 1-ary boolean functions (without fictive variables) are:

- $\text{id}(x) = 1 \iff_{\text{def}} x = 1$ denoted in formulas by x
- $\text{not}(x) = 1 \iff_{\text{def}} x \neq 1$ denoted in formulas by $\bar{x}, \neg x$

The most prominent 2-ary boolean functions (without fictive variables) are:

- $\text{and}(x, y) = 1 \iff_{\text{def}} \min(x, y) = 1$ denoted in formulas by \wedge
- $\text{or}(x, y) = 1 \iff_{\text{def}} \max(x, y) = 1$ denoted in formulas by \vee
- $\text{imp}(x, y) = 1 \iff_{\text{def}} x \leq 1$ denoted in formulas by \rightarrow
- $\text{eq}(x, y) = 1 \iff_{\text{def}} x = y$ denoted in formulas by \leftrightarrow
- $\text{xor}(x, y) = 1 \iff_{\text{def}} x \neq y$ denoted in formulas by \oplus

The following properties are important to describe clones:

- For $b \in \{0, 1\}$, a function f is *b-reproducing* $\iff_{\text{def}} f(b, \dots, b) = b$

Examples: ...

- A function f is *monotone* \iff_{def} for all $\vec{x}, \vec{y} \in \{0, 1\}^n$, $\vec{x} \leq \vec{y}$ implies $f(\vec{x}) \leq f(\vec{y})$

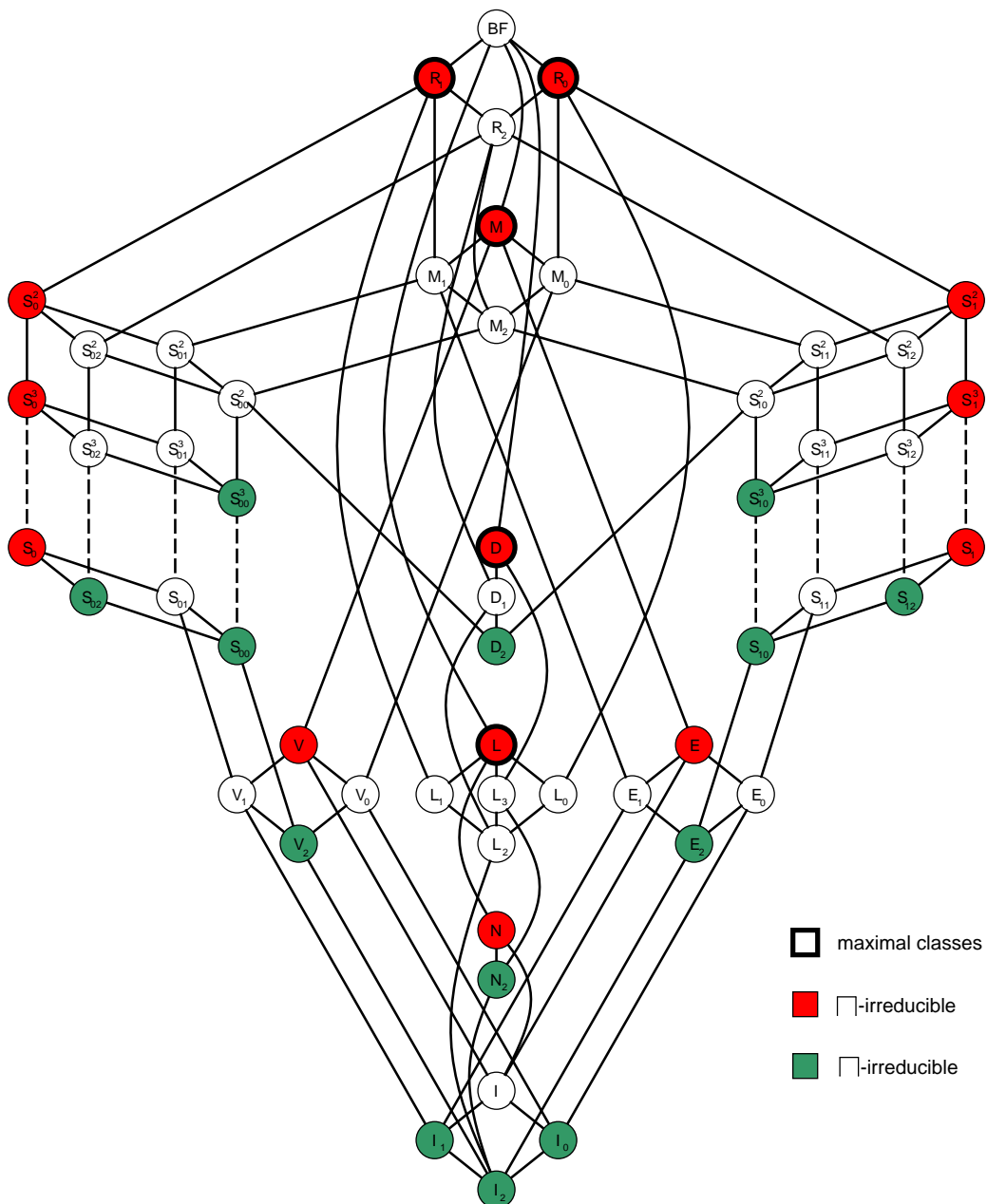
Examples: id , and , or are monotone;
 not , imp , eq , xor are not monotone

- A function f is *selfdual* \iff_{def} for all $(x_1, \dots, x_n) \in \{0, 1\}^n$,
 $f(x_1, \dots, x_n) = \text{not}(f(\text{not}(x_1), \dots, \text{not}(x_n)))$

Examples: id, not are selfdual

there is no selfdual 2-ary function (without fictive variables);

$(x_1 \wedge \bar{x}_2) \vee (x_1 \wedge \bar{x}_3) \vee (\bar{x}_2 \wedge \bar{x}_3)$ denotes a selfdual function.



Class	Definiton	Logical Basis
BF	all boolean functions	$\{\wedge, \neg\}$
R ₀	$\{f \mid f \text{ is 0-reproducing}\}$	$\{\wedge, \oplus\}$
R ₁	$\{f \mid f \text{ is 1-reproducing}\}$	$\{\vee, x \oplus y \oplus 1\}$
R ₂	$R_0 \cap R_1$	$\{\vee, x \wedge (y \oplus z \oplus 1)\}$
M	$\{f \mid f \text{ is monotone}\}$	$\{\wedge, \vee, 0, 1\}$
M ₀	$M \cap R_0$	$\{\wedge, \vee, 0\}$
M ₁	$M \cap R_1$	$\{\wedge, \vee, 1\}$
M ₂	$M \cap R_2$	$\{\wedge, \vee\}$
S ₀ ^k	$\{f \mid f \text{ is 0-separating of degree } k\}$	$\{\rightarrow, \bigwedge_{i=1}^{k+1} (x_1 \vee \dots \vee x_{i-1} \vee x_{i+1} \vee \dots \vee x_{k+1})\}$
S ₀	$\{f \mid f \text{ is 0-separating}\}$	$\{\rightarrow\}$
S ₁ ^k	$\{f \mid f \text{ is 1-separating of degree } k\}$	$\{x \wedge \bar{y}, \bigvee_{i=1}^{k+1} (x_1 \wedge \dots \wedge x_{i-1} \wedge x_{i+1} \wedge \dots \wedge x_{k+1})\}$
S ₁	$\{f \mid f \text{ is 1-separating}\}$	$\{x \wedge \bar{y}\}$
S ₀₂ ^k	$S_0^k \cap R_2$	$\{x \vee (y \wedge \bar{z}), \bigwedge_{i=1}^{k+1} (x_1 \vee \dots \vee x_{i-1} \vee x_{i+1} \vee \dots \vee x_{k+1})\}$
S ₀₂	$S_0 \cap R_2$	$\{x \vee (y \wedge \bar{z})\}$
S ₀₁ ^k	$S_0^k \cap M$	$\{\bigwedge_{i=1}^{k+1} (x_1 \vee \dots \vee x_{i-1} \vee x_{i+1} \vee \dots \vee x_{k+1}), 1\}$
S ₀₁	$S_0 \cap M$	$\{x \vee (y \wedge z), 1\}$
S ₀₀ ^k	$S_0^k \cap R_2 \cap M$	$\{x \vee (y \wedge z), \bigwedge_{i=1}^{k+1} (x_1 \vee \dots \vee x_{i-1} \vee x_{i+1} \vee \dots \vee x_{k+1})\}$
S ₀₀	$S_0 \cap R_2 \cap M$	$\{x \vee (y \wedge z)\}$
S ₁₂ ^k	$S_1^k \cap R_2$	$\{x \wedge (y \vee \bar{z}), \bigvee_{i=1}^{k+1} (x_1 \wedge \dots \wedge x_{i-1} \wedge x_{i+1} \wedge \dots \wedge x_{k+1})\}$
S ₁₂	$S_1 \cap R_2$	$\{x \wedge (y \vee \bar{z})\}$
S ₁₁ ^k	$S_1^k \cap M$	$\{\bigvee_{i=1}^{k+1} (x_1 \wedge \dots \wedge x_{i-1} \wedge x_{i+1} \wedge \dots \wedge x_{k+1}), 0\}$
S ₁₁	$S_1 \cap M$	$\{x \wedge (y \vee z), 0\}$
S ₁₀ ^k	$S_1^k \cap R_2 \cap M$	$\{x \wedge (y \vee z), \bigvee_{i=1}^{k+1} (x_1 \wedge \dots \wedge x_{i-1} \wedge x_{i+1} \wedge \dots \wedge x_{k+1}), 0\}$
S ₁₀	$S_1 \cap R_2 \cap M$	$\{x \wedge (y \vee z)\}$
D	$\{f \mid f \text{ is selfdual}\}$	$\{(x \wedge \bar{y}) \vee (x \wedge \bar{z}) \vee (y \wedge \bar{z})\}$
D ₁	$D \cap R_2$	$\{(x \wedge y) \vee (x \wedge \bar{z}) \vee (y \wedge \bar{z})\}$
D ₂	$D \cap M$	$\{(x \wedge y) \vee (x \wedge z) \vee (y \wedge z)\}$
L	$\{f \mid f \text{ is linear}\}$	$\{\oplus, 1\}$
L ₀	$L \cap R_1$	$\{\oplus\}$
L ₁	$L \cap R_1$	$\{\leftrightarrow\}$
L ₂	$L \cap R_2$	$\{x \oplus y \oplus z\}$
L ₃	$L \cap D$	$\{x \oplus y \oplus z \oplus 1\}$
V	$\{f \mid f \text{ is maximizing or constant}\}$	$\{\vee, 0, 1\}$
V ₀	$\{\{\text{or}\}\} \cup \{\{c_0\}\}$	$\{\vee, 0\}$
V ₁	$\{\{\text{or}\}\} \cup \{\{c_1\}\}$	$\{\vee, 1\}$
V ₂	$\{\{\text{or}\}\}$	$\{\vee\}$
E	$\{f \mid f \text{ is minimizing or constant}\}$	$\{\wedge, 0, 1\}$
E ₀	$\{\{\text{and}\}\} \cup \{\{c_0\}\}$	$\{\wedge, 0\}$
E ₁	$\{\{\text{and}\}\} \cup \{\{c_1\}\}$	$\{\wedge, 1\}$
E ₂	$\{\{\text{and}\}\}$	$\{\wedge\}$
N	$\{\{\text{not}\}\} \cup \{\{c_0\}\} \cup \{\{c_1\}\}$	$\{\neg, 0\}$
N ₂	$\{\{\text{not}\}\}$	$\{\neg\}$
I	$\{\{c_0\}\} \cup \{\{c_1\}\}$	$\{0, 1\}$
I ₀	$\{\{c_0\}\}$	$\{0\}$
I ₁	$\{\{c_1\}\}$	$\{1\}$
I ₂	$\{\emptyset\}$	\emptyset

Table 3.1: Boolean clones.

Examples:

D is the class of selfdual functions; E_2 is the smallest class containing \wedge .
It holds that $E_2 \not\subseteq D$. Moreover, $I_1 \not\subseteq D$; but $[I_1 \cup D] = I_1 \sqcup D = \text{BF}$.

- A function f is *linear* \iff_{def} there exist constants $a_0, a_1, \dots, a_n \in \{0, 1\}$ such that $f(x_1, \dots, x_n) = a_0 \oplus a_1x_1 \oplus a_2x_2 \oplus \dots \oplus a_nx_n$

Examples: 0, 1, id, not, xor, eq are linear.

- For $b \in \{0, 1\}$, a tuple set $T \subseteq \{0, 1\}^n$ is b -separating if and only if there is an $i \in \{1, \dots, n\}$ such that $t_i = b$ for all $(t_1, \dots, t_n) \in T$.

A function f is *b -separating* \iff_{def} $f^{-1}(b)$ is b -separating.

A function f is *b -separating of degree k* \iff_{def} every $T \subseteq f^{-1}(b)$ such that $\|T\| = k$ is b -separating.

Examples: imp, or are 0-separating
and is 1-separating

Theorem 3.9 (Post 1941) *The family of all boolean clones is a countable lattice (with respect to set inclusion).*

3.2.3 Polymorphisms

Alternative approach for general attribute types.

Let D be any attribute type and let $R \subseteq D^n$ be an n -ary relation.

An m -ary D -function $f : D^m \rightarrow D$ is a *polymorphism* of R \iff_{def} for all tuples $x_1, \dots, x_m \in R$ (such that $x_i = (x_i[1], \dots, x_i[n])$,

$$(f(x_1[1], \dots, x_m[1]), \dots, f(x_1[n], \dots, x_m[n])) \in R$$

Let $\text{Pol}(R)$ denote the set of all polymorphisms of R , a set Q of relations define

$$\text{Pol}(Q) =_{\text{def}} \bigcap_{R \in Q} \text{Pol}(R)$$

Example:

$R_0 = \{(0, 0), (1, 1), (0, 1)\}$, i.e., R_0 is the standard total order on $\{0, 1\}$. Note that $x \in R_0 \iff_{\text{def}} x[1] \leq x[2]$. Let $f \in \text{Pol}(R_0)$. Then, for all $x_1, \dots, x_m \in R_0$ is holds that

- $(x_1[1], \dots, x_m[1]) \leq (x_1[2], \dots, x_m[2])$
- $f(x_1[1], \dots, x_m[1]) \leq f(x_1[2], \dots, x_m[2])$

Thus, f is monotone. Hence, $\text{Pol}(R_0) = M$.

