Tree Path Labeling of Set Systems

A Generalization of Consecutive Ones Property

N. S. Narayanaswamy

Department of Computer Science and Engineering
Indian Institute of Technology Madras

21 October 2011

Outline

1. **Introduction**
   - Illustration
   - Motivation
2. **Characterization**
   - Feasible Tree Path Labeling
3. **Computation of Feasible Tree Path Labeling**
   - On k-subdivided Stars
4. **Conclusion**
   - Application

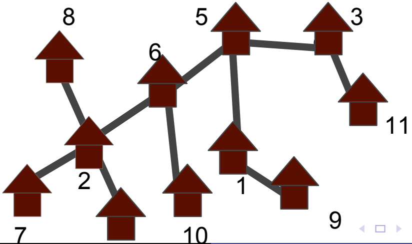
Outline

1. **Introduction**
   - Illustration
   - Motivation
2. **Characterization**
   - Feasible Tree Path Labeling
3. **Computation of Feasible Tree Path Labeling**
   - On k-subdivided Stars
4. **Conclusion**
   - Application

A Study Group Housing problem

An Illustration

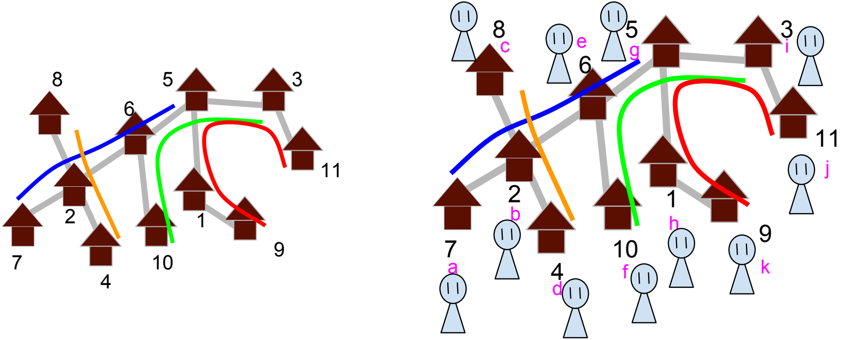
- A set $\{a, b, c, d, e, f, g, h, i, j, k\}$ of n students arrive for a summer course. They form m study groups $\{R, B, O, G\}$.
- A student is in at least one study group, $R = \{g, h, i, j, k\}$, $B = \{a, b, e, g\}$, $O = \{c, b, d\}$, $G = \{e, f, g, i\}$
- There are n single occupancy apartments in the university campus for their accommodation.
- All these apartments are placed such that streets connecting them do not form loops



A Study Group Housing problem

The problem

How should the students be allocated apartments such that each study group has consecutive apartments allocated?



Tree Path Labeling(TPL) of Set Systems

The combinatorial problem terminology

Terminology

- The set of study groups i.e. sets of students → SET SYSTEM / HYPERGRAPH
- The streets with apartments → TARGET TREE
- The route mapping to study groups → TREE PATH LABELING (TPL)
- The apartment allocation → PATH HYPERGRAPH ISOMORPHISM

Tree Path Labeling(TPL) of Set Systems

The combinatorial problem terminology

Terminology

- The set of study groups i.e. sets of students → SET SYSTEM / HYPERGRAPH
- The streets with apartments → TARGET TREE
- The route mapping to study groups → TREE PATH LABELING (TPL)
- The apartment allocation → PATH HYPERGRAPH ISOMORPHISM

Tree Path Labeling(TPL) of Set Systems

The combinatorial problem terminology

Terminology

- The set of study groups i.e. sets of students → SET SYSTEM / HYPERGRAPH
- The streets with apartments → TARGET TREE
- The route mapping to study groups → TREE PATH LABELING (TPL)
- The apartment allocation → PATH HYPERGRAPH ISOMORPHISM

Tree Path Labeling(TPL) of Set Systems

The combinatorial problem terminology

Terminology

- The set of study groups i.e. sets of students → SET SYSTEM / HYPERGRAPH
- The streets with apartments → TARGET TREE
- The route mapping to study groups → TREE PATH LABELING (TPL)
- The apartment allocation → PATH HYPERGRAPH ISOMORPHISM

Tree Path Labeling(TPL) of Set Systems

The combinatorial problem terminology

Terminology

- The set of study groups i.e. sets of students → SET SYSTEM / HYPERGRAPH
- The streets with apartments → TARGET TREE
- The route mapping to study groups → TREE PATH LABELING (TPL)
- The apartment allocation → PATH HYPERGRAPH ISOMORPHISM

Tree Path Labeling of Set Systems

The combinatorial problem

Terminology [contd.]

