Trees, I

李昂生

Discrete Mathematics
U CAS
July, 2018

Outline

Background

- Cayley 1857 used tree to study the types of chemical compounds
- Trees provide a methodology for various subjects of sciences, especially, for graph algorithms
- Trees are the structure for data storage, searching and communications

Trees in Nature

- Family trees
- Evolutionary trees
- Chemical compounds
- Social organisations

Knowledge tree

- Any book is organised as a tree of knowledge

Why trees?

- Why trees are so fundamental?

Structural information theory answers this question.

Definition

- A graph with no cycle is *acyclic*.
- A *forest* is an acyclic graph.
- A *tree* is a connected acyclic graph.
- A *leaf* (or *pendant vertex*) is a vertex of degree 1.
- A *spanning subgraph* of G is a subgraph with vertex set V_G.
- A *spanning tree* of G is a spanning subgraph that is a tree.

Examples of Tree

- A tree is a connected forest.
- Every component of a forest is a tree.
- A graph with no cycles has no odd cycles; hence trees and forests are bipartite.
- Paths are trees.
- A tree is a path if and only if its maximum degree is 2.
- A star is a tree consisting of one vertex adjacent to all the others.

Basic Properties

Lemma 1

(1) *Every tree with at least two vertices has leaves.*

(2) *Deleting a leaf from an n-vertex tree produces a tree of $n - 1$ vertices.*

For (1).
Let $T = (V, E)$ be a tree with $|V| \geq 2$.
Let P be a longest nontrivial path in T. Suppose that P has the form:

$$P: \ u = x_0, x_1, \cdots, x_{l-1}, x_l = v,$$

for some $u \neq v$.
Note: A path is nontrivial, if all the vertices appeared in P are distinct.
By the choice of P, both u and v are leaves of T.

Proof - continued

For (2).
Let v be a leaf of a tree G and let $G' = G - v$.
For $x, y \in V_{G'}$, any path from x to y in G is a path in G'. So G' is connected. Since G has no cycle, so is G'. G' is a tree of $n - 1$ vertices.

Characterisations of Tree

Theorem 2

Let $G = (V, E)$ be a graph of n vertices. Then the following properties are equivalent:

A *G is connected and has no cycles.*

B *G is connected and has $n - 1$ edges.*

C *G has $n - 1$ edges and no cycles.*

D *G has no loops and has, for each $u, v \in V_G$, exactly one path from u to v.*

A $\Rightarrow$ B, C

By induction on n.
Basis step: $n = 1$. An acyclic 1-vertex graph has no edge.
Inductive Step: $n > 1$.
Suppose that the implication holds for graphs with vertices $< n$.
Let G be an n-vertex, acyclic and connected graph. By the basic properties before, there are leaves in G. Let v be a leaf of G, and let $G' = G - v$. Then G' is acyclic and connected. Applying the inductive hypothesis, $e(G') = n - 2$. This shows that $e(G) = n - 1$.

B $\Rightarrow$ A, C

Let $G = (V, E)$ be connected, and $e(G) = n - 1$.
Let G' be the graph obtained from G by deleting one edge from a cycle one by one, until the resulting graph G' is acyclic. Since no edge of a cycle is a cut edge, G' is connected. By the proof of A $\Rightarrow$ B, C, $e(G') = n - 1$. By the assumption, $e(G) = n - 1$, no edge is deleted. $G' = G$ and G is acyclic.

C $\Rightarrow$ A, B

Let G be a graph with $n-1$ edges and no cycles.
Suppose that $G_1, \cdots, G_k$ are all the connected components of G. Since every vertex appears in exactly one of the components. Let n_i be the number of vertices in G_i. Then

$$\sum_{i=1}^{k} n_i = n.$$

Since G has no cycles, each G_i has no cycles. By the proof of A$\Rightarrow$ B, C, for each i, $e(G_i) = n_i - 1$.
Summing over i yields

$$e(G) = \sum_{i=1}^{k} (n_i - 1) = n - k = n - 1,$$

so $k = 1$ and G is connected.

A $\Rightarrow$ D

Suppose that G is connected and has no cycles.
Suppose to the contrary that there is a pair x, y that is connected by more than one path. Let P, Q be two distinct paths between x and y. Then there is a cycle in the union of P and Q. A contradiction.

D $\Rightarrow$ A

Clearly G is connected.
Towards a contradiction, suppose that G has a cycle C. Then there are vertices u, v on C for which there are two paths between u and v. A contradiction.

