Tree Path Labeling of Path Hypergraphs

Generalization of the Consecutive-ones Property

N. S. Narayanaswamy, Anju Srinivasan

Department of Computer Science and Engineering,
Indian Institute of Technology Madras

Feb 2015

An Illustration

- To introduce the combinatorial problem of TPL.

Study Group Accommodation problem

- A set of n **students** arrive for a summer course, say $\{a, b, c, d, e, f, g, h, i, j, k\}$, $n = 11$.

Study Group Accommodation problem

- A set of n **students** arrive for a summer course, say $\{a, b, c, d, e, f, g, h, i, j, k\}$, $n = 11$.
- They form m **study groups**, say $\{R, B, O, G\}$, $m = 4$

Study Group Accommodation problem

- A set of n **students** arrive for a summer course, say $\{a, b, c, d, e, f, g, h, i, j, k\}$, $n = 11$.
- They form m **study groups**, say $\{R, B, O, G\}$, $m = 4$
- A student may be in more than one study group but *will* be in at least one, say $R = \{g, h, i, j, k\}$, $B = \{a, b, e, g\}$, $O = \{c, b, d\}$, $G = \{e, f, g, i\}$

Study Group Accommodation problem

- A set of n **students** arrive for a summer course, say $\{a, b, c, d, e, f, g, h, i, j, k\}$, $n = 11$.
- They form m **study groups**, say $\{R, B, O, G\}$, $m = 4$
- A student may be in more than one study group but *will* be in at least one, say $R = \{g, h, i, j, k\}$, $B = \{a, b, e, g\}$, $O = \{c, b, d\}$, $G = \{e, f, g, i\}$
- There are n single occupancy **apartments** in the university campus for their accommodation.

Study Group Accommodation problem

- A set of n **students** arrive for a summer course, say $\{a, b, c, d, e, f, g, h, i, j, k\}$, $n = 11$.
- They form m **study groups**, say $\{R, B, O, G\}$, $m = 4$
- A student may be in more than one study group but *will* be in at least one, say $R = \{g, h, i, j, k\}$, $B = \{a, b, e, g\}$, $O = \{c, b, d\}$, $G = \{e, f, g, i\}$
- 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 - **streets form a tree**

Study Group Accommodation problem

The problem

How should the students be allocated apartments such that:

- students of each study group are neighbours?

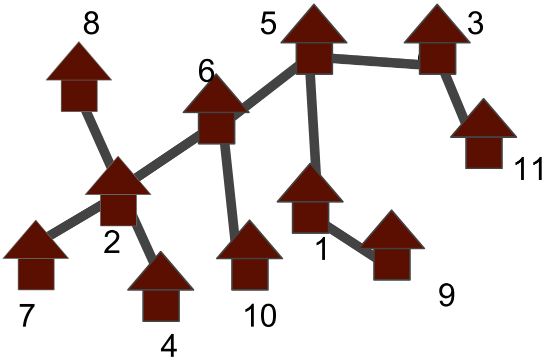
Study Group Accommodation problem

The problem

How should the students be allocated apartments such that:

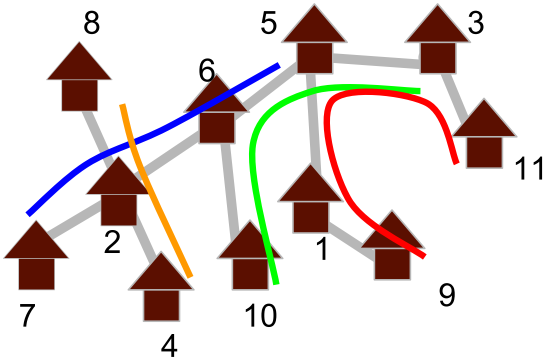
- students of each study group are neighbours?
- i.e. a study group forms **a path in the tree**.