There *exists* an apartment allocation that “fits” the route mapping

Tree Path Labeling of Set Systems

The combinatorial problem

Terminology [contd.]

There *exists* a hypergraph isomorphism that “fits” the TPL

Tree Path Labeling of Set Systems

The combinatorial problem

Terminology [contd.]

There *exists* a hypergraph isomorphism that "fits" the TPL
$\longrightarrow$ the TPL is FEASIBLE

Tree Path Labeling of Set Systems

The combinatorial problem

Terminology [contd.]

There *exists* a hypergraph isomorphism that "fits" the TPL
$\rightarrow$ the TPL is FEASIBLE

There *exists* an apartment allocation that gives the required route mapping

Tree Path Labeling of Set Systems

The combinatorial problem

Terminology [contd.]

There *exists* a hypergraph isomorphism that "fits" the TPL
$\longrightarrow$ the TPL is FEASIBLE

There *exists* a hypergraph isomorphism that gives paths/adjacent vertices in tree

Tree Path Labeling of Set Systems

The combinatorial problem

Terminology [contd.]

There *exists* a hypergraph isomorphism that "fits" the TPL
$\longrightarrow$ the TPL is FEASIBLE

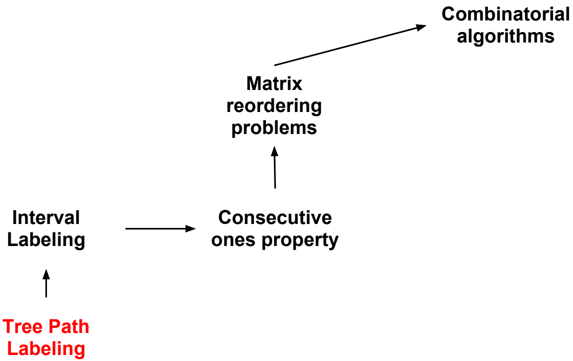
There *exists* a hypergraph isomorphism that gives paths/adjacent vertices in tree
$\longrightarrow$ the hypergraph is a PATH HYPERGRAPH

Outline

1. **Introduction**
   - Illustration
   - Motivation
2. **Characterization**
   - Feasible Tree Path Labeling
3. **Computation of Feasible Tree Path Labeling**
   - On k-subdivided Stars
4. **Conclusion**
   - Application

Consecutive Ones $\longrightarrow$ Path Labeling

The motivation



Consecutive Ones Property(COP)

Does a 0-1 Matrix have COP?

Does there exist a permutation of rows such that ones in each column occur consecutively?

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} \longrightarrow \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

COP is equivalent to interval assignment to a set system

Given,

1. a set $\mathcal{F}$ of subsets of a finite set U of cardinality n,
2. an assignment of intervals of the discrete set $\{1, \ldots, n\}$ to each element of $\mathcal{F}$

Does there exist a bijection $\phi : U \rightarrow \{1, \ldots, n\}$ such that for each element of $\mathcal{F}$, its image under ϕ is same as the interval assigned to it?

Interval Assignment to Set System

An interval assignment $I : \mathcal{F} \rightarrow I$ to $\mathcal{F}$ is called *feasible* if there exists such a bijection ϕ.

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$\mathcal{F} : \{1,3\}, \{1,5\}, \{1,3,5\}, \{2,4,5\} \Rightarrow I : [12], [23], [13], [35]$

Characterization of Feasible Interval Assignments

An interval assignment is feasible if and only if it is an Intersection Cardinality Preserving Interval Assignment (ICPIA)

$I : \mathcal{F} \rightarrow I$ is an ICPIA if

1. $|S_i| = |I(S_i)|$, for all $S_i \in \mathcal{F}$
2. $|S_i \cap S_j| = |I(S_i) \cap I(S_j)|$, for all $S_i, S_j \in \mathcal{F}$

Path Assignment to Set System

Given,

1. a set system $\mathcal{F}$ of a finite set U of cardinality n, a tree T of size n
2. a bijection called *tree path labeling*, ℓ mapping the sets in $\mathcal{F}$ to paths in T

Question

Does there exist a bijection $\phi : U \rightarrow V(T)$ such that for each $S \in \mathcal{F}$, $\{\phi(x) \mid x \in S\} = \ell(S)$?

