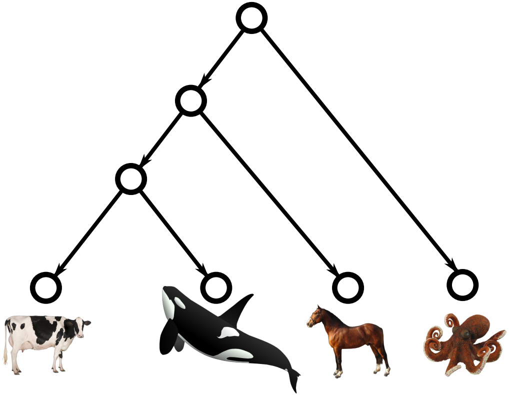
LCA or No LCA:
A short story about simplifying networks

Anna Linderberg and Marc Hellmuth

Department of Mathematics
Faculty of Science
Stockholm University

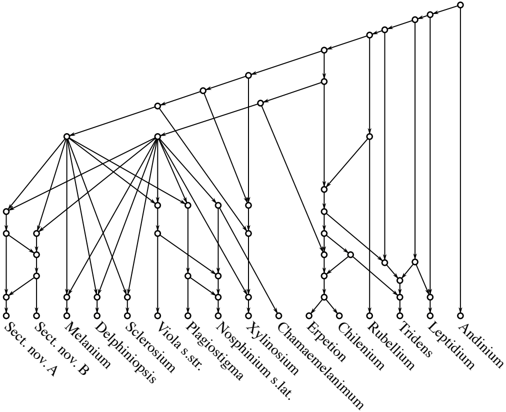
40th TBI Winterseminar in Bled, 2025

Evolutionary Histories . . .



Classical Tree-like Relationships

Evolutionary Histories . . .

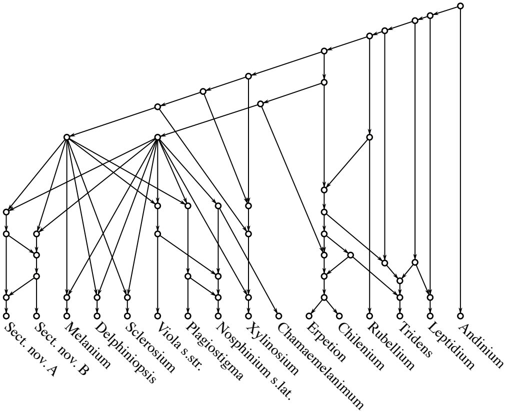


Network of some *Viola* plants*

Well in these inconspicuous plants . . .

*From Gene Trees to a Dated Allopolyploid Network: Insights from the Angiosperm Genus Viola (Violaceae), Marcussen et al., Syst. Biol., 2015

Evolutionary Histories . . .

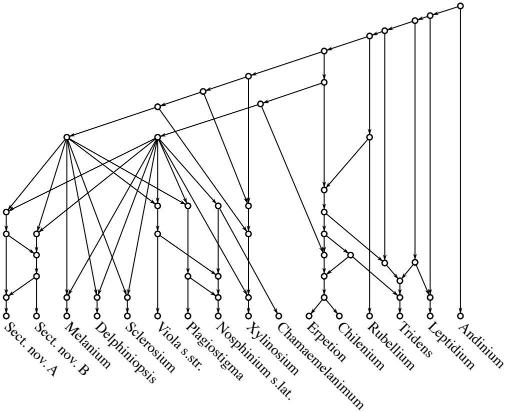


Network of some *Viola* plants*

Well in these inconspicuous plants . . .
. . . there is a lot happening in their sexual reproduction

*From Gene Trees to a Dated Allopolyploid Network: Insights from the Angiosperm Genus Viola (Violaceae), Marcussen et al., Syst. Biol., 2015

Evolutionary Histories . . .



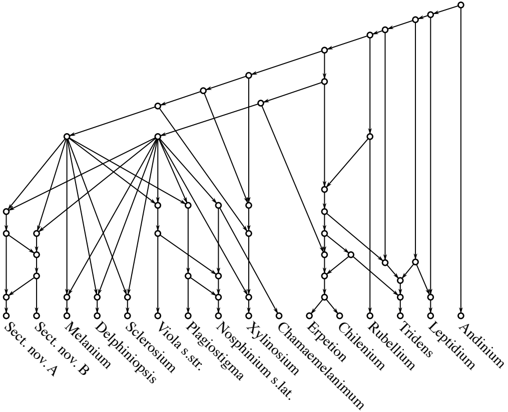
Network of some *Viola* plants*

Well in these inconspicuous plants . . .
. . . there is a lot happening in their sexual reproduction

Hybridization
(when plants decide monogamy is overrated)

* From Gene Trees to a Dated Allopolyploid Network: Insights from the Angiosperm Genus Viola (Violaceae), Marcussen et al., Syst. Biol., 2015

Evolutionary Histories . . .



Network of some *Viola* plants*

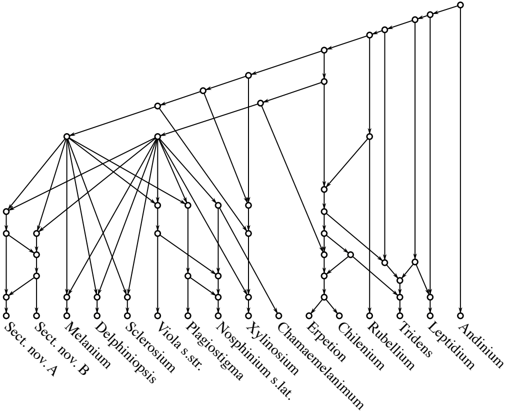
Well in these inconspicuous plants . . .
. . . there is a lot happening in their sexual reproduction

Hybridization
(when plants decide monogamy is overrated)

HGT
(here it gets wild and boundaries are just suggestions)

* From Gene Trees to a Dated Allopolyploid Network: Insights from the Angiosperm Genus Viola (Violaceae), Marcussen et al., Syst. Biol., 2015

Evolutionary Histories . . .



Network of some *Viola* plants*

Well in these inconspicuous plants . . .
. . . there is a lot happening in their sexual reproduction

Hybridization
(when plants decide monogamy is overrated)

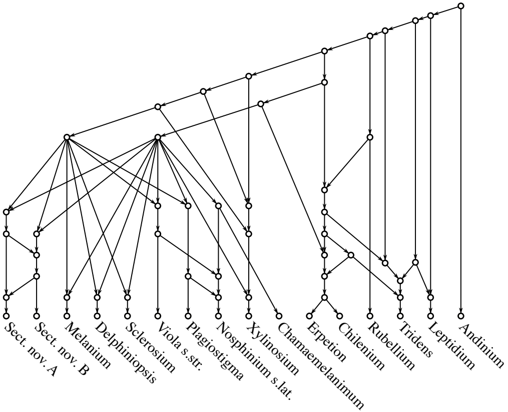
HGT
(here it gets wild and boundaries are just suggestions)

and many more
(now the weired sexual preferences of these cute plants go beyond my limits of imagination)

* From Gene Trees to a Dated Allopolyploid Network: Insights from the Angiosperm Genus Viola (Violaceae), Marcussen et al., Syst. Biol., 2015

Evolutionary Histories . . .

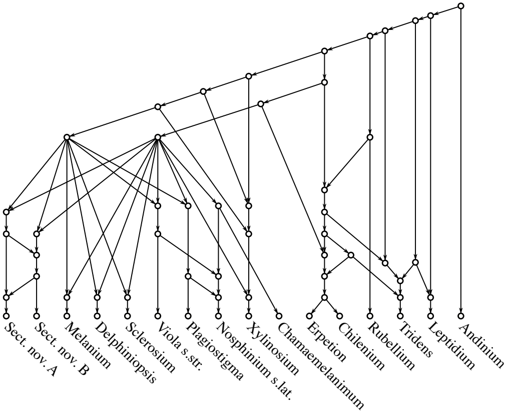
Mechanisms like HGT or hybridiations cannot be fully captured by trees
$\Longrightarrow$ evolutionary history is network-like



Network of some *Viola* plants*

*From Gene Trees to a Dated Allopolyploid Network: Insights from the Angiosperm Genus Viola (Violaceae), Marcussen et al., Syst. Biol., 2015

Evolutionary Histories . . .



Network of some *Viola* plants*

Mechanisms like HGT or hybridiations cannot be fully captured by trees
$\Longrightarrow$ evolutionary history is network-like

How to obtain a phylogenetic network?

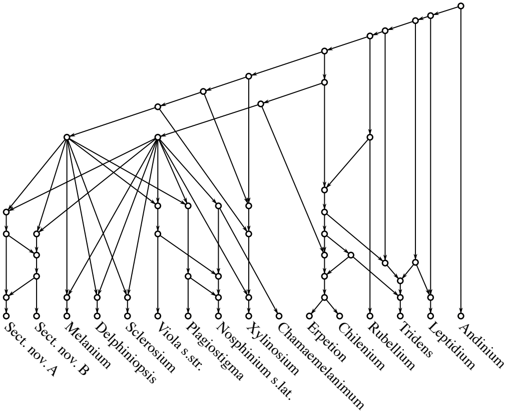
Start with observable data
= genomic sequences of extant taxa
= set $L(N)$ of leaves in network N we want be infer

Based on observable data infer the network $N^{\ddagger}$

* From Gene Trees to a Dated Allopolyploid Network: Insights from the Angiosperm Genus Viola (Violaceae), Marcussen et al., Syst. Biol., 2015

‡ A Survey of Combinatorial Methods for Phylogenetic Networks, Huson and Scornavacca, GBE, 2010

Evolutionary Histories . . .



Network of some *Viola* plants*

Mechanisms like HGT or hybridiations cannot be fully captured by trees
$\Longrightarrow$ evolutionary history is network-like

How to obtain a phylogenetic network?