Study Group Accommodation problem



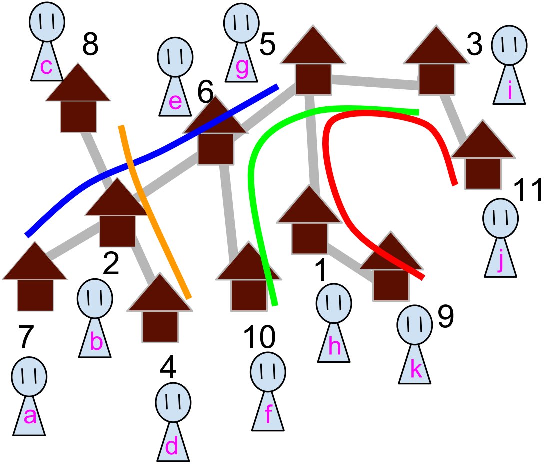
$R = \{g, h, i, j, k\}$
$B = \{a, b, e, g\}$
$O = \{c, b, d\}$
$G = \{e, f, g, i\}$

Study Group Accommodation problem



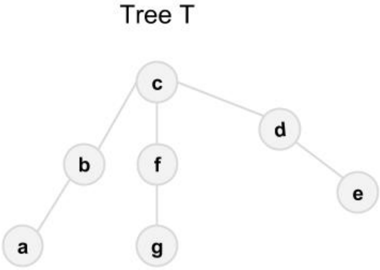
$R = \{g, h, i, j, k\}$
$\rightarrow \{9, 1, 5, 3, 11\}$
$B = \{a, b, e, g\}$
$\rightarrow \{7, 2, 6, 5\}$
$O = \{c, b, d\}$
$\rightarrow \{4, 2, 8\}$
$G = \{e, f, g, i\}$
$\rightarrow \{10, 6, 5, 3\}$

Study Group Accommodation problem



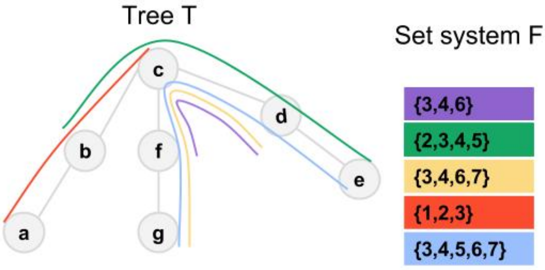
$R = \{g, h, i, j, k\}$
$\rightarrow \{9, 1, 5, 3, 11\}$
$B = \{a, b, e, g\}$
$\rightarrow \{7, 2, 6, 5\}$
$O = \{c, b, d\}$
$\rightarrow \{4, 2, 8\}$
$G = \{e, f, g, i\}$
$\rightarrow \{10, 6, 5, 3\}$

Path Labelling Example

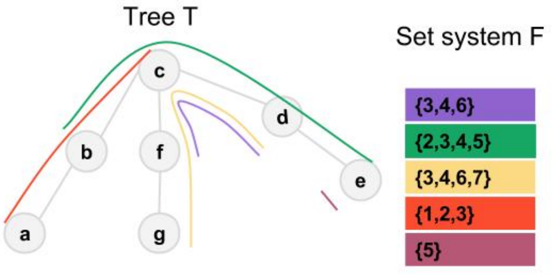


F = { {3,4,6}, { 2,3,4,5}, {3,4,6,7}, {1,2,3}, {3,4,5,6,7} }
P = { {d,c,f}, {e,d,c,b}, { d,c,f,g}, {c,b,a}, {e,d,c,f,g} }

