Prüfer Encoding and a Proof of Cayley's Tree Formula

NGUYEN Tue Tai (ID 062201848)

May 20, 2024

1 Introduction

At the end of Chapter 3 in the lecture note [1], we encounter Cayley's tree formula which returns the number of labeled trees on n vertices for $n \geq 2$. This report will present a proof of this formula by Heinz Prüfer in 1918. The following discussion follows Section 3.7 in [2].

2 Prüfer Encoding

Definition 1. Let $S = \{s_1, s_2, \ldots, s_n\}$ be a completely ordered set such that $s_1 < s_2 < \ldots < s_n$ for $n \geq 1$. A **vertex-labeling** of an n-vertex graph G is a one-to-one assignment of elements in S to the vertices of G. A graph with a vertex-labeling is called a labeled graph.

Given a graph, we can label it with elements from many different sets S. With two sets of labels S_1 and S_2 with the same number of elements, there exists a one-to-one correspondence between them, which preserves the ordering structure. Hence, there is a type of isomorphisms between graphs labeled with different sets of labels.

This report focuses on labeled trees, so we will only consider elements of $\mathcal{T}_n$, the set of all different trees with n labeled vertices, up to isomorphism. Aside, we will sometimes call a vertex by its label, for short. The goal at the end of this report is to prove Cayley's tree formula which gives us the number of elements in $\mathcal{T}_n$. We do this by establishing a bijection between $\mathcal{T}_n$ and another finite set which is easier to count its elements. We define these elements as follows.

Definition 2. Let $S = \{s_1, s_2, \ldots, s_n\}$ be a completely ordered set such that $s_1 < s_2 < \ldots < s_n$ for $n \geq 2$. A **Prüfer sequence** of length $n-2$, is any sequence of elements in S, with repetition allowed. The set of all Prüfer sequences of length $n-2$ is denoted by $\mathcal{P}_{n-2}$.

Originally, a Prüfer sequence of length $n-2$, for $n \geq 2$, is any sequence of integers between 1 and n, with repetition allowed. However, we can always have a one-to-one correspondence between a completely ordered finite set S and the set of numbers $\{1, 2, \ldots, n\}$, which preserves the ordering structure. Hence, there exist "isomorphisms" between Prüfer sequences with different sets S. The set $\mathcal{P}_{n-2}$ is defined up to these "isomorphisms". Working with the more general definition of Prüfer sequences makes the proofs easier later. However, we will use the labels $1, 2, \ldots, n$ for all examples.

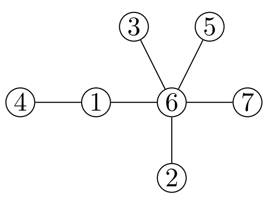
Every labeled trees on $n \geq 2$ can be encoded in a Prüfer sequence by Algorithm 1. We will prove later that this map is a bijection.

Algorithm 1 Prüfer Encoding

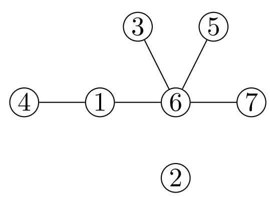
function $f_e(T_0)$ ▷ Input: a labeled n-vertex tree T_0
  $T \leftarrow T_0$
  for $i \leftarrow 1, (n-2)$ **do**
    $v \leftarrow$ the leaf vertex with the smallest label
    $p_i \leftarrow$ the label of the only neighbor w of v
    $T \leftarrow T - v$
  end for
  return $\langle p_1, p_2, \ldots, p_{n-2} \rangle$ ▷ Output: a Prüfer sequence of length $n-2$
end function

Example 3. Here are some examples of using Prüfer Encoding for some labeled graphs.

