Abelian networks and a weak version of Merino's theorem

Swee Hong Chan
Cornell University

Banff International Research Station-CMO

November 8, 2015

Merino's Theorem

Merino's Theorem for undirected graphs

Theorem 1 ([ML97])

Let G be a connected, undirected graph with a chosen vertex $s \in V$. If $Rec(G, s)$ is the set of recurrent states of the sandpile model $Sand(G, s)$, then:

$$\sum_{\mathbf{q} \in Rec(G,s)} y^{lvl(\mathbf{q})} = T_G(1; y),$$

where $T_G(x; y)$ is the Tutte polynomial for the graph G.

Corollary 2 (weak version of Merino's Theorem)

The polynomial $\sum_{\mathbf{q} \in Rec(G,s)} y^{lvl(\mathbf{q})}$ does not depend on the choice of sink $s \in V$.

Merino's Theorem for undirected graphs

Theorem 1 ([ML97])

Let G be a connected, undirected graph with a chosen vertex $s \in V$. If $Rec(G, s)$ is the set of recurrent states of the sandpile model $Sand(G, s)$, then:

$$\sum_{\mathbf{q} \in Rec(G,s)} y^{lvl(\mathbf{q})} = T_G(1; y),$$

where $T_G(x; y)$ is the Tutte polynomial for the graph G.

Corollary 2 (weak version of Merino's Theorem)

The polynomial $\sum_{\mathbf{q} \in Rec(G,s)} y^{lvl(\mathbf{q})}$ does not depend on the choice of sink $s \in V$.

A conjecture of Perrot and Pham

Conjecture 1 ([PV13])

Let G be a strongly connected digraph with a chosen vertex $s \in V$. Let $Rec(G, s, \sim)$ be the set of recurrent classes of the sandpile model on the digraph. If the polynomial $\mathcal{P}(G, s; y)$ is given by:

$$\mathcal{P}(G, s; y) = \sum_{[\mathbf{q}] \in Rec(G, s, \sim)} y^{lvl([\mathbf{q}])},$$

then $\mathcal{P}(G, s; y)$ does not depend on the choice of vertex s.

Weak version of Merino's theorem for abelian networks

Theorem 3 ([Cha14])

Let $\mathcal{M}$ be a Markovian network and let $s \in A$ be a letter of $\mathcal{M}$. Let $Rec(G, s, \sim)$ be the set of recurrent classes of $\mathcal{M}_s$. If the polynomial $\mathcal{P}(\mathcal{M}, s; y)$ is given by:

$$\mathcal{P}(\mathcal{M}, s; y) = \sum_{[\mathbf{q}] \in Rec(\mathcal{M}_s; \sim)} y^{lvl([\mathbf{q}])}.$$

Then $\mathcal{P}(\mathcal{M}, s; y)$ does not depend on the choice of sink s for the Markovian network.

Remark

In particular, Theorem 3 verifies Conjecture 1 by taking $\mathcal{M}$ to be the sandpile model on digraphs.

Weak version of Merino's theorem for abelian networks

Theorem 3 ([Cha14])

Let $\mathcal{M}$ be a Markovian network and let $s \in A$ be a letter of $\mathcal{M}$. Let $Rec(G, s, \sim)$ be the set of recurrent classes of $\mathcal{M}_s$. If the polynomial $\mathcal{P}(\mathcal{M}, s; y)$ is given by:

$$\mathcal{P}(\mathcal{M}, s; y) = \sum_{[\mathbf{q}] \in Rec(\mathcal{M}_s; \sim)} y^{lvl([\mathbf{q}])}.$$

Then $\mathcal{P}(\mathcal{M}, s; y)$ does not depend on the choice of sink s for the Markovian network.

Remark

In particular, Theorem 3 verifies Conjecture 1 by taking $\mathcal{M}$ to be the sandpile model on digraphs.

Merino's Theorem (ct'd)

There are several notions in Theorem 3 that needs to be properly defined, namely:

- Markovian network;
- Sink of a Markovian network;
- Recurrent classes;
- Level of a recurrent class.