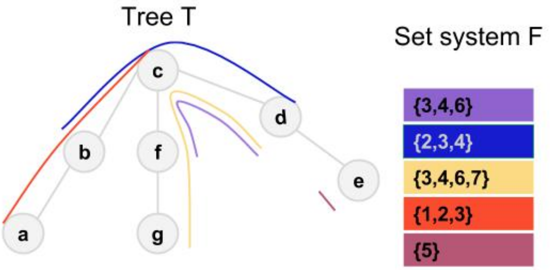
Path Labelling Example



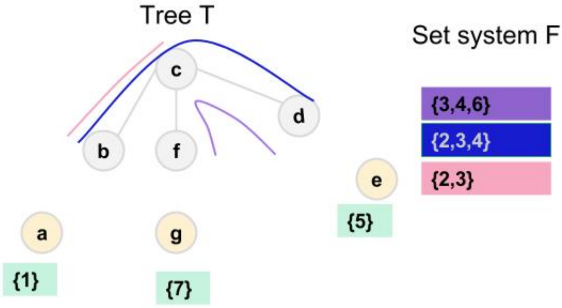
Path Labelling Example



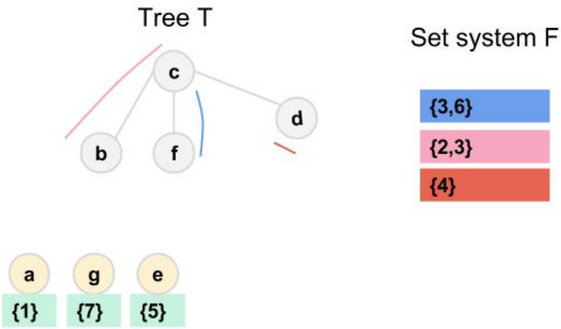
Path Labelling Example



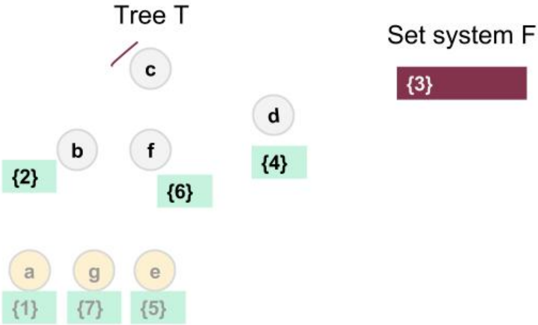
Path Labelling Example



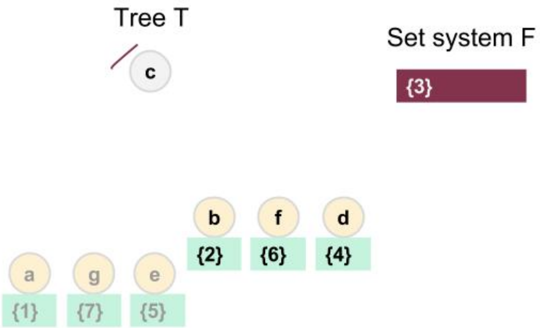
Path Labelling Example



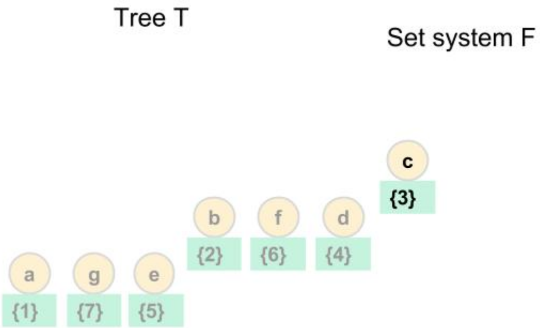
Path Labelling Example



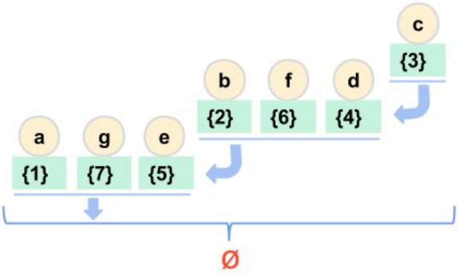
Path Labelling Example



Path Labelling Example



Path Labelling Example



Tree Path Labeling of Path Hypergraphs

- The set of study groups → **Set system / Hypergraph**

Tree Path Labeling of Path Hypergraphs

- The set of study groups → **Set system / Hypergraph**
- The streets with apartments → **Target tree**

Tree Path Labeling of Path Hypergraphs

- The set of study groups → **Set system / Hypergraph**
- The streets with apartments → **Target tree**
- The path mapping to study groups → **Tree Path Labeling (TPL)**

Tree Path Labeling of Path Hypergraphs

- The set of study groups → **Set system / Hypergraph**
- The streets with apartments → **Target tree**
- The path mapping to study groups → **Tree Path Labeling (TPL)**
- The apartment allocation → **Path Hypergraph Isomorphism**

Tree Path Labeling of Path Hypergraphs

There *exists* an apartment allocation that "fits" the path mapping

Tree Path Labeling of Path Hypergraphs

There *exists* a hypergraph isomorphism that "fits" the TPL

Tree Path Labeling of Path Hypergraphs

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

Tree Path Labeling of Path Hypergraphs

There *exists* a hypergraph isomorphism that "fits" the TPL
$\Rightarrow$ the TPL is FEASIBLE
There *exists* an apartment allocation that gives some study group path mapping