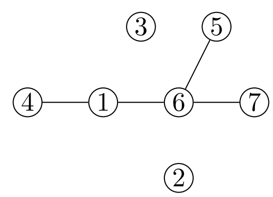
(1) Graph 1:



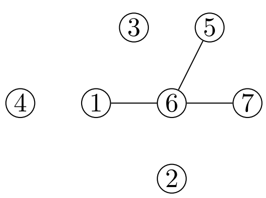
$P = \langle \rangle$



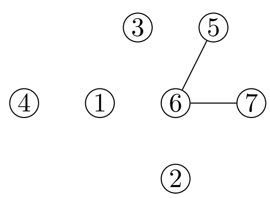
$P = \langle 6 \rangle$



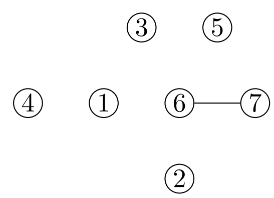
$P = \langle 6, 6 \rangle$



$P = \langle 6, 6, 1 \rangle$

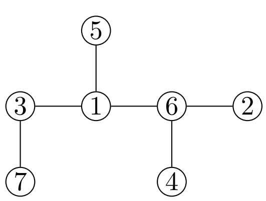


$P = \langle 6, 6, 1, 6 \rangle$

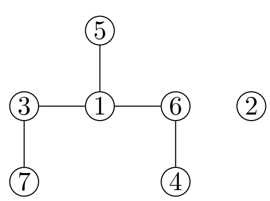


$P = \langle 6, 6, 1, 6, 6 \rangle$

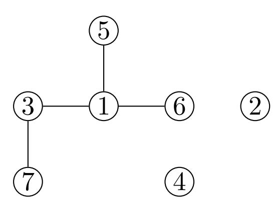
(2) Graph 2:



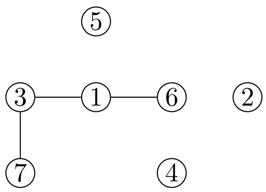
$P = \langle \rangle$



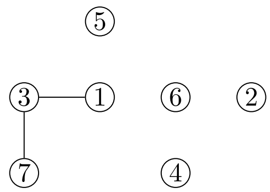
$P = \langle 6 \rangle$



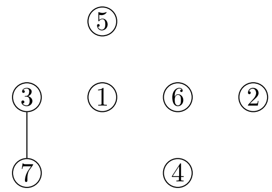
$P = \langle 6, 6 \rangle$



$P = \langle 6, 6, 1 \rangle$

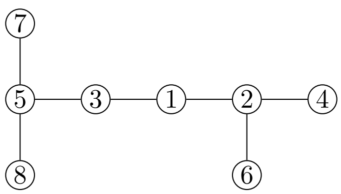


$P = \langle 6, 6, 1, 1 \rangle$

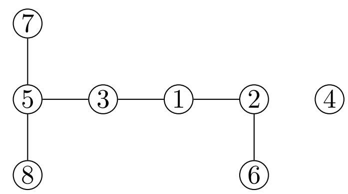


$P = \langle 6, 6, 1, 1, 3 \rangle$

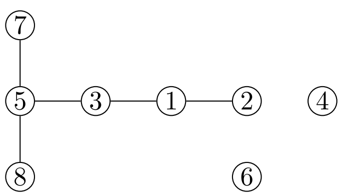
(3) Graph 3:



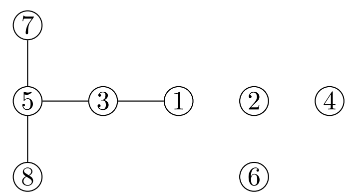
$P = \langle \rangle$



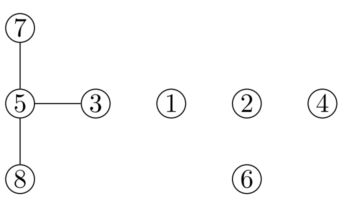
$P = \langle 2 \rangle$



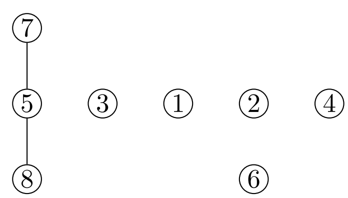
$P = \langle 2, 2 \rangle$



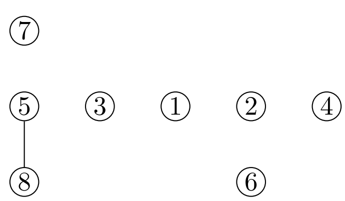
$P = \langle 2, 2, 1 \rangle$



$P = \langle 2, 2, 1, 3 \rangle$



$P = \langle 2, 2, 1, 3, 5 \rangle$



$P = \langle 2, 2, 1, 3, 5, 5 \rangle$

In the examples, we observe that all leave vertices do not appear in the Prüfer sequences encoding the trees. Also, if we count the number of occurrences of each labels in the sequence, they turn out to be equal to the degree of the corresponding vertices minus one. This fact is true for every labeled trees by the following proposition.

Proposition 4. *Let d_s be the number of occurrences of the number s in a Prüfer encoding sequence for a labeled tree T. Then the degree of vertex with the label s in T equals $d_s + 1$.*

Proof. We prove this statement by induction on the number n of vertices. For $n = 2$, the Prüfer sequence is an empty sequence. The statement is true in this case since the degree of each vertex in the tree is 1 and their labels have 0 occurences in the Prüfer sequence.

Now, assume that the statement is true for some $n \geq 3$. Let T be a labeled tree with $n+1$ vertices. Also, let v be the leaf vertex with the smallest label and w be its neighbor. Then, the label p_1 of w is the first element in the Prüfer sequence P of T. Let P_1 be the subsequence of P following p_1. The sequence P_1 is the Prüfer sequence of the tree $T_1 = T - v$. We can apply the induction hypothesis for T_1 since it has n vertices. Hence, if d_s is the number of occurrences of s in P_1, then $\deg_{T_1}(u) = d_s + 1$ where u is the vertex in T_1 with label s. In addition, we have $\deg_T(u) = \deg_{T_1}(u)$ for any $u \neq w$, and we also have $\deg_T(w) = \deg_{T_1}(w) + 1$ since w has one more neighbor v in T. If d_{p_1} is the number of occurrences of p_1 in P_1, then there are $d_{p_1} + 1$ occurrences of p_1 in P. Thus, $\deg_T(w) = \deg_{T_1}(w) + 1 = (d_{p_1} + 1) + 1$. As a result, the statement is true for any vertex in T. ☐

3 Prüfer Decoding

If there is an encoding procedure, then there should be a decoding procedure, which recovers a tree from a Prüfer sequence. The decoding procedure is given in Algorithm 2.

Algorithm 2 Prüfer Decoding

function $f_d(P_0)$ ▷ Input: a Prüfer sequence P_0 of length $n - 2$
    $P \leftarrow P_0$
    $S \leftarrow \{s_1, s_2, \ldots, s_n\}$
    $T \leftarrow$ a labeled graph with n isolated vertices
    for $i \leftarrow 1, (n-2)$ **do**
        $s \leftarrow$ the smallest label in S that is listed not in P
        $p \leftarrow$ the first element in P
        Add to T an edge joining the vertices labeled s and p
        Remove s from S
        Remove the first occurence of p from P
    end for
    Add to T an edge joining the two vertices with the two remaining labels in S
    return T ▷ Output: a labeled n-vertex tree T, which is proved below
end function

The following proposition shows that the output of this procedure is indeed a labeled tree.

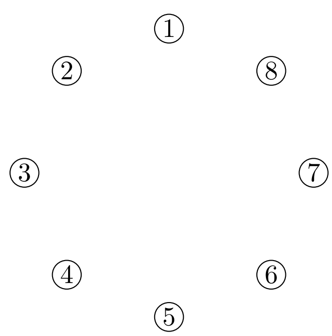
Proposition 5. *The Prüfer decoding procedure defines a function $f_d : \mathcal{P}_{n-2} \to \mathcal{T}_n$.*

Proof. From a Prüfer sequence of length $n-2$, the Prüfer decoding procedure always returns a labeled graph with n vertices. We only need to prove that this graph is a tree. We do this by induction on the number n of vertices. The case $n = 2$ is trivial since we only add an edge joining 2 vertices to obtain a labeled tree on 2 vertices.