Abelian networks

Abelian networks

An **abelian network** $\mathcal{N}$ [BL13] consists of the following data:

- The **state space** $Q = \{\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_m\}$;
- The **alphabet space** $A = \{a_1, a_2, \ldots, a_n\}$;
- For each $a \in A$, a **transition function** $t_a : Q \to Q$ and an **output vector** $\mathbf{o}_a : Q \to \mathbb{N}^A$ that satisfies the *abelian property*, i.e.:

$$\begin{aligned} t_a \circ t_b &= t_b \circ t_a \\ \mathbf{o}_b(t_a(\mathbf{q})) + \mathbf{o}_a(\mathbf{q}) &= \mathbf{o}_a(t_b(\mathbf{q})) + \mathbf{o}_b(\mathbf{q}), \end{aligned}$$

for all $a, b \in A$ and $\mathbf{q} \in Q$.

Abelian networks (ct'd)

We use the pair $\mathbf{x}.\mathbf{q}$ to describe the current state $\mathbf{q} \in Q$ of the network and the number of letters $\mathbf{x} \in \mathbb{Z}^A$ waiting to be executed.

Executing a letter $a \in A$ on $\mathbf{x}.\mathbf{q}$ gives us a new pair $\mathbf{x}'.\mathbf{q}'$, where

$$\begin{aligned} \mathbf{q}' &= t_a(\mathbf{q}); \\ \mathbf{x}' &= \mathbf{x} - \mathbf{1}_a + \mathbf{o}_a(\mathbf{q}). \end{aligned}$$

We use $\pi_a(\mathbf{x}.\mathbf{q})$ as a shorthand the pair $\mathbf{x}'.\mathbf{q}'$.

If $\omega = a_1 a_2 \dots a_r \in A^*$, then executing ω on $\mathbf{x}.\mathbf{q}$ gives us:

$$\pi_\omega(\mathbf{x}.\mathbf{q}) = \pi_{a_r} \circ \pi_{a_{r-1}} \circ \cdots \circ \pi_{a_1}(\mathbf{x}.\mathbf{q}).$$

Abelian networks (ct'd)

We use the pair $\mathbf{x}.\mathbf{q}$ to describe the current state $\mathbf{q} \in Q$ of the network and the number of letters $\mathbf{x} \in \mathbb{Z}^A$ waiting to be executed.

Executing a letter $a \in A$ on $\mathbf{x}.\mathbf{q}$ gives us a new pair $\mathbf{x}'.\mathbf{q}'$, where

$$\begin{aligned}\mathbf{q}' &= t_a(\mathbf{q});\\ \mathbf{x}' &= \mathbf{x} - \mathbf{1}_a + \mathbf{o}_a(\mathbf{q}).\end{aligned}$$

We use $\pi_a(\mathbf{x}.\mathbf{q})$ as a shorthand the pair $\mathbf{x}'.\mathbf{q}'$.

If $\omega = a_1 a_2 \dots a_r \in A^*$, then executing ω on $\mathbf{x}.\mathbf{q}$ gives us:

$$\pi_\omega(\mathbf{x}.\mathbf{q}) = \pi_{a_r} \circ \pi_{a_{r-1}} \circ \cdots \circ \pi_{a_1}(\mathbf{x}.\mathbf{q}).$$

Abelian networks (ct'd)

We use the pair $\mathbf{x}.\mathbf{q}$ to describe the current state $\mathbf{q} \in Q$ of the network and the number of letters $\mathbf{x} \in \mathbb{Z}^A$ waiting to be executed.

Executing a letter $a \in A$ on $\mathbf{x}.\mathbf{q}$ gives us a new pair $\mathbf{x}'.\mathbf{q}'$, where

$$\begin{aligned} \mathbf{q}' &= t_a(\mathbf{q}); \\ \mathbf{x}' &= \mathbf{x} - \mathbf{1}_a + \mathbf{o}_a(\mathbf{q}). \end{aligned}$$

We use $\pi_a(\mathbf{x}.\mathbf{q})$ as a shorthand the pair $\mathbf{x}'.\mathbf{q}'$.

If $\omega = a_1 a_2 \dots a_r \in A^*$, then executing ω on $\mathbf{x}.\mathbf{q}$ gives us:

$$\pi_\omega(\mathbf{x}.\mathbf{q}) = \pi_{a_r} \circ \pi_{a_{r-1}} \circ \cdots \circ \pi_{a_1}(\mathbf{x}.\mathbf{q}).$$

Production matrix

We say that a network $\mathcal{N}$ is **irreducible** if for any $\mathbf{q}, \mathbf{q}' \in Q$ there exists a word $\omega \in A^*$ such that $t_\omega(\mathbf{q}) = \mathbf{q}'$.

