Non-local Correlation Functions in the Spanning Tree Model near the Boundary

Khaydar Nurligareev (joint with A. Povolotsky)

HSE, Moscow

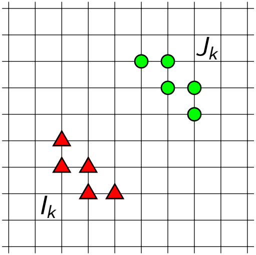
Classical and Quantum Integrable Systems 2021

Sochi, July 27, 2021

Content

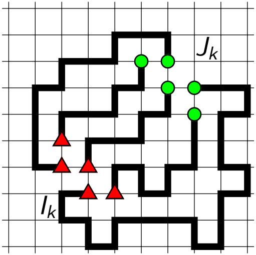
Watermelon configurations

- Let $\mathcal{G} = (V, E)$ be an undirected connected graph with neither loops nor multiple edges.
- $I_k = \{i_1, \ldots, i_k\}$ and $J_k = \{j_1, \ldots, j_k\}$ are two non-intersecting subsets.



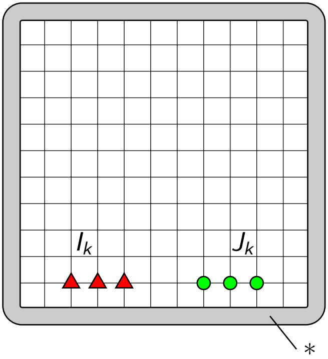
Watermelon configurations

- Let $\mathcal{G} = (V, E)$ be an undirected connected graph with neither loops nor multiple edges.
- $I_k = \{i_1, \ldots, i_k\}$ and $J_k = \{j_1, \ldots, j_k\}$ are two non-intersecting subsets.
- *Watermelon* is a configuration of k disjoint loopless paths from I_k to J_k.



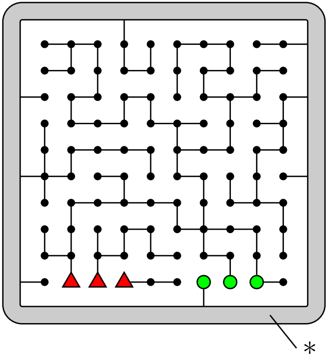
Spanning forests

- Let $\mathcal{G}^* = (V^*, E^*)$, where $V^* = V \cup \{*\}$ and $*$ is a sink.
- Choose I_k and J_k.



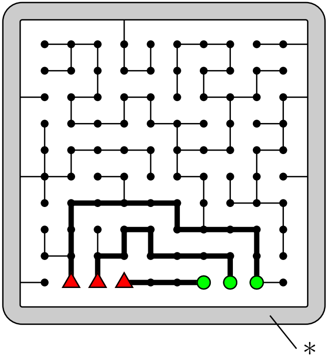
Spanning forests

- Let $\mathcal{G}^* = (V^*, E^*)$, where $V^* = V \cup \{*\}$ and $*$ is a sink.
- Choose I_k and J_k.
- Take a $(k+1)$-connected spanning forest with roots $I_k \cup \{*\}$.
- Consider a uniform measure on the set of all spanning forests.



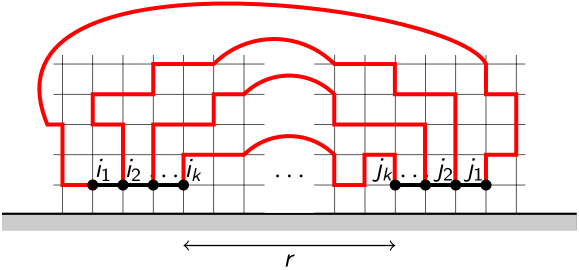
Spanning forests

- Let $\mathcal{G}^* = (V^*, E^*)$, where $V^* = V \cup \{*\}$ and $*$ is a sink.
- Choose I_k and J_k.
- Take a $(k+1)$-connected spanning forest with roots $I_k \cup \{*\}$.
- Consider a uniform measure on the set of all spanning forests.



- **Question**.
  What is the probability to have a watermelon configuration?