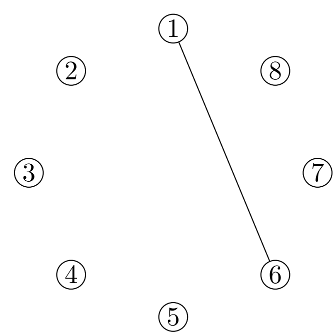
Now, assume for some $n \geq 2$ that the Prüfer decoding procedure always returns a labeled tree on n vertices. Consider a set of labels $S = \{s_1, s_2, \ldots, s_{n+1}\}$ and any Prüfer sequence $P = \langle p_1, p_2, \ldots, p_{n-1}\rangle$ of length $n-1$. In the first step of the decoding procedure, we add an edge e joining p_1 and s_k, where s_k is the smallest label in S that is not in P. Since we remove s_k from S in later iterations, there is no more edge joining s_k and other vertices besides p_1. If we remove s_k from the graph, then the iterations from 2 to n (including the step when joing 2 remaining labels in S) yield a labeled graph on n vertices. Since the procedure from iteration 2 to iteration n is also a decoding procedure with the Prüfer sequence $P_1 = \langle p_2, p_3, \ldots, p_{n-1}\rangle$ and the set of labels $\{s_1, s_2, \ldots, s_{k-1}, s_{k+1}, \ldots, s_{n+1}\}$, the induction hypothesis implies that this procedure yields a labeled tree on n vertices. Adding back the vertex s_k, and the edge e joining s_k and p_1, creates a labeled tree on $n+1$ vertices since we only add a new leave. □

Example 6. Here are examples of Prüfer decoding for recovering labeled trees from Prüfer sequences.

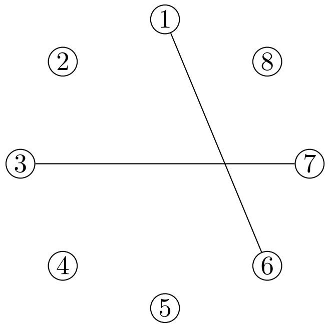
(1)



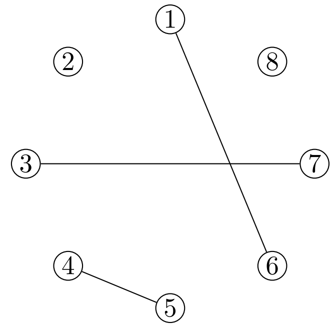
$P = \langle 6, 7, 4, 4, 4, 2\rangle$
$S = \{1, 2, 3, 4, 5, 6, 7, 8\}$



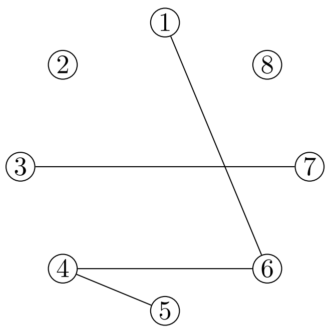
$P = \langle 7, 4, 4, 4, 2\rangle$
$S = \{2, 3, 4, 5, 6, 7, 8\}$



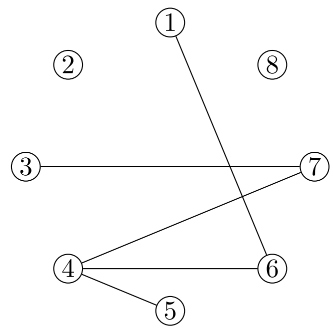
$P = \langle 4, 4, 4, 2 \rangle$
$S = \{2, 4, 5, 6, 7, 8\}$