Corollary

a) Every edge of a tree is a cut-edge.
b) Adding one edge to a tree forms exactly one cycle.
c) Every connected graph contains a spanning tree.

Proof.
For a). A tree has no cycles, so every edge is a cut-edge.
For b). A tree has a unique path linking each pair of vertices, so joining them by an edge creates exactly one cycle.
For c). Iteratively deleting edges from cycles one by one in a connected graph yields a connected acyclic spanning subgraph that is a spanning tree. □

Pair of Spanning Trees - I

Proposition 1

If T, T' are spanning trees of a connected graph G and $e \in E_T \setminus E_{T'}$, then there is an edge $e' \in E_{T'} \setminus E_T$ such that $T - e + e'$ is a spanning tree of G.

Proof.

Every edge of T is a cut-edge. Let U, U' be the two components of $T - e$. Since T' is connected, T' has an edge e' with endpoints in U and U'. Now $T - e + e'$ is connected and has $n_G - 1$ edges, and is a spanning tree of G. $\square$

Pair of Spanning Trees - II

Proposition 2

If T, T' are spanning trees of a connected graph G and $e \in E_T \setminus E_{T'}$, then there is an edge $e' \in E_{T'} \setminus E_T$ such that $T' + e - e'$ is a spanning tree of G.

Proof.

$T' + e$ contains a unique cycle C. T is acyclic, so there is an edge $e' \in C \setminus E_T$. Deleting e' breaks the unique cycle C. So $T' + e - e'$ is connected and acyclic, and is a spanning tree of G. □

Tree as a Subgraph

Proposition 3

If T is a tree with k edges and G is a simple graph with $\delta(G) \geq k$, then T is a subgraph of G, where $\delta(G)$ is the least degree of vertices of G.

We prove by induction on k.
Basis step: $k = 0$.
T is an isolated vertex, which is certainly a subgraph of G.
Induction step: $k > 0$.
Suppose by induction that the proposition holds for all $k' < k$.
T is a tree with $k > 0$ edges. Then there must be leaves in T.
Let v be a leaf in T; and u be a neighbor of v in T.
For $T' = T - v$. By inductive hypothesis, T' is a subgraph of G, since $\delta(G) \geq k > k - 1$.

Proof - continued

T' has $k-1$ edges and k vertices, including u.
By the assumption that $d(u) \geq \delta(G) \geq k$, there must be some vertex w such that wu is an edge of G, w is not a vertex of T'.
Let $T^{''} = T' + wu$. Then $T^{''}$ is a copy of T that is a subgraph of G.

Rooted Trees

A rooted tree is a tree in which one vertex has been designated as the root and every edge is directed away from the root.

Some notations of rooted trees:

- Parent vertex
- Child
- Siblings
- Ancestors
- Descendants
- Leaf
- Internal vertices
- Subtrees

m-ary Trees

- Every internal vertex has m immediate successors.
- $m = 2$ - Binary trees
- Priority tree is the ordered rooted tree.

Trees as Models

- Computer science
- Chemistry
- Geology
- Botany
- Psychology
- Social organisations
- Parallel computation

Questions: Why trees?

Properties of Rooted Trees

Proposition 4

(1) *A tree with n vertices has $n - 1$ edges.*

(2) *A full m-ary tree with i internal vertices contains $n = mi + 1$ vertices.*

(3) *A full m-ary tree with*

(3a) *n vertices has $i = \frac{n-1}{m}$ internal vertices and $l = \frac{(m-1)n+1}{n}$ leaves.*

(3b) *i internal vertices has $n = mi + 1$ vertices and $l = (m - 1)i + 1$ leaves.*

(3c) *l leaves has $n = (ml - 1)/(m - 1)$ vertices and $i = (l - 1)/(m - 1)$ internal vertices.*

Height of a Tree

Definition 3

The *height* of a rooted tree is the length of the longest path from the root to all the other vertices.

Definition 4

A tree is called *balanced*, if all leaves have heights k or $k + 1$, for some fixed k.

Theorem 5

There are at most m^h leaves in an m-ary tree of height h.

By induction on h.

Open question

Are the distances of a random tree uniformly distributed?
How to measure the uniformity, if any?