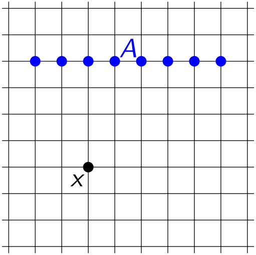
Main question



- Let I_k and J_k be separated by distance r.
- **Main question**. What is the asymptotical behavior of $\mathbb{P}$(watermelon configuration) for $r \to \infty$?

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \ldots, X_{\tau_A})$ is a path corresponding to X_n.



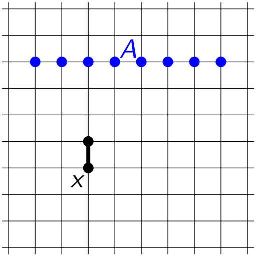
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \ldots, y_m) = (X_{n_0}, \ldots, X_{n_m}),$$

where $\quad n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \ldots, X_{\tau_A})$ is a path corresponding to X_n.



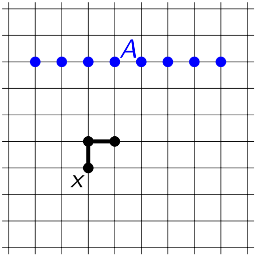
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \ldots, y_m) = (X_{n_0}, \ldots, X_{n_m}),$$

where $\quad n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \ldots, X_{\tau_A})$ is a path corresponding to X_n.



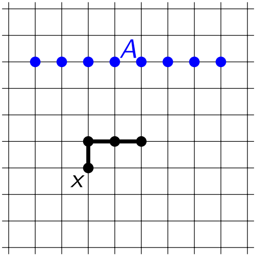
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \ldots, y_m) = (X_{n_0}, \ldots, X_{n_m}),$$

where $\quad n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \ldots, X_{\tau_A})$ is a path corresponding to X_n.



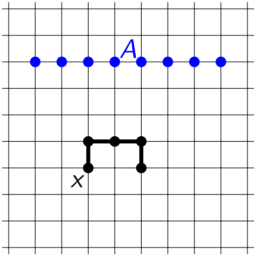
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \ldots, y_m) = (X_{n_0}, \ldots, X_{n_m}),$$

where $\quad n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \dots, X_{\tau_A})$ is a path corresponding to X_n.



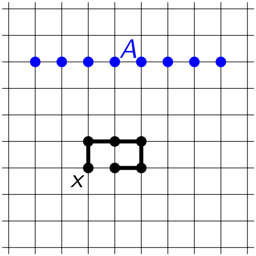
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \dots, y_m) = (X_{n_0}, \dots, X_{n_m}),$$

where $\quad n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \dots, X_{\tau_A})$ is a path corresponding to X_n.



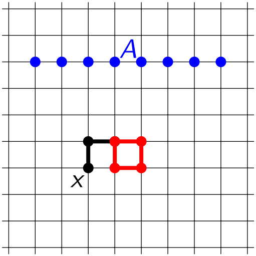
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \dots, y_m) = (X_{n_0}, \dots, X_{n_m}),$$

where $\quad n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \ldots, X_{\tau_A})$ is a path corresponding to X_n.



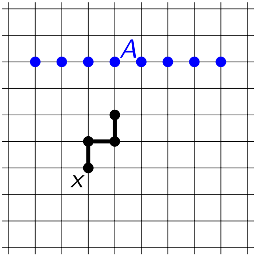
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \ldots, y_m) = (X_{n_0}, \ldots, X_{n_m}),$$

where $\quad n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \ldots, X_{\tau_A})$ is a path corresponding to X_n.



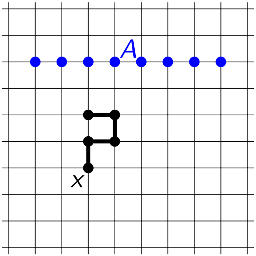
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \ldots, y_m) = (X_{n_0}, \ldots, X_{n_m}),$$

where $n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \ldots, X_{\tau_A})$ is a path corresponding to X_n.



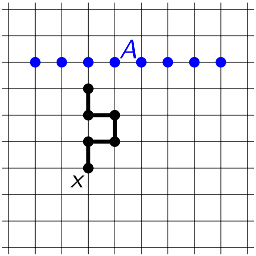
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \ldots, y_m) = (X_{n_0}, \ldots, X_{n_m}),$$

where $\quad n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \ldots, X_{\tau_A})$ is a path corresponding to X_n.



