# A combinatorial game on rooted Galton-Watson trees

Moumanti Podder

Indian Institute of Science Education & Research (IISER) Pune

IIT Kharagpur, Department of Mathematics
Weekly Online Seminar
December 3, 2021

# What's a rooted Galton-Watson tree?

- Start with root vertex $\phi$, and an offspring distribution $\chi$ (which is a probability distribution supported on $\mathbb{N}_0$).
- Let $\phi$ have $X_0$ children where $X_0 \sim \chi$ (i.e. $\mathbf{P}[X_0 = k] = \chi(k)$ for all $k \in \mathbb{N}_0$).
- Conditioned on $X_0 = m$, enumerate the children of $\phi$ as $v_1, \ldots, v_m$. Let $v_i$ have $X_i$ children, where $X_1, \ldots, X_m$ are i.i.d. $\chi$.
- Continue thus. This stochastic process (called a branching process) yields a (random) rooted tree $\mathcal{T}_\chi$.
- Classical result: $\mathcal{T}_\chi$ has a positive probability of being infinite (i.e. of survival) if and only if $\mu := \sum_{k \in \mathbb{N}_0} k\chi(k) > 1$.

# What are combinatorial games?

- The ones I study are played on graphs, and in particular, random graphs (such as rooted Galton-Watson trees or Erdős-Rényi random graphs).
- Usually played between two players (their roles in the game may or may not be symmetric to one another). Let us call these players P1 and P2.
- The graph on which the game is being played is revealed in its entirety to the two players (this means that the players can and must use *look-ahead strategies* to decide their moves). In this sense, these games are *perfect information* games.
- Both players play *optimally*, i.e. if in a game, P1 is destined to win, then she tries to win as quickly (efficiently) as possible, whereas P2 tries to prolong the game as much as she can.

# Why think about such games?

- The short answer is: they are immensely useful in *mathematical logic*, *automaton theory* and *computer science*.
- They help understand local as well as global structures / patterns inside the random graphs / trees the games are being played on.
- As an example, the *Ehrenfeucht-Fraïssé games* help investigate whether two graphs / trees are "equivalent" as far as sentences from *first order logic* of a given quantifier depth are concerned.
- The game I study here is called the *normal game*. I shall illustrate how this game relates with *finite state tree automata*.

# The simplest normal game

- Studied by Alexander E. Holroyd and James Martin in their 2018 paper, "Galton-Watson Games".
- $\mathcal{T}_\chi$ is visualized as a directed graph, where edge $\{u, v\}$ between parent $u$ and child $v$ is assumed directed from $u$ to $v$ (and denoted $(u, v)$).
- A token is placed at some vertex $v$ of $\mathcal{T}_\chi$ at the beginning of the game. We call $v$ the *initial vertex*.
- Players P1 and P2 take turns to move a token along these directed edges. Whoever fails to move (i.e. reaches a leaf vertex), loses the game.

# The version I study: it involves jumps!

- Fix a positive integer $k$.
- Let $\rho$ denote the usual graph metric on $\mathcal{T}_\chi$.
- For any vertex $u$ in $\mathcal{T}_\chi$, let

$$\Gamma_i(u) = \{v \in \mathcal{T}_\chi \setminus \{u\} : \rho(u,v) \leqslant i \text{ and } v \text{ a descendant of } u\}.$$

- When it is P$i$'s turn to move (where $i \in \{1,2\}$), and the token is at some vertex $u$, P$i$ may move the token to any vertex $v \in \Gamma_k(u)$.
- Thus, unlike the simplest version, $v$ here need not be a child of $u$, but needs to be some descendant of $u$ at distance at most $k$ from $u$.

# How do we analyze these games?

- Let us consider the simplest version first.
- Let NW denote the set of all vertices $v$ such that if $v$ is the initial vertex, whoever plays the first round, wins.
- Let NL denote the set of all vertices $v$ such that if $v$ is the initial vertex, whoever plays the first round, loses.
- Let nw and n$\ell$ respectively denote the probabilities of $\phi$ being in NW and NL.
- It is easy to establish recursion relations for nw and n$\ell$ from the description of the game.