Binary Search Tree

Decision Tree

Prefix Free Codes

Given an alphabet Σ, we may encode all symbols in Σ by $0, 1$-strings such that for any $s_1, s_2 \in \Sigma$, if $s_1 \neq s_2$, then $\sigma(s_1) \not\subseteq \sigma(s_2)$, where $\sigma(s)$ is the codeword of s.
Then

$$\{\sigma(s) \mid s \in \Sigma\}$$

is the prefix free encoding of Σ.
The prefix free encoding of Σ allows to determine the word $s = s_1 s_2 \cdots s_k$ from the codeword $\sigma(s)$ of s. That is, if σ is a $0, 1$-string that encodes s, then the first symbol s_1 of s is the unique symbol $y \in \Sigma$ such that the codeword of y is an initial segment of σ. Deleting the initial segment of s_1 from σ, we obtain a code word σ_1. Then the second symbol s_2 of σ is the symbol $z \in \Sigma$ whose codeword is the initial segment of σ_1.
Repeating the procedure, we decode s from the codeword σ.

Prefix Free Codes by Binary Tree

Given a rooted binary tree, we assign a label 0 or 1 to each of the edges such that for a node α of the tree, the immediate successors of α are $\alpha_0 = \alpha\hat{}\langle 0\rangle$ and $\alpha_1 = \alpha\hat{}\langle 1\rangle$ with α_0 to the left of α_1.
Let T be such a tree. Then every leaf of T is denoted by the labels on the path from the root to the leaf, which is a $0, 1$-string.
Let C be the set of all the codewords of the leaves of T. Then C is a prefix free encoding system, since for every $\alpha, \beta \in C$, $\alpha \not\subseteq \beta$, i.e., α is not an initial segment of β.

Prefix Free Encoding

As above, we may construct a prefix free codes of a set Σ to be the set of all the leaves of an ordered rooted binary tree. However, in real application, the elements in Σ may have different frequencies of occurrence. To save the length of codewords, we may assign the symbols in Σ that have higher frequencies to have the codes of shorter lengths, and the symbols in Σ having low frequencies to have the codes of longer lengths.

Understanding of the prefix free codes

- The length of codewords may vary
- We need a way to recognise the end of the current word.
- If no codeword is an initial segment of another, the current word ends as soon as the bits after the end of the last word form a codeword
- Under prefix free coding, the binary codewords corresponding to the leaves of a binary tree, the left/right i.e., $0/1$, edges.
- Suppose that item i has probability p_i and has code of length l_i. Then the expected length of a message is

$$\sum p_i l_i.$$

Huffman's Prefix-Free Coding

Input: Weights or distribution:

$$p_1, p_2, \cdots, p_n.$$

Output: Prefix-Free Code
Idea: If p_i is small, then i is coded by longer $0/1$-string.
Initial Case: $n = 2$
The two codes are:

$$0, 1.$$

Shannon codes

Let $p = (p_1, p_2, \cdots, p_n)$ be a probability distribution of set $N = \{1, 2 \cdots, n\}$.
For each i, let $l_i = \lceil -\log_2 p_i \rceil$. We can construct a prefix-free codes α_i for i such that for each i, the length of α_i is $l(\alpha_i) = l_i$.
(Prove this!)

Recursion

Let $n > 2$.

(1) Let i, j be such that the least two weights are p_i and p_j.

(2) Let $q = p_i + p_j$.

(3) Find the prefix-free codes for

$$p_1, \cdots, p_{i-1}, p_{i+1}, \cdots, p_{j-1}, p_{j+1}, \cdots, p_n, q.$$

(4) Split the item with weight q into two, labelled 0 and 1 with weights p_i and p_j, respectively.

Huffman Codes are Optimal

Theorem 6

Given a probability distribution $\{p_i\}$ on n items, Huffman's algorithm produces the prefix-free codes with minimum expected length.

We prove by induction on n

Basis step: $n = 2$.

1 bit is necessary. Therefore, it is the shortest possible length of codes.

Inductive step: $n > 2$.

Suppose by induction that the result holds for any distribution with $n - 1$ items.

Note that every code assigns items to the leaves of a ordered, rooted binary tree.

Inductive Step

Given a fixed tree with n leaves, we minimise the expected length by greedily assigning the messages with probability

$$p_1 \geq p_2 \geq \cdots \geq p_n$$

to leaves in increasing order of depth.
Thus every optimal code has least likely messages assigned to leaves of greatest depth.
Since every leaf at the greatest depth has another leaf as its sibling and permuting the items at the same depth does not change the expected length.
We assume that the least likely messages appear as siblings at the greatest depth.