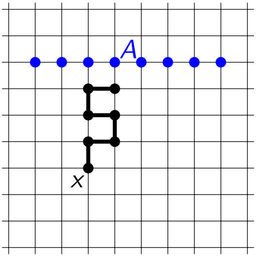
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \ldots, y_m) = (X_{n_0}, \ldots, X_{n_m}),$$

where $\quad n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \dots, X_{\tau_A})$ is a path corresponding to X_n.



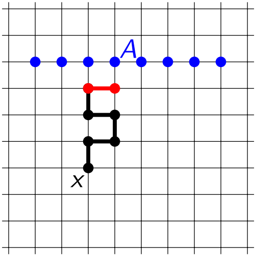
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \dots, y_m) = (X_{n_0}, \dots, X_{n_m}),$$

where $\quad n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \ldots, X_{\tau_A})$ is a path corresponding to X_n.



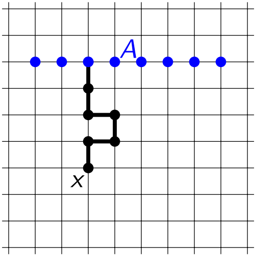
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \ldots, y_m) = (X_{n_0}, \ldots, X_{n_m}),$$

where $n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Loop-erased random walk (LERW)

- $A \subset V$ is a set of vertices,
- X_n is a simple random walk starting at $X_0 = x$,
- $\tau_A = \min\{n \geqslant 0 \colon \xi_n \in A\}$ is a stopping time (hitting time for the set A),
- $\gamma = (X_0, X_1, \dots, X_{\tau_A})$ is a path corresponding to X_n.



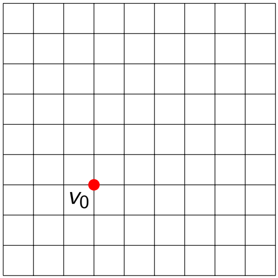
Loop-erased random walk is a path

$$LERW(x, A) = (y_0, \dots, y_m) = (X_{n_0}, \dots, X_{n_m}),$$

where $\quad n_0 = 0, \qquad n_{i+1} = \max\{j \colon \gamma(j) = \gamma(n_i)\} + 1.$

Wilson algorithm for generating uniform spanning tree

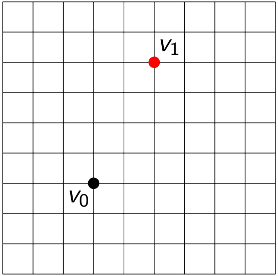
1. Take any vertex $v_0 \in V$.
2. Define $\mathcal{U}_0 = \{v_0\}$.
3. Take any vertex $v_1 \in V \setminus \mathcal{U}_0$.
4. Consider $LERW(v_1, \mathcal{U}_0)$ and define $\mathcal{U}_1 = LERW(v_1, \mathcal{U}_0)$.
5. ...
6. Take any vertex $v_k \in V \setminus \mathcal{U}_{k-1}$.
7. Define $\mathcal{U}_k = LERW(v_k, \mathcal{U}_{k-1}) \cup \mathcal{U}_{k-1}$.
8. At the end, we obtain a spanning tree.



If $|\mathcal{U}_0| > 1$, then we will get a spanning forest.

Wilson algorithm for generating uniform spanning tree

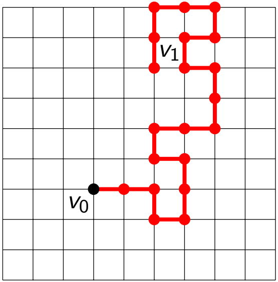
1. Take any vertex $v_0 \in V$.
2. Define $\mathcal{U}_0 = \{v_0\}$.
3. Take any vertex $v_1 \in V \setminus \mathcal{U}_0$.
4. Consider $LERW(v_1, \mathcal{U}_0)$ and define $\mathcal{U}_1 = LERW(v_1, \mathcal{U}_0)$.
5. ...
6. Take any vertex $v_k \in V \setminus \mathcal{U}_{k-1}$.
7. Define $\mathcal{U}_k = LERW(v_k, \mathcal{U}_{k-1}) \cup \mathcal{U}_{k-1}$.
8. At the end, we obtain a spanning tree.