The **production matrix** $P \in \mathrm{Mat}(A, \mathbb{Q})$ is a square matrix indexed by A such that the column of P associated to $a \in A$ is equal to:

$$P_a = \lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^{n} \mathbf{o}_a((t_a)^i(\mathbf{q})).$$

Halting networks

An irreducible abelian network $\mathcal{H}$ is called a **halting network** if the production matrix P has Perron eigenvalue strictly less than 1.

If $\mathcal{H}$ is a halting network, then for each $a \in A$ there exists a **global action** $\tau_a : Q \to Q$ that sends one state of $\mathcal{H}$ to another state.

A state $\mathbf{q} \in Q$ of $\mathcal{H}$ is **recurrent** if for all $\mathbf{q}' \in Q$ there exists a word $\omega = a_1 \dots a_r \in A^*$ such that

$$\tau_{a_r} \circ \cdots \tau_{a_1}(\mathbf{q}') = \mathbf{q}.$$

Halting networks

An irreducible abelian network $\mathcal{H}$ is called a **halting network** if the production matrix P has Perron eigenvalue strictly less than 1.

If $\mathcal{H}$ is a halting network, then for each $a \in A$ there exists a **global action** $\tau_a : Q \to Q$ that sends one state of $\mathcal{H}$ to another state.

A state $\mathbf{q} \in Q$ of $\mathcal{H}$ is **recurrent** if for all $\mathbf{q}' \in Q$ there exists a word $\omega = a_1 \dots a_r \in A^*$ such that

$$\tau_{a_r} \circ \cdots \tau_{a_1}(\mathbf{q}') = \mathbf{q}.$$

Markovian networks

Markovian networks

We say that an irreducible abelian network $\mathcal{M}$ is **Markovian** if

- the production matrix P is an irreducible matrix;
- $\mathbf{1}^\top \cdot P = \mathbf{1}^\top$.

Examples of Markovian networks

Sandpile network

Let $G = (V, E)$ be a strongly-connected digraph and let $V = \{1, \ldots, n\}$.

The alphabet space A of Sand(G) is the vertex set V.
The state space Q of Sand(G) is $\prod_{i=1}^{n} \mathbb{Z}_{\text{outdeg}(i)}$.

For each $i \in V$, the transition function $t_i : Q \to Q$ is defined by

$$t_i(z_1, z_2, \ldots, z_n) = (z_1, \ldots, z_{i-1}, z_i + 1, z_{i+1}, \ldots, z_n).$$

The output vector $\mathbf{o}_i : Q \to \mathbb{N}^A$ is defined by:

$$\mathbf{o}_i(z_1, \ldots, z_n) = \begin{cases} \mathbf{0} & \text{if } z_i \neq \text{outdeg}(i) - 1; \\ (a_{ji})_{j \in V} & \text{if } z_i = \text{outdeg}(i) - 1. \end{cases}$$

Examples of Markovian networks

Sandpile network

Let $G = (V, E)$ be a strongly-connected digraph and let $V = \{1, \ldots, n\}$.

The alphabet space A of $\mathrm{Sand}(G)$ is the vertex set V.
The state space Q of $\mathrm{Sand}(G)$ is $\prod_{i=1}^{n} \mathbb{Z}_{\mathrm{outdeg}(i)}$.

For each $i \in V$, the transition function $t_i : Q \to Q$ is defined by

$$t_i(z_1, z_2, \ldots, z_n) = (z_1, \ldots, z_{i-1}, z_i + 1, z_{i+1}, \ldots, z_n).$$

The output vector $\mathbf{o}_i : Q \to \mathbb{N}^A$ is defined by:

$$\mathbf{o}_i(z_1, \ldots, z_n) = \begin{cases} \mathbf{0} & \text{if } z_i \neq \mathrm{outdeg}(i) - 1; \\ (a_{ji})_{j \in V} & \text{if } z_i = \mathrm{outdeg}(i) - 1. \end{cases}$$

Examples of Markovian networks (ct'd)

Rotor network

The alphabet space A of Rotor(G) is the vertex set V.
The state space Q of Rotor(G) is $\prod_{i=1}^{n} \mathbb{Z}_{\text{outdeg}(i)}$.
For each $i \in V$, the transition function $t_i : Q \to Q$ is defined by

$$t_i(z_1, z_2, \ldots, z_n) = (z_1, \ldots, z_{i-1}, z_i + 1, z_{i+1}, \ldots, z_n).$$