Inductive Step - continued

Let T be an optimal tree for $p_1, \cdots, p_n$ with the least likely items p_n and p_{n-1} located at sibling leaves at the greatest depth.
Let T' be the tree obtained from T by deleting these leaves, and let $q_i = p_i$ for $i \leq n-2$, and $q_{n-1} = p_{n-1} + p_n$. The tree T' gives a code for $\{q_i\}$. Let k be the height of the leaf assigned for q_{n-1}.
The expected length for T is the expected length for T' plus q_{n-1}, since if k is the depth of the leaf assigned q_{n-1}, we lose kq_{n-1} and gain $(k+1)(p_{n-1}+p_n)$ in moving from T' to T.
Therefore, the expected length for $\{p_i\}$ is at least the expected length for $\{q_i\}$ plus $p_{n-1}+p_n$.
Key:

$$l^*(p) \geq l(T'; q) + p_{n-1} + p_n \geq l^*(q) + p_{n-1} + p_n,$$

where $l^*(p)$ is the least expected length.

Inductive Step - continued

By inductive hypothesis, the Huffman algorithm finds an optimal tree T' for $\{q_i\}$. For $\{p_i\}$, the Huffman algorithm first replaces $\{p_{n-1}, p_n\}$ by q_{n-1}. The algorithm gives a tree with length to be the least expected legth for $\{q_i\}$ plus $p_{n-1} + p_n$. Therefore, the least expected length for $\{p_i\}$ must be be at most the least expected length for $\{q_i\}$ plus $p_{n-1} + p_n$, which has been achieved by the Huffman algorithm.

Shannon's entropy

Definition 7

Given a probability $p = (p_1, p_2, \cdots, p_n)$, the Shannon entropy of p is defined by

$$H(p) = -\sum_{i=1}^{n} p_i \log_2 p_i. \tag{1}$$

Intuition of Shannon's entropy

Intuition

- $H(p)$ is the measure of the information (uncertainty) contained in the probability distribution p.
- The probability distribution p is unstructured.
- $H(p)$ is a number characterising the global uncertainty of p.
- $H(p)$ fails to support clearing of a data
- $H(p)$ is a one-dimensional metric

Question: How to define the metric of information (uncertainty) embedded in a physical system or graph that supports the analysis of the graph?

Shannon's entropy is the lower bound

Theorem 8

For every code with binary digits, the expected length is at least the entropy of $\{p_i\}$*, which is*

$$-\sum_{i=1}^{n} p_i \log_2 p_i.$$

Proof - 1

Proof.

$$\begin{aligned} L - H(p) &= \sum_{i=1}^{n} p_i l_i + \sum_{i=1}^{n} p_i \log_2 p_i \\ &= \sum_{i=1}^{n} p_i \log_2 \frac{p_i}{2^{-l_i}} \end{aligned}$$

□

Proof - 2

Proof.

Set $q_i = 2^{-l_i}$. Then $\sum_{i=1}^{n} q_i = \sum_{i=1}^{n} 2^{-l_i} \leq 1$ (you are asked to prove), and

$$\begin{aligned}
-(L - H(p)) &= -\sum_{i=1}^{n} p_i \log_2 \frac{p_i}{q_i} \\
&= \sum_{i=1}^{n} p_i \log_2 \frac{q_i}{p_i} \\
&\leq \log_2(\sum_{i=1}^{n} p_i \frac{q_i}{p_i}), \text{log is convex,} \\
&\leq 0, \text{since} \sum_{i=1}^{n} q_i \leq 1.
\end{aligned}$$

This shows that $L \geq H(p)$, equality holds if and only if

Priority tree

Definition 9

A priority tree T is a rooted tree with root node λ.
Suppose that $\alpha \in T$ is a node in T, then possible outcomes of α are

$$0 <_{\mathrm{L}} 1 <_{\mathrm{L}} \cdots <_{\mathrm{L}} k$$

for some natural number k.
Then the immediate successors of α in T are the possible outcomes of α listed as that in the possible outcomes of α.

Coding tree of a graph

Definition 10

(Li, Pan) Let $G = (V, E)$ be a graph. We say that a priority tree T is a coding tree of G, if for every tree node $\alpha \in T$, there is a set $X_\alpha \subseteq V$ associated with α, satisfying:

1. For the root node λ, the associated set of vertices $X_\lambda = V$.
2. For every node $\alpha \in T$, suppose that $\alpha\hat{}\langle 0\rangle, \alpha\hat{}\langle 1\rangle, \cdots, \alpha\hat{}\langle k\rangle$ are all the immediate successors of α in T. Let $\beta_i = \alpha\hat{}\langle i\rangle$, then
   $\{X_{\beta_0}, \cdots, X_{\beta_k}\}$ is a partition of X_α.
3. For every leave node $\gamma \in T$, X_γ is a singleton, in which case, if $X_\gamma = \{v\}$, then we say that γ is a codeword of v.
4. For every vertex $v \in V$, there is a unique tree node $\gamma \in T$ such that $X_\gamma = \{v\}$, i.e., v has a codeword γ in T.

