Examples of radius and diameter

Note that the diameter is the maximum of the vertex eccentricities, whereas the radius is the minimum.

Examples of radius and diameter

Note that the diameter is the maximum of the vertex eccentricities, whereas the radius is the minimum.

What are the diameter and the radius of:

- the Petersen graph?

Examples of radius and diameter

Note that the diameter is the maximum of the vertex eccentricities, whereas the radius is the minimum.

What are the diameter and the radius of:

- the Petersen graph? $\text{diam} = \text{rad} = 2.$

Examples of radius and diameter

Note that the diameter is the maximum of the vertex eccentricities, whereas the radius is the minimum.

What are the diameter and the radius of:

- the Petersen graph? diam = rad = 2.
- the hypercube Q_k?

Examples of radius and diameter

Note that the diameter is the maximum of the vertex eccentricities, whereas the radius is the minimum.

What are the diameter and the radius of:

- the Petersen graph? $\quad$ diam $=$ rad $= 2$.
- the hypercube Q_k? $\quad$ diam $=$ rad $= k$.

Examples of radius and diameter

Note that the diameter is the maximum of the vertex eccentricities, whereas the radius is the minimum.

What are the diameter and the radius of:

- the Petersen graph? diam = rad = 2.
- the hypercube Q_k? diam = rad = k.
- the cycle C_n?

Examples of radius and diameter

Note that the diameter is the maximum of the vertex eccentricities, whereas the radius is the minimum.

What are the diameter and the radius of:

- the Petersen graph? diam $=$ rad $= 2$.
- the hypercube Q_k? diam $=$ rad $= k$.
- the cycle C_n? diam $=$ rad $= \lfloor n/2 \rfloor$.

Examples of radius and diameter

Note that the diameter is the maximum of the vertex eccentricities, whereas the radius is the minimum.

What are the diameter and the radius of:

- the Petersen graph? diam $=$ rad $= 2$.
- the hypercube Q_k? diam $=$ rad $= k$.
- the cycle C_n? diam $=$ rad $= \lfloor n/2 \rfloor$.

For $n \geq 3$, what are the n-vertex trees of smallest/largest diameter?

Examples of radius and diameter

Note that the diameter is the maximum of the vertex eccentricities, whereas the radius is the minimum.

What are the diameter and the radius of:

- the Petersen graph? diam = rad = 2.
- the hypercube Q_k? diam = rad = k.
- the cycle C_n? diam = rad = $\lfloor n/2 \rfloor$.

For $n \geq 3$, what are the n-vertex trees of smallest/largest diameter?

Theorem

If G is a simple graph with $\mathrm{diam}(G) \geq 3$, then $\mathrm{diam}(\overline{G}) \leq 3$.

Examples of radius and diameter

Note that the diameter is the maximum of the vertex eccentricities, whereas the radius is the minimum.

What are the diameter and the radius of:

- the Petersen graph? diam $=$ rad $= 2$.
- the hypercube Q_k? diam $=$ rad $= k$.
- the cycle C_n? diam $=$ rad $= \lfloor n/2 \rfloor$.

For $n \geq 3$, what are the n-vertex trees of smallest/largest diameter?

Theorem

If G is a simple graph with $diam(G) \geq 3$, then $diam(\overline{G}) \leq 3$.

Definition

The **center** of G is the subgraph induced by the vertices of minimum eccentricity.

2.2 Spanning trees and enumeration

There are $2^{\binom{n}{2}}$ simple graphs with vertex set $[n] := \{1, 2, \ldots, n\}$.

2.2 Spanning trees and enumeration

There are $2^{\binom{n}{2}}$ simple graphs with vertex set $[n] := \{1, 2, \ldots, n\}$.
How many of these are trees?

2.2 Spanning trees and enumeration

There are $2^{\binom{n}{2}}$ simple graphs with vertex set $[n] := \{1, 2, \ldots, n\}$.
How many of these are trees?

n	2	3	4	5	...
#trees	1	3			

2.2 Spanning trees and enumeration

There are $2^{\binom{n}{2}}$ simple graphs with vertex set $[n] := \{1, 2, \ldots, n\}$.
How many of these are trees?

n	2	3	4	5	$\ldots$
#trees	1	3	16		

2.2 Spanning trees and enumeration

There are $2^{\binom{n}{2}}$ simple graphs with vertex set $[n] := \{1, 2, \ldots, n\}$.
How many of these are trees?

n	2	3	4	5	$\ldots$
#trees	1	3	16	125	$\ldots$

2.2 Spanning trees and enumeration

There are $2^{\binom{n}{2}}$ simple graphs with vertex set $[n] := \{1, 2, \ldots, n\}$. How many of these are trees?

n	2	3	4	5	...
#trees	1	3	16	125	...

We often call these *labeled* trees, meaning that the vertices are labeled with $1, 2, \ldots, n$.