Tree Path Labeling of Path Hypergraphs

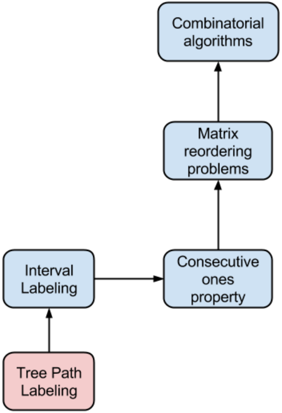
There *exists* a hypergraph isomorphism that "fits" the TPL
$\Rightarrow$ the TPL is FEASIBLE
There *exists* a hypergraph isomorphism that gives at least one feasible TPL

Tree Path Labeling of Path Hypergraphs

There *exists* a hypergraph isomorphism that "fits" the TPL
⇒ the TPL is FEASIBLE
There *exists* a hypergraph isomorphism that gives at least one feasible TPL
⇒ the hypergraph is a PATH HYPERGRAPH

Consecutive Ones → Path Labeling

The motivation



Feasible Tree Path Labeling

Input	A hypergraph $\mathcal{F}$ with vertex set U, a tree T, a set of paths $\mathcal{P}$ from T and a bijection $\ell : \mathcal{F} \to \mathcal{P}$.
Question	Does there exist a bijection $\phi : U \to V(T)$ such that ϕ when applied on any hyperedge in $\mathcal{F}$ will give the path mapped to it by the given tree path labeling ℓ. i.e., $\ell(S) = \{\phi(x) \mid x \in S\}$, for every hyperedge $S \in \mathcal{F}$.

Feasible Tree Path Labeling

- Is the given TPL ℓ of hypergraph $\mathcal{F}$ on tree T feasible?
- What is the hypergraph isomorphism $\phi : U \rightarrow V(T)$?
- Solvable in polynomial time.

Compute Feasible Tree Path Labeling

`Input`	A hypergraph $\mathcal{F}$ with vertex set U and a tree T.
`Question`	Does there exist a set of paths $\mathcal{P}$ from T and a bijection $\ell : \mathcal{F} \rightarrow \mathcal{P}$, such that Feasible Tree Path Labeling returns **true** on $(\mathcal{F}, T, \ell)$.

Compute Feasible Tree Path Labeling

- Is the given hypergraph $\mathcal{F}$ a path hypergraph w.r.t. target tree T?
- i.e. find at least one feasible tree path labeling $\ell : \mathcal{F} \to P$, P is a set of paths on T.
- Complexity is inconclusive for arbitrary trees, polynomial time for certain classes of trees.

Compute Interval Labeling

Compute Feasible Tree Path Labeling when target tree is an interval or path P_n

Compute Interval Labeling

- Is the given hypergraph $\mathcal{F}$ an interval hypergraph?
- Equivalent to consecutive ones property checking
- Finding an interval labelling
- Solvable in polynomial time.

Feasible Tree Path Labeling

Input	A hypergraph $\mathcal{F}$ with vertex set U, a tree T, a set of paths $\mathcal{P}$ from T and a bijection $\ell : \mathcal{F} \rightarrow \mathcal{P}$.
Question	Does there exist a bijection $\phi : U \rightarrow V(T)$ such that ϕ when applied on any hyperedge in $\mathcal{F}$ will give the path mapped to it by the given tree path labeling ℓ. i.e., $\ell(S) = \{\phi(x) \mid x \in S\}$, for every hyperedge $S \in \mathcal{F}$.