Theorem 3.10 (Geiger 1968; Bodvarchuk, Kaluzhnin, Kotov, Romov 1969)

Let D be any finite attribute type and let \mathcal{F} be any class of D -functions. Then, \mathcal{F} is a clone \iff there exists a set Q of relations over D such that $\mathcal{F} = \text{Pol}(Q)$.

3.3 Schedule Frameworks

No mathematical theory available which explains the meaning of modularity for update schedules.

List of schedule types:

- *Sync* is the class of *synchronous* schedules, i.e., the class of mappings

$$\alpha : \{1, \dots, T\} \rightarrow \{V\} \text{ for } T \in \mathbb{N}_+ \cup \{\infty\}, \text{ graph } G = (V, E)$$

- *Async* is the class of *asynchronous* schedules, i.e., the class of mappings

$$\alpha : \{1, \dots, T\} \rightarrow V \text{ for } T \in \mathbb{N}_+ \cup \{\infty\}, \text{ graph } G = (V, E)$$

- *Seq* is the class of *sequential* schedules, i.e., the class of mappings

$$\alpha : \{1, \dots, T\} \rightarrow V \text{ for } T \in \mathbb{N}_+ \cup \{\infty\}, \text{ graph } G = (V, E),$$

such that $\alpha(t) = \alpha(t + k \cdot \|V\|)$ for all $k \in \mathbb{N}$.

Note that: sequential schedule α is *fair* if and only if α is surjective.

- *Semi-synchronous* schedule are schedules not belonging to $\text{Sync} \cup \text{Async}$.

3.4 More Ensembles

(1) Random boolean networks (Kauffman networks)

- (BF, Reg_k^- , Sync) ensemble where Reg_k^- is the class of all directed graphs with loops an in-degree k .
- “random” means “chosen uniformly at random among all graphs on n vertices”.

(2) Canalizing ensemble

- (NCF, \mathcal{G} , \mathcal{S}) where NCF is the class of all “nested canalizing functions” over $D = \{0, 1\}$:

$f : \{0, 1\}^n \rightarrow \{0, 1\}$ is NCF

$$\iff f(x_1, \dots, x_n) = x_{\pi(1)}^* \circ_1 \left(x_{\pi(2)}^* \circ_2 \left(\dots \left(x_{\pi(n-1)}^* \circ_{n-1} x_{\pi(n)}^* \right) \dots \right) \right)$$

where π is a permutation, x^* is a literal x or \bar{x} , and $\circ_i \in \{\wedge, \vee\}$.

- local transitions of the tryptophan operon are NCF.

(3) Hopfield networks

- (TF, \mathcal{G} , Sync) where TF is the class of all “threshold functions” over $D = \{0, 1\}$:

$$f(x_1, \dots, x_n) = \begin{cases} 1 & \text{if } \sum_{i=1}^n w_i x_i > \mathcal{V} \\ 0 & \text{otherwise} \end{cases}$$

for appropriate $\mathcal{V} > 0$, $w_i > 0$.

(4) Contagion networks

Phase Spaces

4

Note: 4.1 ‘Generators’, 4.2 ‘Structure’: later

4.3 Similarity

4.3.1 Functional Equivalence

Let $(X, E, R)_{i \in [0, \infty]}$ be any process.

Let L be a set of local transitions with interdependence structure E .

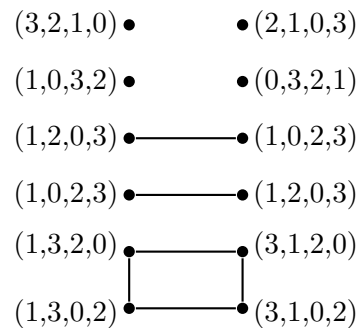
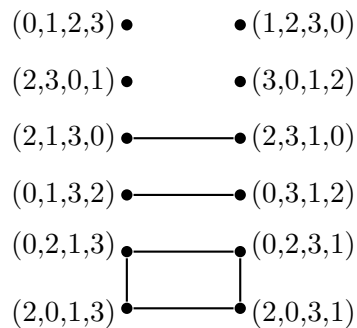
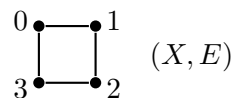
We want to compare permutations π, π' of X such that LSDs (L, π) and (L, π') induce the same phase space as a labelled directed graph, i.e.,

$$\mathbf{F}_{(L, \pi)} = \mathbf{F}_{(L, \pi')}$$

Functional equivalence is based on update orders:

- let X be a population, without loss of generality (w.l.o.g.) $X = \{0, 1, \dots, n - 1\}$
- let E be an interdependence structure
- let S_X denote the symmetric group of X , i.e., the set all permutations $\pi : X \rightarrow X$
- for different $\pi, \pi' \in S_X$ we say that π and π' are *adjacent* if and only if there is a k such that $\{\pi(k), \pi(k + 1)\} \notin E$ and $\pi(i) = \pi'(i)$ for $i \notin \{k, k + 1\}$
- the *update graph* $U(X, E)$ consists of vertex set S_X and edge set $\{(\pi, \pi') \mid \pi \text{ and } \pi' \text{ are adjacent}\}$

Example: update graph for Circ_4



- We define an equivalence relation on S_X with respect to $U = U(X, E)$

$$\pi \sim_U \pi' =_{\text{def}} \pi \text{ and } \pi' \text{ are connected by a path in } U$$

- We consider equivalence class of π : $[\pi]_U =_{\text{def}} \{\pi' \mid \pi \sim_U \pi'\}$

$$S_X / \sim_U = \{[\pi]_U \mid \pi \in S_X\}$$

Proposition 4.1 *Let $G = (X, E)$ be an undirected graph and let $U = U(X, E)$ be the update graph. Then, there exists a bijective mapping*

$$f_G : S_X / \sim_U \rightarrow \text{Acyc}(G)$$

where $\text{Acyc}(G)$ is the set of all acyclic orientations of G .

Proof: We first construct a mapping $\tilde{f}_G : S_X \rightarrow \text{Acyc}(G)$.

Any permutation $\pi \in S_X$ induces a linear ordering \leq_π on X by

$$i \leq_\pi j \iff_{\text{def}} \pi(i) \leq \pi(j)$$

Any linear ordering \leq_π on X induces an acyclic orientation for each $\{i, j\} \in E$ set

$$i \rightarrow j \iff_{\text{def}} i <_\pi j$$

Let \tilde{f}_G map each permutation to the according orientation.

We have to prove that $\tilde{f}_G(\pi) = \tilde{f}_G(\pi')$ for $\pi \sim_U \pi'$. It suffices to show $\tilde{f}_G(\pi) = \tilde{f}_G(\pi')$ for adjacent π, π' (general case by induction):

If π and π' are adjacent, they differ in exactly two consecutive entries not connected by an edge in E . Thus, $\tilde{f}_G(\pi) = \tilde{f}_G(\pi')$.

Define $f_G : S_X / \sim_U \rightarrow \text{Acyc}(G)$ by $f_G([\pi]_U) = \tilde{f}_G(\pi)$.

Observe that f_G is injective (Δ).

It remains to show that f_G is surjective. Consider an acyclic orientation of G . For vertex $i \in X$ define

$$\text{rank}(i) =_{\text{def}} \begin{array}{l} \text{length of a longest directed path to } i \\ \text{(with respect to given acyclic orientation)} \end{array}$$

Note that $\text{rank}(i) = \text{rank}(j)$ implies $\{i, j\} \notin E$ for $i \neq j$.

Define $H =_{\text{def}} \{h \mid \text{rank}^{-1}(h) \neq \emptyset\}$ and for $h \in H$:

$$\text{rank}^{-1}(h) =_{\text{def}} (i_1, \dots, i_{m_h}),$$

where $\text{rank}(i_j) = h$, $i_j < i_k$ for $j < k$. Furthermore, consider

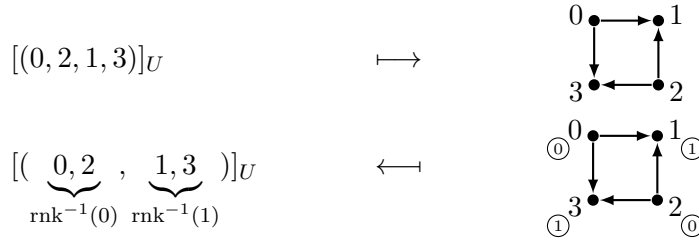
$$[(\text{rk}^{-1}(0), \text{rk}^{-1}(1), \dots, \text{rk}^{-1}(t))]_U$$

with $t = \max H$.

Then, clearly, f_G maps $[(\text{rk}^{-1}(0), \dots, \text{rk}^{-1}(t))]_U$ to the given orientation.

Thus, f_G is surjective. Hence, f_G is bijective. ■

Example: Consider Circ_4



Proposition 4.2 Let $G = (X, E)$ be an undirected graph and let $\pi, \pi' \in S_X$. If $\pi \sim_U \pi'$ then for all sets L of local transition functions,

$$\begin{aligned}
 \mathbf{F}_L[\pi] &= \mathbf{F}_L[\pi'] \\
 (\mathbf{F}_{(L, \pi)} &= \mathbf{F}_{(L, \pi')})
 \end{aligned}$$

Proposition 4.3 For any undirected graph $G = (X, E)$ and any set L of local transition functions on E ,

$$\|\{\mathbf{F}_{(L, \pi)} \mid \pi \in S_X\}\| \leq \|\text{Acyc}(G)\|,$$

and the bound is sharp.

Proof: Inequality is immediate from Proposition 1 and 2; Sharpness is exercise. ■

$$\left[\begin{array}{l} \|\{\mathbf{F}_{(L, \pi)} \mid \pi \in S_X\}\| \leq \{[\pi]_U \mid \pi \in S_X\} \quad (\text{Prop. 2}) \\ \qquad \qquad \qquad = \|S_X / \sim_U\| \\ \qquad \qquad \qquad = \|\text{Acyc}(G)\| \quad (\text{Prop. 1}) \end{array} \right]$$

Example:

It hold that $\|\text{Acyc}(\text{Circ}_n)\| = 2^n - 2$ (only two of possible orientations of Circ_4 are not acyclic).

Thus, there are at most $2^n - 2$ different phase spaces / LSD on Circ_n .

How to compute $|\text{Acyc}(G)|$?

Use chromatic polynomial: Let $G = (V, E)$ be an undirected graph. A *vertex coloring* with k colors $1, \dots, k$ is a mapping $f : V \rightarrow \{1, \dots, k\}$ such that $f(u) \neq f(v)$ if $\{u, v\} \in E$.

Define $P_G(k)$ to be the number of different vertex colorings with k colors of G .

Example: Let $G = K_n$. That is $P_G(k) = 0$ for $k < n$.

Moreover, $P_G(n) = n!$. It holds that $P_G(x) = x^n$.

Lemma 4.4 *Let G, H be undirected graphs.*

- (1) *If G is a one-vertex graph, $P_G(k) = k$.*
- (2) *$P_{G \oplus H}(k) = P_G(k) \cdot P_H(k)$*
- (3) *$P_G(k) = P_{G-e}(k) - P_{G/e}(k)$*

Example:

Let T be a tree with n vertices. Let u be an arbitrary leaf of T and $e = \{u, v\}$ be the edge connecting u with T . Then, it holds

$$\begin{aligned} P_T(x) &= P_{T-e}(x) - P_{T/e}(x) \\ &= P_{T'}(x) \cdot x - P_{T''(x)} \end{aligned}$$

Here, T' is a tree with $n-1$ vertices, T'' is a tree with $n-1$ vertices. Actually, $T' \simeq T''$. We conclude

$$P_T(x) = P_{T'}(x) \cdot (x-1)$$

By iteration, we obtain: $P_T(x) = x(x-1)^{n-1}$.

Thus, each tree with n vertices has the same chromatic polynomial independent of its structure. Moreover, a graph G with n vertices is a tree if and only if $P_G(x) = x(x-1)^{n-1}$.

Lemma 4.5 *Let G be an undirected graph. Suppose there are graphs G_1, G_2 such that $G = G_1 \cup G_2$ and $G_1 \cap G_2 = K_n$. Then,*

$$P_G(x) = \frac{P_{G_1}(x) \cdot P_{G_2}(x)}{P_{K_n}(x)}$$

Proof: Each vertex coloring f of G corresponds to exactly one pair (f_1, f_2) of colorings of G_1 and G_2 which are identical on K_n . So, let f_1 be a k -coloring of G_1 .