# The recursions for the simplest version

- For a vertex $u$ to be in NW, it must have at least one child $v$, such that if the player who plays the first round (say, P1) moves the token to $v$ in the first round, then the game that begins with $v$ as the initial vertex (and whose first round is played by P2) is lost by P2. This means that $v \in \mathrm{NL}$.
- Thus $u \in \mathrm{NW}$ if and only if at least one child of $u$ is in NL. This yields

$$\mathrm{nw} = \sum_{m=1}^{\infty} \{1 - (1 - \mathrm{n}\ell)^m\} \chi(m) = 1 - G(1 - \mathrm{n}\ell),$$

where $G$ is the probability generating function of $\chi$.

# The recursions for the simplest version, continued

- For $u$ to be in NL, no matter which child $v$ of $u$ P1 (who plays the first round) moves the token to from the initial vertex $u$, the game that begins with $v$ as the initial vertex and P2 playing the first round is won by P2. That is, $v \in \mathrm{NW}$.
- Thus $u \in \mathrm{NL}$ if and only if *every* child $v$ of $u$ is in NW. This yields

$$\mathrm{n}\ell = \sum_{m=0}^{\infty} (\mathrm{nw})^m \chi(m) = G(\mathrm{nw}).$$

# Connection with finite state tree automata

- A finite state automaton is a kind of state machine used in mathematical computing.
- A finite state tree automaton comprises a *finite* set $\Sigma$ of "states" or "colours", and a "rule" $\Gamma$, which is a function from $\mathbb{N}_0^{\Sigma}$ to $\Sigma$.
- If a vertex $u$ has $m_i$ many children that are in state $i$, for all $i \in \Sigma$, then the state of the parent vertex $u$ is given by

$$\Gamma(m_i : i \in \Sigma).$$

- We now see the desired connection: let $\Sigma = \{\mathrm{NW}, \mathrm{NL}\}$. Then

$$\Gamma(m_{\mathrm{NW}}, m_{\mathrm{NL}}) = \mathrm{NW} \text{ as long as } m_{\mathrm{NL}} \geqslant 1,$$

whereas

$$\Gamma(m_{\mathrm{NW}}, m_{\mathrm{NL}}) = \mathrm{NL} \text{ as long as } m_{\mathrm{NL}} = 0.$$

# The more complicated recursions for the jump version

- For $u$ to be in NW, at least one $v \in \Gamma_k(u)$ must be in NL.
- This leads to defining several new classes of vertices:
  1. A vertex $v$ is in $\mathcal{C}_{0,1}$ if it has at least one child in NL.
  2. For all $2 \leqslant i \leqslant k-1$, a vertex $v$ is in $\mathcal{C}_{0,i}$ if it has at least one child in $\mathcal{C}_{0,i-1}$, but $v$ itself is *not* in $\bigcup_{j=1}^{i-1} \mathcal{C}_{0,j}$.
- These definitions imply that $u$ is in NW if and only if it has at least one child $v$ that is either in NL or in $\bigcup_{j=1}^{k-1} \mathcal{C}_{0,j}$.
- Notice that by definition, $\mathcal{C}_{0,1}, \ldots, \mathcal{C}_{0,k-1}$ are all pairwise disjoint, and each is disjoint from NL because if a vertex belongs to $\mathcal{C}_{0,i}$, then it is also in NW.
- Let $\mathbf{P}[\phi \in \mathcal{C}_{0,i}] = p_{0,i}$. We then have

$$\mathrm{nw} = \sum_{m=1}^{\infty} \left\{ 1 - \left( 1 - \mathrm{n}\ell - \sum_{i=0}^{k-1} p_{0,i} \right)^m \right\} \chi(m)$$