Start with observable data
= genomic sequences of extant taxa
= set $L(N)$ of leaves in network N we want be infer

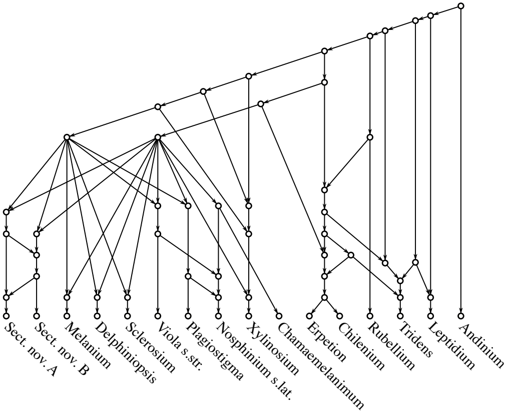
Based on observable data infer the network N‡

Networks inferred from genomic data can . . .
. . . be highly complex and tangled
. . . contain redundant information
. . . contain information that is not supported by “observable data”

* From Gene Trees to a Dated Allopolyploid Network: Insights from the Angiosperm Genus Viola (Violaceae), Marcussen et al., Syst. Biol., 2015

‡ A Survey of Combinatorial Methods for Phylogenetic Networks, Huson and Scornavacca, GBE, 2010

Evolutionary Histories . . .



Network of some *Viola* plants*

Mechanisms like HGT or hybridiations cannot be fully captured by trees
$\Longrightarrow$ evolutionary history is network-like

How to obtain a phylogenetic network?

Start with observable data
= genomic sequences of extant taxa
= set $L(N)$ of leaves in network N we want be infer

Based on observable data infer the network $N^{\ddagger}$

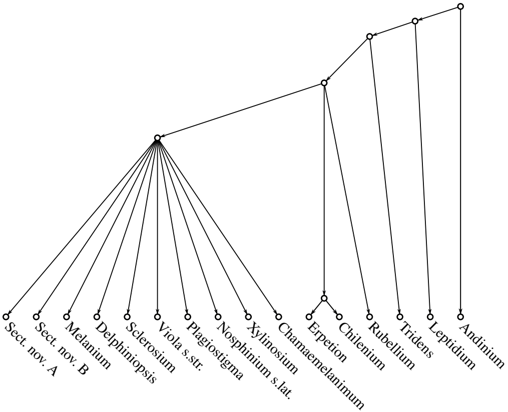
Networks inferred from genomic data can . . .
. . . be highly complex and tangled
. . . contain redundant information
. . . contain information that is not supported by "observable data"

$\Longrightarrow$ Find methods to simplify networks
while keeping the main structural features

* From Gene Trees to a Dated Allopolyploid Network: Insights from the Angiosperm Genus Viola (Violaceae), Marcussen et al., Syst. Biol., 2015

‡ A Survey of Combinatorial Methods for Phylogenetic Networks, Huson and Scornavacca, GBE, 2010

Evolutionary Histories ...



Lowest Stable Ancestor (LSA) tree†
to show "trend of evolution"

Mechanisms like HGT or hybridiations cannot be fully captured by trees
$\Longrightarrow$ evolutionary history is network-like

How to obtain a phylogenetic network?

Start with observable data
= genomic sequences of extant taxa
= set $L(N)$ of leaves in network N we want be infer

Based on observable data infer the network $N^{\ddagger}$

Networks inferred from genomic data can ...
... be highly complex and tangled
... contain redundant information
... contain information that is not supported by "observable data"

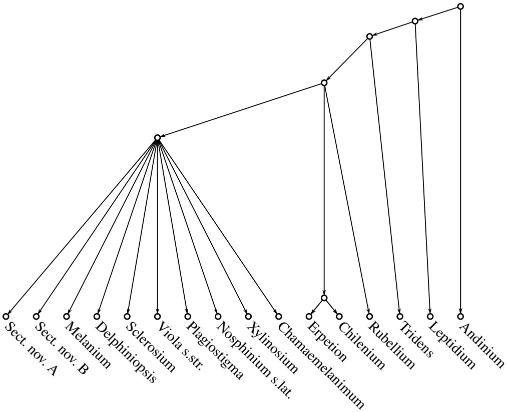
$\Longrightarrow$ Find methods to simplify networks
while keeping the main structural features

* From Gene Trees to a Dated Allopolyploid Network: Insights from the Angiosperm Genus Viola (Violaceae), Marcussen et al., Syst. Biol., 2015

† Transformations to Simplify Phylogenetic Networks, Heiss, Huson and Steel, Bulletin of Math. Bio., 2025

‡ A Survey of Combinatorial Methods for Phylogenetic Networks, Huson and Scornavacca, GBE, 2010

Evolutionary Histories . . .



Lowest Stable Ancestor (LSA) tree†
to show "trend of evolution"

Mechanisms like HGT or hybridiations cannot be fully captured by trees
$\Longrightarrow$ evolutionary history is network-like

How to obtain a phylogenetic network?

Start with observable data
= genomic sequences of extant taxa
= set $L(N)$ of leaves in network N we want be infer

Based on observable data infer the network $N^{\ddagger}$

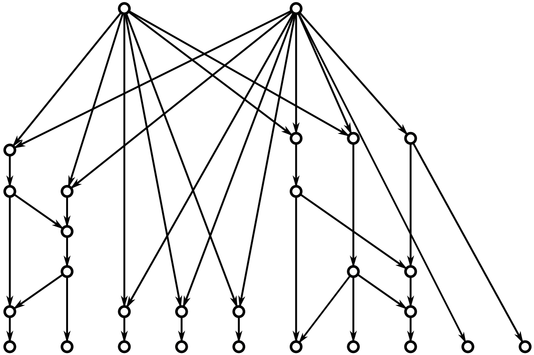
Networks inferred from genomic data can . . .
. . . be highly complex and tangled
. . . contain redundant information
. . . contain information that is not supported by "observable data"

$\Longrightarrow$ Find methods to simplify networks
while keeping the main structural features

The method we propose aims to simplify by eliminating vertices that are not supported by "observable data"

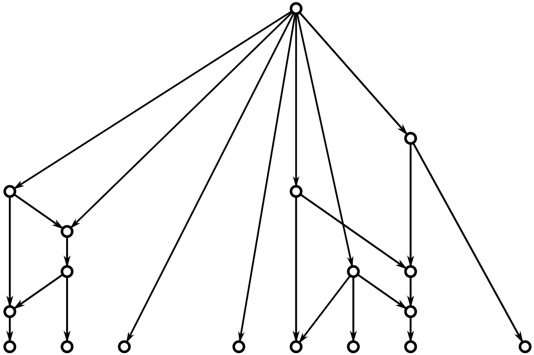
* From Gene Trees to a Dated Allopolyploid Network: Insights from the Angiosperm Genus Viola (Violaceae), Marcussen et al., Syst. Biol., 2015
† Transformations to Simplify Phylogenetic Networks, Heiss, Huson and Steel, Bulletin of Math. Bio., 2025
‡ A Survey of Combinatorial Methods for Phylogenetic Networks, Huson and Scornavacca, GBE, 2010

Some Basics (AKA unavoidable defs and some math)



DAG = Directed Acyclic Graph.

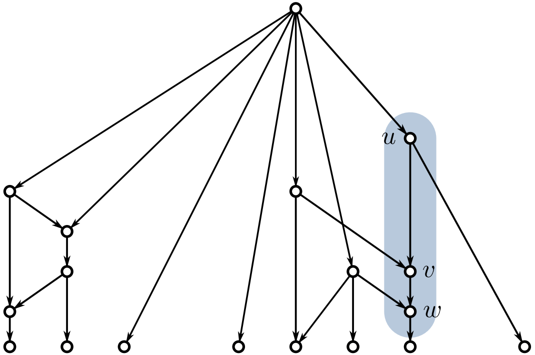
Some Basics (AKA unavoidable defs and some math)



DAG = Directed Acyclic Graph.

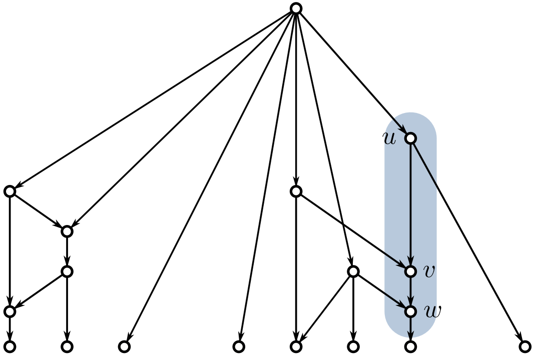
If a DAG has unique source (=root), it is a network

Some Basics (AKA unavoidable defs and some math)



u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

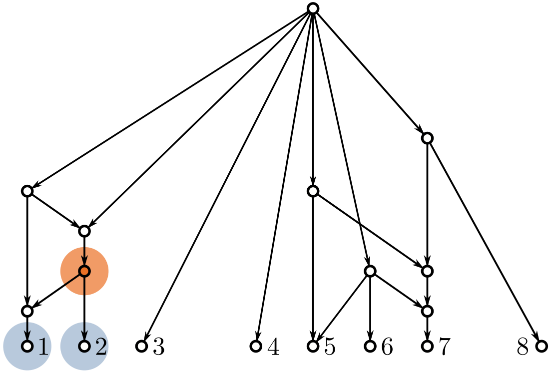
Some Basics (AKA unavoidable defs and some math)



u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set LCA(A) comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

Some Basics (AKA unavoidable defs and some math)