2.2 Spanning trees and enumeration

There are $2^{\binom{n}{2}}$ simple graphs with vertex set $[n] := \{1, 2, \ldots, n\}$. How many of these are trees?

n	2	3	4	5	$\ldots$
#trees	1	3	16	125	$\ldots$

We often call these *labeled* trees, meaning that the vertices are labeled with $1, 2, \ldots, n$.

Theorem (Cayley)

The number of labeled trees with n vertices is n^{n-2}.

2.2 Spanning trees and enumeration

There are $2^{\binom{n}{2}}$ simple graphs with vertex set $[n] := \{1, 2, \ldots, n\}$. How many of these are trees?

n	2	3	4	5	...
#trees	1	3	16	125	...

We often call these *labeled* trees, meaning that the vertices are labeled with $1, 2, \ldots, n$.

Theorem (Cayley)

The number of labeled trees with n vertices is n^{n-2}.

We will give a bijective proof, by encoding each tree with a unique sequence of length $n-2$ with entries in $[n]$, called the **Prüfer code**.

Prüfer code

Input: A labeled tree T with n vertices.

Output: A sequence $(a_1, a_2, \ldots, a_{n-2})$ where $a_i \in [n]$ for all i.

Prüfer code

Input: A labeled tree T with n vertices.

Output: A sequence $(a_1, a_2, \ldots, a_{n-2})$ where $a_i \in [n]$ for all i.

For i from 1 to $n - 2$:

- Find the leaf v with the smallest label.
- Let a_i be the label of the neighbor of this leaf.
- Remove v from the tree to create a new tree.

Recovering a tree from its Prüfer code

Note: the leaves of T are precisely the elements that do not appear in $(a_1, a_2, \ldots, a_{n-2})$.

Recovering a tree from its Prüfer code

Note: the leaves of T are precisely the elements that do not appear in $(a_1, a_2, \ldots, a_{n-2})$.

The first deleted leaf x_1 must be the smallest such element, and it is adjacent to a_1.

Recovering a tree from its Prüfer code

Note: the leaves of T are precisely the elements that do not appear in $(a_1, a_2, \ldots, a_{n-2})$.

The first deleted leaf x_1 must be the smallest such element, and it is adjacent to a_1.

The second deleted leaf x_2 must be the smallest element $\neq x_1$ that does not appear in $(a_2, \ldots, a_{n-2})$, and it is adjacent to a_2.

Recovering a tree from its Prüfer code

Note: the leaves of T are precisely the elements that do not appear in $(a_1, a_2, \ldots, a_{n-2})$.

The first deleted leaf x_1 must be the smallest such element, and it is adjacent to a_1.

The second deleted leaf x_2 must be the smallest element $\neq x_1$ that does not appear in $(a_2, \ldots, a_{n-2})$, and it is adjacent to a_2.

In general, the ith deleted leaf x_i must be the smallest element not in the set $\{x_1, \ldots, x_{i-1}, a_i, a_{i+1}, \ldots, a_{n-2}\}$, and it is adjacent to a_i.

Recovering a tree from its Prüfer code

Note: the leaves of T are precisely the elements that do not appear in $(a_1, a_2, \ldots, a_{n-2})$.

The first deleted leaf x_1 must be the smallest such element, and it is adjacent to a_1.

The second deleted leaf x_2 must be the smallest element $\neq x_1$ that does not appear in $(a_2, \ldots, a_{n-2})$, and it is adjacent to a_2.

In general, the ith deleted leaf x_i must be the smallest element not in the set $\{x_1, \ldots, x_{i-1}, a_i, a_{i+1}, \ldots, a_{n-2}\}$, and it is adjacent to a_i.

To recover T, we repeat the above procedure for i from 1 to $n-2$. Finally, we join the two vertices not in $\{x_1, \ldots, x_{n-2}\}$.

Remarks on the Prüfer code

We saw that the leaves of T are the elements that do not appear in the Prüfer code.

Remarks on the Prüfer code

We saw that the leaves of T are the elements that do not appear in the Prüfer code.

More generally, how can we determine the degree of a vertex?

Remarks on the Prüfer code

We saw that the leaves of T are the elements that do not appear in the Prüfer code.

More generally, how can we determine the degree of a vertex?

The degree of a vertex is the number of times that it appears in the Prüfer code, plus 1.

Remarks on the Prüfer code

We saw that the leaves of T are the elements that do not appear in the Prüfer code.

More generally, how can we determine the degree of a vertex?

The degree of a vertex is the number of times that it appears in the Prüfer code, plus 1.

Proposition

Given positive integers $d_1, d_2, \ldots, d_n$ summing to $2n-2$, there are exactly

$$\frac{(n-2)!}{\prod_{i=1}^{n}(d_i-1)!}$$

trees with vertex set $[n]$ such that vertex i has degree d_i for each i.

Remarks on the Prüfer code

We saw that the leaves of T are the elements that do not appear in the Prüfer code.

More generally, how can we determine the degree of a vertex?

The degree of a vertex is the number of times that it appears in the Prüfer code, plus 1.