For each $i \in V$, choose a cyclic ordering $n_{i,0}, n_{i,1}, \ldots, n_{i,\text{outdeg}(i)-1}$ for the neighbors of i in G. The output vector $\mathbf{o}_i : Q \to \mathbb{N}^A$ is defined by:

$$\mathbf{o}_i(z_1, \ldots, z_n) = \mathbf{1}_{n_{i,z_i+1}}.$$

Examples of Markovian networks (ct'd)

Rotor network

The alphabet space A of Rotor(G) is the vertex set V.
The state space Q of Rotor(G) is $\prod_{i=1}^{n} \mathbb{Z}_{\text{outdeg}(i)}$.
For each $i \in V$, the transition function $t_i : Q \to Q$ is defined by

$$t_i(z_1, z_2, \ldots, z_n) = (z_1, \ldots, z_{i-1}, z_i + 1, z_{i+1}, \ldots, z_n).$$

For each $i \in V$, choose a cyclic ordering $n_{i,0}, n_{i,1}, \ldots, n_{i,\text{outdeg}(i)-1}$ for the neighbors of i in G. The output vector $\mathbf{o}_i : Q \to \mathbb{N}^A$ is defined by:

$$\mathbf{o}_i(z_1, \ldots, z_n) = \mathbf{1}_{n_{i,z_i+1}}.$$

Level of a state

Level of a state

Let $\mathcal{M}$ be a Markovian network, choose an initial state $\mathbf{q}_0 \in Q$. The **level** of a state $\mathbf{q} \in Q$ is defined by $\mathrm{lvl}(\mathbf{q}) = \mathbf{1} \cdot \mathbf{x} + 1$, where $\mathbf{x} \in \mathbb{Z}^A$ satisfies $\pi_\omega(\mathbf{x}.\mathbf{q}_0) = \mathbf{0}.\mathbf{q}$.

Remark

The level of a state is well defined and does not depend on the choice of vector $\mathbf{x}$ if $\mathcal{M}$ is Markovian.

Level of a state

Let $\mathcal{M}$ be a Markovian network, choose an initial state $\mathbf{q}_0 \in Q$. The **level** of a state $\mathbf{q} \in Q$ is defined by $\mathrm{lvl}(\mathbf{q}) = \mathbf{1} \cdot \mathbf{x} + 1$, where $\mathbf{x} \in \mathbb{Z}^A$ satisfies $\pi_\omega(\mathbf{x}.\mathbf{q}_0) = \mathbf{0}.\mathbf{q}$.

Remark

The level of a state is well defined and does not depend on the choice of vector $\mathbf{x}$ if $\mathcal{M}$ is Markovian.

Sink of a Markovian network

Markovian network with sink

Let $\mathcal{M}$ be a Markovian network, and let $s \in A$ be a letter of $\mathcal{M}$. The **Markovian network with sink**, denoted by $\mathcal{M}_s$, is the abelian network constructed from $\mathcal{M}$ by removing all the alphabets of type s that is produced by the network.

Example of Markovian networks with sink

Sandpile network with sink

Let $\mathcal{M} = \mathsf{Sand}(G)$, and let $s \in V(G)$.

The alphabet space A of $\mathsf{Sand}(G, s)$ is the vertex set V.
The state space Q of $\mathsf{Sand}(G, s)$ is $\prod_{i=1}^{n} \mathbb{Z}_{\mathrm{outdeg}(i)}$.
For each $i \in V$, the transition function $t_i : Q \to Q$ is defined by

$$t_i(z_1, z_2, \ldots, z_n) = (z_1, \ldots, z_{i-1}, z_i + 1, z_{i+1}, \ldots, z_n).$$

The output vector $\mathbf{o}_i : Q \to \mathbb{N}^A$ is defined by:

$$\mathbf{o}_i(z_1, \ldots, z_n) = \begin{cases} \mathbf{0} & \text{if } z_i \neq \mathrm{outdeg}(i) - 1; \\ (a_{ji} 1_{j \neq s})_{j \in V} & \text{if } z_i = \mathrm{outdeg}(i) - 1. \end{cases}$$

Example of Markovian networks with sink

Sandpile network with sink

Let $\mathcal{M} = \mathrm{Sand}(G)$, and let $s \in V(G)$.