1. A tree path labeling of a set system is called *feasible* if there exists such a bijection ϕ
2. If the path assignment is feasible, then all way intersection cardinalities are preserved
3. For trees that are paths, pairwise intersection cardinality is sufficient to guarantee that all way intersection cardinality is preserved. Can we extend this idea to aribtrary trees?

Outline

Feasible Tree Path Labeling

An Algorithmic Characterization

Feasible Tree Path Labeling

A path labeling is feasible if and only if it is an ICPPL and it successfully passes the filtering algorithms 1 and 2.

Path Labeling $l : \mathcal{F} \rightarrow \mathcal{P}$ for set system $\mathcal{F}$ and path system $\mathcal{P}$. l is an Intersection Cardinality Preserving Path Labeling (ICPPL) if

1. $|S_i| = |l(S_i)|$, for all $S_i \in \mathcal{F}$
2. $|S_i \cap S_j| = |l(S_i) \cap l(S_j)|$, for all $S_i, S_j \in \mathcal{F}$
3. $|S_i \cap S_j \cap S_k| = |l(S_i) \cap l(S_j) \cap l(S_k)|$, for all $S_i, S_j, S_k \in \mathcal{F}$

Given an ICPPL,

1. Filter 1 processes the set and path system such that there are no 2 sets whose images share a leaf of the tree
2. Filter 2 assigns set elements to leaves of the tree
3. Prune the tree of its leaves and recurse

Filter 1: Process Sets whose images share a leaf

Algorithm 1 Refine ICPPL `filter_1` $(\mathcal{F}, \ell, T)$

while $\exists S_1, S_2 \in \mathcal{F}$ whose images share a leaf of T **do**
  Remove S_1, S_2 from $\mathcal{F}$
  Add the "filtered" sets $\{S_1 \cap S_2, S_1 \setminus S_2, S_2 \setminus S_1\}$ to $\mathcal{F}$
  Retain path labeling of sets other than S_1 and S_2
  $\ell(S_1 \cap S_2) \leftarrow \ell(S_1) \cap \ell(S_2)$
  $\ell(S_1 \setminus S_2) \leftarrow \ell(S_1) \setminus \ell(S_2)$
  $\ell(S_2 \setminus S_1) \leftarrow \ell(S_2) \setminus \ell(S_1)$
  if $(\mathcal{F}, \ell)$ does not satisfy condition (3) of ICPPL **then**
    exit
  end if
end while
return $\mathcal{F}, \ell$

Filter 2: Assign Set Elements to Leaves

Algorithm 2 Leaf Labeling `filter_2(`$\mathcal{F}$, ℓ, T`)`

Path images are such that no two path images share a leaf
while there is a leaf $v \in T$ and a $S_1 \in \mathcal{F}$ s.t $v \in \ell(S_1)$ **do**
  Remove S_1 from $\mathcal{F}$
  Retain path labeling of sets in $\mathcal{F}$ other than S_1
  Let X be the set of elements present only in S_1
  if X is empty **then**
    exit
  end if
  $x \leftarrow$ arbitrary element from X
  Add $\{x\}$ and $S_1 \setminus \{x\}$ to $\mathcal{F}$
  $\ell(\{x\}) \leftarrow \{v\}$
  $\ell(S_1 \setminus \{x\}) \leftarrow \ell(S_1) \setminus \{v\}$
end while
return $\mathcal{F}, \ell$

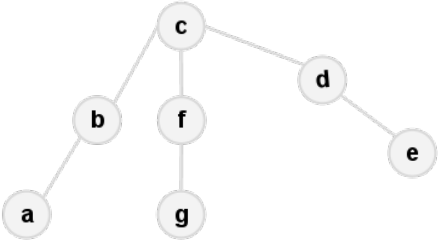
Feasible Tree Path Labeling

Algorithm 3 `get-feasible-permutation`$(\mathcal{F}, \ell, T)$

Call `filter_1` to pre-process $(\mathcal{F}, \ell)$
Call `filter_2` to obtain pre-images for leaves of T
for every leaf $v \in T$ **do**
  $\phi(x) \leftarrow v$ where $x \in \ell^{-1}(\{v\})$
end for
$L \leftarrow \{v \mid v \text{ is a leaf in } T\}$
For each $v \in L$, remove $\{v\}$ and $l(\{v\}) = \{x\}$ from $\mathcal{F}$ and ℓ
$T \leftarrow T \setminus L$
if T is not empty **then**
  $\phi \leftarrow \phi \cup$ `get-feasible-permutation`$(\mathcal{F}, \ell, T)$
end if
return ϕ