Then, there are $P_{G_2}(k)/P_{K_n}(k)$ k -colorings of G_2 which are identical on K_n with f_1 . ■

Example:

We want to compute the chromatic polynomial for k^n . We obtain the following recursion:

$$\begin{aligned} P_{k^n}(x) &= P_{k^n-e}(x) - P_{k^n/e}(x) \\ &= \frac{P_{k^{n-1}}(x)^2}{P_{k^{n-2}}(x)} - P_{k^{n-1}}(x) \\ &= \frac{P_{k^{n-1}}(x)}{P_{k^{n-2}}(x)} (P_{k^{n-1}}(x) - P_{k^{n-2}}(x)) \end{aligned}$$

By induction we can prove that $P_{k^n}(x) = x^n$:

- $n = 1$: $P_{k^1}(x) = x = x^1$
- $n = 2$: $P_{k^2}(x) = x(x-1) = x^2$
- $n > 2$: $P_{k^n}(x) = \frac{x^{n-1}}{x^{n-2}} (x^{n-1} - x^{n-2})$

$$= (x - (n-1) + 1) x^{n-2} ((x - (n-1) + 1) - 1)$$

$$= x^{n-2} (x - (n-2))(x - (n-1))$$

$$= x^n$$

We give a different interpretation of $P_G(x)$

Proposition 4.6 *Let $G = (V, E)$ be an undirected graph. Then, $P_G(k)$ is equal to the numbers of pairs (f, O) where $f : V \rightarrow \{1, \dots, k\}$ and O is an orientation of G such that:*

- (i) *orientation O is acyclic*
- (ii) *if $u \rightarrow v$ in orientation O then $f(u) > f(v)$*

Proof: Consider a pair (f, O) satisfying (i), (ii). From (ii) it follows that $f(u) \neq f(v)$ for $\{u, v\} \in E$. Thus, f is a vertex coloring with k colors. Moreover, (ii) implies (i).

Conversely, if f is a vertex coloring with k colors then f defines a unique acyclic orientation O by $u \rightarrow v$ if and only if $f(u) > f(v)$. Hence, the number of allowed pairs (f, O) is the number of vertex colorings with colors $1, \dots, k$ and is, thus, $P_G(k)$. ■

Proposition 6 suggests the following modification:

Let $G = (V, E)$ be an undirected graph and let $k \in \{1, \dots, n\}$ where $n = ||V||$. Define $\bar{P}_G(k)$ to be the number of pairs (f, O) where $f : V \rightarrow \{1, \dots, k\}$ and O is an orientation of G such that:

- (i) orientation O is acyclic

(ii) if $u \rightarrow v$ in orientation O then $f(u) > f(v)$ (we say that f is compatible with O)

Lemma 4.7 *Let G, H be undirected graphs.*

(1) *If G is one-vertex graph then $\bar{P}_G(k) = k$.*

(2) $\bar{P}_{G \oplus H}(k) = \bar{P}_G(k) \cdot \bar{P}_H(k)$

(3) $\bar{P}_G(k) = \bar{P}_{G-e}(k) + \bar{P}_{G/e}(k)$ for any $e \in E$

Proof:

(1), (2): obvious

(3): Let $f : V \rightarrow \{1, \dots, k\}$ be a mapping and let O be an acyclic orientation of $G-e$ compatible with f , where $e = \{u, v\} \in E$. Let O_1 be the orientation of G obtained by adjoining $u \rightarrow v$ to O , and O_2 that is obtained by adjoining $v \rightarrow u$ to O .

We will show that for each pair (f, O) exactly one of O_1 and O_2 is an acyclic orientation compatible with f , except for $\bar{P}_{G/e}(k)$ of the pairs, in which case both O_1 and O_2 are acyclic orientations compatible with f . Thus, $\bar{P}_{G-e}(k) = \bar{P}_G(k) - \bar{P}_{G/e}(k)$.

Consider the following cases:

(a) $f(u) > f(v)$: Then, O_2 is not compatible with f while O_1 is compatible. Moreover, O_1 is acyclic since if $u \rightarrow v \rightarrow w_1 \rightarrow w_2 \rightarrow \dots \rightarrow u$ were a directed cycle in O_1 , we would have $f(u) > f(v) \geq f(w_1) \geq f(w_2) \geq \dots \geq f(u)$, which is a contradiction.

(b) $f(u) < f(v)$: symmetrical to (a).

(c) $f(u) = f(v)$: Both O_1 and O_2 are compatible with f . Then, at least one of them is acyclic. If not then:

* O_1 contains a cycle $u \rightarrow v \rightarrow w_1 \rightarrow w_2 \rightarrow \dots \rightarrow u$,

* O_2 contains a cycle $v \rightarrow u \rightarrow w'_1 \rightarrow w'_2 \rightarrow \dots \rightarrow v$.

Hence, O contains a cycle $v \rightarrow w_1 \rightarrow w_2 \rightarrow \dots \rightarrow u \rightarrow w'_1 \rightarrow w'_2 \rightarrow \dots \rightarrow u$ which is not possible.

It remains to prove that O_1 and O_2 are acyclic for exactly $\bar{P}_{G/e}(k)$ pairs (f, O) with $f(u) = f(v)$. Define $\Phi(f, O) =_{\text{def}} (f', O')$ such that $f' : V(G/e) \rightarrow \{1, \dots, k\}$ (note that $f(u) = f(v)$) and O' is an acyclic orientation of G/e compatible with f' . Let z be the vertex obtained by identifying u and v . Define f' to be

$$f'(w) =_{\text{def}} \begin{cases} f(w) & \text{if } w \in V \setminus \{u, v\} \\ f(u) & \text{if } w = z \end{cases}$$

Define O' by $w_1 \rightarrow w_2$ in O' if and only if $w_1 \rightarrow w_2$ in O . Then Φ is a bijection. ■

Theorem 4.8 (Stanley 1973) For each graph $G = (V, E)$ such that $\|V\| = n$, $\bar{P}_G(k) = (-1)^n P_G(-k)$.

Proof: By induction on n :

- $n = 1$: $\bar{P}_G(k) = k = (-1)^1(-k)$
- If G is the empty graph with n vertices.
Then, $\bar{P}_G(k) = k^n = (-1)^n(-k)^n$.

Suppose $\|E\| \geq 1$. Then, for some edge $e \in E$

$$\begin{aligned} \bar{P}_G(k) &= \bar{P}_{G-e}(k) + \bar{P}_{G/e}(k) \\ &= (-1)^n P_{G-e}(-k) + (-1)^{n-1} P_{G/e}(-k) \\ &= (-1)^n (P_{G-e}(-k) - P_{G/e}(-k)) \\ &= (-1)^n P_G(-k) \end{aligned} \quad \blacksquare$$

Corollary 4.9 $\|\text{Acyc}(G)\| = (-1)^n P_G(-1)$.

Proof: $\|\text{Acyc}(G)\| = \bar{P}_G(1) = (-1)^n P_G(-1)$. ■

Example:

We want to compute $\|\text{Acyc}(\text{Circ}_n)\|$, $n \geq 3$.

First, we prove that $P_{\text{Circ}_n}(x) = (x-1)^n + (-1)^n(x-1)$ by induction on $n \geq 3$.

- $n = 3$: $P_{\text{Circ}_3}(x) = x(x-1)(x-2)$
 $= x^3 - 3x^2 + 2x$
 $= x^3 - 3x^2 + 3x - 1 - (x-1)$
 $= (x-1)^3 + (-1)^3(x-1)$
- $n > 3$: $P_{\text{Circ}_n}(x) = P_{\text{Circ}_n-e}(x) - P_{\text{Circ}_n/e}(x)$
 $= x(x-1)^{n-1} - ((x-1)^{n-1} + (-1)^{n-1}(x-1))$
 $= (x-1)^n(x-1) - (-1)^{n-1}(x-1)$
 $= (x-1)^n + (-1)^n(x-1)$

From corollary 9, we obtain $\|\text{Acyc}(\text{Circ}_n)\| = 2^n - 2$:

If n is even then $\bar{P}_{\text{Circ}_n}(1) = P_{\text{Circ}_n}(-1) = 2^n - 2$

If n is odd then $\bar{P}_{\text{Circ}_n}(1) = -P_{\text{Circ}_n}(-1) = 2^n - 2 = -(-2^n - (-2))$

4.3.2 Black-Box Equivalence

Want to extend functional equivalence to arbitrary LSD, i.e., to arbitrary schedules.

Definition 4.10 . Let $G = (X, E)$ be an undirected graph. Let $\alpha : \{1, \dots, T\} \rightarrow \mathcal{P}(X)$ and $\alpha' : \{1, \dots, T'\} \rightarrow \mathcal{P}(X)$ be update schedules. Then, $\alpha \equiv_{\text{bb}} \alpha' \iff_{\text{def}}$ for all attribute types D and all sets L of D -functions,

$$\mathbf{F}_{L,\alpha}^{(\cdot,T)} = \mathbf{F}_{L,\alpha'}^{(\cdot,T')}$$

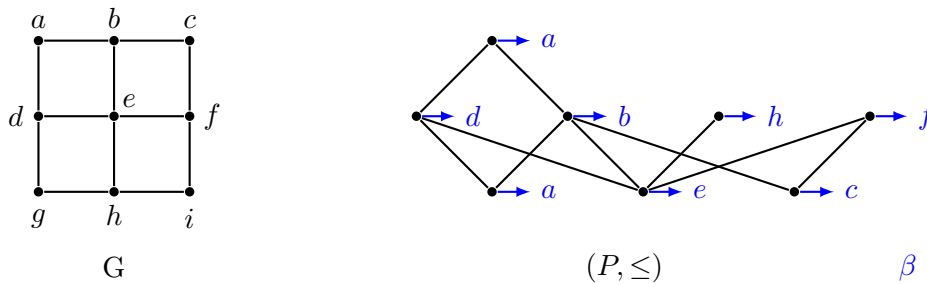
Clearly for all permutation $\pi, \pi' : \pi \sim_{U(X,E)} \pi' \iff \pi \equiv_{\text{bb}} \pi'$.

We want to determine “best” simulations (according to \equiv_{bb}) for sequential updates.

Definition 4.11 . A triple (G, P, β) is called a **pograph** (or poset model for graphs) if and only if $G = (X, E)$ is an undirected graph, $P = (P, \leq)$ is a finite poset, and $\beta : P \rightarrow X$ is a mapping satisfying $i, j \in P$:

- (i) $i \leq_p j \longrightarrow \{\beta(i), \beta(j)\} \in E$
- (ii) $\{\beta(i), \beta(j)\} \in E \longrightarrow i \leq j \vee j \leq i$

Example:



Theorem 4.12 (Laubenbacher, Pareigis 2006) Let $G = (X, E)$ be an undirected graph and let $\alpha : \{1, \dots, T\} \rightarrow X$ be an asynchronous schedule. Then, there exists a finite poset P with $\|P\| = T$, a mapping $\beta : P \rightarrow X$, and a bijective and monotone mapping $\gamma : P \rightarrow \{1, \dots, T\}$ such that

- (i) (G, P, β) is a pograph
- (ii) $\alpha = \beta \circ \gamma^{-1}$

Proof: [sketch] Construct a canonical representation of (G, α) :
 Define a poset $(P(G, \alpha), \leq_\alpha)$ as follows:

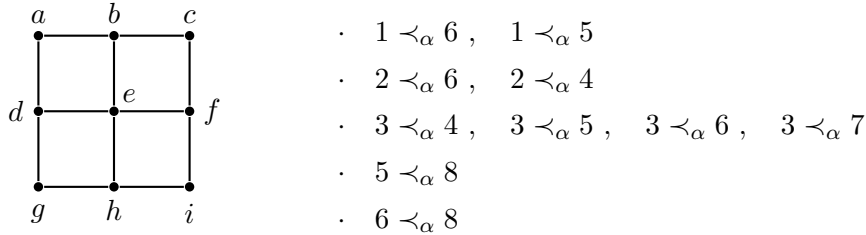
- $P(G, \alpha) =_{\text{def}} \{1, \dots, T\}$
- \leq_α is reflexive and transitive closure of precedence \prec_α :

$$i \prec_\alpha j \iff i \leq j \text{ and } \{\alpha(i), \alpha(j)\} \in E$$

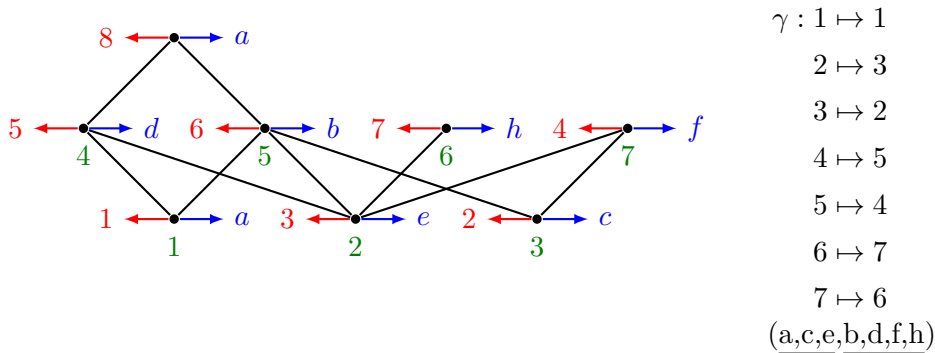
Prove that $(P(G, \alpha), \leq_\alpha)$ is a poset.
 Prove the existence of β and γ . ■

Example: G from above, $\alpha = (a, c, e, f, d, b, h, a)$, $T = 8$

We obtain:



Hasse diagram for $(P(G, \alpha), \leq_\alpha)$:



Thus, $\beta \circ \gamma^{-1} = \alpha$.

Given the pograph for (G, α) we define the *level-wise schedule*:

- let P be a finite poset
- $h_P(i)$ is the *height* of $i \in P$, i.e., the maximum length of a decreasing chain with i as its max. element

$$h_P(i) =_{\text{def}} \max \{r \mid (\exists i_1, \dots, i_r \in P) [i_r = i \wedge (\forall 1 \leq t < r) [i_t <_P i_{t+1}]]\}$$

- $h(P)$ is the *height of P* :

$$h(P) =_{\text{def}} \max_{i \in P} h_P(i)$$

- Given (G, P, β) define λ_P for $i \in \{1, \dots, T\}$ as follows

$$\lambda_P(i) =_{\text{def}} \{\beta(j) \mid j \in P \text{ and } h_p(j) = i\}$$

Example: For (G, P, β) from above:

$$\begin{aligned} \lambda_P(1) &= \{a, c, e\} \\ \lambda_P(2) &= \{b, d, f, h\} \\ \lambda_P(3) &= \{a\} \end{aligned}$$

Note that λ_P is not asynchronous.