A characterization of feasible TPL

Intersection Cardinality Preserving Path Labeling (ICPPL)

A path labeling $(\mathcal{F}, \ell)$ on the given tree T s.t.

$$|S_1 \cap S_2 \cap S_3| = |\ell(S_1) \cap \ell(S_2) \cap \ell(S_3)|$$

for all not necessarily distinct $S_1, S_2, S_3 \in \mathcal{F}$

If T is a path, ICPPL is referred to as an Intersection Cardinality Preserving Interval Assignment (ICPIA)

A characterization of feasible TPL

Intersection Cardinality Preserving Path Labeling (ICPPL)

A path labeling $(\mathcal{F}, \ell)$ on the given tree T s.t.

$$|S_1 \cap S_2 \cap S_3| = |\ell(S_1) \cap \ell(S_2) \cap \ell(S_3)|$$

for all not necessarily distinct $S_1, S_2, S_3 \in \mathcal{F}$

If T is a path, ICPPL is referred to as an Intersection Cardinality Preserving Interval Assignment (ICPIA)

Theorem

A path labeling $(\mathcal{F}, \ell)$ on tree T is feasible iff it is an ICPPL.

Finding path hypergraph isomorphism ϕ

Given an ICPPL $(\mathcal{F}, \ell)$ on tree T

- Uses two filters to refine $(\mathcal{F}, \ell)$

Finding path hypergraph isomorphism ϕ

Given an ICPPL $(\mathcal{F}, \ell)$ on tree T

- Uses two filters to refine $(\mathcal{F}, \ell)$
- `filter common leaf` - ensures that the resulting ICPPL has no two path labels sharing a leaf.

Finding path hypergraph isomorphism ϕ

Given an ICPPL $(\mathcal{F}, \ell)$ on tree T

- Uses two filters to refine $(\mathcal{F}, \ell)$
- `filter common leaf` - ensures that the resulting ICPPL has no two path labels sharing a leaf.
- `filter fix leaf` - finds the pre-image of each leaf in T.

Finding path hypergraph isomorphism ϕ

Given an ICPPL $(\mathcal{F}, \ell)$ on tree T

- Uses two filters to refine $(\mathcal{F}, \ell)$
- `filter common leaf` - ensures that the resulting ICPPL has no two path labels sharing a leaf.
- `filter fix leaf` - finds the pre-image of each leaf in T.
- Remove leaves from T and their preimages from $\mathcal{F}$. Repeat filters until T becomes a path.

Finding path hypergraph isomorphism ϕ

Given an ICPPL $(\mathcal{F}, \ell)$ on tree T

- Uses two filters to refine $(\mathcal{F}, \ell)$
- `filter common leaf` - ensures that the resulting ICPPL has no two path labels sharing a leaf.
- `filter fix leaf` - finds the pre-image of each leaf in T.
- Remove leaves from T and their preimages from $\mathcal{F}$. Repeat filters until T becomes a path.
- When T is a path, problem becomes interval assignment. Use ICPIA

Finding path hypergraph isomorphism ϕ

`filter common leaf` $(\mathcal{F}, \ell)$

- Pick any two paths P_1, P_2 in $(\mathcal{F}, \ell)$ that share a leaf.

Finding path hypergraph isomorphism ϕ

`filter common leaf` $(\mathcal{F}, \ell)$

- Pick any two paths P_1, P_2 in $(\mathcal{F}, \ell)$ that share a leaf.
- Remove S_1, S_2, P_1, P_2 from $(\mathcal{F}, \ell)$

Finding path hypergraph isomorphism ϕ

filter common leaf $(\mathcal{F}, \ell)$

- Pick any two paths P_1, P_2 in $(\mathcal{F}, \ell)$ that share a leaf.
- Remove S_1, S_2, P_1, P_2 from $(\mathcal{F}, \ell)$
- Add to $(\mathcal{F}, \ell)$:
  $\ell(S_1 \setminus S_2) = P_1 \setminus P_2$
  $\ell(S_2 \setminus S_1) = P_2 \setminus P_1$
  $\ell(S_1 \cap S_2) = P_1 \cap P_2$

Finding path hypergraph isomorphism ϕ

`filter common leaf` $(\mathcal{F}, \ell)$

- Pick any two paths P_1, P_2 in $(\mathcal{F}, \ell)$ that share a leaf.
- Remove S_1, S_2, P_1, P_2 from $(\mathcal{F}, \ell)$
- Add to $(\mathcal{F}, \ell)$:
  $\ell(S_1 \setminus S_2) = P_1 \setminus P_2$
  $\ell(S_2 \setminus S_1) = P_2 \setminus P_1$
  $\ell(S_1 \cap S_2) = P_1 \cap P_2$
- Repeat till no two paths share a leaf.

