Color Refinement for Relational Structures

Benjamin Scheidt Nicole Schweikardt

Humboldt-Universität zu Berlin

AlMoTh'25

arXiv:2407.16022

What the next 20 minutes will be about

Color Refinement

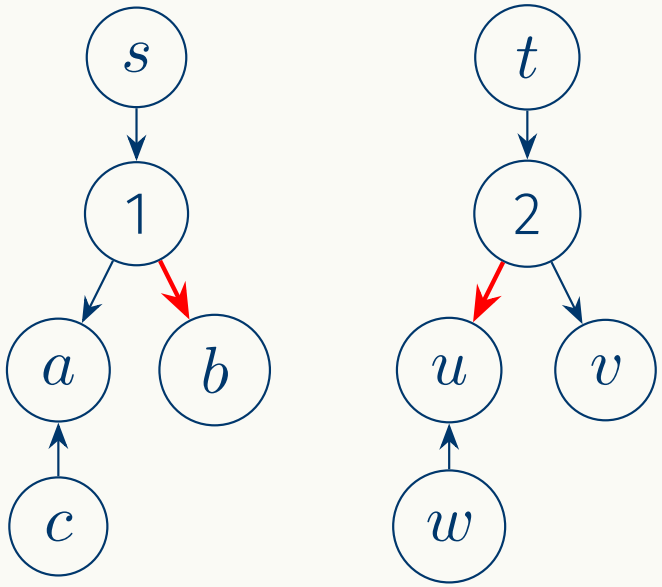
How to lift CR to Relational Structures

Doing it Efficiently

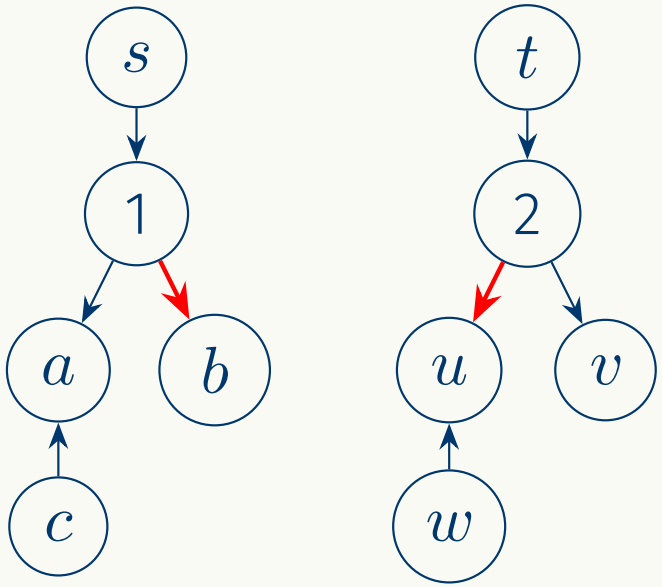
Simpler Approaches?