Lemma 4.13 (Laubenbacher, Paraeigis 2006) *Let (G, P, β) be a pograph, $\|P\| = T$. Let $\gamma, \gamma' : P \rightarrow \{1, \dots, T\}$ be bijective and monotone mappings. Then, $\beta \circ \gamma^{-1} \equiv_{\text{bb}} \beta \circ \gamma'^{-1}$.*

Proposition 4.14 *Let $G = (X, E)$ be an undirected graph and let L be a set of local transition functions. Let $U \subseteq X$ be an independent set, i.e., $G[U]$ is empty. Then, for all $\alpha : \{1, \dots, T\} \rightarrow \mathcal{P}_+(U)$ such that $2 \leq T \leq \|U\|$, $\alpha(i) \cap \alpha(j) = \emptyset$ for $i \neq j$, and $\alpha(1) \cup \dots \cup \alpha(T) = U$,*

$$\mathbf{F}_L[U] = \prod_{i=1}^T \mathbf{F}_L[\alpha(i)]$$

Proposition 4.15 *Let G be an undirected graph. Let α be an asynchronous schedule. Then, $\alpha \equiv_{\text{bb}} \lambda_{P(G, \alpha)}$.*

Proof: Let $G = (X, E)$, $\alpha : \{1, \dots, T\} \rightarrow X$ (permutation). W.l.o.g. $X = \{1, \dots, n\}$. By Theorem 12, $(G, P(G, \alpha), \alpha)$ is a pograph. Define $\gamma =_{\text{def}} \text{id}_{P(G, \alpha)}$, i.e., γ is bijective and monotone. It holds that $\alpha \circ \gamma^{-1} = \alpha$.

Define $\gamma' : P(G, \alpha) \rightarrow \{1, \dots, T\}$ for all $j \in P(G, \alpha)$ by

$$\gamma'(j) =_{\text{def}} \|\{k \mid h_{P(G, \alpha)}(k) < h_{P(G, \alpha)}(j) \text{ or } h_{P(G, \alpha)}(k) = h_{P(G, \alpha)}(j) \wedge \alpha(k) \leq \alpha(j)\}\|$$

Then, γ' is bijective and monotone on $P(G, \alpha)$.

By Proposition 13, $\alpha \circ \gamma^{-1} \equiv_{\text{bb}} \alpha \circ \gamma'^{-1}$.

By construction of $\lambda_{P(G, \alpha)}$, each set $\lambda_{P(G, \alpha)}(i)$ is an independent set in G . Thus, by Proposition 14, $\alpha \circ \gamma'^{-1} \equiv_{\text{bb}} \lambda_{P(G, \alpha)}$.

Hence, $\alpha \equiv_{\text{bb}} \lambda_{P(G, \alpha)}$. ■

Proposition 4.16 *Let G be an undirected graph. Let α be an asynchronous schedule. Then, for all schedules $\alpha' \equiv_{\text{bb}} \alpha$ it holds that $\text{ord}(\alpha') \geq \text{ord}(\lambda_{P(G,\alpha)})$.*

Proof: Let $G = (X, E)$, $\alpha : \{1, \dots, T\} \rightarrow X$, $\text{ord}(\alpha) = T$. Consider attribute type $D = \{1, \dots, T\}$. For each $i \in X$ define

$$f_i(z_1, \dots, z_n) =_{\text{def}} 1 + \max\{z_j \mid j = i \text{ or } \{i, j\} \in E\}$$

Define $L = \{f_1, \dots, f_n\}$.

Observe that the maximum component of

- $\mathbf{F}_{(L, \lambda_{P(G,\alpha)})}(0, \dots, 0 ; \text{ord}(\lambda_{P(G,\alpha)}))$ is $h(P(G, \alpha)) = \text{ord}(\lambda_{P(G,\alpha)})$
- $\mathbf{F}_{(L, \alpha')}(0, \dots, 0 ; \text{ord}(\alpha'))$ is $\text{ord}(\alpha')$

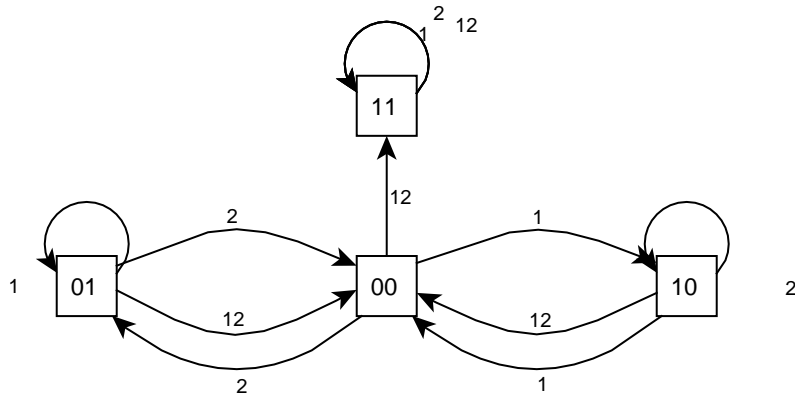
Hence, $\text{ord}(\alpha') \geq \text{ord}(\lambda_{P(G,\alpha)})$ (otherwise: $\alpha' \not\equiv_{\text{bb}} \alpha$). ■

Proposition 4.17 *Let $G = (X, E)$ be an undirected graph and let α be a schedule of order T such that there is an edge $e \in E$ with $e \leq \alpha(i)$ for some $i \in \{1, \dots, T\}$. Then, there is no asynchronous schedule α' such that $\alpha' \equiv_{\text{bb}} \alpha$ on G .*

Proof: We first consider a simple LSD. Let $H = (\{1, 2\}, \{\{1, 2\}\})$ be a single-edge graph. Define f_j for $j \in \{1, 2\}$ by

$$f_j(z_1, z_2) =_{\text{def}} \begin{cases} 1 & \text{if } z_1 = z_2 \\ 0 & \text{otherwise} \end{cases}$$

Then, the global transition function for $L = \{f_1, f_2\}$ is as follows:



Now suppose α is a schedule of order T such that $\alpha(i) = \{1, 2\}$ for some $i \in \{1, \dots, T\}$. Then, we obtain $\|\mathbf{F}_{(L,\alpha)}^{-1}(1, 1)\| \geq 2$.

- (i) $(1, 1) \in \mathbf{F}_{(L,\alpha)}^{-1}(1, 1)$
- (ii) Let i be minimal subject to $\alpha(i) = \{1, 2\}$. The sequence $(\alpha(1), \dots, \alpha(i-1))$ can be viewed as a word over $\{1, 2\}$. Let (a, b) be the configuration that leads to $(0, 0)$ for schedule/word $(\alpha(i-1), \dots, \alpha(1))$. Clearly, $(a, b) \neq (1, 1)$ but $\mathbf{F}_{(L,\alpha)}(a, b) = (1, 1)$.

Let β be any asynchronous schedule. Then, $\mathbf{F}_{(L,\beta)}^{-1}(1, 1) = \{(1, 1)\}$. Thus, $\alpha \not\equiv_{\text{bb}} \beta$ on H .

Consider any graph $G = (X, E)$ with $E \neq \emptyset$ and a schedule α of order T . Let $e = \{u, v\} \in E$ be an edge such that $\{u, v\} \subseteq \alpha(i)$ for some $i \in \{1, \dots, T\}$. Set $f_w =_{\text{def}} 0$ for $w \notin \{u, v\}$ and simulate the LSD from above for subgraph $\{u, v\}$.

Hence, there is no asynchronous schedule equivalent to α on G . ■

Corollary 4.18 *Let G be a graph and let α be a schedule. There is an asynchronous schedule α' such that $\alpha' \equiv_{\text{bb}} \alpha$ on G if and only if for $i \in \{1, \dots, \text{ord}(\alpha)\}$, $\alpha(i)$ is an independent set in G .*

4.3.3 Dynamical Equivalence

Dynamical equivalence refers to LSD's having isomorphic phase-spaces.

Definition 4.19 . *Let D be an attribute type. Let $P = (X, E, D^n)_{i \in [0, \infty]}$ be any state process. Let (L_1, α_1) and (L_2, α_2) be LSD's on X . Then, (L_1, α_1) and (L_2, α_2) are **dynamically equivalent** if and only if there is a bijection $\varphi : D^n \rightarrow D^n$ such that for all $x \in D^n$*

$$\mathbf{F}_{(L_1, \alpha_1)}(\varphi(x)) = \varphi(\mathbf{F}_{(L_2, \alpha_2)}(x))$$

Proposition 4.20 *Let φ be the Euler φ -function.*

For $n \geq 3$, there are at most

$$(i) \frac{1}{2n} \sum_{d/n} \varphi(d) (2^{n/d} - 2) + 2^{n/2}/4, \text{ if } n \text{ is even}$$

$$(ii) \frac{1}{2n} \sum_{d/n} \varphi(d) (2^{n/d} - 2), \text{ if } n \text{ is odd}$$

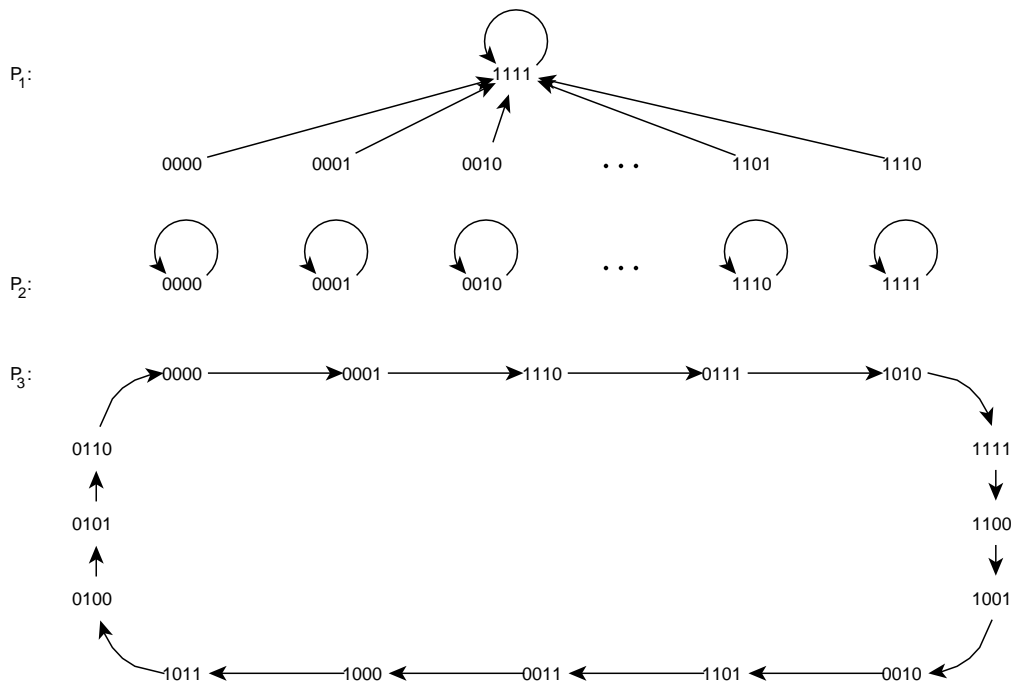
dynamically non-equivalent permutations on Circ_n .

4.4 Features

Features are relevant patterns in phase spaces.

4.4.1 Chaos

Consider the following three phase spaces for state processes over population size 4 with attribute type $D = \{0, 1\}$.

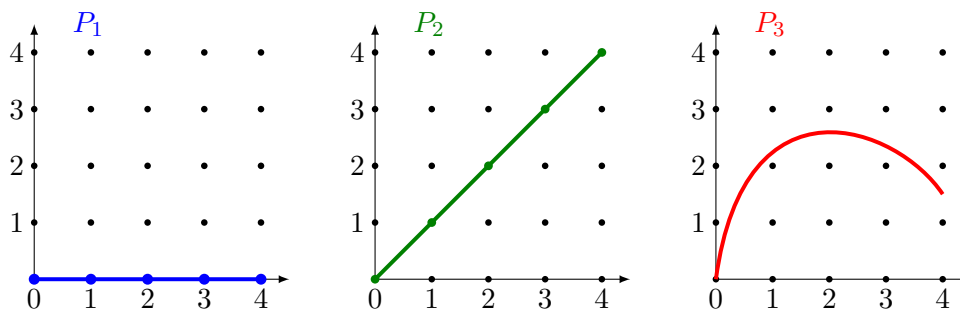


Derrida plot:

- let $d_H(\vec{x}, \vec{y})$ denote the Hamming distance between configurations \vec{x} and \vec{y} , i.e., $d_H(\vec{x}, \vec{y}) =_{\text{def}} |\{i \mid x_i \neq y_i\}|$
- let $\mathbf{F} : D^n \rightarrow D^n$ be a (time-invariant) global state dynamic
- Derrida relation $\mathcal{D} = \mathcal{D}(\mathbf{F})$ (as multi-set):

$$\mathcal{D} =_{\text{def}} \{(h_1, h_2) \mid \text{there are states } \vec{x}, \vec{y} \text{ such that } d_H(\vec{x}, \vec{y}) = h_1 \text{ and } d_H(\mathbf{F}(\vec{x}), \mathbf{F}(\vec{y})) = h_2\}$$

- plot \mathcal{D} as a diagram:



- colored lines are interpolations of average resulting distances

Intuition: The more pairs above the diagonal, the more chaos.