An Example: Input Instance

Tree T



F = { {3,4,6}, { 2,3,4,5}, {3,4,6,7}, {1,2,3}, {3,4,5,6,7} }
P = { {d,c,f}, {e,d,c,b}, { d,c,f,g}, {c,b,a}, {e,d,c,f,g} }

Figure: Set System and Corresponding Paths (labeling is in the given order)

An Example: Output of Filtering Algorithms

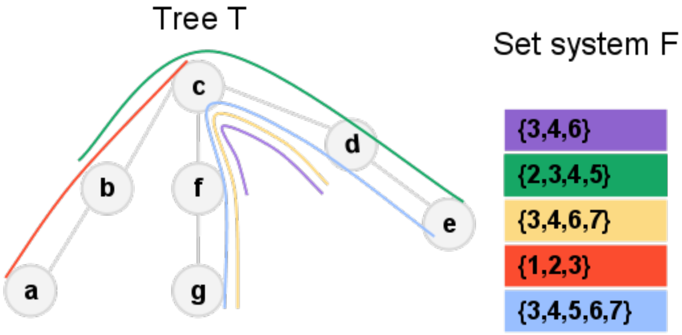


Figure: Paths *gfcd*, *gfcde* share leaf *g*

An Example: Output of Filtering Algorithms

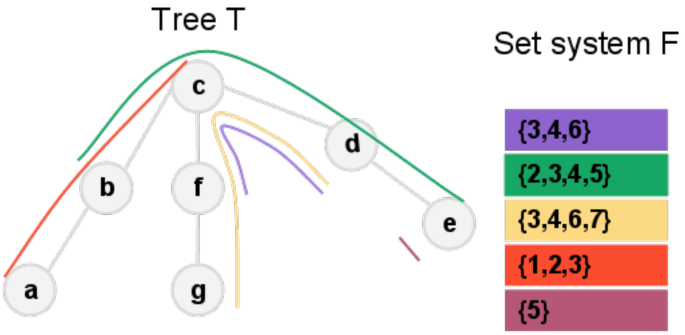


Figure: New set $\{5\}$ created. *bcde*, *e* share leaf *e*.

An Example: Output of Filtering Algorithms

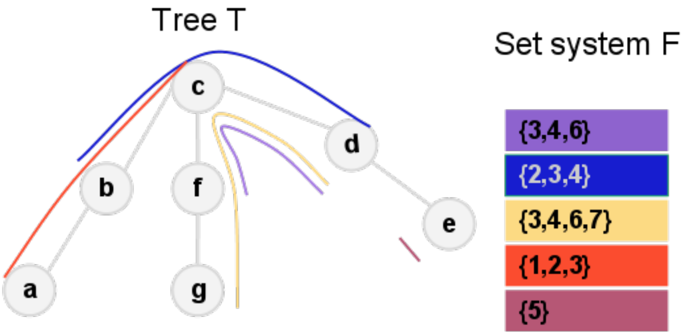


Figure: New set $\{2, 3, 4\}$ created. No more leaves shared.

An Example: Output of Filtering Algorithms

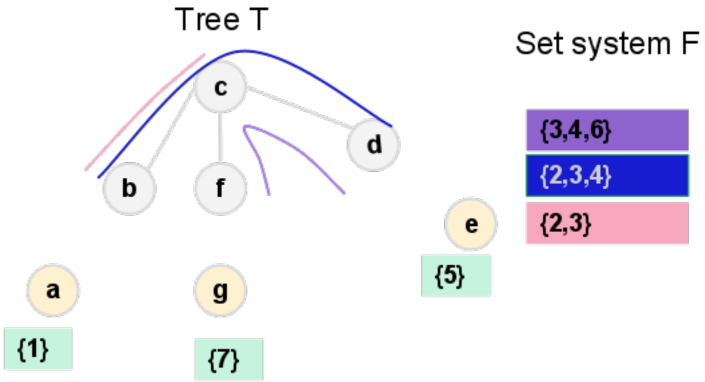


Figure: Leaf assignment done and leaves pruned. Paths *bcd*, *fcd* share leaf *d*.