Outlook

Color Refinement: Iteratively compute stable coloring γ_∞

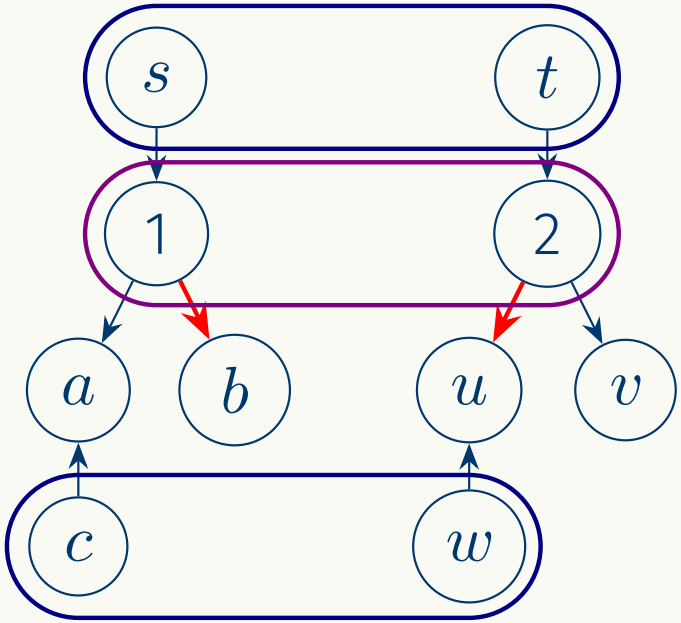


Color Refinement: Iteratively compute stable coloring γ_∞



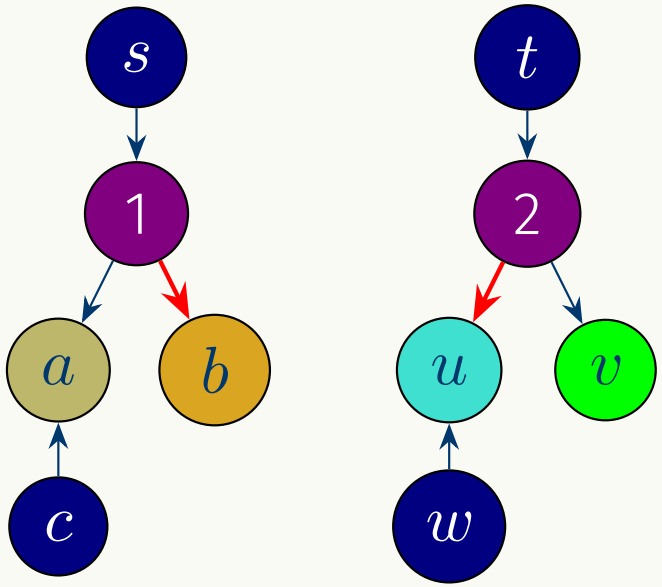
$\gamma_{i+1}(u) \neq \gamma_{i+1}(v) \iff$ ex. edge and vertex colors λ, α s.t. num. of λ-neighbors of color α differ.

Color Refinement: Iteratively compute stable coloring γ_∞



$\gamma_{i+1}(u) \neq \gamma_{i+1}(v) \iff$ ex. edge and vertex colors λ, α s.t. num. of λ-neighbors of color α differ.

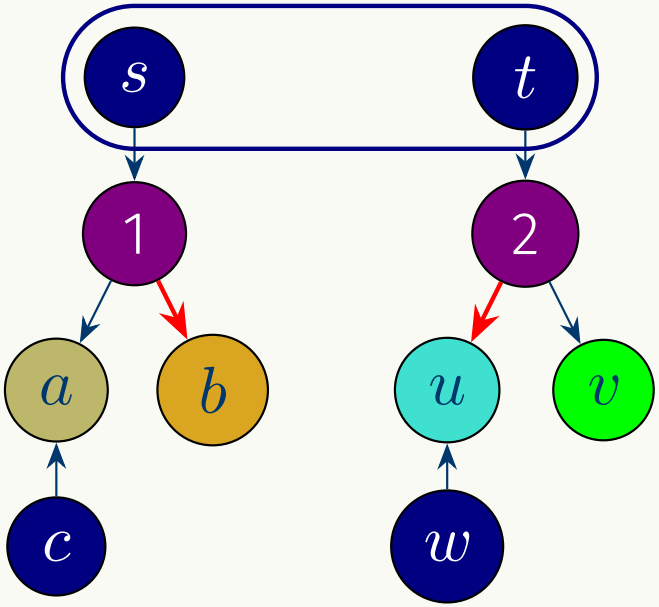
Color Refinement: Iteratively compute stable coloring γ_∞



$$\gamma_{i+1}(u) \neq \gamma_{i+1}(v) \iff$$

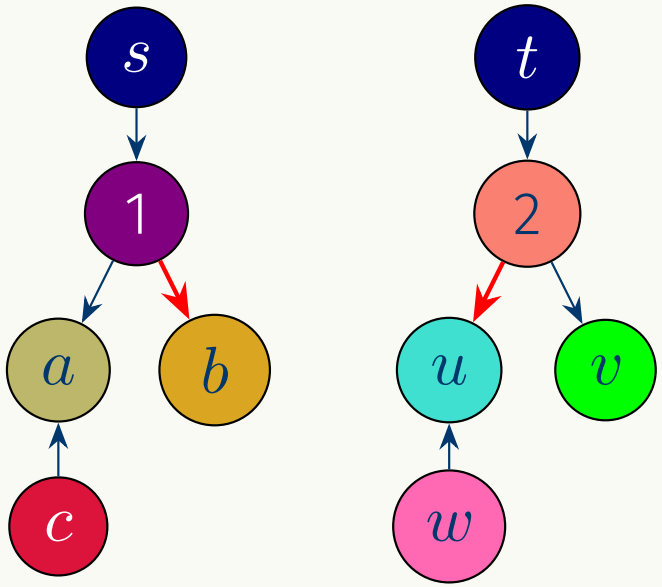
ex. edge and vertex colors λ, α s.t.
num. of λ-neighbors of color α differ.