Try to figure out an appropriate parameter for measuring chaos:

Use linear regression: Suppose \mathcal{D} consists of N pairs $(x_1, y_1), \dots, (x_N, y_N)$; define

$$L(\beta) =_{\text{def}} \sum_{i=1}^N (y_i - \beta x_i)^2$$

$$L'(\beta) = \sum_{i=1}^N 2(y_i - \beta x_i)^2 (-x_i)$$

$$L''(\beta) = \sum_{i=1}^N 2x_i^2 > 0, \text{ i.e., any zero of } L'(\beta) \text{ is a minimum.}$$

$$\text{Thus, } \beta = \frac{\sum_{i=1}^N y_i x_i}{\sum_{i=1}^N x_i^2} \text{ is optimal slope}$$

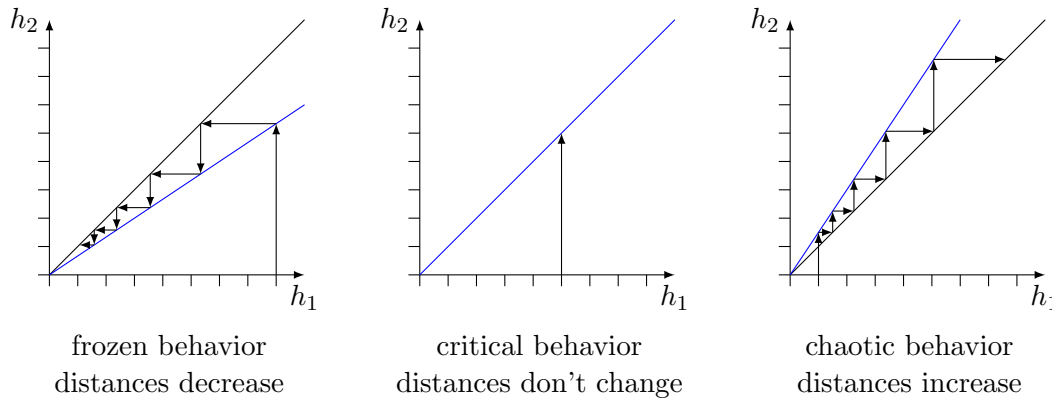
Measuring the angle gives a scaled parameter β^* :

$$\beta =^*_{\text{def}} \left(\frac{4}{\pi} \arctan \beta \right) - 1, \text{ i.e., } -1 \leq \beta^* \leq 1$$

Interpretation:

- $\beta^* \approx -1$: *frozen* behavior
- $\beta^* \approx 0$: *critical* behavior
- $\beta^* \approx 1$: *chaotic* behavior

Slope β is a statistical measure of how different states evolve over time
(for $h_1 \rightarrow \infty$, i.e., $n \rightarrow \infty$)



Example:

- Phase space P_1 shows frozen behavior: $\beta^* = -1$
- Phase space P_2 shows critical behavior: $\beta^* = 0$
- Phase space P_3 shows almost critical: $\beta = 0.8441358 \dots$
 $\beta^* = -0.107357 \dots$

Remark:

For $n < \infty$ we have $\sum_{i=1}^N x_i y_i \leq \sum_{i=1}^N x_i^2$, i.e., $\beta^* \leq 0$

Why? It holds $\sum_{i=1}^N y_i^2 \leq \sum_{i=1}^N x_i^2$ with equality for \mathbf{F} being a permutation. Thus,

$$\begin{aligned}
 0 &\leq \sum_{i=1}^N (x_i - y_i)^2 = \sum_{i=1}^N (x_i^2 - 2x_i y_i + y_i^2) \\
 &\leq 2 \left(- \sum_{i=1}^N x_i y_i + \sum_{i=1}^N x_i^2 \right)
 \end{aligned}$$

Want to determine system parameters inducing a certain phase behavior.

Consider a random boolean network consisting of n actors with attribute type $D = \{0, 1\}$ each depending on exactly k actors of the population (multiple dependencies are allowed), i.e., (BF, Reg_k^- , Sync)-ensemble.

Suppose we are given $h_1, h_2 \in \{0, \dots, n\}$. Consider randomly chosen states $\vec{x} = (x_1, \dots, x_n)$ and $\vec{y} = (y_1, \dots, y_n)$ such that $d_H(\vec{x}, \vec{y}) = h_1$.

Define $P_t(h_2, h_1)$ to be the probability that Hamming distance of the images \vec{x}', \vec{y}' of \vec{x}, \vec{y} after t synchronous update steps is h_2 .

- $t = 1$: Let \vec{x} and \vec{y} be two states. Define the sets

$$\begin{aligned} A &=_{\text{def}} \{i \mid x_i = y_i\} \\ B &=_{\text{def}} \{i \mid x_i \neq y_i\} \end{aligned}$$

That is, $\|A\| = n - h_1, \|B\| = h_1$. Define $Q(n_0)$ to be the probability that n_0 actors depend completely on k actors from A . We obtain

$$Q(n_0) = \binom{h}{n_0} \left[\left(\frac{n - h_1}{n} \right)^k \right]^{n_0} \left[\left(1 - \left(\frac{n - h_1}{n} \right)^k \right) \right]^{n - n_0}$$

These n_0 actors have the same states in \vec{x}' and \vec{y}' . For the remaining $n - n_0$ actors, with probability $\frac{1}{2}$ states are equal in \vec{x}' and \vec{y}' and with probability $\frac{1}{2}$ states are different in \vec{x}' and \vec{y}' .

Thus, using $x = \frac{h_1}{n}, y = \frac{h_2}{n}$

$$\begin{aligned} P_1(h_2, h_1) &= \sum_{n_0=0}^n Q(n_0) \binom{n - n_0}{h_2} \underbrace{\left(\frac{1}{2} \right)^{h_2} \left(\frac{1}{2} \right)^{n - n_0 - h_2}}_{\left(\frac{1}{2} \right)^{n - n_0}} \\ &= \sum_{n_0=0}^n \binom{n - n_0}{h_2} \left(\frac{1}{2} \right)^{n - n_0} \binom{n}{n_0} [(1 - x)^k]^{n_0} \left[(1 - (1 - x)^k) \right]^{n - n_0} \\ &= \sum_{n_0=0}^n \binom{n}{h_2} \binom{n - h_2}{n_0} [(1 - x)^k]^{h_0} \left[\frac{1}{2} (1 - (1 - x)^k) \right]^{n - n_0} \\ &= \binom{n}{h_2} \left[\frac{1}{2} (1 - (1 - x)^k) \right]^{h_2} \sum_{n_0=0}^{n - h_2} \binom{n - h_2}{n_0} [(1 - x)^k]^{n_0} \left[\frac{1}{2} (1 - (1 - x)^k) \right] \\ &= \binom{n}{h_2} \left[\frac{1}{2} (1 - (1 - x)^k) \right]^{h_2} \left[(1 - x)^k + \frac{1}{2} (1 - (1 - x)^k) \right]^{n - h_2} \\ &= \binom{n}{h_2} \left(\frac{1}{2} \right)^n (1 - (1 - x)^k)^{h_2} (1 + (1 - x)^k)^{n - h_2} \end{aligned}$$

The probability $P_1(h_2, h_1)$ is maximal for $2y = 1 - (1 - x)^k$:

We set $z =_{\text{def}} (1 - x)^k$ and consider $f(z) =_{\text{def}} (1 - z)^{h_2}(1 + z)^{n-h_2}$

Hence,

$$\begin{aligned} f'(z) &= h_2(1 - z)^{h_2-1}(-1)(1 + z)^{n-h_2} + (1 - z)^{h_2}(n - h_2)(1 + z)^{n-h_2-1} \\ &= (1 - z)^{h_2-1}(1 + z)^{n-h_2-1} \underbrace{[-h_2(1 + z) + (n - h_2)(1 - z)]}_{\substack{=-h_2-h_2z+n-h_2-nz+h_2z \\ =n-2h_2-nz}} \end{aligned}$$

We obtain the following zeroes:

$$z_0 = 1, \quad z_1 = -1, \quad \text{and } z_2 = 1 - 2\frac{h_2}{n} = 1 - 2y$$

Zeroes z_0, z_1 correspond to minima; from z_2 we conclude $2y = 1 - (1 - x)^k$ for maximum.

- $t > 1$: We use an annealed approximation:

$$\tilde{P}_t(h_{t+1}, h_1) = \sum_{h_2=0}^n \cdots \sum_{h_k=0}^n P_1(h_2, h_1) \cdot P_1(h_3, h_2) \cdots P(h_{t+1}, h_t)$$

A similar analysis yields a high concentration of \tilde{P} for

$$y_t = \frac{1 - (1 - y_{t-1})^k}{2} \quad (y_0 = x(!) \quad y_1 = y)$$

where y_1 is the value above (for $t = 1$).

We obtain the following cases for k :

- $k = 1$: $\lim_{t \rightarrow \infty} y_t = 0$ since $y_t = \frac{y_{t-1}}{2}$
- $k = 2$: $\lim_{t \rightarrow \infty} y_t = 0$ since $y = \frac{1 - (1 - y)^2}{2}$ has solution

$$\begin{aligned} 2y &= 1 - (1 - 2y + y^2) \\ &= 2y - y^2, \text{ i.e.,} \\ &\text{only solution } 0 \leq y \leq 1 \text{ is } 0. \end{aligned}$$
- $k = 3$: $\lim_{t \rightarrow \infty} y_t = y^* > 0$ since $2y = 1 - (1 - y)^3$

$$\begin{aligned} &= 1 - (1 - 3y + 3y^2 - y^3), \\ \text{i.e., } 2y &= 3y - 3y^2 + y^3, \\ \text{i.e., } 0 &= 1 - 3y^2 + y^3 \end{aligned}$$

interpretation:

- $k \geq 3$: random boolean networks show chaotic behavior.
- $k = 2$: critical behavior
- $k = 1$: frozen behavior

Analysis of the behavior for k :

$$\begin{aligned}
 2 \frac{h_2}{n} &= 1 - \left(1 - \frac{h_1}{n}\right)^k \\
 &= 1 - \sum_{l=0}^k \binom{k}{l} \left(-\frac{h_1}{n}\right)^l \\
 &= 1 - \left(1 - k \cdot \frac{h_1}{n} + \sum_{l=2}^k \binom{k}{l} \left(-\frac{h_1}{n}\right)^l\right) \\
 &= k \cdot \frac{h_1}{n} - \sum_{l=2}^k \binom{k}{l} \left(-\frac{h_1}{n}\right)^l \\
 \text{That is: } h_2 &= \frac{k}{2} \cdot h_1 - \frac{1}{2} \sum_{l=2}^k \binom{k}{l} \frac{(-h_1)^l}{n^{l-1}}
 \end{aligned}$$

For n approaching ∞ we obtain:

$$\lim_{n \rightarrow \infty} h_2 = \frac{k}{2} \cdot h_1$$

We have three cases (for n large).

- $k = 1$: $h_2 \approx \frac{1}{2} \cdot h_1$,i.e., \approx frozen behavior
- $k = 2$: $h_2 \approx h_1$,i.e., \approx critical behavior
- $k = 3$: $h_2 \approx \frac{3}{2} \cdot h_1$,i.e., \approx chaotic behavior

Exploring exponential-size phase spaces is intractable, e.g., for boolean domain $D = \{0, 1\}$ and $n = 300$ actors, there are $2^{300} > 10^{90}$ states ($\approx 10^{80}$ atoms in the universe).

Which dynamics (networks, transitions, schedules) allow “simulation short-cuts”?

5.1 Fixed Points

Definition 5.1 . Let $G = (X, E)$ be an undirected graph and let $L = \{f_1, \dots, f_n\}$ be a set of local transitions over domain D .

- (1) A configuration $x \in D^n$ is said to be a **local fixed point (LFP)** of (G, L) for $U \subseteq X \iff_{\text{def}} \mathbf{F}_L[U](x) = x$
- (2) A configuration $x \in D^n$ is said to be a **fixed point (FP)** of $(G, L) \iff_{\text{def}} x$ is LFP for X

Proposition 5.2 Let $G = (X, E)$ be an undirected graph, L a set of local transitions over D . Let $x \in D^n$ be a configuration.

- (1) If x is LFP for $U' \subseteq X$ and x is LFP for $U'' \subseteq X$, then x is LFP for $U' \cup U''$
- (2) x is LFP for $U \subseteq X \iff x$ is LFP for all $U' \subseteq U$

Corollary 5.3 Let $G = (X, E)$ be undirected graph, L a set of local transitions over D . A configuration $x \in D^n$ is FP of $(G, L) \iff$ for all update schedules

$$\alpha : \{1, \dots, T\} \rightarrow \mathcal{P}(X), \mathbf{F}_{(L, \alpha)}(x) = x$$

Consider the following computational problem:

Let \mathcal{F} be a class of functions.

Let \mathcal{G} be a class of graphs.

Problem: $\text{FP}(\mathcal{F}, \mathcal{G})$

Input: undirected graph $G \in \mathcal{G}$ of size n , set $L = \{f_1, \dots, f_n\} \subseteq \mathcal{F}$ of local transition functions

Question: Is there a fixed point of (G, L) ? (If the answer is “yes”, find some?)

Remark: We do not want to check whether $G \in \mathcal{G}$ or $f_i \in \mathcal{F}$; membership is always guaranteed.

5.1.1 Boolean Ensembles

Complexity depends on input representation.

Three cases for boolean function:

- FP_T : local transitions given by lookup tables
- FP_F : local transitions given by formulas (over bases)
- FP_C : local transitions given by circuits (over bases)

Input sizes of $(G, \{f_1, \dots, f_n\})$ for $G = (X, E)$:

$$\|X\| + \|E\| + \sum_{i \in X} |f_i|$$

Size $|f_i|$ for $i \in X$ is as follows:

- FP_T : tables with 2^{1+d_i} rows (d_i id degree of i); each row has $2 + d_i$ entries, i.e.,

$$|f_i| = \Theta\left((2 + d_i) 2^{1+d_i}\right)$$

alternative representation: Wolfram number = bit-vector, e.g.,
Rule 146 corresponds to $(10010010)_2$, i.e.,

x1	x2	x3	f
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

- FP_F : number of symbols encoding a certain basis + number of variables occurring in the formula.
- FP_C : number of wires connecting the gates of the circuit.