An Example: Output of Filtering Algorithms

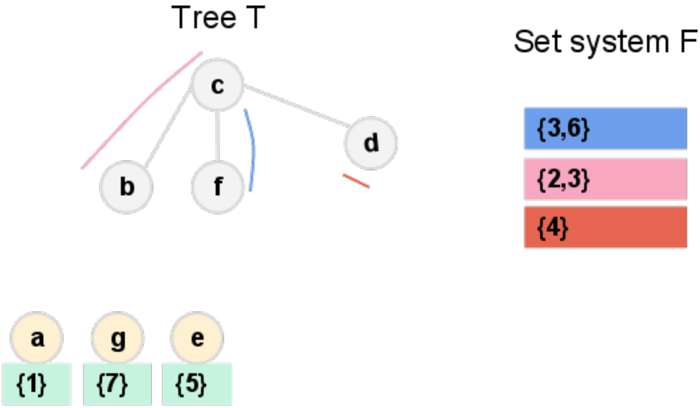


Figure: New set $\{4\}$ created. No more leaves shared.

An Example: Output of Filtering Algorithms

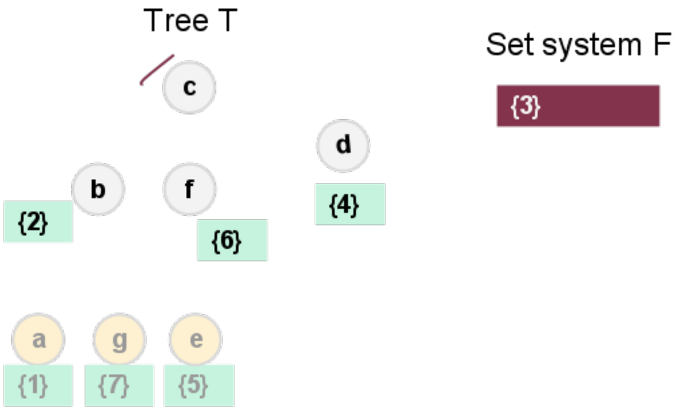


Figure: Leaf assignment done and leaves pruned.

An Example: Output of Filtering Algorithms

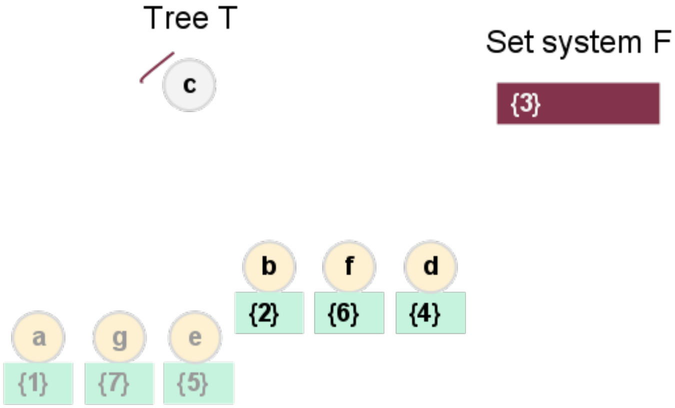


Figure: Only single node left in T.

An Example: Output of Filtering Algorithms

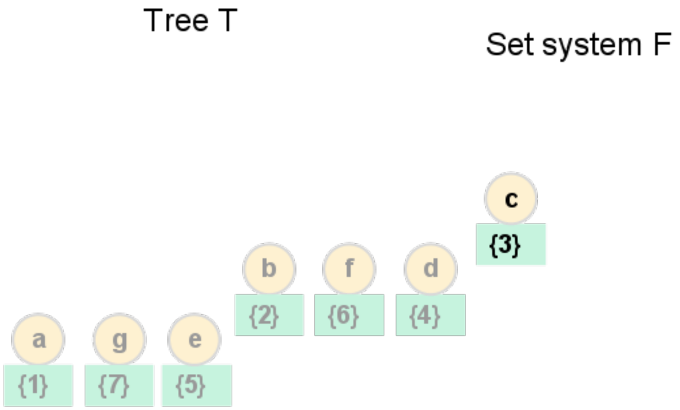


Figure: Labeling is obvious.