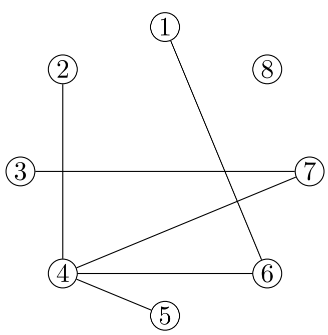
$P = \langle 4, 4, 2 \rangle$
$S = \{2, 4, 6, 7, 8\}$



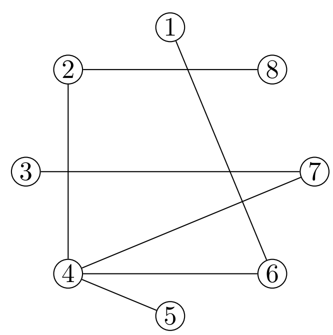
$P = \langle 4, 2 \rangle$
$S = \{2, 4, 7, 8\}$



$P = \langle 2 \rangle$
$S = \{2, 4, 8\}$

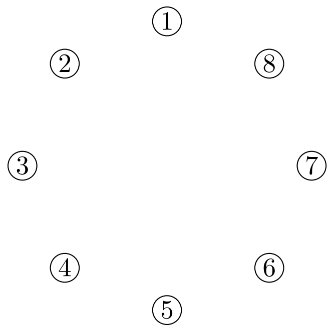


$P = \langle \rangle$
$S = \{2, 8\}$

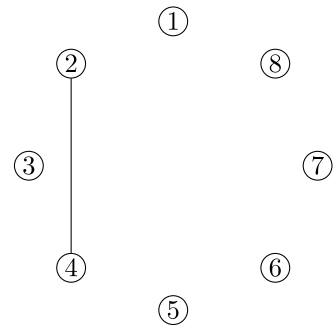


Resulting tree T

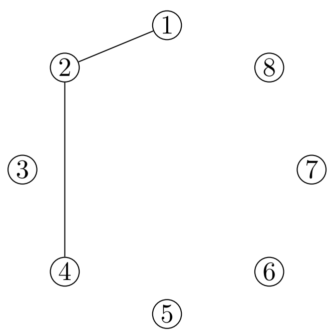
(2)



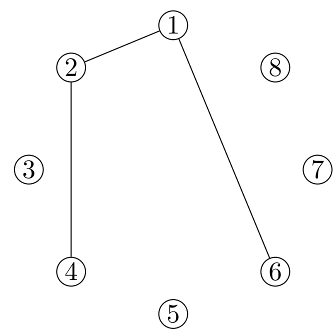
$P = \langle 2, 1, 1, 3, 5, 5 \rangle$
$S = \{1, 2, 3, 4, 5, 6, 7, 8\}$



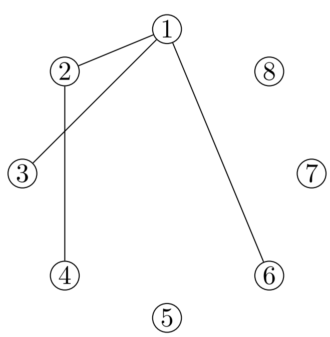
$P = \langle 1, 1, 3, 5, 5 \rangle$
$S = \{1, 2, 3, 5, 6, 7, 8\}$



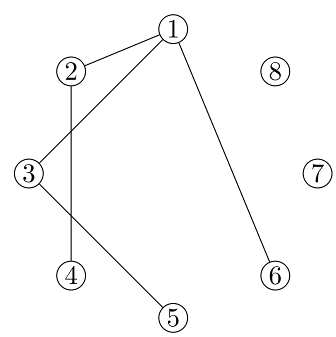
$P = \langle 1, 3, 5, 5 \rangle$
$S = \{1, 3, 5, 6, 7, 8\}$



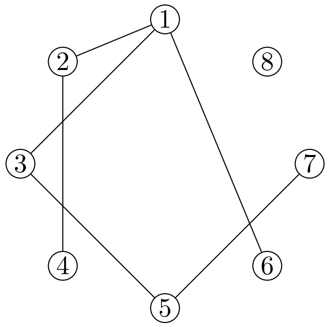
$P = \langle 3, 5, 5 \rangle$
$S = \{1, 3, 5, 7, 8\}$