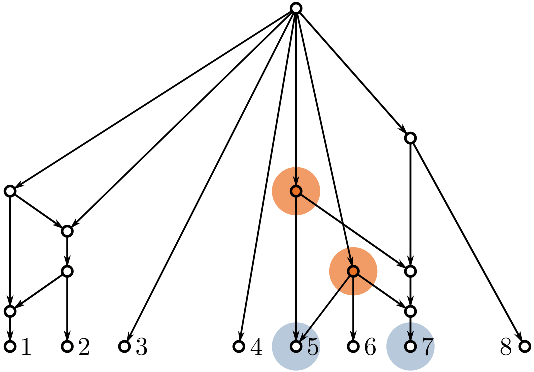


u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set LCA(A) comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

LCA$(\{1,2\})$

Some Basics (AKA unavoidable defs and some math)

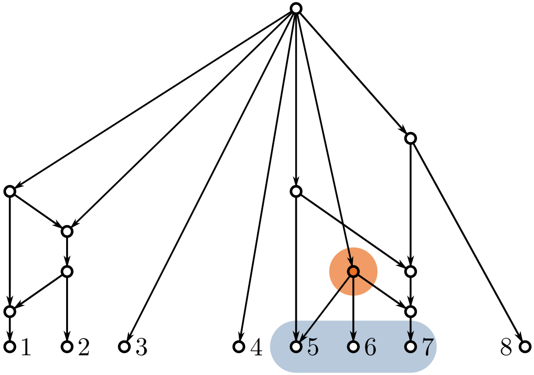


u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set $\mathrm{LCA}(A)$ comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

$$\mathrm{LCA}(\{5,7\})$$

Some Basics (AKA unavoidable defs and some math)

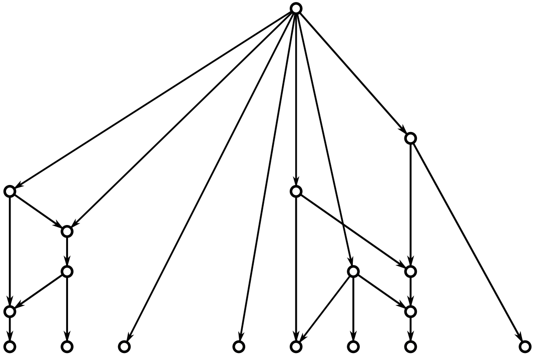


u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set $\mathrm{LCA}(A)$ comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

$$\mathrm{LCA}(\{5,6,7\})$$

Some Basics (AKA unavoidable defs and some math)



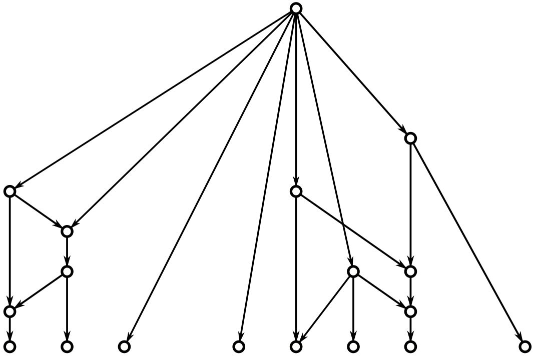
u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set $\mathrm{LCA}(A)$ comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

We want to simplify by eliminating vertices that are not supported by observable data

What does "supported" mean?

Some Basics (AKA unavoidable defs and some math)



u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

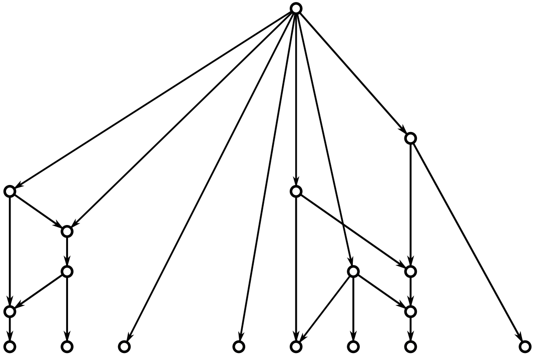
Least Common Ancestors:
for $A \subseteq L(G)$, the set $\mathrm{LCA}(A)$ comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

We want to simplify by eliminating vertices that are not supported by observable data

What does "supported" mean?

A vertex v is an LCA-vertex (= supported), if v is the LCA of some subset of leaves

Some Basics (AKA unavoidable defs and some math)



u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set $\mathrm{LCA}(A)$ comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

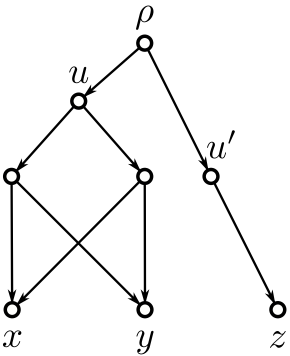
We want to simplify by eliminating vertices that are not supported by observable data

What does "supported" mean?

A vertex v is an LCA-vertex (= supported), if v is the LCA of some subset of leaves

A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices

Some Basics (AKA unavoidable defs and some math)



u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set $\mathrm{LCA}(A)$ comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

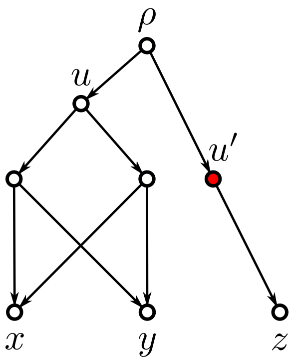
We want to simplify by eliminating vertices that are not supported by observable data

What does "supported" mean?

A vertex v is an LCA-vertex (= supported),
if v is the LCA of some subset of leaves

A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices

Some Basics (AKA unavoidable defs and some math)



u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set $\mathrm{LCA}(A)$ comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

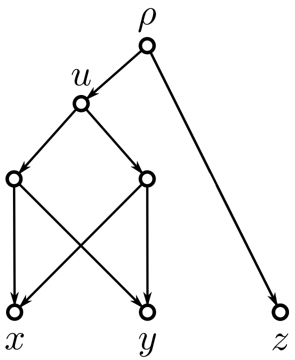
We want to simplify by eliminating vertices that are not supported by observable data

What does "supported" mean?

A vertex v is an LCA-vertex (= supported),
if v is the LCA of some subset of leaves

A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices

Some Basics (AKA unavoidable defs and some math)



u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set $\mathrm{LCA}(A)$ comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

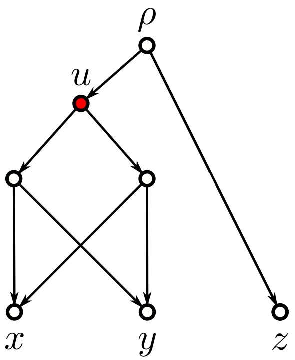
We want to simplify by eliminating vertices that are not supported by observable data

What does "supported" mean?

A vertex v is an LCA-vertex (= supported), if v is the LCA of some subset of leaves

A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices

Some Basics (AKA unavoidable defs and some math)



u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set LCA(A) comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

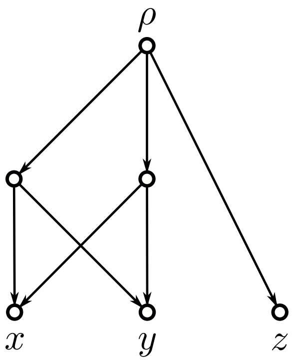
We want to simplify by eliminating vertices that are not supported by observable data

What does "supported" mean?

A vertex v is an LCA-vertex (= supported), if v is the LCA of some subset of leaves

A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices

Some Basics (AKA unavoidable defs and some math)



u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set LCA(A) comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

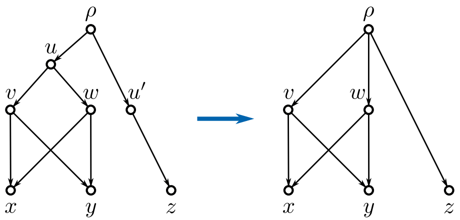
We want to simplify by eliminating vertices that are not supported by observable data

What does "supported" mean?

A vertex v is an LCA-vertex (= supported),
if v is the LCA of some subset of leaves

A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices

Some Basics (AKA unavoidable defs and some math)



u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set $\mathrm{LCA}(A)$ comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

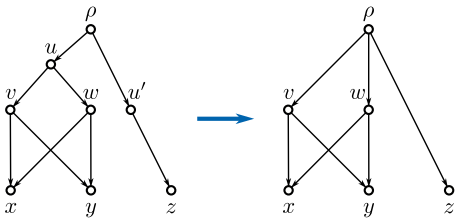
We want to simplify by eliminating vertices that are not supported by observable data

What does "supported" mean?

A vertex v is an LCA-vertex (= supported), if v is the LCA of some subset of leaves

A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices

Some Basics (AKA unavoidable defs and some math)



u is ancestor of w if there is a directed path $u \rightsquigarrow w$
Denoted $w \preceq u$.

Least Common Ancestors:
for $A \subseteq L(G)$, the set $\mathrm{LCA}(A)$ comprises all $\preceq$-minimal vertices that are ancestors of all leaves in A

We want to simplify by eliminating vertices that are not supported by observable data

What does "supported" mean?

A vertex v is an LCA-vertex (= supported), if v is the LCA of some subset of leaves

A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices

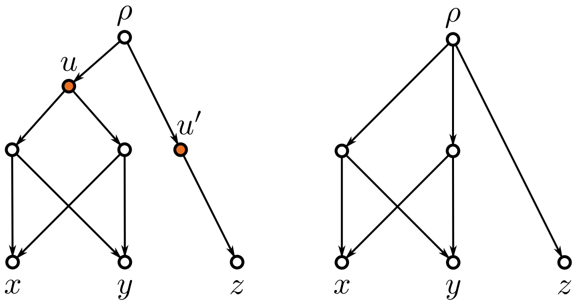
This begs several questions:

- How to determine unsupported vertices and how to get rid of them?
  Testing all $A \subseteq L(G)$ and check if $v \in \mathrm{LCA}(A)$ is not an option!
- Can we characterize LCA-RELEVANT DAGs?