Finding path hypergraph isomorphism ϕ

`filter common leaf` $(\mathcal{F}, \ell)$

Lemma

Let $(\mathcal{F}', \ell')$ be the resulting labeling after applying `filter common leaf` *to TPL $(\mathcal{F}, \ell)$. If $(\mathcal{F}, \ell)$ is an ICPPL, $(\mathcal{F}', \ell')$ is also an ICPPL.*

Finding path hypergraph isomorphism ϕ

`filter common leaf` $(\mathcal{F}, \ell)$

Lemma

Let $(\mathcal{F}', \ell')$ be the resulting labeling after applying `filter common leaf` *to TPL $(\mathcal{F}, \ell)$. If $(\mathcal{F}, \ell)$ is an ICPPL, $(\mathcal{F}', \ell')$ is also an ICPPL.*

Proof.

- Induction on iteration of the filter.
- Invariants: $\ell_j(S)$ is a path, ℓ_j maintains ICPPL's intersection cardinality equations.
- ICPPL also preserves 4-way intersection cardinalities.

□

Finding path hypergraph isomorphism ϕ

`filter fix leaf` $(\mathcal{F}, \ell)$

- A leaf is unique to a path

Finding path hypergraph isomorphism ϕ

`filter fix leaf` $(\mathcal{F}, \ell)$

- A leaf is unique to a path
- Pick a leaf v, let it be on path P. Let $\ell(S) = P$

Finding path hypergraph isomorphism ϕ

filter fix leaf $(\mathcal{F}, \ell)$

- A leaf is unique to a path
- Pick a leaf v, let it be on path P. Let $\ell(S) = P$
- Pick an element x from S which is not present in any other set. i.e. $x \in S \setminus \cup_{S_i \neq S} S_i$

Finding path hypergraph isomorphism ϕ

`filter fix leaf` $(\mathcal{F}, \ell)$

- A leaf is unique to a path
- Pick a leaf v, let it be on path P. Let $\ell(S) = P$
- Pick an element x from S which is not present in any other set. i.e. $x \in S \setminus \cup_{S_i \neq S} S_i$
- Remove S, P from $(\mathcal{F}, \ell)$

Finding path hypergraph isomorphism ϕ

`filter fix leaf` $(\mathcal{F}, \ell)$

- A leaf is unique to a path
- Pick a leaf v, let it be on path P. Let $\ell(S) = P$
- Pick an element x from S which is not present in any other set. i.e. $x \in S \setminus \cup_{S_i \neq S} S_i$
- Remove S, P from $(\mathcal{F}, \ell)$
- Add $\ell(S \setminus x) = P \setminus v$. Define $\phi(x) = v$

Finding path hypergraph isomorphism ϕ

`filter fix leaf` $(\mathcal{F}, \ell)$

- A leaf is unique to a path
- Pick a leaf v, let it be on path P. Let $\ell(S) = P$
- Pick an element x from S which is not present in any other set. i.e. $x \in S \setminus \cup_{S_i \neq S} S_i$
- Remove S, P from $(\mathcal{F}, \ell)$
- Add $\ell(S \setminus x) = P \setminus v$. Define $\phi(x) = v$
- Remove leaf v from T

Finding path hypergraph isomorphism ϕ

`filter fix leaf` $(\mathcal{F}, \ell)$

- A leaf is unique to a path
- Pick a leaf v, let it be on path P. Let $\ell(S) = P$
- Pick an element x from S which is not present in any other set. i.e. $x \in S \setminus \cup_{S_i \neq S} S_i$
- Remove S, P from $(\mathcal{F}, \ell)$
- Add $\ell(S \setminus x) = P \setminus v$. Define $\phi(x) = v$
- Remove leaf v from T
- Repeat till there are no more unique paths for leaves. Call `filter common leaf` .