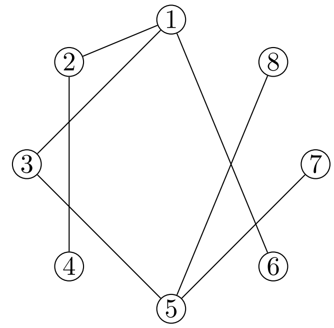
$P = \langle 5, 5 \rangle$
$S = \{3, 5, 7, 8\}$



$P = \langle 5 \rangle$
$S = \{5, 7, 8\}$

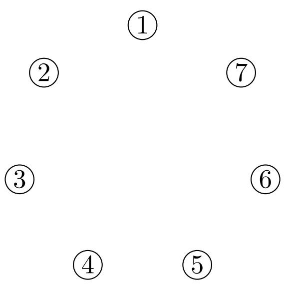


$P = \langle \rangle$
$S = \{5, 8\}$

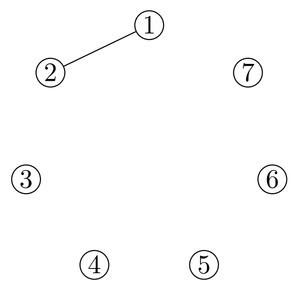


Resulting tree T

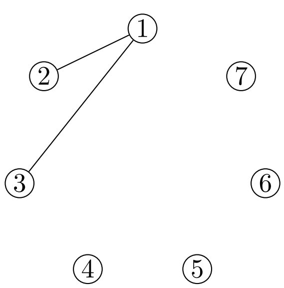
(3)



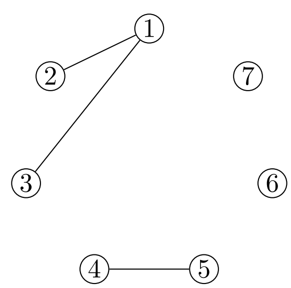
$P = \langle 1, 1, 5, 1, 5 \rangle$
$S = \{1, 2, 3, 4, 5, 6, 7\}$



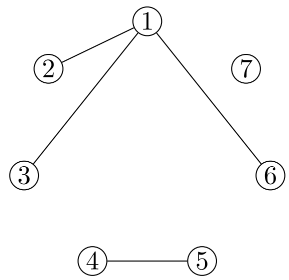
$P = \langle 1, 5, 1, 5 \rangle$
$S = \{1, 3, 4, 5, 6, 7\}$



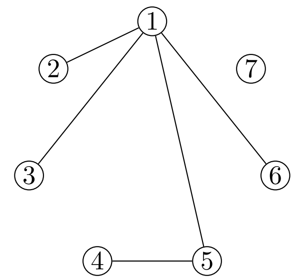
$P = \langle 5, 1, 5 \rangle$
$S = \{1, 4, 5, 6, 7\}$



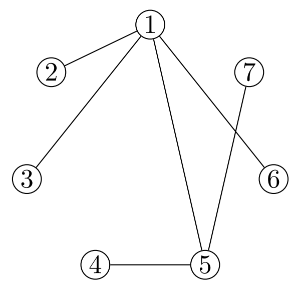
$P = \langle 1, 5 \rangle$
$S = \{1, 5, 6, 7\}$



$P = \langle 5 \rangle$
$S = \{1, 5, 7\}$



$P = \langle \rangle$
$S = \{5, 7\}$



Resulting tree T

The following proposition shows that the encoding function f_e is a bijection with the inverse f_d.