Results: Characterization

Questions:

- How to determine unsupported vertices and how to get rid of them ?
- Can we characterize LCA-RELEVANT DAGs?
  [A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices]



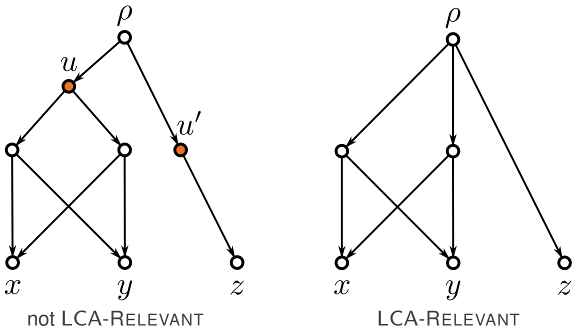
not LCA-RELEVANT LCA-RELEVANT

Results: Characterization

Questions:

- How to determine unsupported vertices and how to get rid of them ?
- Can we characterize LCA-RELEVANT DAGs?
  [A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices]

Lemma. *A vertex v is not an* LCA*-vertex* $\iff$ v *has a child* w *with* $\mathtt{C}(v) = \mathtt{C}(w)$.

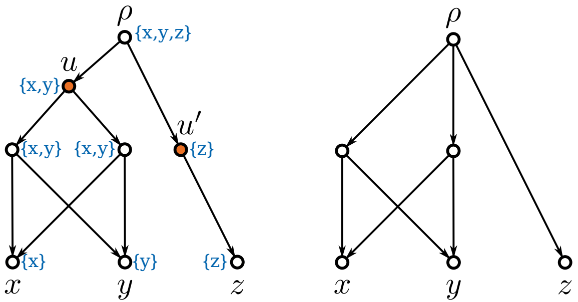


Results: Characterization

Questions:

- How to determine unsupported vertices and how to get rid of them ?
- Can we characterize LCA-RELEVANT DAGs?
  [A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices]

Lemma. *A vertex v is not an* LCA*-vertex* $\iff$ *v has a child w with $\mathtt{C}(v) = \mathtt{C}(w)$.*

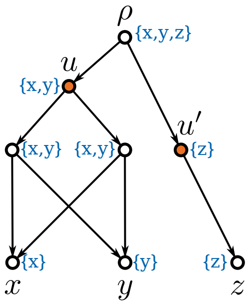


Results: Characterization

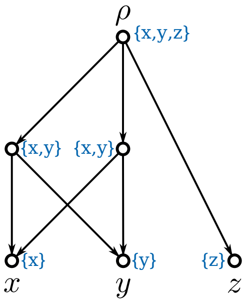
Questions:

- How to determine unsupported vertices and how to get rid of them ?
- Can we characterize LCA-RELEVANT DAGs?
  [A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices]

Lemma. *A vertex v is not an* LCA*-vertex* $\iff$ v *has a child* w *with* $\mathtt{C}(v) = \mathtt{C}(w)$.



not LCA-RELEVANT



LCA-RELEVANT

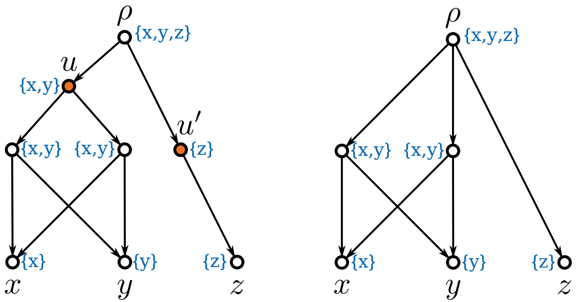
Results: Characterization

Questions:

- How to determine unsupported vertices and how to get rid of them ?
- Can we characterize LCA-RELEVANT DAGs?
  [A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices]

Lemma. *A vertex v is not an* LCA*-vertex* $\iff$ v *has a child* w *with* $\mathrm{C}(v) = \mathrm{C}(w)$.

*All non-*LCA*-vertices can be determined in polynomial-time.*



not LCA-RELEVANT LCA-RELEVANT

Results: Characterization

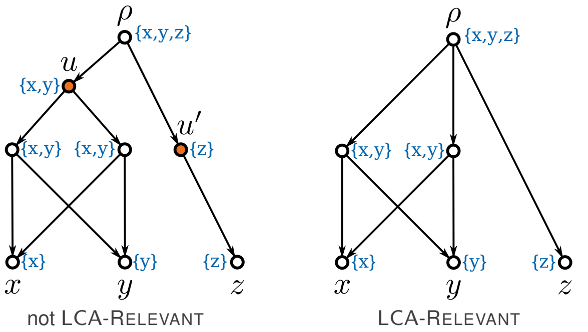
Questions:

- How to determine unsupported vertices and how to get rid of them ?
- Can we characterize LCA-RELEVANT DAGs?
  [A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices]

Lemma. *A vertex v is not an LCA-vertex $\iff$ v has a child w with $\mathrm{C}(v) = \mathrm{C}(w)$.*

All non-LCA-vertices can be determined in polynomial-time.

Theorem. *A DAG G is LCA-RELEVANT $\iff$ no two adjacent vertices of G have the same cluster.*



Results: Characterization

Questions:

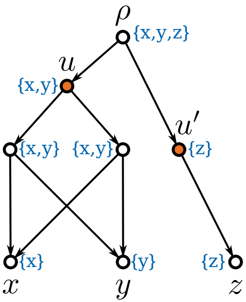
- How to determine unsupported vertices ✓ and how to get rid of them ?
- Can we characterize LCA-RELEVANT DAGs? ✓

  [A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices]

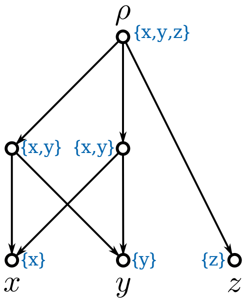
Lemma. *A vertex v is not an LCA-vertex $\iff$ v has a child w with $\mathtt{C}(v) = \mathtt{C}(w)$.*

All non-LCA-vertices can be determined in polynomial-time.

Theorem. *A DAG G is LCA-RELEVANT $\iff$ no two adjacent vertices of G have the same cluster.*



not LCA-RELEVANT



LCA-RELEVANT

Results: Characterization

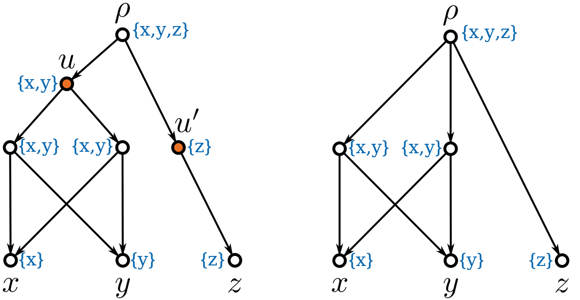
Questions:

- How to determine unsupported vertices ✓ and **how to get rid of them** ?
- Can we characterize LCA-RELEVANT DAGs?✓
  [A DAG G is LCA-RELEVANT if all its vertices are LCA-vertices]

Lemma. *A vertex v is not an LCA-vertex $\iff$*
v has a child w with $\mathtt{C}(v) = \mathtt{C}(w)$.

All non-LCA-vertices can be determined in polynomial-time.

Theorem. *A DAG G is LCA-RELEVANT $\iff$*
no two adjacent vertices of G have the same cluster.



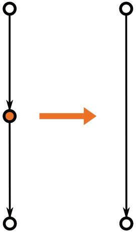
not LCA-RELEVANT   LCA-RELEVANT

Results: The $\ominus$-operator

If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.

Results: The $\ominus$-operator

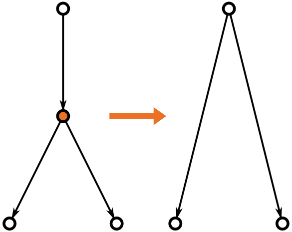
If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.



The $\ominus$-operator generalizes the concept of “vertex-suppression”

Results: The $\ominus$-operator

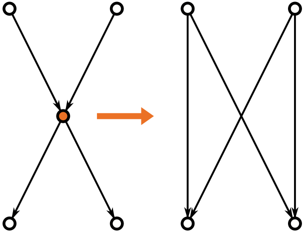
If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.



The $\ominus$-operator generalizes the concept of “vertex-suppression”

Results: The $\ominus$-operator

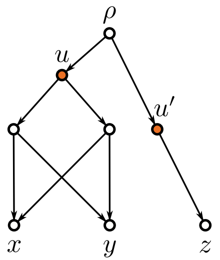
If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.



The $\ominus$-operator generalizes the concept of "vertex-suppression"

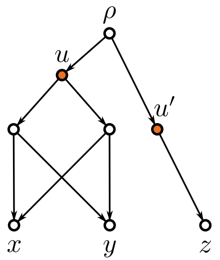
Results: The $\ominus$-operator

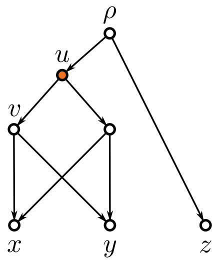
If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.



Results: The $\ominus$-operator

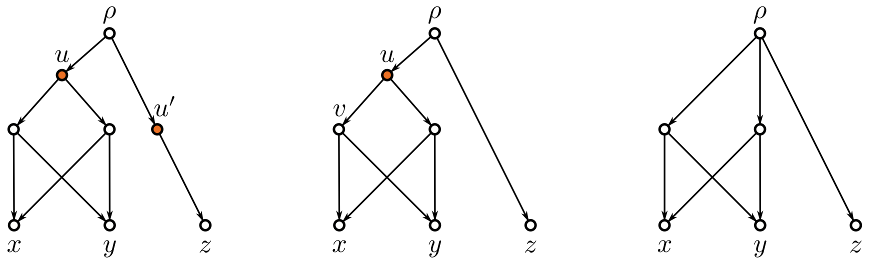
If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.