The alphabet space A of $\mathrm{Sand}(G, s)$ is the vertex set V.
The state space Q of $\mathrm{Sand}(G, s)$ is $\prod_{i=1}^{n} \mathbb{Z}_{\mathrm{outdeg}(i)}$.
For each $i \in V$, the transition function $t_i : Q \to Q$ is defined by

$$t_i(z_1, z_2, \ldots, z_n) = (z_1, \ldots, z_{i-1}, z_i + 1, z_{i+1}, \ldots, z_n).$$

The output vector $\mathbf{o}_i : Q \to \mathbb{N}^A$ is defined by:

$$\mathbf{o}_i(z_1, \ldots, z_n) = \begin{cases} \mathbf{0} & \text{if } z_i \neq \mathrm{outdeg}(i) - 1; \\ (a_{ji} 1_{j \neq s})_{j \in V} & \text{if } z_i = \mathrm{outdeg}(i) - 1. \end{cases}$$

Example of Markovian networks with sink

Sandpile network with sink

Let $\mathcal{M} = \mathrm{Sand}(G)$, and let $s \in V(G)$.

The alphabet space A of $\mathrm{Sand}(G, s)$ is the vertex set V.
The state space Q of $\mathrm{Sand}(G, s)$ is $\prod_{i=1}^{n} \mathbb{Z}_{\mathrm{outdeg}(i)}$.
For each $i \in V$, the transition function $t_i : Q \to Q$ is defined by

$$t_i(z_1, z_2, \ldots, z_n) = (z_1, \ldots, z_{i-1}, z_i + 1, z_{i+1}, \ldots, z_n).$$

The output vector $\mathbf{o}_i : Q \to \mathbb{N}^A$ is defined by:

$$\mathbf{o}_i(z_1, \ldots, z_n) = \begin{cases} \mathbf{0} & \text{if } z_i \neq \mathrm{outdeg}(i) - 1; \\ (a_{ji} 1_{j \neq s})_{j \in V} & \text{if } z_i = \mathrm{outdeg}(i) - 1. \end{cases}$$

Markovian networks with sink (ct'd)

- The state space of $\mathcal{M}_s$ is equal to the state space of $\mathcal{M}$.
  Hence we can define the level of a state $\mathbf{q}$ of $\mathcal{M}_s$ to be equal to the level of $\mathbf{q}$ in $\mathcal{M}$.
- The production matrix of $\mathcal{M}_s$ has Perron eigenvalue strictly less than 1, so $\mathcal{M}_s$ is a halting network.
  Therefore, it makes sense to talk about the recurrent states of the network $\mathcal{M}_s$.

Markovian networks with sink (ct'd)

- The state space of $\mathcal{M}_s$ is equal to the state space of $\mathcal{M}$. Hence we can define the level of a state $\mathbf{q}$ of $\mathcal{M}_s$ to be equal to the level of $\mathbf{q}$ in $\mathcal{M}$.
- The production matrix of $\mathcal{M}_s$ has Perron eigenvalue strictly less than 1, so $\mathcal{M}_s$ is a halting network.

  Therefore, it makes sense to talk about the recurrent states of the network $\mathcal{M}_s$.

Markovian networks with sink (ct'd)

- The state space of $\mathcal{M}_s$ is equal to the state space of $\mathcal{M}$. Hence we can define the level of a state $\mathbf{q}$ of $\mathcal{M}_s$ to be equal to the level of $\mathbf{q}$ in $\mathcal{M}$.
- The production matrix of $\mathcal{M}_s$ has Perron eigenvalue strictly less than 1, so $\mathcal{M}_s$ is a halting network.

Therefore, it makes sense to talk about the recurrent states of the network $\mathcal{M}_s$.

Markovian networks with sink (ct'd)

- The state space of $\mathcal{M}_s$ is equal to the state space of $\mathcal{M}$. Hence we can define the level of a state $\mathbf{q}$ of $\mathcal{M}_s$ to be equal to the level of $\mathbf{q}$ in $\mathcal{M}$.
- The production matrix of $\mathcal{M}_s$ has Perron eigenvalue strictly less than 1, so $\mathcal{M}_s$ is a halting network.
  Therefore, it makes sense to talk about the recurrent states of the network $\mathcal{M}_s$.