Proposition

Given positive integers $d_1, d_2, \ldots, d_n$ summing to $2n-2$, there are exactly

$$\frac{(n-2)!}{\prod_{i=1}^{n}(d_i - 1)!}$$

trees with vertex set $[n]$ such that vertex i has degree d_i for each i.

Example: The number of trees with vertex set $[7]$ with degrees $(d_1, \ldots, d_7) = (3, 1, 2, 1, 3, 1, 1)$ is

Remarks on the Prüfer code

We saw that the leaves of T are the elements that do not appear in the Prüfer code.

More generally, how can we determine the degree of a vertex?

The degree of a vertex is the number of times that it appears in the Prüfer code, plus 1.

Proposition

Given positive integers $d_1, d_2, \ldots, d_n$ summing to $2n-2$, there are exactly

$$\frac{(n-2)!}{\prod_{i=1}^{n}(d_i - 1)!}$$

trees with vertex set $[n]$ such that vertex i has degree d_i for each i.

Example: The number of trees with vertex set [7] with degrees $(d_1, \ldots, d_7) = (3, 1, 2, 1, 3, 1, 1)$ is 30.

Spanning trees in graphs

Recall that a spanning tree of a connected graph G is a subgraph with vertex set $V(G)$ that is a tree.

Spanning trees in graphs

Recall that a spanning tree of a connected graph G is a subgraph with vertex set $V(G)$ that is a tree.

Question: Given a graph G, how many spanning trees does it have?

Let $\tau(G)$ denote the number of spanning trees of G.

Spanning trees in graphs

Recall that a spanning tree of a connected graph G is a subgraph with vertex set $V(G)$ that is a tree.

Question: Given a graph G, how many spanning trees does it have?

Let $\tau(G)$ denote the number of spanning trees of G.

Examples:

$\tau(K_n) =$

Spanning trees in graphs

Recall that a spanning tree of a connected graph G is a subgraph with vertex set $V(G)$ that is a tree.

Question: Given a graph G, how many spanning trees does it have?

Let $\tau(G)$ denote the number of spanning trees of G.

Examples:

$\tau(K_n) = n^{n-2}$

...

Contraction of an edge

Definition

Let G be a graph and let $e = uv$ be an edge. The contraction of e is the operation that replaces e with a single vertex, which is incident to those edges that were incident to either u or v in G.

Contraction of an edge

Definition

Let G be a graph and let $e = uv$ be an edge. The contraction of e is the operation that replaces e with a single vertex, which is incident to those edges that were incident to either u or v in G.

Denote by $G \cdot e$ the resulting graph.

Contraction of an edge

Definition

Let G be a graph and let $e = uv$ be an edge. The contraction of e is the operation that replaces e with a single vertex, which is incident to those edges that were incident to either u or v in G.

Denote by $G \cdot e$ the resulting graph.

- This construction may create loops and multiple edges.

Contraction of an edge

Definition

Let G be a graph and let $e = uv$ be an edge. The contraction of e is the operation that replaces e with a single vertex, which is incident to those edges that were incident to either u or v in G.

Denote by $G \cdot e$ the resulting graph.

- This construction may create loops and multiple edges.
- $G \cdot e$ has one fewer edge than G.

Deletion-contraction method

How do the numbers $\tau(G)$, $\tau(G - e)$, and $\tau(G \cdot e)$ relate to each other?

Deletion-contraction method

How do the numbers $\tau(G)$, $\tau(G - e)$, and $\tau(G \cdot e)$ relate to each other?

Proposition (Deletion-contraction recurrence)

If $e \in E(G)$ is not a loop, then

$$\tau(G) = \tau(G - e) + \tau(G \cdot e).$$

Deletion-contraction method

How do the numbers $\tau(G)$, $\tau(G-e)$, and $\tau(G \cdot e)$ relate to each other?

Proposition (Deletion-contraction recurrence)

If $e \in E(G)$ is not a loop, then

$$\tau(G) = \tau(G-e) + \tau(G \cdot e).$$

[Example]

Deletion-contraction method

How do the numbers $\tau(G)$, $\tau(G-e)$, and $\tau(G \cdot e)$ relate to each other?

Proposition (Deletion-contraction recurrence)

If $e \in E(G)$ is not a loop, then

$$\tau(G) = \tau(G-e) + \tau(G \cdot e).$$

[Example]

- If e is a loop, one can just delete it since it does not affect the number of spanning trees.

Deletion-contraction method

How do the numbers $\tau(G)$, $\tau(G - e)$, and $\tau(G \cdot e)$ relate to each other?

Proposition (Deletion-contraction recurrence)

If $e \in E(G)$ is not a loop, then

$$\tau(G) = \tau(G - e) + \tau(G \cdot e).$$

[Example]

- If e is a loop, one can just delete it since it does not affect the number of spanning trees.
- With this recurrence, one can in theory compute $\tau(G)$ for any graph recursively, but it's computationally impractical, since one would have to compute up to $2^{e(G)}$ terms.