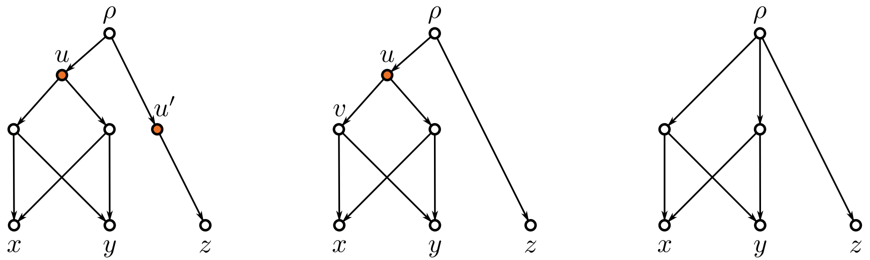
Results: The $\ominus$-operator

If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.



Results: The $\ominus$-operator

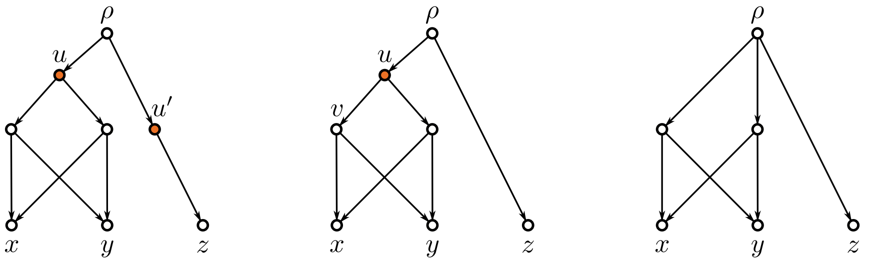
If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.



Theorem. *If W is the set of all non-*LCA *vertices of G, then $G \ominus W$ is* LCA-RELEVANT *and satisfies the following properties:*

Results: The $\ominus$-operator

If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.

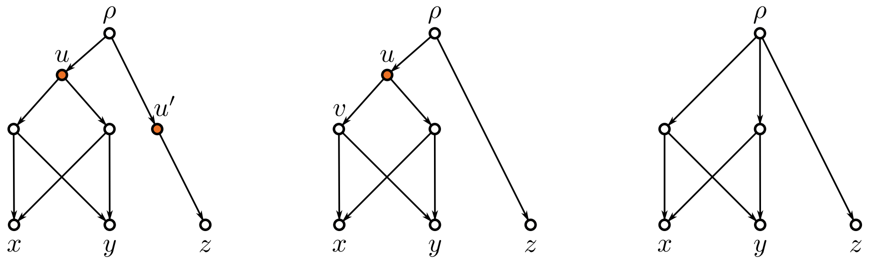


Theorem. *If W is the set of all non-*LCA *vertices of G, then $G \ominus W$ is* LCA-RELEVANT *and satisfies the following properties:*

- $V(G \ominus W) = V(G) \setminus W$ and $L(G \ominus W) = L(G)$ *[no new vertices introduced amd same leaf-set]*

Results: The $\ominus$-operator

If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.

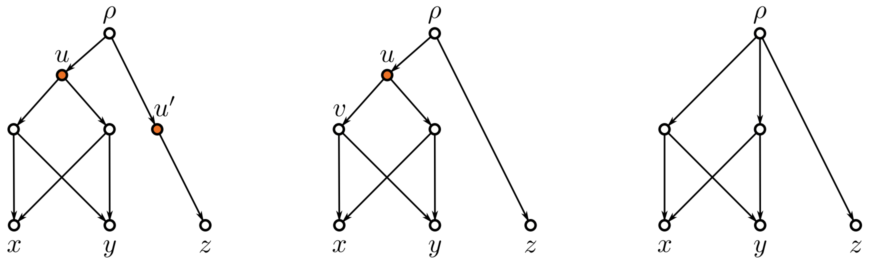


Theorem. *If W is the set of all non-*LCA *vertices of G, then $G \ominus W$ is* LCA-RELEVANT *and satisfies the following properties:*

- $V(G \ominus W) = V(G) \setminus W$ and $L(G \ominus W) = L(G)$ *[no new vertices introduced amd same leaf-set]*

Results: The $\ominus$-operator

If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.

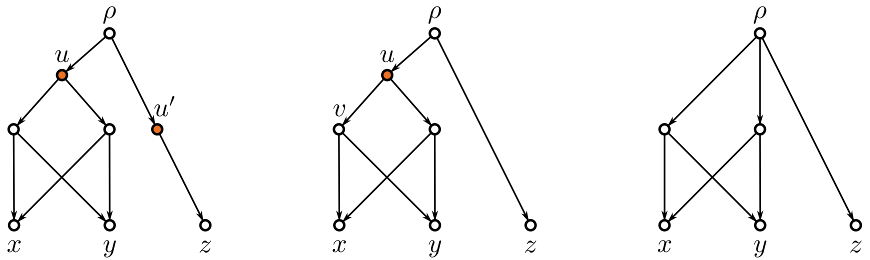


Theorem. *If W is the set of all non-*LCA *vertices of G, then $G \ominus W$ is* LCA-Relevant *and satisfies the following properties:*

- $V(G \ominus W) = V(G) \setminus W$ *and* $L(G \ominus W) = L(G)$ *[no new vertices introduced amd same leaf-set]*
- $u \preceq_G w$ *if and only if* $u \preceq_{G \ominus W} w$ *for all* $u, w \in V(G) \setminus W$ *[ancestor relationship perserved]*

Results: The $\ominus$-operator

If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.

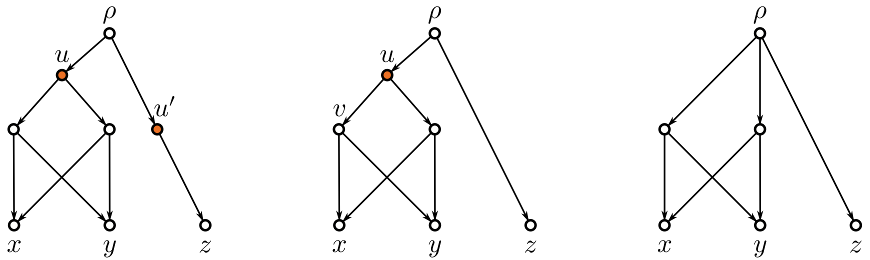


Theorem. *If W is the set of all non-*LCA *vertices of G, then $G \ominus W$ is* LCA-RELEVANT *and satisfies the following properties:*

- $V(G \ominus W) = V(G) \setminus W$ *and* $L(G \ominus W) = L(G)$ *[no new vertices introduced amd same leaf-set]*
- $u \preceq_G w$ *if and only if* $u \preceq_{G \ominus W} w$ *for all* $u, w \in V(G) \setminus W$ *[ancestor relationship perserved]*
- $\mathrm{LCA}_G(A) = \mathrm{LCA}_{G \ominus W}(A)$ *for all* $A \subseteq L(G \ominus W)$ *[the same sets of LCA's]*

Results: The $\ominus$-operator

If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.

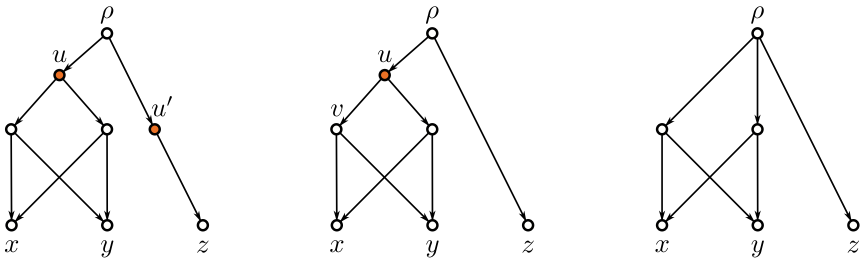


Theorem. *If W is the set of all non-*LCA *vertices of G, then $G \ominus W$ is* LCA-RELEVANT *and satisfies the following properties:*

- $V(G \ominus W) = V(G) \setminus W$ *and* $L(G \ominus W) = L(G)$ *[no new vertices introduced amd same leaf-set]*
- $u \preceq_G w$ *if and only if* $u \preceq_{G \ominus W} w$ *for all* $u, w \in V(G) \setminus W$ *[ancestor relationship perserved]*
- $\mathrm{LCA}_G(A) = \mathrm{LCA}_{G \ominus W}(A)$ *for all* $A \subseteq L(G \ominus W)$ *[the same sets of LCA's]*
- $\mathfrak{C}_G = \mathfrak{C}_{G \ominus W}$ *[same clusters]*

Results: The $\ominus$-operator

If v is a vertex of G, then $G \ominus v$ is obtained by adding an edge from each parent of v to each child of v, and then removing v.