If $|\mathcal{U}_0| > 1$, then we will get a spanning forest.

Wilson algorithm for generating uniform spanning tree

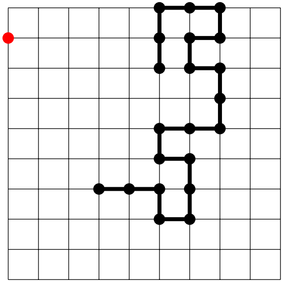
1. Take any vertex $v_0 \in V$.
2. Define $\mathcal{U}_0 = \{v_0\}$.
3. Take any vertex $v_1 \in V \setminus \mathcal{U}_0$.
4. Consider $LERW(v_1, \mathcal{U}_0)$ and define $\mathcal{U}_1 = LERW(v_1, \mathcal{U}_0)$.
5. ...
6. Take any vertex $v_k \in V \setminus \mathcal{U}_{k-1}$.
7. Define $\mathcal{U}_k = LERW(v_k, \mathcal{U}_{k-1}) \cup \mathcal{U}_{k-1}$.
8. At the end, we obtain a spanning tree.



If $|\mathcal{U}_0| > 1$, then we will get a spanning forest.

Wilson algorithm for generating uniform spanning tree

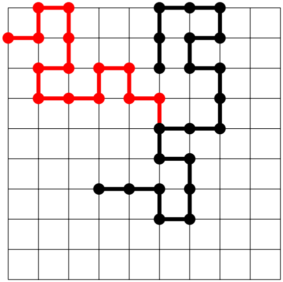
1. Take any vertex $v_0 \in V$.
2. Define $\mathcal{U}_0 = \{v_0\}$.
3. Take any vertex $v_1 \in V \setminus \mathcal{U}_0$.
4. Consider $LERW(v_1, \mathcal{U}_0)$ and define $\mathcal{U}_1 = LERW(v_1, \mathcal{U}_0)$.
5. ...
6. Take any vertex $v_k \in V \setminus \mathcal{U}_{k-1}$.
7. Define $\mathcal{U}_k = LERW(v_k, \mathcal{U}_{k-1}) \cup \mathcal{U}_{k-1}$.
8. At the end, we obtain a spanning tree.



If $|\mathcal{U}_0| > 1$, then we will get a spanning forest.

Wilson algorithm for generating uniform spanning tree

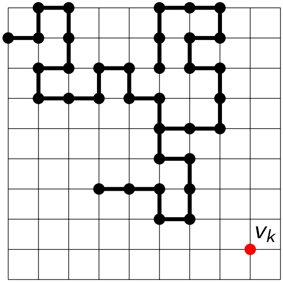
1. Take any vertex $v_0 \in V$.
2. Define $\mathcal{U}_0 = \{v_0\}$.
3. Take any vertex $v_1 \in V \setminus \mathcal{U}_0$.
4. Consider $LERW(v_1, \mathcal{U}_0)$ and define $\mathcal{U}_1 = LERW(v_1, \mathcal{U}_0)$.
5. ...
6. Take any vertex $v_k \in V \setminus \mathcal{U}_{k-1}$.
7. Define $\mathcal{U}_k = LERW(v_k, \mathcal{U}_{k-1}) \cup \mathcal{U}_{k-1}$.
8. At the end, we obtain a spanning tree.



If $|\mathcal{U}_0| > 1$, then we will get a spanning forest.

Wilson algorithm for generating uniform spanning tree

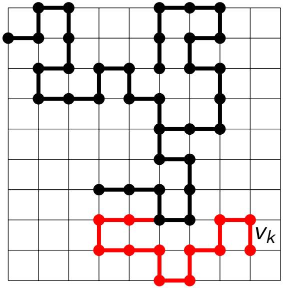
1. Take any vertex $v_0 \in V$.
2. Define $\mathcal{U}_0 = \{v_0\}$.
3. Take any vertex $v_1 \in V \setminus \mathcal{U}_0$.
4. Consider $LERW(v_1, \mathcal{U}_0)$ and define $\mathcal{U}_1 = LERW(v_1, \mathcal{U}_0)$.
5. ...
6. Take any vertex $v_k \in V \setminus \mathcal{U}_{k-1}$.
7. Define $\mathcal{U}_k = LERW(v_k, \mathcal{U}_{k-1}) \cup \mathcal{U}_{k-1}$.
8. At the end, we obtain a spanning tree.