Color Refinement: Iteratively compute stable coloring γ_∞



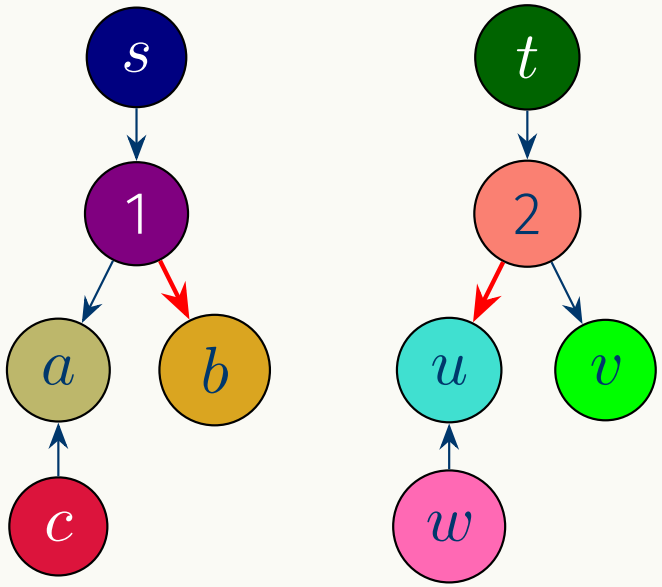
$\gamma_{i+1}(u) \neq \gamma_{i+1}(v) \iff$ ex. edge and vertex colors λ, α s.t. num. of λ-neighbors of color α differ.

Color Refinement: Iteratively compute stable coloring γ_∞



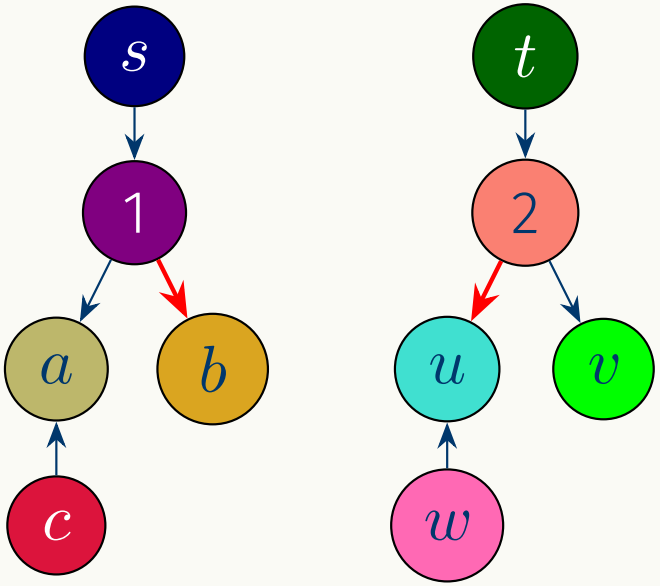
$\gamma_{i+1}(u) \neq \gamma_{i+1}(v) \iff$ ex. edge and vertex colors λ, α s.t. num. of λ-neighbors of color α differ.

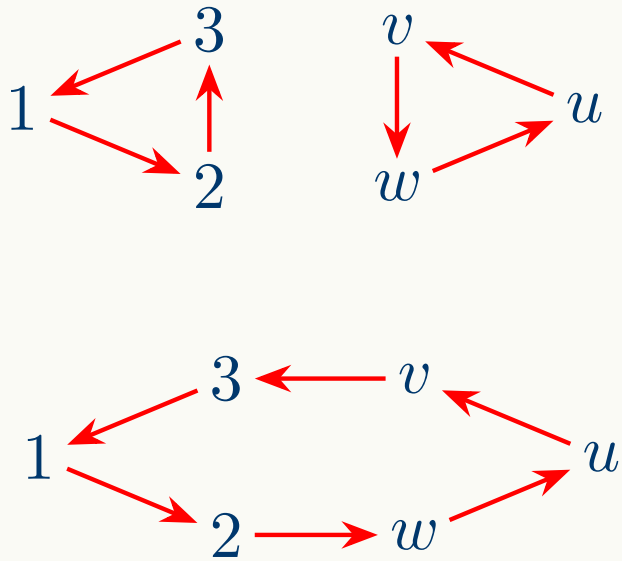
Color Refinement: Iteratively compute stable coloring γ_∞



$\gamma_{i+1}(u) \neq \gamma_{i+1}(v) \iff$ ex. edge and vertex colors λ, α s.t. num. of λ-neighbors of color α differ.