Goal: Identifying “islands of tractability”

Decision problem $L \subseteq \Sigma^*$ is called

- *intractable* $\iff_{\text{def}} L$ is NP-hard, i.e., $\text{SAT} \leq_m^p L$
- *tractable* $\iff_{\text{def}} L$ is solvable in polynomial time

Remarks:

1. For sets $A, B \subseteq \Sigma^*$, $A \leq_m^p B \iff_{\text{def}}$ there is a function f computable in polynomial time such that for all $x \in \Sigma^*$, $x \in A \iff f(x) \in B$.
2. Difference “intractable/tractable” under assumption $P \neq NP$
3. Observe that for $\mathcal{F} \subseteq \mathcal{F}'$, $\mathcal{G} \subseteq \mathcal{G}'$, it holds that

$$\begin{aligned} \text{FP}(\mathcal{F}, \mathcal{G}) &\leq_m^p \text{FP}(\mathcal{F}', \mathcal{G}) \text{ via } f = \text{id} \\ \text{FP}(\mathcal{F}, \mathcal{G}) &\leq_m^p \text{FP}(\mathcal{F}, \mathcal{G}') \text{ via } f = \text{id} \end{aligned}$$

We start with lookup-tables:

Theorem 5.4 (Dichotomy theorem for tables) *Let \mathcal{F} be a boolean clone and let \mathcal{G} be a class of graphs closed under taking minors. If \mathcal{F} contains all selfdual functions ($\mathcal{F} \subseteq D$) and \mathcal{G} contains all planar graphs ($\mathcal{G} \supseteq \text{Forb}_{\preceq}(K_{3,3}, K^5)$) then $\text{FP}_T(\mathcal{F}, \mathcal{G})$ is intractable. Otherwise, $\text{FP}_T(\mathcal{F}, \mathcal{G})$ is tractable.*

Proof postponed until number of lemmas proven...

Consider following clones:

- R_1 , the class of 1-reproducing functions
- R_0 , the class of 0-reproducing functions
- M , the class of monotone functions
- L , the class of linear functions

Lemma 5.5 $\text{FP}_T(R_1, \text{Forb}_{\preceq}(\emptyset))$ is solvable in polynomial time.

Proof: $(1, \dots, 1) \in \{0, 1\}^n$ is always a fixed point. ■

Lemma 5.6 $\text{FP}_T(R_0, \text{Forb}_{\preceq}(\emptyset))$ is solvable in polynomial time.

Proof: $(0, \dots, 0) \in \{0, 1\}^n$ is always a fixed point. ■

Lemma 5.7 $FP_T(M, \text{Forb}_{\preceq}(\emptyset))$ is solvable in polynomial time.

Proof: Let $G = (X, E)$ be an undirected graph. Let L be a set of local transitions f_1, \dots, f_n such that $f_I \in M$.

Observe that for $x, y \in \{0, 1\}^n$: $x \leq y \implies \mathbf{F}_L[X](x) \leq \mathbf{F}_L[X](y)$.

To find a fixed point proceed as follows:

- (1) Start with configuration $x^{(0)} =_{\text{def}} (0, \dots, 0)$
- (2) Iteratively compute $x^{(k)} = \mathbf{F}_L[X](x^{(k-1)})$
- (3) If $x^{(k)} = x^{(k-1)}$ then $x^{(k)}$ is a fixed point

Since $(\{0, 1\}^n, \leq)$ has top element (maximum), the algorithm converges, i.e., there is always a fixed point. Complexity analysis:

- Computing $x^{(k)}$ takes $\mathcal{O}(\sum_{i \in X} |f_i|)$ steps
- number of iterations: $\leq n$

Overall, algorithm runs in time $\mathcal{O}(N \cdot \log N)$ ■

Lemma 5.8 $FP_T(L, \text{Forb}_{\preceq}(\emptyset))$ is solvable in polynomial time.

Proof: Let $G = (X, E)$ be an undirected graph, $n = ||X||$. Let $f_1, \dots, f_n \in L$.

First, we transform $f \in L$ in a formula.

Suppose $f \in L$, i.e., $f(x_1, \dots, x_k) = a_0 \oplus a_1 x_1 \oplus \dots \oplus a_k x_k$.

For each tuple $(a_0, a_1, \dots, a_k) \in \{0, 1\}^k$ check whether

$$f(x_1, \dots, x_k) = a_0 \oplus a_1 x_1 \oplus \dots \oplus a_k x_k;$$

if “yes” then formula H_f is found.

Complexity analysis:

- $\mathcal{O}(k \cdot 2^k)$ per tuple
- 2^{k+1} tuple (a_0, a_1, \dots, a_k)

Overall, complexity (in k): $\mathcal{O}(k \cdot 2^{2k})$. Thus, H_f can be found in time $\mathcal{O}(N^2)$.

Now, consider system of linear equations:

$$x_i = f_i(x_{i_0}, x_{i_1}, \dots, x_{i_k}) \text{ where } k = d_i, i \in \{1, \dots, n\}$$

Use Gaußelimination to solve the system in time $\mathcal{O}(n^3)$. ■

Remark:

Number of fixed points is 2^r where r is rank of the system's matrix.

To restrict graph classes, we use CSPs:

A *constraint satisfaction problem* (CSP) is a triple (X, D, \mathcal{C}) such that

- (i) $X = \{x_1, \dots, x_n\}$ is a set of variables
- (ii) D is the (finite) domain of variables
- (iii) \mathcal{C} is a set of constraints Rx_{i_1}, \dots, x_{i_k} with associated relations Ri_1, \dots, i_k , i.e., \mathcal{C} is a list of pairs $\langle Rx_{i_1}, \dots, x_{i_k}, Ri_1, \dots, i_k \rangle$

A solution or CSP (X, D, \mathcal{C}) is an assignment $I : X \rightarrow D$ such that

$$(I(x_{i_1}), \dots, I(x_{i_k})) \in Ri_1, \dots, i_k \text{ for all } Rx_{i_1}, \dots, x_{i_k} \in \mathcal{C}.$$

Example: We are given $H = (x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2) \wedge (x_2 \vee \bar{x}_3 \vee x_4)$. Then, CSP for H :

- $X = \{x_1, x_2, x_3, x_4\}$
- $D = \{0, 1\}$
- $\mathcal{C}_H = \{Rx_1x_2x_3, Rx_1x_2, Rx_2x_3x_4\}$ with relations
 - $R_{123} = \{(0, 0, 0), (0, 0, 1), (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1), (0, 1, 0)\}$
 - $R_{12} = \{(0, 0), (0, 1), (1, 1)\}$
 - $R_{234} = \{(0, 0, 0), (0, 0, 1), (0, 1, 0), (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1)\}$

Satisfying assignment $I : \{x_1, x_2, x_3, x_4\} \rightarrow \{0, 1\}$ such that

$$I(x_1) = 1, I(x_2) = 2, I(x_3) = I(x_4) = 0$$

is also a solution for (X, D, \mathcal{C}) :

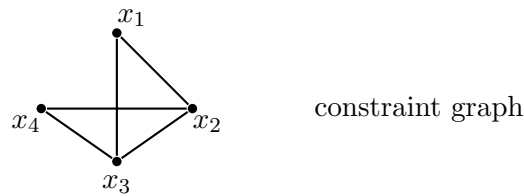
$$\begin{aligned} (1, 1, 0) \in R_{123} & \quad \text{i.e.,} \quad (1, 1, 0) \text{ satisfies } Rx_1x_2x_3 \\ (1, 1) \in R_{12} & \quad \text{i.e.,} \quad (1, 1) \text{ satisfies } Rx_1x_2 \\ (1, 0, 0) \in R_{234} & \quad \text{i.e.,} \quad (1, 0, 0) \text{ satisfies } Rx_2x_3x_4 \end{aligned}$$

That is: H is satisfiable $\iff (X, D, \mathcal{C})$ is solvable. As a consequence: CSP is NP-hard

The *constraint graph* $\Gamma(X, \mathcal{C})$ for (X, D, \mathcal{C}) consists of

- vertex set X
- edge set $E =_{\text{def}} \{\{x_i, x_j\} \mid x_i, x_j \text{ occur in some constraint of } \mathcal{C}\}$

Example: CSP from above:



Theorem 5.9 (Freuder 1990) *A solution of a CSP (X, D, \mathcal{C}) can be found in time $\mathcal{O}(\|D\|^w \cdot p(\|X\| + |\mathcal{C}|))$ for some polynomial p , where w is the treewidth of the constraint graph $\Gamma(X, \mathcal{C})$.*

Lemma 5.10 *Let X be a planar graph. Then, FP_T (BF, $\text{Forb}_{\preceq}(X)$) is solvable in polynomial time.*

Proof: We encode FP problems as CSPs:

Let $G = (V, E)$ be an undirected graph. Let $L = \{f_1, \dots, f_n\}$ be a set of local transition functions. Define $\text{CSP}(G, L) = (X, D, \mathcal{C})$ to be the CSP given as follows:

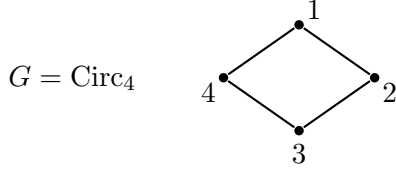
- $X =_{\text{def}} \{x_1, \dots, x_n\}$
- $D =_{\text{def}} \bigcup_{i \in V} D_i$ where

$$D_i =_{\text{def}} \{(I, i) \mid I : N_G^0(i) \rightarrow \{0, 1\} \text{ such that } f_i(I(i_0), \dots, I(i_k)) = I(i)\}$$

- $\mathcal{C} =_{\text{def}} \{E x_i x_j \mid \{i, j\} \in E, i \leq j\}$ where for $i \leq j$

$$E_{ij} =_{\text{def}} \{((I_i, i), (I_j, j)) \mid I_i(k) = I_j(k) \text{ for all } k \in N_G^0(i) \cap N_G^0(j)\} \quad \blacksquare$$

Example:



$$f_i(x_{i-1}, x_i, x_{i+1}) \\ = x_{i-1} \oplus \bar{x}_i \oplus x_{i+1}$$

x_{i-1}	x_i	x_{i+1}	f_i
0	0	0	1
0	<u>0</u>	1	0 [*a]
0	1	0	0
0	<u>1</u>	1	1 [*b]
1	<u>0</u>	0	0 [*c]
1	0	1	1
1	<u>1</u>	0	1 [*d]
1	1	1	0

(*: local FPs)

$\text{CSP}(G, L)$:

- $X = \{ x_0, x_1, x_2, x_3 \}$
 - $D = \{ (0^3, 0^0, 1^1, 0) \rightarrow (0^0, 0^1, 1^2, 1) \rightarrow (0^1, 0^2, 1^3, 2) \rightarrow (0^2, 0^3, 1^0, 3), [*a]$
 $(0^3, 1^0, 1^1, 0) \rightarrow (0^0, 1^1, 1^2, 1) \rightarrow (0^1, 1^2, 1^3, 2) \rightarrow (0^2, 1^3, 1^0, 3), [*b]$
 $(1^3, 0^0, 0^1, 0) \rightarrow (1^0, 0^1, 0^2, 1) \rightarrow (1^1, 0^2, 0^3, 2) \rightarrow (1^2, 0^3, 0^0, 3), [*c]$
 $(1^3, 1^0, 0^1, 0) \rightarrow (1^0, 1^1, 0^2, 1) \rightarrow (1^1, 1^2, 0^3, 2) \rightarrow (1^2, 1^3, 0^0, 3) [*d] \}$
 - $\mathcal{C} = \{ Ex_0x_1, Ex_1x_2, Ex_2x_3, Ex_0x_3 \}$
- $$E_{01} = \{ ((0^3, 0^0, 1^1, 0), (0^0, 1^1, 1^2, 1)), \\ ((0^3, 1^0, 1^1, 0), (1^0, 1^1, 0^2, 1)), \\ ((1^3, 0^0, 0^1, 0), (0^0, 0^1, 1^2, 1)), \\ ((1^3, 1^0, 0^1, 0), (1^0, 0^1, 0^2, 1)) \}$$

(G, L) has FP $0 \mapsto 1, 1 \mapsto 1, 2 \mapsto 0, 3 \mapsto 0$

Define $I'(x_0) = ((0^3, 1^0, 1^1), 0)$

Let $n = ||V||$, $m = ||E||$. Then, the size of $\text{CSP}(G, L)$ is as follows:

- $|X| = n$
- $|D| = \mathcal{O}\left(\sum_{i \in V} (1 + d_i) \cdot 2^{1+d_i}\right)$
- length of constraints and constraint relations:

$$\leq \mathcal{C} \left(\sum_{\{i,j\} \in E} (1 + d_i + 1 + d_j) \cdot 2^{1+d_i} \cdot 2^{1+d_j} \right) \\ \leq \mathcal{C} \left(\sum_{i \in V} (1 + d_i) \cdot 2^{1+d_i} \right)^2$$

Hence, $|\text{CSP}(G, L)| = \mathcal{O}(N^2)$, i.e., encoding f can be computed in polynomial time ($f : (G, L) \mapsto \text{CSP}(G, L)$).

Claim. (G,L) has FP \Leftrightarrow CSP (G,L) has a solution.

\Rightarrow Suppose $I : V \rightarrow \{0,1\}$ is FP of (G,L) .

Define an assignment $I' : \{x_1, \dots, x_n\} \rightarrow D$ by

$$I'(x_i) = (I[N_G^0(i)], i)$$

Let $i \in V$ and let I_i denote $I[N_G^0(i)]$. Let $E_{x_i x_j}$ be any constraint of CSP (G,L) and E_{ij} the associated relation, i.e., $\{i, j\} \in E$.

Since I is FP, we have $(I_i, i), (I_j, j) \in D$.

Let $k \in N_G^0(i) \cap N_G^0(j)$. Then,

$$\begin{aligned} I_i(k) &= (I[N_G^0(i)])(k) \\ &= (I[N_G^0(j)])(k) = I_j(k), \end{aligned}$$

i.e., $((I_i, i), (I_j, j)) \in E_{ij}$

\Leftarrow Suppose I is a solution of CSP (G, L) .