If $|\mathcal{U}_0| > 1$, then we will get a spanning forest.

Wilson algorithm for generating uniform spanning tree

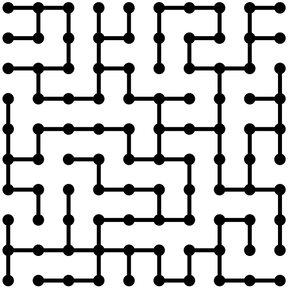
1. Take any vertex $v_0 \in V$.
2. Define $\mathcal{U}_0 = \{v_0\}$.
3. Take any vertex $v_1 \in V \setminus \mathcal{U}_0$.
4. Consider $LERW(v_1, \mathcal{U}_0)$ and define $\mathcal{U}_1 = LERW(v_1, \mathcal{U}_0)$.
5. ...
6. Take any vertex $v_k \in V \setminus \mathcal{U}_{k-1}$.
7. Define $\mathcal{U}_k = LERW(v_k, \mathcal{U}_{k-1}) \cup \mathcal{U}_{k-1}$.
8. At the end, we obtain a spanning tree.



If $|\mathcal{U}_0| > 1$, then we will get a spanning forest.

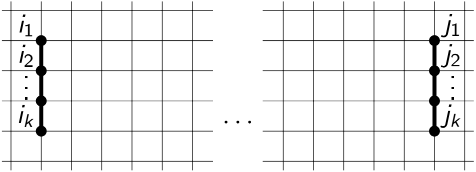
Wilson algorithm for generating uniform spanning tree

1. Take any vertex $v_0 \in V$.
2. Define $\mathcal{U}_0 = \{v_0\}$.
3. Take any vertex $v_1 \in V \setminus \mathcal{U}_0$.
4. Consider $LERW(v_1, \mathcal{U}_0)$ and define $\mathcal{U}_1 = LERW(v_1, \mathcal{U}_0)$.
5. ...
6. Take any vertex $v_k \in V \setminus \mathcal{U}_{k-1}$.
7. Define $\mathcal{U}_k = LERW(v_k, \mathcal{U}_{k-1}) \cup \mathcal{U}_{k-1}$.
8. At the end, we obtain a spanning tree.



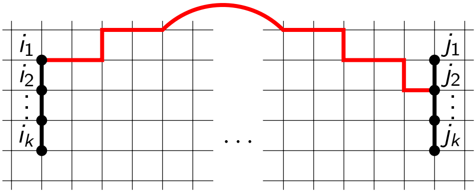
If $|\mathcal{U}_0| > 1$, then we will get a spanning forest.

LERW and watermelons



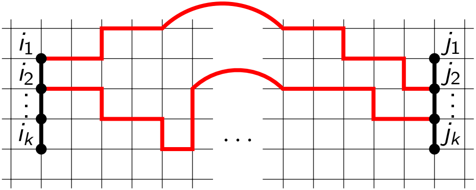
Every k-leg watermelon can be considered as k loop-erased random walks from J_k to I_k.

LERW and watermelons



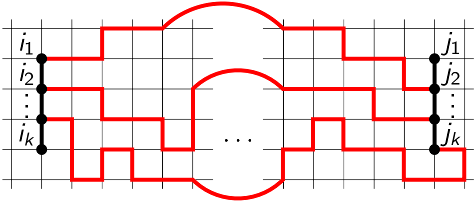
Every k-leg watermelon can be considered as k loop-erased random walks from J_k to I_k.

LERW and watermelons



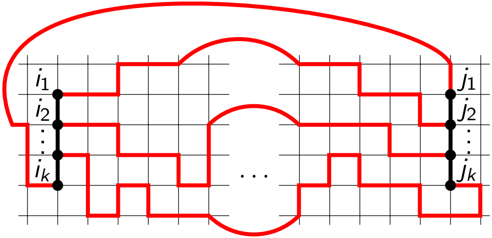
Every k-leg watermelon can be considered as k loop-erased random walks from J_k to I_k.