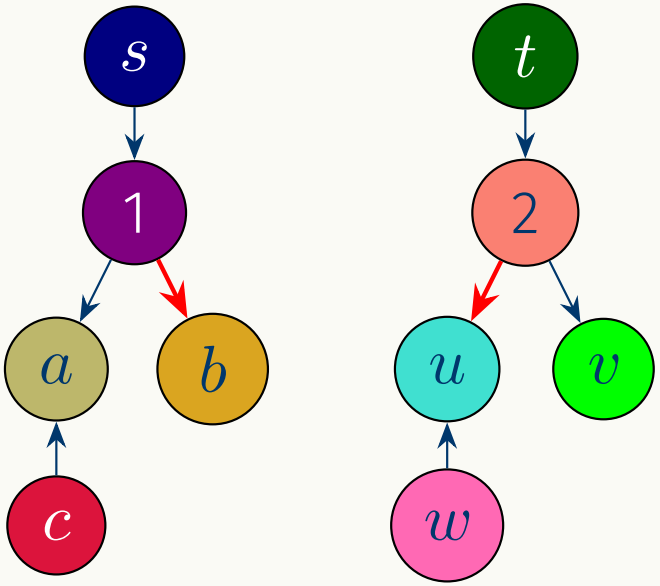
Color Refinement: Iteratively compute stable coloring γ_∞

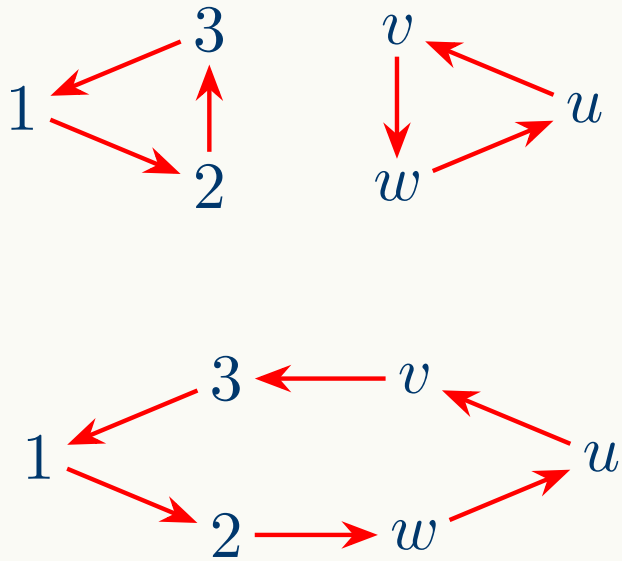




$\gamma_{i+1}(u) \neq \gamma_{i+1}(v) \iff$ ex. edge and vertex colors λ, α s.t. num. of λ-neighbors of color α differ.