Recurrent class

Recurrent class

Let $\mathcal{M}$ be a Markovian network and let $s \in A$. We define an equivalence relation $\sim$ on $\mathrm{Rec}(\mathcal{M}_s)$ by $\mathbf{q} \sim \mathbf{q}'$ for $\mathbf{q}, \mathbf{q}' \in \mathrm{Rec}(\mathcal{M}_s)$ if $\mathbf{q}' = (\tau_s)^k(\mathbf{q})$ for some $k \in \mathbb{N}$.

A **recurrent class** $[\mathbf{q}]$ is the equivalence class of an element $\mathbf{q} \in \mathrm{Rec}(\mathcal{M}_s)$.

The **level** of a recurrent class $[\mathbf{q}]$ is defined by

$$\mathrm{lvl}([\mathbf{q}]) = \max_{\mathbf{q} \in [\mathbf{q}]} \mathrm{lvl}(\mathbf{q}).$$

Recurrent class

Let $\mathcal{M}$ be a Markovian network and let $s \in A$. We define an equivalence relation $\sim$ on $\mathrm{Rec}(\mathcal{M}_s)$ by $\mathbf{q} \sim \mathbf{q}'$ for $\mathbf{q}, \mathbf{q}' \in \mathrm{Rec}(\mathcal{M}_s)$ if $\mathbf{q}' = (\tau_s)^k(\mathbf{q})$ for some $k \in \mathbb{N}$.

A **recurrent class** $[\mathbf{q}]$ is the equivalence class of an element $\mathbf{q} \in \mathrm{Rec}(\mathcal{M}_s)$.

The **level** of a recurrent class $[\mathbf{q}]$ is defined by

$$\mathrm{lvl}([\mathbf{q}]) = \max_{\mathbf{q} \in [\mathbf{q}]} \mathrm{lvl}(\mathbf{q}).$$

Recurrent class

Let $\mathcal{M}$ be a Markovian network and let $s \in A$. We define an equivalence relation $\sim$ on $\mathrm{Rec}(\mathcal{M}_s)$ by $\mathbf{q} \sim \mathbf{q}'$ for $\mathbf{q}, \mathbf{q}' \in \mathrm{Rec}(\mathcal{M}_s)$ if $\mathbf{q}' = (\tau_s)^k(\mathbf{q})$ for some $k \in \mathbb{N}$.

A **recurrent class** $[\mathbf{q}]$ is the equivalence class of an element $\mathbf{q} \in \mathrm{Rec}(\mathcal{M}_s)$.
The **level** of a recurrent class $[\mathbf{q}]$ is defined by

$$\mathrm{lvl}([\mathbf{q}]) = \max_{\mathbf{q} \in [\mathbf{q}]} \mathrm{lvl}(\mathbf{q}).$$

A weak version of Merino's Theorem

A weak version of Merino's Theorem

Theorem 3

If $\mathcal{M}$ is a Markovian network and $s \in A$ is a letter of $\mathcal{M}$, then the polynomial $\mathcal{P}(\mathcal{M}, s; y)$ does not depend on the choice of s:

$$\mathcal{P}(\mathcal{M}, s; y) = \sum_{[\mathbf{q}] \in Rec(\mathcal{M}_s; \sim)} y^{lvl([\mathbf{q}])}.$$

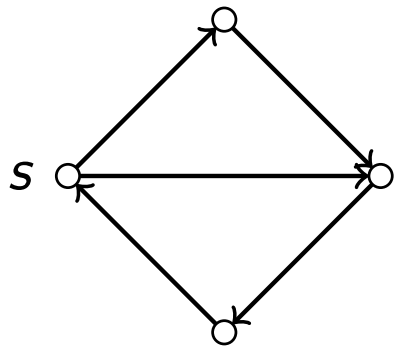
(Full) Merino's Theorem for Eulerian digraphs

In order to state Merino's Theorem for Eulerian digraphs, we need to introduce a generalization of one variable Tutte polynomial for directed graphs.