LERW and watermelons



Every k-leg watermelon can be considered as k loop-erased random walks from J_k to I_k.

LERW and watermelons



Every k-leg watermelon can be considered as k loop-erased random walks from J_k to I_k.

CFT predictions

- **In the bulk**, Duplantier and Saleur (1987) predicted

$$\nu^{bulk} = \frac{k^2 - 1}{2}$$

with the help of the Coulomb gas approach.

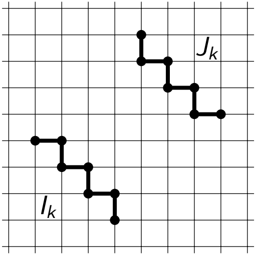
- **For the half-plane**, Duplantier and Saleur (1986) predicted

$$\nu^{hp} = k(k-1).$$

Watermelons in the bulk

Let

- $\mathcal{G}$ be a square lattice (bulk case),
- k be odd,
- I_k and J_k have the form of *fence*.

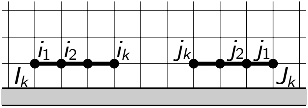


Theorem (Ivashkevich, Hu, 2005; Gorsky, Nechaev, Poghosyan, Priezzhev, 2013)

$$F(r) \sim C \cdot r^{-\nu^{bulk}} \cdot \ln r, \qquad \textit{where } \nu^{bulk} = \frac{k^2 - 1}{2}.$$

Watermelons on the half-plane, open boundary

- $\mathcal{G}$ is a square lattice on the half-plane,
- I_k and J_k have the form of segments located near the boundary,
- absorbing boundary conditions.



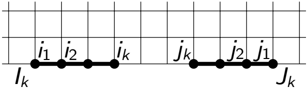
Theorem

$$\mathbb{P}(\textit{watermelon configuration}) \sim C^{op} \cdot r^{-k(k+1)},$$

$$C^{op} = \frac{\prod_{s=1}^{k} (s!)^2}{p_k^{op}(\pi) \cdot k!}, \qquad p_k^{op}(x) \textit{ is a polynomial of degree } k.$$

Watermelons on the half-plane, closed boundary

- $\mathcal{G}$ is a square lattice on the half-plane,
- I_k and J_k have the form of segments located near the boundary,



- reflecting boundary conditions.

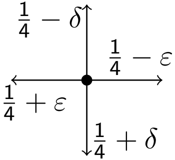
Theorem

$$\mathbb{P}(\textit{watermelon configuration}) \sim C^{cl} \cdot r^{-k(k-1)},$$

$$C^{cl} = \frac{\prod_{s=1}^{k-1}(s!)^2}{p_k^{cl}(\pi)\cdot(k-1)!}, \qquad p_k^{cl}(x) \text{ is a polynomial of degree } k-1.$$

Predictions for stretched watermelons

For the anisotropic case, we have the universality class of the vicious walkers model.



- **In the bulk**, Fisher (1984) predicted

$$\nu^{bulk,\parallel} = \frac{k^2}{2}.$$

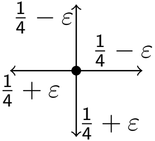
- **For the half-plane**, depending on boundary conditions, Guttmann, Owczarek, and Viennot (1998) predicted

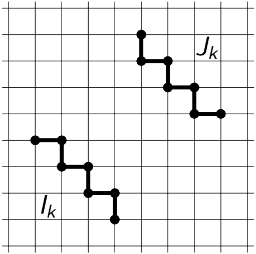
$$\nu^{op,\parallel} = k\left(k + \frac{1}{2}\right) \qquad \text{for absorbing boundary conditions,}$$

$$\nu^{cl,\parallel} = k\left(k - \frac{1}{2}\right) \qquad \text{for reflecting boundary conditions.}$$

Elongated watermelons in the bulk

- $\mathcal{G}$ is an elongated square lattice,
- I_k and J_k have the form of *fence*.
- k is odd,
- $\delta = \varepsilon$.