Color Refinement: Iteratively compute stable coloring γ_∞

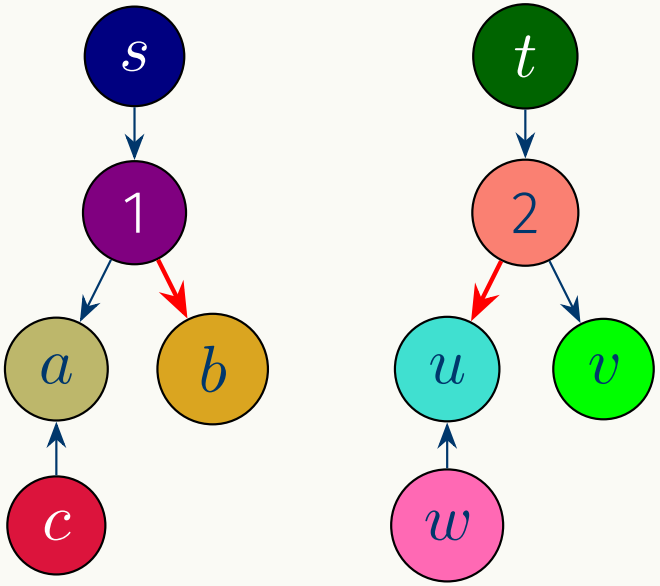


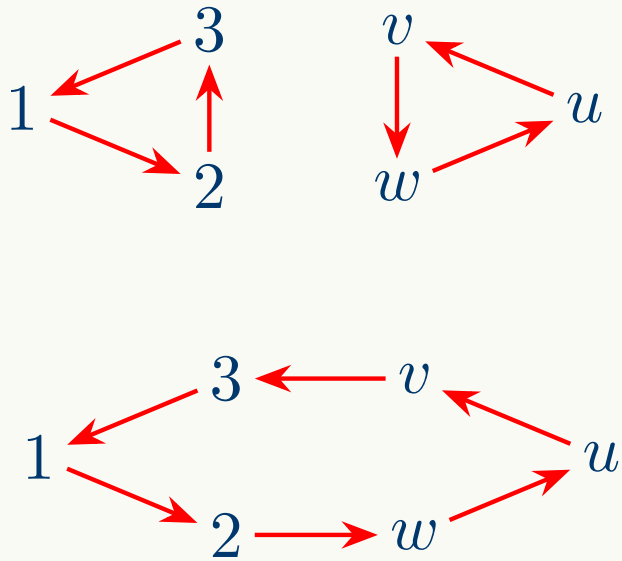


$\gamma_{i+1}(u) \neq \gamma_{i+1}(v) \iff$ ex. edge and vertex colors λ, α s.t. num. of λ-neighbors of color α differ.

Can be used to *distinguish* graphs and *classify* vertices.

Color Refinement: Iteratively compute stable coloring γ_∞





$\gamma_{i+1}(u) \neq \gamma_{i+1}(v) \iff$ ex. edge and vertex colors λ, α s.t. num. of λ-neighbors of color α differ.

Can be used to *distinguish* graphs and *classify* vertices.

Wanted: similar method for more complex objects.

In this talk: Relational structures.

But how to identify the “right” approach?

Key Properties of Color Refinement

Theorem (Dvořák 2010; Dell, Grohe, Rattan 2018; Immerman, Lander 1990; and others)

The following are equivalent:

- *Color Refinement distinguishes G and H.*

Key Properties of Color Refinement

Theorem (Dvořák 2010; Dell, Grohe, Rattan 2018; Immerman, Lander 1990; and others)

The following are equivalent:

- *Color Refinement distinguishes G and H.*
- *There is a tree F s.t.* $\hom(F, G) \neq \hom(F, H)$.

Key Properties of Color Refinement

Theorem (Dvořák 2010; Dell, Grohe, Rattan 2018; Immerman, Lander 1990; and others)

The following are equivalent:

- *Color Refinement distinguishes G and H.*
- *There is a tree F s.t.* $\hom(F, G) \neq \hom(F, H)$.
- *There is a sentence $\varphi \in \mathsf{C}^2$ s.t. $G \models \varphi$ and $H \not\models \varphi$.*

Key Properties of Color Refinement

Theorem (Dvořák 2010; Dell, Grohe, Rattan 2018; Immerman, Lander 1990; and others)

The following are equivalent:

- *Color Refinement distinguishes G and H.*
- *There is a tree F s.t.* $\hom(F, G) \neq \hom(F, H)$.
- *There is a sentence $\varphi \in \mathsf{C}^2$ s.t. $G \models \varphi$ and $H \not\models \varphi$.*

CR can be implemented to run in time $\mathcal{O}\big((n+m)\log n\big)$.

Key Properties of Color Refinement

Theorem (Dvořák 2010; Dell, Grohe, Rattan 2018; Immerman, Lander 1990; and others)

The following are equivalent:

- *Color Refinement distinguishes G and H.*
- *There is a tree F s.t.* $\hom(F, G) \neq \hom(F, H)$.
- *There is a sentence $\varphi \in \mathsf{C}^2$ s.t. $G \models \varphi$ and $H \not\models \varphi$.*

CR can be implemented to run in time $\mathcal{O}\big((n+m)\log n\big)$.

Want for "Relational Color Refinement" (RCR) to be equivalent:

- Relational Color Refinement distinguishes $\mathcal{A}$ and $\mathcal{B}$.