Finding path hypergraph isomorphism ϕ

`filter fix leaf` $(\mathcal{F}, \ell)$

- A leaf is unique to a path
- Pick a leaf v, let it be on path P. Let $\ell(S) = P$
- Pick an element x from S which is not present in any other set. i.e. $x \in S \setminus \cup_{S_i \neq S} S_i$
- Remove S, P from $(\mathcal{F}, \ell)$
- Add $\ell(S \setminus x) = P \setminus v$. Define $\phi(x) = v$
- Remove leaf v from T
- Repeat till there are no more unique paths for leaves. Call `filter common leaf` .
- End if T is empty

Finding path hypergraph isomorphism ϕ

`filter fix leaf` $(\mathcal{F}, \ell)$

Critical part is finding $x \in S \setminus \cup_{S_i \neq S} S_i$

Finding path hypergraph isomorphism ϕ

`filter fix leaf` $(\mathcal{F}, \ell)$

Critical part is finding $x \in S \setminus \cup_{S_i \neq S} S_i$

Lemma

If $\ell(S)$ uniquely has a leaf, S_{priv} is non-empty where $S_{priv} = S \setminus \cup_{S_i \neq S} S_i$.

Finding path hypergraph isomorphism ϕ

filter fix leaf $(\mathcal{F}, \ell)$

Critical part is finding $x \in S \setminus \cup_{S_i \neq S} S_i$

Lemma

If $\ell(S)$ uniquely has a leaf, S_{priv} is non-empty where $S_{priv} = S \setminus \cup_{S_i \neq S} S_i$.

Proof.

- Let $\mathcal{F}' = S \cap S_i$ and $\ell'(S \cap S_i) = P \cap P_i$ for all $S_i \in \mathcal{F}$, $\ell(S_i) = P_i$.
- $S_{two} = supp(\mathcal{F}'), P_{two} = supp(\ell')$
- $(\mathcal{F}', \ell')$ is an ICPIA. Therefore $|S_{two}| = |P_{two}|$. Hence $|S_{priv}| = |P_{priv}|$. We know P has at least a leaf.

Compute Feasible Tree Path Labeling

`Input`	A hypergraph $\mathcal{F}$ with vertex set U and a tree T.
`Question`	Does there exist a set of paths $\mathcal{P}$ from T and a bijection $\ell : \mathcal{F} \rightarrow \mathcal{P}$, such that Feasible Tree Path Labeling returns **true** on $(\mathcal{F}, T, \ell)$.

Special case

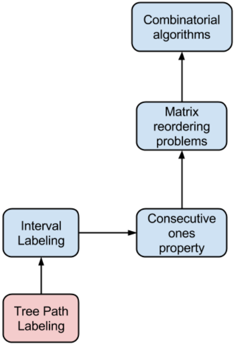
Interval assignment problem / COP

1. T is a path $\Longrightarrow$ paths in T are intervals
2. Only pairwise intersection cardinality needs to be preserved $\Longrightarrow$ ICPIA
3. Higher level intersection cardinalities preserved by **Helly Property**

This problem is equivalent to Consecutive Ones Property of binary matrices

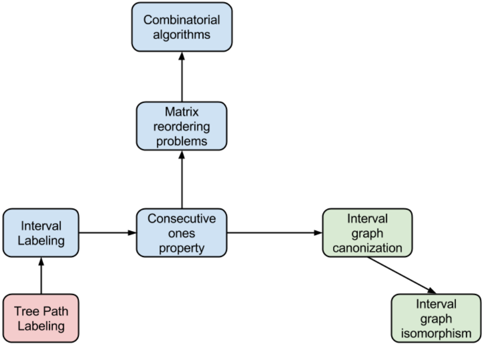
Path Labeling → Graph Isomorphism

Application



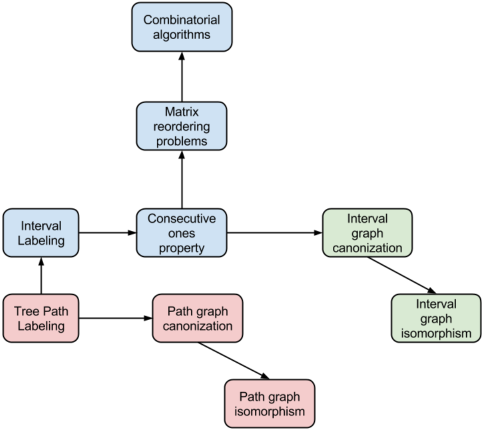
Path Labeling → Graph Isomorphism

Application



Path Labeling → Graph Isomorphism

Application



Thank you

Questions?