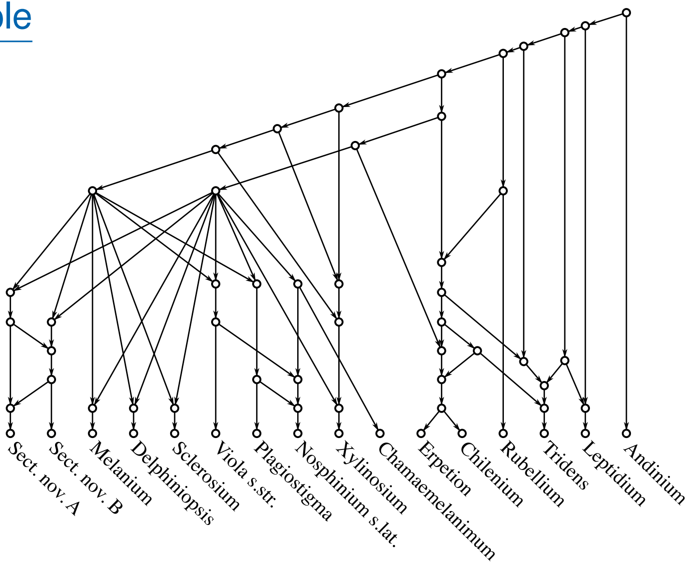
Theorem. *If W is the set of all non-*LCA *vertices of G, then $G \ominus W$ is* LCA-Relevant *and satisfies the following properties:*

- $V(G \ominus W) = V(G) \setminus W$ *and* $L(G \ominus W) = L(G)$ *[no new vertices introduced amd same leaf-set]*
- $u \preceq_G w$ *if and only if* $u \preceq_{G \ominus W} w$ *for all* $u, w \in V(G) \setminus W$ *[ancestor relationship perserved]*
- $\mathrm{LCA}_G(A) = \mathrm{LCA}_{G \ominus W}(A)$ *for all* $A \subseteq L(G \ominus W)$ *[the same sets of LCA's]*
- $\mathfrak{C}_G = \mathfrak{C}_{G \ominus W}$ *[same clusters]*

Moreover, W is the minimum-sized set for which $G \ominus W$ is LCA-Relevant *and $G \ominus W$ can be computed in polynomial time.*

Results: An example

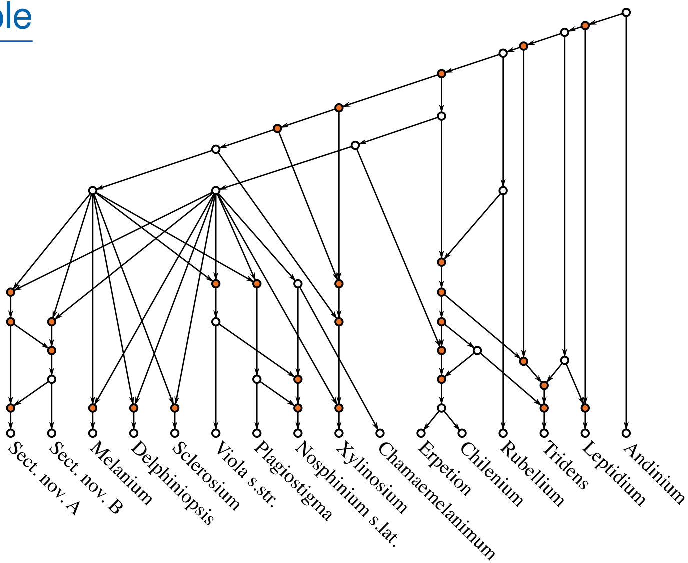
Back to our Plant-Tinder-Elites . . .



Network N

Results: An example

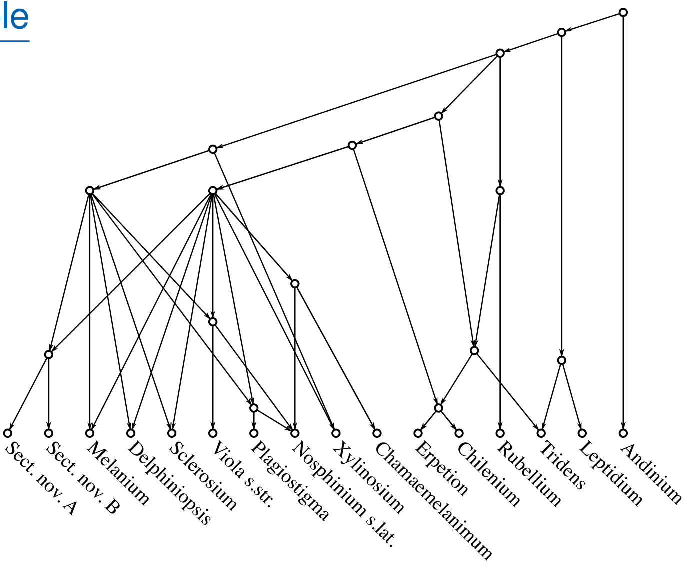
Back to our Plant-Tinder-Elites . . .



Network N with highlighted *non*-LCA vertices

Results: An example

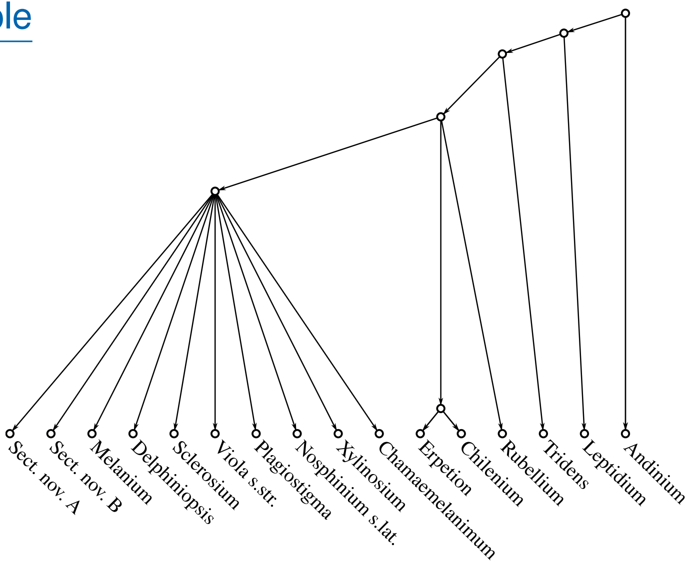
Back to our Plant-Tinder-Elites . . .



The unique LCA-RELEVANT version $N \ominus W$ of N

Results: An example

Back to our Plant-Tinder-Elites . . .



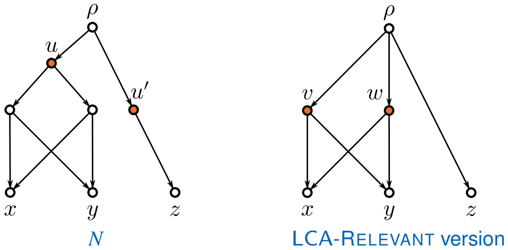
LSA-tree of N to show “trend of evolution”

Results: Let's strengthen LCA-RELEVANT

A DAG G is LCA$^{\mathrm{u}}$-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., LCA$(A) = \{v\}$

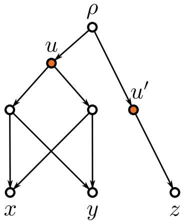
Results: Let's strengthen LCA-Relevant

A DAG G is LCA$^{\mathrm{u}}$-Relevant if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., LCA$(A) = \{v\}$

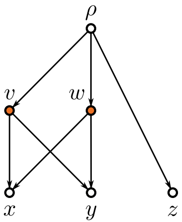


Results: Let's strengthen LCA-RELEVANT

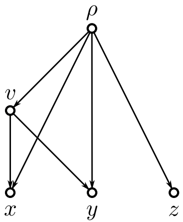
A DAG G is LCA$^{\text{u}}$-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., LCA$(A) = \{v\}$



N



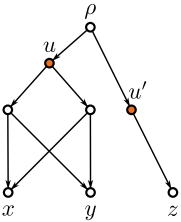
LCA-RELEVANT version



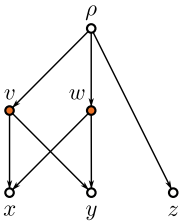
an LCA$^{\text{u}}$-RELEVANT version

Results: Let's strengthen LCA-RELEVANT

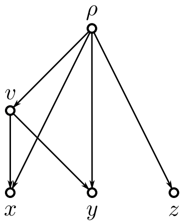
A DAG G is LCA$^{\mathrm{u}}$-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



N



LCA-RELEVANT version

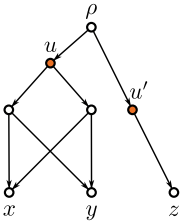


an LCA$^{\mathrm{u}}$-RELEVANT version

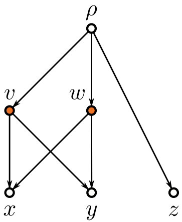
Theorem.
For a network or DAG G the following are equivalent:

Results: Let's strengthen LCA-RELEVANT

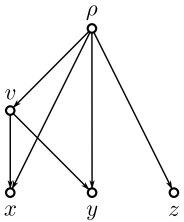
A DAG G is LCA$^{\mathrm{u}}$-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



N



LCA-RELEVANT version



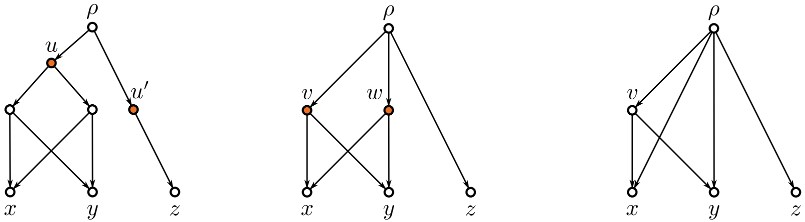
an LCA$^{\mathrm{u}}$-RELEVANT version

Theorem.
For a network or DAG G the following are equivalent:

- *G is* LCA$^{\mathrm{u}}$-RELEVANT

Results: Let's strengthen LCA-RELEVANT

A DAG G is LCA$^{\mathrm{u}}$-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



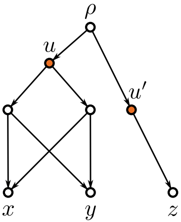
N LCA-RELEVANT version an LCA$^{\mathrm{u}}$-RELEVANT version

Theorem.
For a network or DAG G the following are equivalent:

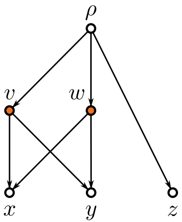
- *G is* LCA$^{\mathrm{u}}$-RELEVANT
- *G is* LCA-RELEVANT *and satisfies (PCC)*

Results: Let's strengthen LCA-Relevant

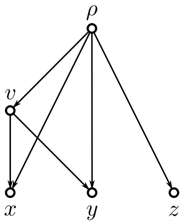
A DAG G is LCA^{u}-Relevant if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



N



LCA-Relevant version

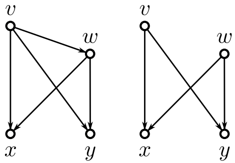


an LCA^{u}-Relevant version

Theorem.
For a network or DAG G the following are equivalent:

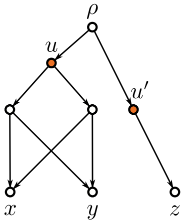
- *G is LCA^{u}-Relevant*
- *G is LCA-Relevant and satisfies (PCC)*

A DAG satisfies (PCC): there is a vw-path if and only if $\mathtt{C}(v)$ and $\mathtt{C}(w)$ are comparable w.r.t $\subseteq$.