$$= 1 - G\left( 1 - \mathrm{n}\ell - \sum_{i=0}^{k-1} p_{0,i} \right).$$

# Recursions for the jump version, continued

- For $u$ to be in NL, *every* $v \in \Gamma_k(u)$ must be in NW.
- This leads to defining the following new classes of vertices: for $0 \leqslant i < j \leqslant k$, $\mathcal{C}_{i,j}$ is the set of all vertices $v$ such that
  1. $\Gamma_i(v) \subset \mathrm{NW}$,
  2. $\Gamma_{j-1}(v) \cap \mathrm{NL} = \emptyset$,
  3. there exists some vertex in $\Gamma_j(v) \setminus \Gamma_{j-1}(v)$ that is in NL.
- Note that $\mathcal{C}_{0,j}$ is obtained by setting $i = 0$ in the above definition, for each $j$.
- Let $p_{i,j} = \mathbf{P}[\phi \in \mathcal{C}_{i,j}]$ for all $0 \leqslant i < j \leqslant k$.
- We then see that $u \in \mathrm{NL}$ if and only if *every* child of $u$ is in $\mathcal{C}_{k-1,k}$. Hence

$$\mathrm{n}\ell = \sum_{m=0}^{\infty} (p_{k-1,k})^m \chi(m) = G(p_{k-1,k}).$$

# What these recursions lead to

- We can establish recursions relating $\mathcal{C}_{i,j}$ with $\mathcal{C}_{i-1,\ell}$ for all $\ell \geqslant j-1$, and thereby, recursions connecting all the $p_{i,j}$s with each other.
- These recursions, combined together, allow us to write $\mathrm{n}\ell = H(\mathrm{n}\ell)$ for a rather complicated function $H$.
- In fact, by considering subsets $\mathrm{NL}^{(n)} \subset \mathrm{NL}$ that comprise vertices $v$ such that a game starting at $v$ lasts $< n$ many rounds, for $n \in \mathbb{N}$, and applying the recursions to these more refined subsets, we can conclude that $\mathrm{n}\ell$ is the smallest fixed point of $H$ in $[0, 1]$.
- We can then obtain nw as a function of $\mathrm{n}\ell$.

# Is there a connection between the jump version and automata?

- Consider a generalized notion of finite state tree automata: we are now provided the states (in $\Sigma$) to which all the vertices of $\Gamma_k(u)$ belong.
- The state of $u$ is then determined from the states of all vertices in $\Gamma_k(u)$.
- In our set-up, let $\Sigma = \{\mathrm{NW}, \mathrm{NL}\}$, and the rule of the automaton states that:
  1. $u \in \mathrm{NW}$ if at least one vertex in $\Gamma_k(u)$ is in NL,
  2. and $u \in \mathrm{NL}$ if evert vertex in $\Gamma_k(u)$ is in NW.

# Further results I have so far

- A popular offspring distribution to consider for rooted Galton-Watson trees is Poisson($\lambda$) for various values of $\lambda > 0$. Recall that in this case,

$$\chi(i) = e^{-\lambda} \cdot \frac{\lambda^i}{i!}.$$

- We consider ND to be the set of all vertices $v$ *not* in $\mathrm{NW} \cup \mathrm{NL}$, i.e. if such a $v$ is an initial vertex, the game ends in a draw. Let $\mathrm{nd} = \mathbf{P}[\phi \in \mathrm{ND}]$.
- From the previously described recursions, one can find a necessary and sufficient condition for nd to be positive.
- I show that $\mathrm{nd} = \mathrm{nd}(\lambda) > 0$ for all $\lambda$ sufficiently large.
- In fact, I establish that $\mathrm{n}\ell = \mathrm{n}\ell(\lambda) \to 0$ as $\lambda \to \infty$, which in turn ensures that $\mathrm{nw} = \mathrm{nw}(\lambda) \to 0$ as $\lambda \to \infty$, and thus $\mathrm{nd} = \mathrm{nd}(\lambda) \to 1$ as $\lambda \to \infty$.

# Thank you!