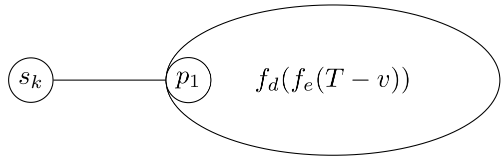
Proposition 7. *The decoding function* $f_d : \mathcal{P}_{n-2} \to \mathcal{T}_n$ *is the inverse of the encoding function* $f_e : \mathcal{T}_n \to \mathcal{P}_{n-2}$.

Proof. In order to prove this statement, we need to prove that $f_d \circ f_e = \mathrm{id}_{\mathcal{T}_n}$ and $f_e \circ f_d = \mathrm{id}_{\mathcal{P}_{n-2}}$. We prove these by induction on the number n of vertices.

(i) *Proof of* $f_d \circ f_e = \mathrm{id}_{\mathcal{T}_n}$

For $n = 2$, $f_e(T)$ is an empty sequence, where T is a labeled tree on 2 vertices, labeled by 1 and 2. Then, $f_d(f_e(T)) = T$ since we only add an edge joining 1 and 2 at the last step of the decoding procedure.

Assume for some $n \geq 2$ that $f_d(f_e(T)) = T$ for any labeled tree T on n vertices. Now, let T be any labeled tree on $n+1$ vertices, v be the leaf vertex with the smallest label s_k, and p_1 be the label of the neighbor w of v. Then, we have $f_e(T) = \langle p_1, f_e(T - v) \rangle$. By Proposition 4, the number of occurrences of s_k in $f_e(T)$ is $d_k = \deg(v) - 1 = 0$, and hence, s_k does not appear in $f_e(T)$. Thus, with s_k being the smallest label that is not in $f_e(T)$, the labeled tree $f_d(f_e(T)) = f_d(\langle p_1, f_e(T - v) \rangle)$ comprises a leaf vertex labeled s_k and an edge joining it to a vertex labeled p_1 in $f_d(f_e(T - v))$. Since $T - v$ contains n labeled vertices, one has $f_d(f_e(T - v)) = T - v$ by the induction hypothesis. As a result, $f_d(f_e(T))$ consists of the leaf vertex labelel s_k, the edge joining it to the vertex w in $T - v$, and therefore, $f_d(f_e(T)) = T$.

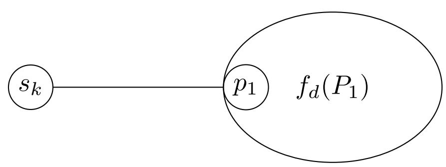


$f_d(f_e(T))$

(ii) *Proof of* $f_e \circ f_d = \mathrm{id}_{\mathcal{P}_{n-2}}$

For $n = 2$, if P is an empty sequence, then $f_d(P)$ is a labeled tree on 2 vertices since we only add an edge joining the two vertices at the last step of the decoding procedure. Hence, $f_e(f_d(P)) = P$ since we do nothing in the encoding procedure.

Assume for some $n \geq 2$ that $f_e(f_d(P)) = P$ for any Prüfer sequence P of length $n-2$ with a set of n labels S. Now, let $S = \{s_1, s_2, \ldots, s_{n+1}\}$ be a set of $n+1$ labels, $P = \langle p_1, p_2, \ldots, p_{n-1}\rangle$ be a Prüfer sequence of length $n-1$, $P_1 = \langle p_2, p_3, \ldots, p_{n-1}\rangle$ be its subsequence, and s_k be the smallest label that is not in P. Then, $f_d(P)$ consists of a vertex v labeled s_k and an edge joining it to a vertex w labeled p_1 in $f_d(P_1)$. Since any labels smaller than s_k appear in P, their degrees are greater than 1 by Proposition 4. Hence, v is the leaf vertex with the smallest label. Since the neigbor w of v is labeled p_1, we have $f_e(f_d(P)) = \langle p_1, f_e(f_d(P_1))\rangle$. By the induction hypothesis, $f_e(f_d(P_1)) = P_1$ since P_1 is a Prüfer sequence P of length $n-2$ with the set of n labels $S_1 = \{s_1, \ldots, s_{k-1}, s_{k+1}, \ldots, s_{n+1}\}$. As a result, $f_e(f_d(P)) = \langle p_1, P_1\rangle = P$.