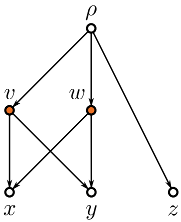


Results: Let's strengthen LCA-RELEVANT

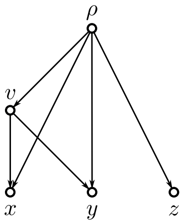
A DAG G is LCA^{u}-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



N



LCA-RELEVANT version



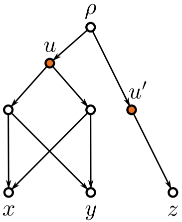
an LCA^{u}-RELEVANT version

Theorem.
For a network or DAG G the following are equivalent:

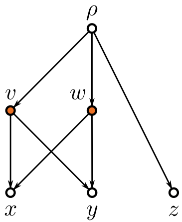
- *G is LCA^{u}-RELEVANT*
- *G is LCA-RELEVANT and satisfies (PCC)*
- *If you remove all shortcuts from G, what you obtain is isomorphic to the Hasse diagram of $\mathfrak{C}_G$*

Results: Let's strengthen LCA-RELEVANT

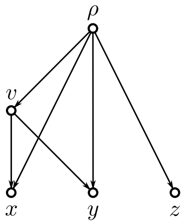
A DAG G is LCA^{u}-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



N



LCA-RELEVANT version

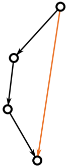


an LCA^{u}-RELEVANT version

Theorem.
For a network or DAG G the following are equivalent:

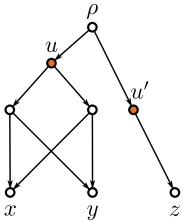
- *G is LCA^{u}-RELEVANT*
- *G is LCA-RELEVANT and satisfies (PCC)*
- *If you remove all shortcuts from G, what you obtain is isomorphic to the Hasse diagram of $\mathfrak{C}_G$*

A shortcut is an edge (u, v) for which there is some other uv-path in G.

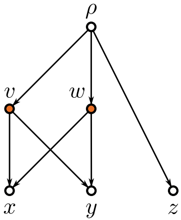


Results: Let's strengthen LCA-RELEVANT

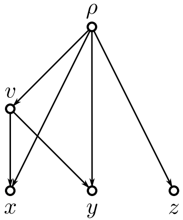
A DAG G is LCA$^{\mathrm{u}}$-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



N



LCA-RELEVANT version



an LCA$^{\mathrm{u}}$-RELEVANT version

Theorem.
For a network or DAG G the following are equivalent:

- *G is LCA$^{\mathrm{u}}$-RELEVANT*
- *G is LCA-RELEVANT and satisfies (PCC)*
- *If you remove all shortcuts from G, what you obtain is isomorphic to the Hasse diagram of $\mathfrak{C}_G$*

$\{x, y, z, w\}$

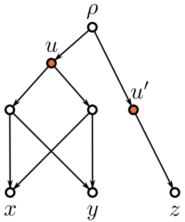
$\{x, y, z\}$

$\{x, y\}$ $\{y, z\}$

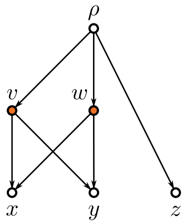
$\{x\}$ $\{y\}$ $\{z\}$ $\{w\}$

Results: Let's strengthen LCA-RELEVANT

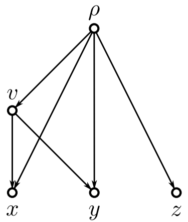
A DAG G is LCA$^{\mathrm{u}}$-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



N



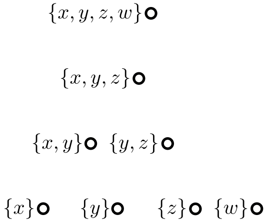
LCA-RELEVANT version



an LCA$^{\mathrm{u}}$-RELEVANT version

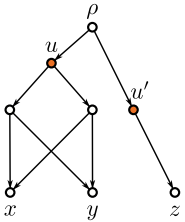
Theorem.
For a network or DAG G the following are equivalent:

- *G is* LCA$^{\mathrm{u}}$-RELEVANT
- *G is* LCA-RELEVANT *and satisfies (PCC)*
- *If you remove all shortcuts from G, what you obtain is isomorphic to the Hasse diagram of $\mathfrak{C}_G$*

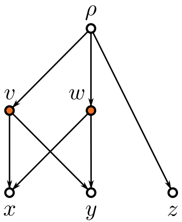


Results: Let's strengthen LCA-RELEVANT

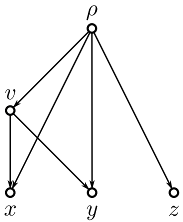
A DAG G is LCA^{u}-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



N



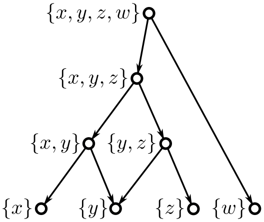
LCA-RELEVANT version



an LCA^{u}-RELEVANT version

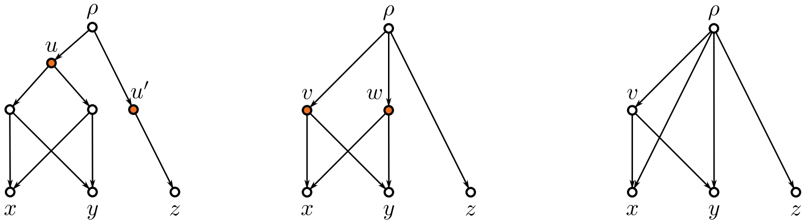
Theorem.
For a network or DAG G the following are equivalent:

- *G is LCA^{u}-RELEVANT*
- *G is LCA-RELEVANT and satisfies (PCC)*
- *If you remove all shortcuts from G, what you obtain is isomorphic to the Hasse diagram of $\mathfrak{C}_G$*



Results: Let's strengthen LCA-RELEVANT

A DAG G is LCA$^{\mathrm{u}}$-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



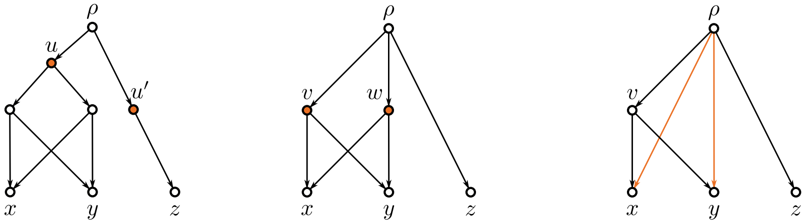
N | LCA-RELEVANT version | an LCA$^{\mathrm{u}}$-RELEVANT version

Theorem.
For a network or DAG G the following are equivalent:

- *G is* LCA$^{\mathrm{u}}$-RELEVANT
- *G is* LCA-RELEVANT *and satisfies (PCC)*
- *If you remove all shortcuts from G, what you obtain is isomorphic to the Hasse diagram of $\mathfrak{C}_G$*

Results: Let's strengthen LCA-RELEVANT

A DAG G is LCA[u]-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



N LCA-RELEVANT version an LCA[u]-RELEVANT version

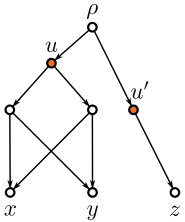
Theorem.

For a network or DAG G the following are equivalent:

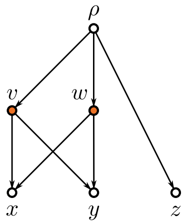
- *G is LCA[u]-RELEVANT*
- *G is LCA-RELEVANT and satisfies (PCC)*
- *If you remove all shortcuts from G, what you obtain is isomorphic to the Hasse diagram of $\mathfrak{C}_G$*

Results: Let's strengthen LCA-RELEVANT

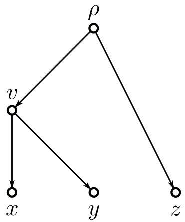
A DAG G is LCA$^{\mathrm{u}}$-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



N



LCA-RELEVANT version



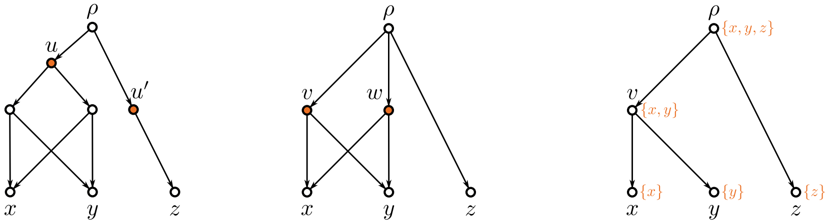
an LCA$^{\mathrm{u}}$-RELEVANT version

Theorem.
For a network or DAG G the following are equivalent:

- *G is* LCA$^{\mathrm{u}}$-RELEVANT
- *G is* LCA-RELEVANT *and satisfies (PCC)*
- *If you remove all shortcuts from G, what you obtain is isomorphic to the Hasse diagram of $\mathfrak{C}_G$*

Results: Let's strengthen LCA-RELEVANT

A DAG G is LCA$^{\mathrm{u}}$-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



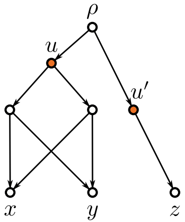
N LCA-RELEVANT version an LCA$^{\mathrm{u}}$-RELEVANT version

Theorem.
For a network or DAG G the following are equivalent:

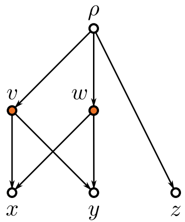
- *G is* LCA$^{\mathrm{u}}$-RELEVANT
- *G is* LCA-RELEVANT *and satisfies (PCC)*
- *If you remove all shortcuts from G, what you obtain is isomorphic to the Hasse diagram of $\mathfrak{C}_G$*

Results: Let's strengthen LCA-RELEVANT

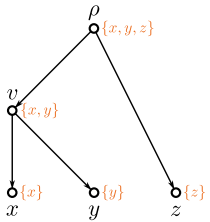
A DAG G is LCA$^{\mathrm{u}}$-RELEVANT if each vertex v is the unique LCA for some $A \subseteq L(G)$, i.e., $\mathrm{LCA}(A) = \{v\}$



N



LCA-RELEVANT version



an LCA$^{\mathrm{u}}$-RELEVANT version

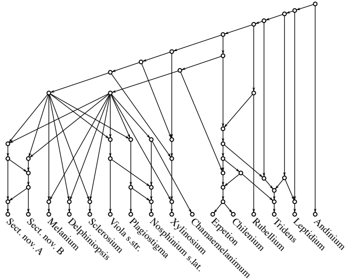
Theorem.
For a network or DAG G the following are equivalent:

- *G is* LCA$^{\mathrm{u}}$-RELEVANT
- *G is* LCA-RELEVANT *and satisfies (PCC)*
- *If you remove all shortcuts from G, what you obtain is isomorphic to the Hasse diagram of $\mathfrak{C}_G$*

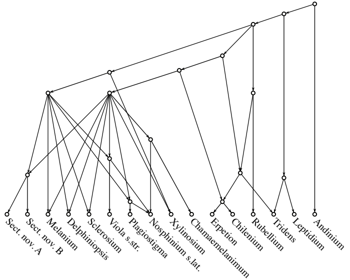
Computing an LCA$^{\mathrm{u}}$-RELEVANT *version of any DAG G can be done in polynomial-time*

Results: An example

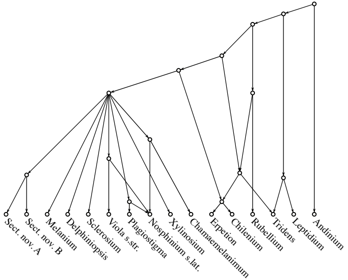
Back to our Plant-Tinder-Elites . . .



N



the LCA-RELEVANT version of N



an LCA^{u}-RELEVANT version of N (all shortcuts removed)

All these simplifications can be computed in polynomial-time and are implemented as python-tool `LCA-SimpliDAG`

Lindeberg and Hellmuth, Simplifying and Characterizing DAGs and Phylogenetic Networks via Least Common Ancestor Constraints, Bulletin of Math. Bio. (submitted), 2024

https://github.com/AnnaLindeberg/LeastCommonAncestor-relevant.git

Other results and open problems

- Generally worked with a generalization: a DAG is $\mathscr{I}$-LCA-RELEVANT if for every vertex $v \in V(G)$ there is some $A \subseteq L(G)$ s.t. $|A| \in \mathscr{I}$ and $v \in \mathrm{LCA}(A)$.

Lindeberg and Hellmuth, Simplifying and Characterizing DAGs and Phylogenetic Networks via Least Common Ancestor Constraints, Bulletin of Math. Bio. (submitted), 2024

Other results and open problems

- Generally worked with a generalization: a DAG is $\mathscr{I}$-LCA-RELEVANT if for every vertex $v \in V(G)$ there is some $A \subseteq L(G)$ s.t. $|A| \in \mathscr{I}$ and $v \in \mathsf{LCA}(A)$.
- Our results also fit rather nicely together with an axiomatic framework recently introduced in the context of another type of simplification (resulting in so-called LSA-trees).

Lindeberg and Hellmuth, Simplifying and Characterizing DAGs and Phylogenetic Networks via Least Common Ancestor Constraints, Bulletin of Math. Bio. (submitted), 2024

Other results and open problems

- Generally worked with a generalization: a DAG is $\mathscr{I}$-LCA-Relevant if for every vertex $v \in V(G)$ there is some $A \subseteq L(G)$ s.t. $|A| \in \mathscr{I}$ and $v \in \mathrm{LCA}(A)$.
- Our results also fit rather nicely together with an axiomatic framework recently introduced in the context of another type of simplification (resulting in so-called LSA-trees).

Still lots of interesting open todos:

Lindeberg and Hellmuth, Simplifying and Characterizing DAGs and Phylogenetic Networks via Least Common Ancestor Constraints, Bulletin of Math. Bio. (submitted), 2024

Other results and open problems

- Generally worked with a generalization: a DAG is $\mathscr{I}$-LCA-RELEVANT if for every vertex $v \in V(G)$ there is some $A \subseteq L(G)$ s.t. $|A| \in \mathscr{I}$ and $v \in \mathrm{LCA}(A)$.
- Our results also fit rather nicely together with an axiomatic framework recently introduced in the context of another type of simplification (resulting in so-called LSA-trees).

Still lots of interesting open todos:

- For $G \ominus W$ being LCA-RELEVANT the set W is uniquely determined and $\mathfrak{C}_{G \ominus W} = \mathfrak{C}_G$

Lindeberg and Hellmuth, Simplifying and Characterizing DAGs and Phylogenetic Networks via Least Common Ancestor Constraints, Bulletin of Math. Bio. (submitted), 2024

Other results and open problems

- Generally worked with a generalization: a DAG is $\mathscr{I}$-LCA-RELEVANT if for every vertex $v \in V(G)$ there is some $A \subseteq L(G)$ s.t. $|A| \in \mathscr{I}$ and $v \in \mathrm{LCA}(A)$.
- Our results also fit rather nicely together with an axiomatic framework recently introduced in the context of another type of simplification (resulting in so-called LSA-trees).

Still lots of interesting open todos:

- For $G \ominus W$ being LCA-RELEVANT the set W is uniquely determined and $\mathfrak{C}_{G \ominus W} = \mathfrak{C}_G$

  For $G \ominus W$ being LCA$^{\mathrm{u}}$-RELEVANT, the set W is not uniquely determined and $\mathfrak{C}_{G \ominus W} \subseteq \mathfrak{C}_G$

Lindeberg and Hellmuth, Simplifying and Characterizing DAGs and Phylogenetic Networks via Least Common Ancestor Constraints, Bulletin of Math. Bio. (submitted), 2024

Other results and open problems

- Generally worked with a generalization: a DAG is $\mathscr{I}$-LCA-RELEVANT if for every vertex $v \in V(G)$ there is some $A \subseteq L(G)$ s.t. $|A| \in \mathscr{I}$ and $v \in \mathrm{LCA}(A)$.
- Our results also fit rather nicely together with an axiomatic framework recently introduced in the context of another type of simplification (resulting in so-called LSA-trees).

Still lots of interesting open todos:

- For $G \ominus W$ being LCA-RELEVANT the set W is uniquely determined and $\mathfrak{C}_{G \ominus W} = \mathfrak{C}_G$

  For $G \ominus W$ being $\mathrm{LCA}^{\mathrm{u}}$-RELEVANT, the set W is not uniquely determined and $\mathfrak{C}_{G \ominus W} \subseteq \mathfrak{C}_G$

  How to find min-sized $|W|$? Or how to find W s.t. $|\mathfrak{C}_G \setminus \mathfrak{C}_{G \ominus W}|$ is minimized?

Lindeberg and Hellmuth, Simplifying and Characterizing DAGs and Phylogenetic Networks via Least Common Ancestor Constraints, Bulletin of Math. Bio. (submitted), 2024

Other results and open problems

- Generally worked with a generalization: a DAG is $\mathscr{I}$-LCA-RELEVANT if for every vertex $v \in V(G)$ there is some $A \subseteq L(G)$ s.t. $|A| \in \mathscr{I}$ and $v \in \mathrm{LCA}(A)$.
- Our results also fit rather nicely together with an axiomatic framework recently introduced in the context of another type of simplification (resulting in so-called LSA-trees).

Still lots of interesting open todos:

- For $G \ominus W$ being LCA-RELEVANT the set W is uniquely determined and $\mathfrak{C}_{G\ominus W} = \mathfrak{C}_G$

  For $G \ominus W$ being LCA$^{\mathrm{u}}$-RELEVANT, the set W is not uniquely determined and $\mathfrak{C}_{G\ominus W} \subseteq \mathfrak{C}_G$

  How to find min-sized $|W|$? Or how to find W s.t. $|\mathfrak{C}_G \setminus \mathfrak{C}_{G\ominus W}|$ is minimized?
- Is it too much to require vertices to be LCA's of leaves?

  Recursive Def:

  Every leaf in G is pertinent.

  A vertex is pertinent if it is the LCA of pertinent vertices.

  What is the structure of DAGs in which all vertices are pertinent and how to determine such vertices?

Lindeberg and Hellmuth, Simplifying and Characterizing DAGs and Phylogenetic Networks via Least Common Ancestor Constraints, Bulletin of Math. Bio. (submitted), 2024

Joint work with

Anna Lindeberg

Joint work with

Anna Lindeberg

Thank You!