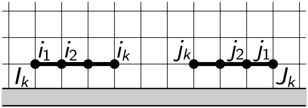
Theorem (Gorsky, Nechaev, Poghosyan, Priezzhev, 2013)

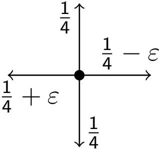
If $\varepsilon \to 1/4$, *then*

$$\mathbb{P}(\textit{watermelon configuration}) \sim C \cdot r^{-\nu^{bulk,\parallel}}, \qquad \textit{where } \nu^{bulk,\parallel} = \frac{k^2}{2}.$$

Elongated watermelons on the half-plane, open boundary

- $\mathcal{G}$ is a horizontally elongated square lattice on the half-plane,
- I_k and J_k are segments located near the boundary,
- absorbing boundary conditions.





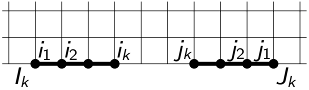
Theorem

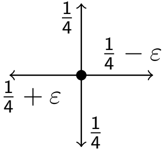
If $\varepsilon \to 1/4$, then

$$\mathbb{P}(\textit{watermelon configuration}) \sim C^{op} \cdot r^{-k\left(k+\frac{1}{2}\right)}.$$

Elongated watermelons on the half-plane, closed boundary

- $\mathcal{G}$ is a horizontally elongated square lattice on the half-plane,
- I_k and J_k have are segments located near the boundary,
- reflecting boundary conditions.





Theorem

If $\varepsilon \to 1/4$, *then*

$$\mathbb{P}(\textit{watermelon configuration}) \sim C^{cl} \cdot r^{-k\left(k-\frac{1}{2}\right)}.$$

Matrix Tree Theorem

- $\mathcal{G} = (V, E)$ is a finite connected (directed) graph without loops and multiple edges.
- $\mathcal{G}^* = (V^*, E^*)$, where $V^* = V \cup \{*\}$, $*$ is a sink.
- *Discrete Laplacian* is the matrix $\Delta = (\Delta_{ij})_{i,j \in V}$,

$$\Delta_{ij} = \begin{cases} \deg i, & \text{if } \ i = j; \\ -1, & \text{if } \ i \neq j, \ \ ij \in V; \\ 0, & \text{if } \ i \neq j, \ \ ij \notin V. \end{cases}$$

Theorem (Kirchhoff, 1848)

$$\#\{\textit{spanning trees of } \mathcal{G}^* \textit{ rooted to } *\} = \det \Delta.$$

All Minors Matrix Tree Theorem

- $I = \{i_1, \dots, i_k\}$, $J = \{j_1, \dots, j_k\}$ and $R = \{r_1, \dots, r_n\}$ are three disjoint subsets of V,
- $\rho_\sigma = i_1 j_{\sigma(1)} | \dots | i_k j_{\sigma(k)} | r_1 | \dots | r_n | *$ is partial pairing, $\sigma \in S_k$,
- $Z[\rho_\sigma]$ is the number of spanning forests on $\mathcal{G}^*$ such that
  - each component is rooted to $I \cup R \cup \{*\}$,
  - i_m and $j_{\sigma(m)}$ are in the same component.

Theorem (Chen, 1976)

Let Δ be invertible, $G = \Delta^{-1}$. Then

$$\det \Delta \cdot \det G^{I \cup R}_{J \cup R} = \sum_{\sigma \in S_k} (-1)^\sigma Z[\rho_\sigma].$$

Watermelon probability and Green functions

$$\mathbb{P}(\text{watermelon configuration}) = \frac{\det G_{J_k}^{I_k}}{\det G_{I_k}^{I_k}}$$

- G — Green function
- $G_{J_k}^{I_k}$ and $G_{I_k}^{I_k}$ are matrices $k \times k$

$$G^{op}_{(x;y_1,y_2)} = \frac{1}{\pi^2}\int\limits_0^\pi d\alpha \int\limits_0^\pi d\beta \, \frac{\cos x\alpha \sin y_1\beta \sin y_2\beta}{2-(\cos\alpha+\cos\beta)}.$$

$$G^{cl}_{(x;y_1,y_2)} = \frac{1}{\pi^2}\int\limits_0^\pi d\alpha \int\limits_0^\pi d\beta \, \frac{\cos x\alpha \cos\left(y_1-1/2\right)\beta \cos\left(y_2-1/2\right)\beta - 1}{2-(\cos\alpha+\cos\beta)}.$$

Thank you for the attention!