$f_d(P)$

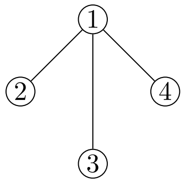
□

After preparing all the ingredients, we can finally prove Cayley's tree formula.

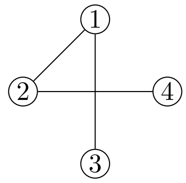
Theorem 8 (Cayley's Tree Formula)**.** *The number of elements in* $\mathcal{T}_n$ *is* n^{n-2}*.*

Proof. By the preceding statements, we have established a one-to-one correspodence f_e from $\mathcal{T}_n$ to $\mathcal{P}_{n-2}$, so they have same number of elements. To create a Prüfer sequence P, we can independently choose one label from n elements of S for each position in P. Hence, the number of different Prüfer sequences of length $n-2$ is n^{n-2} by the rule of product. □

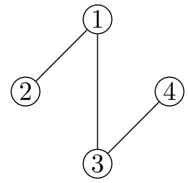
Example 9. We can list all labeled trees on 4 vertices by writing down all Prüfer sequences of length 2 and use the Prüfer decoding procedure. There are indeed 4^{4-2} or 16 labeled trees on 4 vertices.



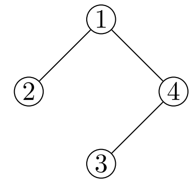
$P = \langle 1, 1 \rangle$



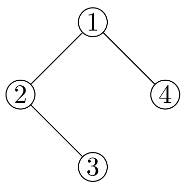
$P = \langle 1, 2 \rangle$



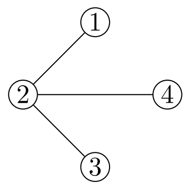
$P = \langle 1, 3 \rangle$



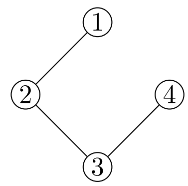
$P = \langle 1, 4 \rangle$



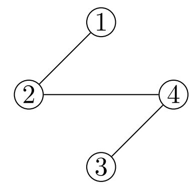
$P = \langle 2, 1 \rangle$



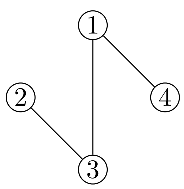
$P = \langle 2, 2 \rangle$



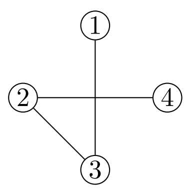
$P = \langle 2, 3 \rangle$



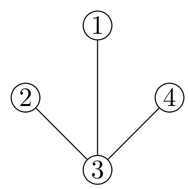
$P = \langle 2, 4 \rangle$



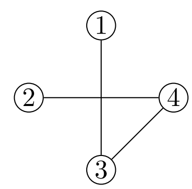
$P = \langle 3, 1 \rangle$



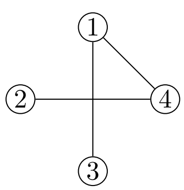
$P = \langle 3, 2 \rangle$



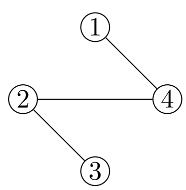
$P = \langle 3, 3 \rangle$



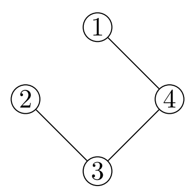
$P = \langle 3, 4 \rangle$



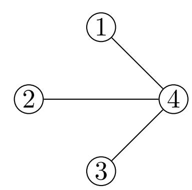
$P = \langle 4, 1 \rangle$



$P = \langle 4, 2 \rangle$



$P = \langle 4, 3 \rangle$



$P = \langle 4, 4 \rangle$

References

[1] Serge Richard. *Special Mathematics Lecture: Graph Theory*, 2024.

[2] J.L. Gross, J. Yellen, M. Anderson. *Graph theory and its applications*, CRC press, 2019.