An Example: Output of Filtering Algorithms

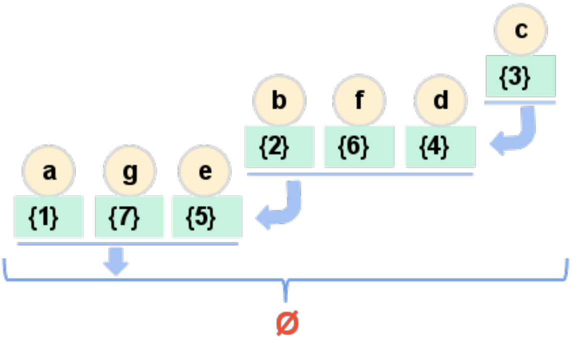


Figure: Recursively computed bijection ϕ

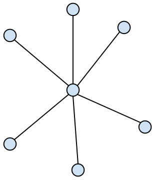
Outline

TPL from a k-subdivided star

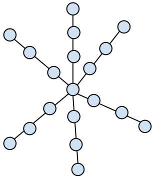
Computing a feasible TPL

Given hypergraph $\mathcal{F}$ with certain properties and a k-subdivided star T, can we find a feasible TPL ℓ to T?

A k-subdivided star is a star with all its edges subdivided exactly k times.



$k = 0$



$k = 2$

TPL from a k-subdivided star
Special case: Interval assignment(COP) problem

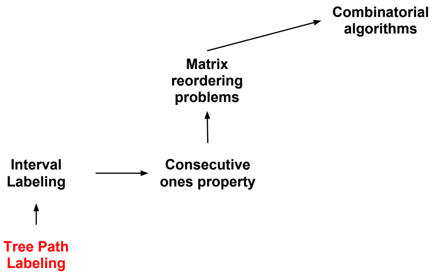
1. T is a path $\implies$ paths in T are intervals
2. Only pairwise intersection cardinality needs to be preserved: ICPIA
3. Higher level intersection cardinalities preserved by **Helly Property**
4. *filter_1*,*filter_2* do not need the **exit** conditions.

This problem is equivalent to Consecutive Ones Property of binary matrices

Outline

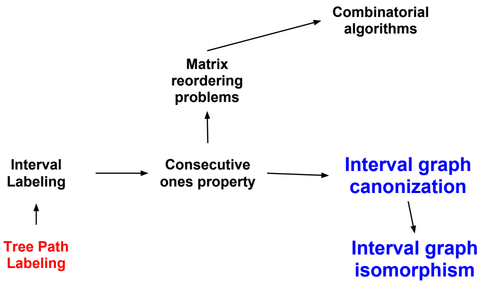
Path Labeling $\longrightarrow$ Graph Isomorphism

Application



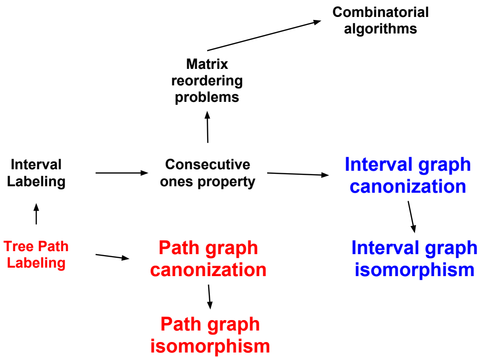
Path Labeling → Graph Isomorphism

Application



Path Labeling $\longrightarrow$ Graph Isomorphism

Application



References

1. Martin Charles Golumbic: **Algorithmic graph theory and perfect graphs**, volume 57 of *Annals of Discrete Mathematics*, Elsevier Science B.V., 2004.
2. N.S. Narayanaswamy and R.Subashini: **A new characterization of matrices with the consecutive ones property**, *Discrete Applied Mathematics*, 157(18):3721–3727, 2009.
3. Fanica Gavril: **A recognition algorithm for the intersection graphs of paths in trees**, *Discrete Mathematics*, 23(3):211 – 227, 1978.
4. Johannes Köbler, Sebastian Kuhnert, Bastian Laubner, and Oleg Verbitsky: Interval graphs: **Canonical representation in logspace**, *Electronic Colloquium on Computational Complexity (ECCC)*, 17:43, 2010.
5. Alejandro A. Schaffer: **A faster algorithm to recognize undirected path graphs**, *Discrete Applied Mathematics*, 43:261–295, 1993.

Thank You!

Questions?