Let $\underbrace{(I_i, i)}_{I(x_i)} \in D$ be the pair assigned to x_i by I .

Define a configuration $I' : V \rightarrow \{0,1\}$ by $I'(i) =_{\text{def}} I_i(i)$

It suffices to show $I'[N_G^0(i)] = I_i$ for $i \in V$. (*)

Let $j \in N_G^0(i)$. If $j = i$ then there is nothing to show.

Let $j \neq i$. Since $\{i, j\} \in E$, $I_i(k) = I_j(k)$ for all $k \in N_G^0(i) \cap N_G^0(j)$. Thus,

$$I'(j) = I_j(j) = I_i(j).$$

Hence, (*) is true.

Finally, observe that $G \cong \Gamma(X, \mathcal{C})$.

Since $G \in \text{Forb}_{\leq}(X)$ for planar X , G has treewidth $\leq w_x$.

So, $\Gamma(X, \mathcal{C})$ has treewidth $\leq w_x$.

Consequently, CSP (G,L) can be solved in time $\mathcal{O}(N^{w_x} p(n + N^2))$ by theorem 9.

Hence, FP of (G,L) can be found in $\mathcal{O}(\text{poly}(N))$. ■

We turn to intractable cases.

Reduction is from Planar 3-SAT:

Input: 3CNF $H = C_1 \wedge \dots \wedge C_m$ having variables x_1, \dots, x_n such that graph representation

$\Gamma(H) = (\{x_1, \dots, x_n, C_1, \dots, C_m\}, \{\{x_i, C_j\} \mid x_i \text{ occurs in } C_j\})$
is a planar (bipartite) graph.

Question: Is H satisfiable?

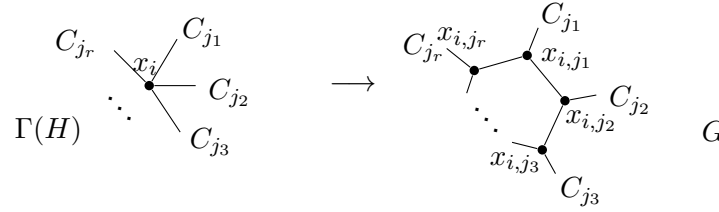
Proposition 5.11 (Lichtenstein 1982) *Planar 3-SAT is NP-complete.*

Lemma 5.12 $FP_T(BF, \text{Forb}_{\leq}(K_{3,3}, K^5))$ is NP-complete, even restricted to graphs having maximum vertex-degree ≤ 3 .

Proof: Let $H = C_1 \wedge, \dots, C_m$ be a 3CNF, x_1, \dots, x_n variables, and $\Gamma(H)$ a planar graph representation where $V(\Gamma(H)) = U \cup W$, $U = \{x_1, \dots, x_n\}$, $W = \{C_1, \dots, C_m\}$.

Construct the following pair (G,L):

- compute an embedding of $\Gamma(H)$ in the plane (in linear time)
- replace vertices of U as follows: Let $x_i \in U$ and suppose $C_{j_1}, \dots, C_{j_r} \in W$ are x_i 's neighbors clockwise ordered. Replace x_i by a cycle $\{x_{i,j_1}, \dots, x_{i,j_r}\}$ such that



- let G be the graph obtained from $\Gamma(H)$ by all replacements.

Observations:

- (i) G is planar, i.e., $G \in \text{Forb}_{\leq}(K_{3,3}, K^5)$
- (ii) $\Delta(G) \leq 3$
- (iii) G can be computed in polynomial time in $|H|$

- Define local transitions on vertices of G :

$$\begin{aligned}
 f_{C_i}(I(C_i), \overbrace{I(x_{i_1, j_1})}^{z_{i_1, j_1}}, \overbrace{I(x_{i_2, j_2})}^{z_{i_2, j_2}}, \overbrace{I(x_{i_3, j_3})}^{z_{i_3, j_3}}) \\
 &=_{\text{def}} \begin{cases} 1 & \text{if } z_{i_1, j_1}, \dots, z_{i_3, j_3} \text{ satisfies } C_i \\ \text{not}(I(C_i)) & \text{otherwise} \end{cases} \\
 f_{x_{i,j}}(I(C_j), I(x_{i,k_0}), \dots, I(x_{i,k_r}))^{(r < 3)} \\
 &=_{\text{def}} \begin{cases} I(x_{ij}) & \text{if } I(x_{i,k_0}) = \dots = I(x_{i,k_r}) \\ \text{not}(I(x_{ij})) & \text{otherwise} \end{cases}
 \end{aligned}$$

Note: f can be computed on polynomial time, since $\Delta(G) = 3$

It holds that:

$$\begin{aligned}
 I \text{ is FP of } (G,L) &\iff I(x_{i,j_1}) = \dots = I(x_{i,j_r}) \text{ for all } i \in \{1, \dots, n\} \text{ and} \\
 &I(C_j) = 1 \text{ for all } j \in \{1, \dots, m\}
 \end{aligned}$$

Then, H is satisfiable $\iff (G,L)$ has a fixed point.

Hence, Planar 3-SAT $\leq_m^n FP_T (BF, \text{Forb}_{\preceq}(K_{3,3}, K^5))$ ■

Proposition 5.13 *Let $n \in \mathbb{N}_+$. For each $f : \{0,1\}^k \rightarrow \{0,1\}$, the function $sd_n(f) : \{0,1\}^{k+n+1} \rightarrow \{0,1\}$ defined by*

$$sd_n(f)(x_1, \dots, x_k, y_1, \dots, y_n, z) =_{\text{def}} \begin{cases} f(x_1, \dots, x_k) & \text{if } y_1 = \dots = y_n = 0 \\ f(\bar{x}_1, \dots, \bar{x}_k) & \text{if } y_1 = \dots = y_n = 1 \\ \bar{z} & \text{otherwise} \end{cases}$$

Proof: Exercise

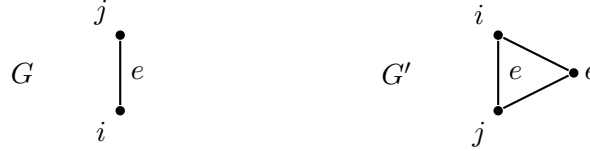
Lemma 5.14 $FP_T (D, \text{Forb}_{\preceq}(K_{3,3}, K^5))$ is NP-complete.

Proof: Let $G = (V, E)$ be a planar graph, $L = \{f_1, \dots, f_n\}$ set of local transitions. Construct (G, L') as follows:

- $G' = (V', E')$ is given by

$$V' =_{\text{def}} V \cup E$$

$$E' =_{\text{def}} E \cup \{\{i, e\} \mid i \in V, e \in E, i \in e\}$$



Note: G' is planar, since G is planar.

- Define local transitions f' on V' :

- For $i \in V$, let $\{i_0, \dots, i_k\} = N_{G'}^0(i) \cup V$, $\{e_1, \dots, e_k\} = N_{G'}^0(i) \cap E$:

$$f'_i(x_{i_0}, \dots, x_{i_n}, x_{e_1}, \dots, x_{e_n}) =_{\text{def}} sd_k(f)(x_{i_0}, \dots, x_{i_k}, x_{e_1}, \dots, x_{e_k}, x_i)$$

Note: f'_i is selfdual, i.e., $f \in D$

Note: degree of i is $2k$ where k is degree of i in G , so f'_i can be computed in polynomial time

- For $e = \{i, j\} \in E$ define

$$f'_e(x_i, x_j, x_{\{i,j\}}) =_{\text{def}} x_{\{i,j\}}$$

Note: f'_e is selfdual

It holds (G, L) has FP if and only if (G', L') has FP \rightarrow proof in script ■

Proof: [Theorem 4] Let \mathbf{F} be a boolean clone. Let \mathcal{G} be a graph class closed under taking minors.

If $\mathbf{F} \supseteq D$ and $\mathcal{G} \supseteq \text{Forb}_{\preceq}(K_{3,3}, K^5)$ then $\text{FP}_T(\mathbf{F}, \mathcal{G})$ is NP-complete (Lemma 14).

Suppose $\mathbf{F} \not\supseteq D$ or $\mathcal{G} \not\supseteq \text{Forb}_{\preceq}(K_{3,3}, K^5)$.

Case 1: $\mathbf{F} \not\supseteq D$, i.e., $\mathbf{F} \not\supseteq R_0, R_1, M, L$. In each of these cases, $\text{FP}_T(\mathbf{F}, \mathcal{G})$ is tractable. (Lemma 5-8).

Case 2: $\mathcal{G} \not\supseteq \text{Forb}_{\preceq}(K_{3,3}, K^5)$. Let $\mathcal{G} = \text{Forb}_{\preceq}(X_1, \dots, X_n)$.

Then, there is $X_i \in \{X_1, \dots, X_n\}$ such that X_i is planar.

Hence, $\mathcal{G} \subseteq \text{Forb}_{\preceq}(X_i)$. By Lemma 10, $\text{FP}_T(\mathcal{F}, \mathcal{G})$ is tractable. ■

Mathematical Tools

A

In this chapter we discuss relevant terminology and notation for sets, relations, and graphs, some fundamental algorithms, and a few other mathematical preliminaries.

A.1 Sets and relations

We denote the set of integers by \mathbb{Z} , the set of non-negative integers by \mathbb{N} , and the set of positive integers by \mathbb{N}_+ . \mathbb{Z}_2 denotes the Galois field $\text{GF}[2]$.

Sets

The empty set is denoted by \emptyset . For an arbitrary set A , $\mathcal{P}(A)$ denotes the power set of A , i.e., the family of all subsets of A , and $\mathcal{P}_+(A)$ denotes the set $\mathcal{P}(A) \setminus \{\emptyset\}$. For an arbitrary finite set A , its cardinality is denoted by $\|A\|$. Let A and B be any sets. Then $A \setminus B$ denotes the difference of A with B , i.e., the set of all elements that are in A but not in B . $A \times B$ denotes the cartesian product, i.e., the set of all pairs (a, b) with $a \in A$ and $b \in B$. For $m \in \mathbb{N}_+$, define $A^m \stackrel{\text{def}}{=} \underbrace{A \times \cdots \times A}_{m \text{ times}}$. Let M be any fixed basic set. For a set

$A \subseteq M$, its complement in the basic set M is denoted by \bar{A} , i.e., $\bar{A} = M \setminus A$. A multiset A is allowed to contain elements many times. The multiplicity of an element x in a multiset A is the number of occurrences of x in A . The cardinality of a multiset A is also denoted by $\|A\|$.

Functions

Let M and M' be any sets, and let $f : M \rightarrow M'$ by any function. The domain of f which we denote by D_f is the set of all $x \in M$ such that $f(x)$ is defined. A function f is total if the domain of f is M . For a set $A \subseteq D_f$, let $f(A) = \{f(x) \mid x \in A\}$ denote the image of A under f . In particular, the range of f which is denoted by R_f is the set $f(D_f)$. For a set $A \subseteq M$, the restriction of a total function f to A is denoted by $f[A]$. The inverse of f is denoted by f^{-1} , i.e., $f^{-1} : M' \rightarrow \mathcal{P}(M)$ such that for all $y \in M'$, $f^{-1}(y) = \{x \in M \mid f(x) = y\}$. If $f^{-1}(y)$ is at most a singleton then we omit the braces. The pre-image of A under f is the set $f^{-1}(A) = \{x \in M \mid f(x) \in A\}$.

We use two notations for composition of functions. If f and f' are functions with $f : M \rightarrow M'$ and $f' : M' \rightarrow M''$, then $(f' \circ f)$ is the function mapping from M to

M'' which is defined for all $x \in M$ as $(f' \circ f)(x) \stackrel{\text{def}}{=} f'(f(x))$. In contrast, we use $f \cdot f'$ to denote $f' \circ f$.

A function $f : M \rightarrow M'$ is bijective if f is surjective, i.e., $R_f = M'$ and injective, i.e., for all $y \in R_f$, $f^{-1}(y)$ is a singleton. Suppose $M' = M$ and M is finite. In this case a bijective function f is a permutation. Suppose $M = \{1, 2, \dots, n\}$. A *cycle* $(i_1 i_2 \dots i_k)$ of length k of the permutation $\pi : M \rightarrow M$ is a sequence (i_1, i_2, \dots, i_k) such that $\pi(i_j) = i_{j+1}$ for $1 \leq j < k$ and $\pi(i_k) = i_1$. Each permutation allows a decomposition into cycles.

Orders

In more detail the following can be found in any textbook (e.g., [Grä78, DP90]) about theory of orders and lattices.

Let P be any set. A *partial order* on P (or order, for short) is a binary relation \leq on P that is reflexive, antisymmetric, and transitive. The set P equipped with a partial order \leq is said to be a *partially ordered set* (for short, *poset*). Usually, we talk about the poset P . Where it is necessary we write (P, \leq) to specify the order. A poset P is a *chain* if for all $x, y \in P$ it holds that $x \leq y$ or $y \leq x$ (i.e., any two elements are comparable with respect to \leq). Such an order is also called a *total order*. A poset P is an *antichain* if for all $x, y \in P$ it holds that $x \leq y$ implies that $x = y$ (i.e., no two elements are comparable with respect to \leq).

We consider \mathbb{N} to be ordered by standard total order on the natural numbers. If a set A is partially ordered by \leq then A^m can be considered to be ordered by the *vector-ordering*, i.e., $(x_1, \dots, x_m) \leq (y_1, \dots, y_m)$ if and only if for all $i \in \{1, \dots, m\}$, $x_i \leq y_i$.