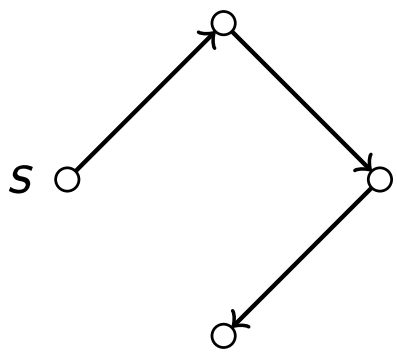
Rooted spanning tree

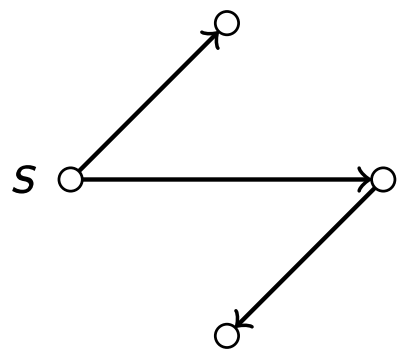
Let G be a strongly connected digraph, and let s be a vertex of $V(G)$.

An **s-rooted spanning tree** is a subgraph of G such that there is a unique directed path from s to v for each vertex $v \in V(G)$.



s-rooted spanning trees



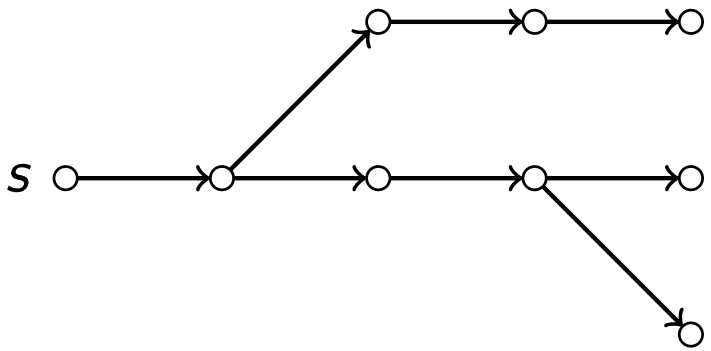


External activity

Fix a total order $<_e$ on $E(G)$.

Let T be an s-rooted spanning tree, and let $e \in E(G) \setminus E(T)$.

The graph $T \cup \{e\}$ has a unique circuit C.

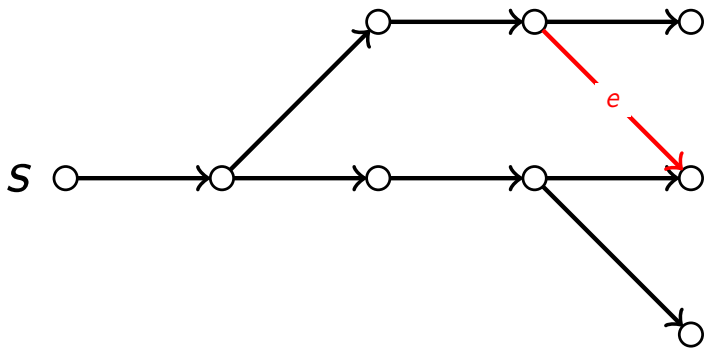


External activity

Fix a total order $<_e$ on $E(G)$.

Let T be an s-rooted spanning tree, and let $e \in E(G) \setminus E(T)$.

The graph $T \cup \{e\}$ has a unique circuit C.

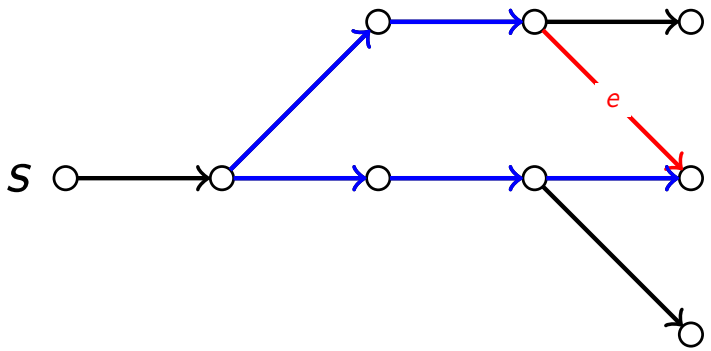


External activity

Fix a total order $<_e$ on $E(G)$.

Let T be an s-rooted spanning tree, and let $e \in E(G) \setminus E(T)$.

The graph $T \cup \{e\}$ has a unique circuit C.