Structural Information by Coding Tree

Definition 11

(Structural information of a graph by a coding tree, Li, Pan) For an undirected and connected network $G = (V, E)$, suppose that $\mathcal{T}$ is a coding tree of G. We define the structural information of G by $\mathcal{T}$ as follows:

(1) For every $\alpha \in \mathcal{T}$, if $\alpha \neq \lambda$, then define

$$\mathcal{H}^{\mathcal{T}}(G) = \sum_{\alpha \in \mathcal{T}, \alpha \neq \lambda} -\frac{g_\alpha}{2m} \log_2 \frac{V_\alpha}{V_{\alpha^-}}, \tag{2}$$

where g_α is the number of edges from nodes in T_α to nodes outside T_α, V_β is the volume of set T_β, namely, the sum of the degrees of all the nodes in T_β.

Structural Information

Definition 12

(Structural information, Li, Pan) Let $G = (V, E)$ be a connected network.

(1) We define the structural information of G as follows:

$$\mathcal{H}(G) = \min_{\mathcal{T}}\{\mathcal{H}^{\mathcal{T}}(G)\}, \tag{3}$$

where $\mathcal{T}$ ranges over all of the coding trees of G.

(2) We say that $\mathcal{T}$ is a *knowledge tree* of G, if:

$$\mathcal{H}^{\mathcal{T}}(G) = \mathcal{H}(G). \tag{4}$$

K-dimensional Structural Information

Definition 13

(K-dimensional structural information, Li, Pan) Let $G = (V, E)$ be a connected network.

(1) We define the K-dimensional structural information of G as follows:

$$\mathcal{H}^K(G) = \min_{\mathcal{T}}\{\mathcal{H}^{\mathcal{T}}(G)\}, \tag{5}$$

where $\mathcal{T}$ ranges over all of the coding trees of G of height at most K.

(2) Given a K-level coding tree $\mathcal{T}$ of G, we say that $\mathcal{T}$ is the *K-dimensional knowledge tree* of G, if:

$$\mathcal{H}^{\mathcal{T}}(G) = \mathcal{H}^K(G). \tag{6}$$

Fundamental theory

Angsheng Li, Yicheng Pan, Structural Information and Dynamical Complexity of Networks, IEEE Transactions on Information Theory, Vol. 62, No. 6, pp 3290 - 3339, 2016.

Contributions and significance

- Priority tree coding of graphs
- Measure the information in any data, instead of just in probability distribution in Shannon's definition
- Decodes the essential structure during the process we are measuring the information in a noisy data structure. That is, structural information gives us not only a number, but decodes the structure in which noises and random variations are maximally excluded.
- Structural information is high-dimensional defined
- Structural information is both globally and locally measurable
- The structural information of a structured noisy data G determines and decodes the knowledge of G by excluding the perturbations by noises and random variations.
- Structural information minimization provides a principle for analysing noisy data.

Preorder Traversal

Definition 14

Let T be an ordered rooted tree with root r. Suppose that $T_1, \cdots, T_k$ are the subtrees at r from left to right in T. The preorder traversal proceeds as follows:

1) Visit r
2) Visit $T_1, T_2, \cdots, T_k$ in preorder, with order as they are listed.

Inorder Traversal

Suppose that $T_1, \cdots, T_k$ are the subtrees at root r from left to right.
The inorder traversal proceeds:

1. Visit T_1 in inorder
2. Visit r
3. Visit T_2 in inorder, then T_3 in inorder, etc, and finally, T_k in inorder.

Postorder Traversal

Let T be an ordered rooted tree with root r.

1. If T consists only of r, then visit r.
2. Otherwise. Suppose that $T_1, T_2, \cdots, T_k$ are the subtrees at r from left to right. Then the postorder traversal is visit T_1 in postorder, $\cdots$, visit T_k in post order, and finally visit r.

Infix Form

Consider:

$$(x + y)^2 + (x - 4)/3$$

Express the form by a tree such that the inorder traversal of the tree is the expression.
Then,

- The prefix traversal of the tree is called the *Polish notation*
- The postorder traversal of the tree gives *reverse Polish notation*.

谢谢！