An important tool for representing posets is the *covering relation* \prec . Let P be a poset and let $x, y \in P$. We say that x is covered by y (or y covers x), and write $x \prec y$, if $x < y$ and $x \leq z < y$ implies that $x = z$. The latter condition is demanding that there be no element z of P with $x < z < y$. A finite poset P can be drawn in a diagram consisting of points (representing the elements of P) and interconnecting lines (indicating the covering relation) as follows: To each element x in P associate a point $P(x)$ in the picture which is above all points $P(y)$ associated to elements y less than x , and connect points $P(x)$ and $P(y)$ by a line if and only if $x \prec y$. A poset can have different representation by diagrams.

Let P and P' be posets. A map $\varphi : P \rightarrow P'$ is said to be *monotone* (or order-preserving) if $x \leq y$ in P implies $\varphi(x) \leq \varphi(y)$ in P' . We say that φ is an *(order-)isomorphism* if φ is monotone, injective, and surjective. Two posets P and P' are *isomorphic*, in symbols $P \cong P'$, if there exists an isomorphism $\varphi : P \rightarrow P'$. Isomorphic poset shall be considered to be not essentially different: Two finite posets are isomorphic if and only if they can be drawn with identical diagrams.

Words

Sometimes we make no difference between m -tuples (x_1, \dots, x_m) over a finite set M and words $x_1 \dots x_m$ of length m over M . Such finite sets are called alphabets. Let Σ be a finite alphabet. Σ^* is the set of all finite words that can be built with letters from Σ . For $x, y \in \Sigma^*$, $x \cdot y$ (or xy for short) denotes the concatenation of x and y . The empty word is denoted by ε . For a word $x \in \Sigma^*$, $|x|$ denotes the length of x . For $n \in \mathbb{N}$, Σ^n is the set of all words $x \in \Sigma^*$ such that with $|x| = n$. For a word $x = x_1 \dots x_n \in \Sigma^*$ any word $x_1 \dots x_k$ such that $k \leq n$ is called a prefix of x . We use regular expressions to describe subsets of Σ^* (see, e.g., [HMU01]).

A.2 Graph theory

A graph $G = (V, E)$ consists of a set V of vertices and a set E of edges joining pairs of vertices. The vertex set and edge set of a graph G are denoted by $V(G)$ and $E(G)$, respectively. The cardinality of V is usually denoted by n , the cardinality of E by m . If two vertices are joined by an edge, they are adjacent and we call them *neighbors*. Graphs can be undirected and directed. In undirected graphs, the order in which vertices are joined is irrelevant. An undirected edge joining vertices $u, v \in V$ is denoted by $\{u, v\}$. In directed graphs, each directed edge has an *origin* and a *destination*. An edge with origin $u \in V$ and destination $v \in V$ is represented by an ordered pair (u, v) . For a directed graph $G = (V, E)$, the *underlying undirected graph* is the undirected graph with vertex set V that has an undirected edge between two vertices $u, v \in V$ if (u, v) or (v, u) is in E .

Multigraphs

In both undirected and directed graphs, we may allow the edge set E to contain the same edge several times, i.e., E can be a multiset. If an edge occurs several times in E , the copies of that edge are called *parallel edges*. Graphs with parallel edges are also called *multigraphs*. A graph is called *simple*, if each of its edges is contained in E only once, i.e., if the graph does not have parallel edges. An edge joining a vertex to itself, is called a *loop*. A graph is called *loopless* if it has no loops. In general, we assume all graphs to be loopless unless specified otherwise.

Degrees

The *degree* of a vertex v in an undirected graph $G = (V, E)$, denoted by d_v , is the number of edges in E joining v . If G is a multigraph, parallel edges are counted according to their multiplicity in E . The set of neighbors of v is denoted by $N(v)$. $N^0(v)$ denotes the vertex set $N(v) \cup \{v\}$. If the graph under consideration is not clear from the context, these notations can be augmented by specifying the graph as an index. For example, $N_G(v)$ denotes the neighborhood of v in G .

Subgraphs

A graph $G' = (V', E')$ is a *subgraph* of the graph $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. Sometimes we denote this by $G' \subseteq G$. It is a *(vertex-)induced subgraph* if E' contains all edges $e \in E$ that join vertices in V' . The induced subgraph of $G = (V, E)$ with vertex set $V' \subseteq V$ is denoted by $G[V']$. The *(edge-)induced subgraph* with edge set $E' \subseteq E$, denoted by $G[E']$, is the subgraph $G' = (V', E')$ of G , where V' is the set of all vertices in V that are joined by at least one edge in E' .

Walks, paths, and cycles

A *walk* from x_0 to x_k in a graph $G = (V, E)$ is a sequence $x_0, e_1, x_1, e_2, x_2, \dots, x_{k-1}, e_k, x_k$ alternating between vertices and edges of G , where $e_i = \{x_{i-1}, x_i\}$ in the undirected case and $e_i = (x_{i-1}, x_i)$ in the directed case. The length of a walk is the number of edges on the walk. As shorthands we use (x_0, x_1, \dots, x_k) and (e_1, e_2, \dots, e_k) to denote a walk. The walk is called a *path* if $x_i \neq x_j$ for $i \neq j$. A walk with $x_0 = x_k$ is called a *cycle* if $e_i \neq e_j$ for $i \neq j$. A cycle is a *simple cycle* if $x_i \neq x_k$ for $0 \leq i < j \leq k - 1$.

Special graphs

A *tree* is a connected (for a definition see below) undirected graph not containing a cycle. An undirected graph $G = (V, E)$ is called *complete* if it contains all possible pairs of vertices as edges. A complete graph with n vertices is denoted by K^n . A K^n is called a *clique*. A K^2 is a graph of two vertices with one edge joining them. A K^3 is also called a *triangle* or *triad*. A graph without edges is called *empty*. An *independent set* within a graph $G = (V, E)$ is a vertex set $U \subseteq V$ such that $G[U]$ is empty. A graph $G = (V, E)$ is called *bipartite* if there are independent vertex sets $V_1, V_2 \subseteq V$ such that V_1 and V_2 are disjoint and $V_1 \cup V_2 = V$. We denote by $E(V_1, V_2)$ the set of edges joining vertices from V_1 with vertices from V_2 . If $E(V_1, V_2) = V_1 \times V_2$ then G is called a *complete bipartite graph*. Such a graph is denoted by K_{n_1, n_2} if V_1 consists of n_1 vertices and V_2 of n_2 vertices. A $K_{1, n}$ is also called a *star*. For two graphs $G = (V, E)$ and $G' = (V', E')$ we denote by $G \oplus G'$ the graph consisting of the disjoint union of the graphs G and G' .

Graph classes

Two graphs $G = (V, E)$ and $G' = (V', E')$ are *isomorphic*, denoted by $G \simeq G'$, if there is a bijective mapping $\varphi : V \rightarrow V'$ such that for all vertices $u, v \in V$ the following is true: in the case that G and G' are directed graphs it holds that $(u, v) \in E \Leftrightarrow (\varphi(u), \varphi(v)) \in E'$, and in the case that G and G' are undirected graphs it holds that $\{u, v\} \in E \Leftrightarrow \{\varphi(u), \varphi(v)\} \in E'$. A set of graphs is called a *graph class* if for each graph G in the class all graphs isomorphic to G belong to the class as well.

A.3 Algorithmics

Most results of this work relate to algorithms. In the following we mention essential problems and concepts which are needed more than once.

For two functions $f : \mathbb{N} \rightarrow \mathbb{N}$ and $g : \mathbb{N} \rightarrow \mathbb{N}$ we say that f is in $O(g)$ if there are constant $n_0, c \in \mathbb{N}_+$ such that for all $n \geq n_0$, $f(n) \leq c \cdot g(n)$. We say that f is in $\Omega(g)$ if g is in $O(f)$. We say that f is in $\Theta(g)$ if f is in $O(g) \cap \Omega(g)$.

Connected components

An undirected graph $G = (V, E)$ is *connected* if every vertex can be reached from every other vertex, i.e., if there is a path from every vertex to every other vertex. A graph consisting of a single vertex is also taken to be connected. Graphs that are not connected are called *disconnected*. For a given undirected graph $G = (V, E)$, a *connected component* of G is an induced subgraphs $G' = (V', E')$ that is connected and maximal, i.e., there is no connected subgraph $G'' = (V'', E'')$ such that $V'' \supset V'$. Checking whether a graph is connected and finding all its connected components can be done in time $O(n + m)$ using depth-first search or breadth-first search.

A directed graph $G = (V, E)$ is *strongly connected* if there is a directed path from every vertex to every other vertex. A *strongly connected component* of a directed graph G is an induced subgraph that is strongly connected and maximal. The strongly connected components of a directed graph can be computed in time $O(n + m)$ using a depth-first search.

NP-completeness

It is important to consider the running-time of an algorithm for a given problem. Usually, one wants to give an upper bound on the running time of the algorithm for inputs of a certain size. If the running-time of an algorithm is $O(n^k)$ for some $k \in \mathbb{N}$ and for inputs of size n , we say that the algorithm runs in polynomial time. For graph problems, the running-time is usually specified as a function of n and m , the number of vertices and edges of the graph, respectively. For many problems, however, no polynomial-time algorithm has been discovered. Although one cannot rule out the possible existence of polynomial-time algorithms for such problems, the theory of NP-completeness provides means to give evidence for the computational intractability of a problem.

A decision problem is in the complexity class NP if there is a nondeterministic Turing machine that solves the problem in polynomial time. That is to say that the answer to a problem instance is “yes” if there exists a solution in the set of all possible solutions to the instance which is of polynomial size. Moreover, the test whether a potential solution is an actual solution must be performed in polynomial time. Note that a decision problem

is usually considered to consist of the set of the “yes”-instances. A decision problem is NP-hard if every problem in NP can be reduced to it via a polynomial-time many-one reduction. (A polynomial-time many-one reduction from a set A to a set B is a function computable in polynomial time such that for all instances x , $x \in A \Leftrightarrow f(x) \in B$.) Problems that are NP-hard and belong to NP are called NP-*complete*. A polynomial-time algorithm for an NP-hard problem would imply polynomial-time algorithms for all problems NP—something that is considered very unlikely. Therefore, the NP-hardness of a problem is considered substantial evidence for the computational difficulty of the problem.

A standard example of an NP-complete problem is 3SAT, i.e., checking whether a given propositional formula given as a 3CNF has a satisfying assignment. To be more precise, a k CNF is a formula $H = C_1 \wedge \cdots \wedge C_m$ consisting of clauses C_i each of which has the form $C_i = l_{i1} \vee l_{i2} \vee \cdots \vee l_{ik}$ where l_{ij} is either a positive or a negative literal. A positive literal is some variable, say x_k , and a negative literal is the negation of some variable, say $\overline{x_k}$.

The class of complements of NP sets is denoted by coNP, i.e., $\text{coNP} = \{\overline{A} \mid A \in \text{NP}\}$.

For optimization problems (where the goal is to compute a feasible solution that maximizes or minimizes some objective function), we say that the problem is NP-*hard* if the corresponding decision problem (checking whether a solution with objective value better than a given value k exists) is NP-hard.

#P-completeness

A complexity class closely related to NP is the class #P which has been introduced in [Val79a, Val79b] to provide evidence for the computational intractability of counting problems. The class #P consists of all problems of the form “compute $f(x)$ ” where $f(x)$ is the number of accepting paths of a nondeterministic Turing machine running in polynomial time. Equivalently, a #P-function counts the number of solutions to instances of an NP-problem. We say that a function f is #P-*complete* if it belongs to #P and every function $g \in \#P$ is polynomial-time Turing reducible to f , i.e., g can be computed by a deterministic polynomial-time Turing machines which is allowed to make queries to f and answering these queries is done within one step (see, e.g., [HMU01, HO02]). The canonical example of a #P-complete problem is #3SAT, i.e., counting the number of satisfying assignments of a propositional formula given as a 3CNF. One of the most prominent #P-complete problem is counting the number of perfect matchings in a bipartite graph [Val79b]. As in the case of NP, if there is a polynomial-time algorithm for computing some #P-complete function from #P then there are polynomial-time algorithms for all #P-functions—which is equally considered unlikely. In particular, such a polynomial-time algorithm would imply that $\text{P} = \text{NP}$.

Bibliography

- [Bar93] Yaneer Bar-Yam. *Dynamics of Complex Systems*. The Advanced Book Program, Addison Wesley, Reading, MA, 1997.
- [Kau93] Stuart A. Kauffman. *The Origins of Order. Self-Organization and Selection in Evolution*. Oxford University Press, Oxford, 1993.
- [MR08] Henning S. Mortveit and Christian M. Reidys. *An Introduction to Sequential Dynamical Systems*. Springer, New York, NY, 2008.
- [Sch93] Heinz Georg Schuster. *Deterministic Chaos*. VCH, Weinheim, 1994.
- [Wo102] Stephen Wolfram. *A New Kind of Science*. Wolfram Media, Champaign, IL, 2002.
- [Grä78] George A. Grätzer. *General Lattice Theory*. Akademie-Verlag, Berlin, 1978.
- [DP90] Brian A. Davey and Hilary A. Priestley. *Introduction to Lattices and Orders*. Cambridge University Press, Cambridge, 1990.
- [HO02] Lane A. Hemaspaandra and Mitsunori Ogihara. *The Complexity Theory Companion*. An EATCS series. Springer-Verlag, Berlin, 2002.
- [HMU01] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation* 2nd edition. Addison Wesley, Reading, MA, 2001.
- [Stan73] Richard P. Stanley. Acyclic Orientations of Graphs. *Discrete Mathematics*, **5**:171–178, 1973.
- [DP86] B. Derrida and Y. Pomeau. Randomized Networks of Automata: A Simple Annealed Approximation. *Europhysics Letters*, **1**(2):45–49, 1986.
- [Val79a] Leslie G. Valiant. The Complexity of Enumeration and Reliability Problems. *SIAM Journal on Computing*, **8**(3):410–421, 1979.
- [Val79b] Leslie G. Valiant. The Complexity of Computing the Permanent. *Theoretical Computer Science*, **8**(2):189–201, 1979