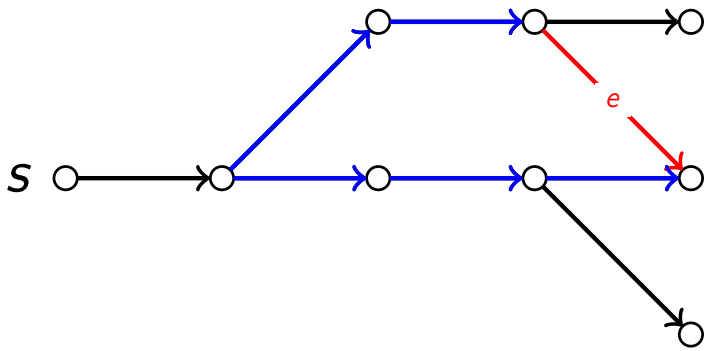


External activity

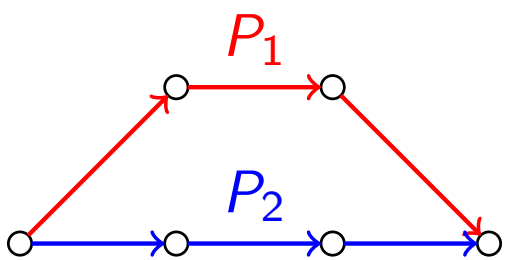
Fix a total order $<_e$ on $E(G)$.

Let T be an s-rooted spanning tree, and let $e \in E(G) \setminus E(T)$.

The graph $T \cup \{e\}$ has a unique circuit C.

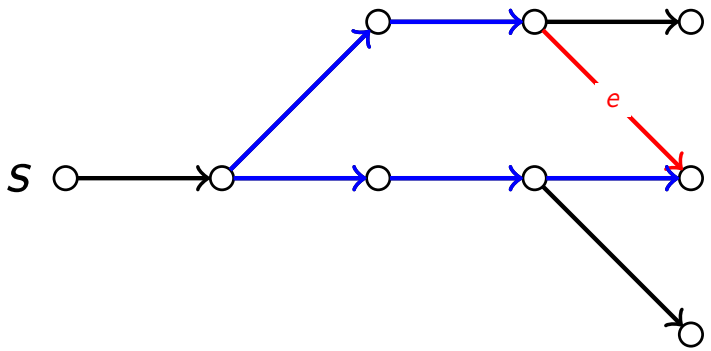


The circuit C decomposes to two directed paths P_1 and P_2. Let P_1 be the path that contains e.

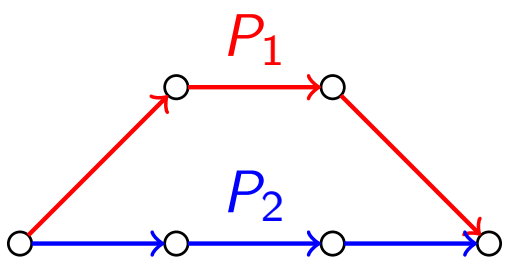


External activity

Fix a total order $<_e$ on $E(G)$.
Let T be an s-rooted spanning tree, and let $e \in E(G) \setminus E(T)$.
The graph $T \cup \{e\}$ has a unique circuit C.



The circuit C decomposes to two directed paths P_1 and P_2. Let P_1 be the path that contains e.



The edge e is **externally active w.r.t.T** if the smallest edge of $E(C)$ is contained in $E(P_1)$.

Greedoid polynomial

The **external activity** of T, denoted by $\mathrm{ext}(T)$, is the number of edges that are externally active w.r.t T.

The **greedoid polynomial** of G w.r.t root vertex s [BKL85] is the polynomial

$$\lambda(G, s; y) = \sum_{T \in \mathrm{RST}(G,s)} y^{\mathrm{ext}(T)}.$$

Merino's Theorem for Eulerian digraphs

Theorem 4 (C., 2015+)

If G is a connected Eulerian digraph and s is a vertex of G, then the following two polynomials are equal:

$$\mathcal{P}(Sand(G,s);y) = \lambda(G,s;y).$$

Thank you!


Anders Björner, Bernhard Korte, and László Lovász.
Homotopy properties of greedoids.
Adv. in Appl. Math., 6(4):447–494, 1985.


B. Bond and L. Levine.
Abelian networks I. Foundations and examples.
ArXiv e-prints, September 2013.


S. H. Chan.
Abelian sandpile model and Biggs-Merino polynomial for directed graphs.
ArXiv e-prints, December 2014.


Criel Merino López.
Chip firing and the Tutte polynomial.
Ann. Comb., 1(3):253–259, 1997.


K. Perrot and T. Van Pham.
Chip-firing game and partial Tutte polynomial for Eulerian digraphs.
ArXiv e-prints, June 2013.