Key Properties of Color Refinement

Theorem (Dvořák 2010; Dell, Grohe, Rattan 2018; Immerman, Lander 1990; and others)

The following are equivalent:

- *Color Refinement distinguishes G and H.*
- *There is a tree F s.t.* $\hom(F, G) \neq \hom(F, H)$.
- *There is a sentence $\varphi \in \mathsf{C}^2$ s.t. $G \models \varphi$ and $H \not\models \varphi$.*

CR can be implemented to run in time $\mathcal{O}\big((n+m)\log n\big)$.

Want for "Relational Color Refinement" (RCR) to be equivalent:

- Relational Color Refinement distinguishes $\mathcal{A}$ and $\mathcal{B}$.
- There is an "acyclic" $\mathcal{C}$ s.t. $\hom(\mathcal{C}, \mathcal{A}) \neq \hom(\mathcal{C}, \mathcal{B})$.

Key Properties of Color Refinement

Theorem (Dvořák 2010; Dell, Grohe, Rattan 2018; Immerman, Lander 1990; and others)

The following are equivalent:

- *Color Refinement distinguishes G and H.*
- *There is a tree F s.t.* $\hom(F, G) \neq \hom(F, H)$.
- *There is a sentence $\varphi \in \mathsf{C}^2$ s.t. $G \models \varphi$ and $H \not\models \varphi$.*

CR can be implemented to run in time $\mathcal{O}\big((n+m)\log n\big)$.

Want for "Relational Color Refinement" (RCR) to be equivalent:

- Relational Color Refinement distinguishes $\mathcal{A}$ and $\mathcal{B}$.
- There is an "acyclic" $\mathcal{C}$ s.t. $\hom(\mathcal{C}, \mathcal{A}) \neq \hom(\mathcal{C}, \mathcal{B})$.
- There is a sentence $\varphi \in \mathsf{L}$ s.t. $\mathcal{A} \models \varphi$ and $\mathcal{B} \not\models \varphi$.

Key Properties of Color Refinement

Theorem (Dvořák 2010; Dell, Grohe, Rattan 2018; Immerman, Lander 1990; and others)

The following are equivalent:

- *Color Refinement distinguishes G and H.*
- *There is a tree F s.t. $\hom(F, G) \neq \hom(F, H)$.*
- *There is a sentence $\varphi \in \mathsf{C}^2$ s.t. $G \models \varphi$ and $H \not\models \varphi$.*

CR can be implemented to run in time $\mathcal{O}\big((n+m)\log n\big)$.

Want for "Relational Color Refinement" (RCR) to be equivalent:

- Relational Color Refinement distinguishes $\mathcal{A}$ and $\mathcal{B}$.
- There is an "acyclic" $\mathcal{C}$ s.t. $\hom(\mathcal{C}, \mathcal{A}) \neq \hom(\mathcal{C}, \mathcal{B})$.
- There is a sentence $\varphi \in \mathsf{L}$ s.t. $\mathcal{A} \models \varphi$ and $\mathcal{B} \not\models \varphi$.

And RCR should run in time $\mathcal{O}\big(\|\mathcal{A}\| \cdot \log \|\mathcal{A}\|\big)$.

Our Result

Theorem (Scheidt, Schweikardt 2024)

The following are equivalent:

- *Relational Color Refinement distinguishes $\mathcal{A}$ and $\mathcal{B}$.*
- *There is an acyclic $\mathcal{C}$ s.t. $\hom(\mathcal{C}, \mathcal{A}) \neq \hom(\mathcal{C}, \mathcal{B})$.*
- *There is a $\varphi \in \mathrm{GF}(\mathrm{C})$ s.t. $\mathcal{A} \models \varphi$, and $\mathcal{B} \not\models \varphi$.*

Our Result

Theorem (Scheidt, Schweikardt 2024)

The following are equivalent:

- *Relational Color Refinement distinguishes $\mathcal{A}$ and $\mathcal{B}$.*
- *There is an acyclic $\mathcal{C}$ s.t. $\hom(\mathcal{C}, \mathcal{A}) \neq \hom(\mathcal{C}, \mathcal{B})$.*
- *There is a $\varphi \in \mathsf{GF}(\mathsf{C})$ s.t. $\mathcal{A} \models \varphi$, and $\mathcal{B} \not\models \varphi$.*
- *Spoiler has a win. strategy for the Guarded-Game on $\mathcal{A}$, $\mathcal{B}$.*

Our Result

Theorem (Scheidt, Schweikardt 2024)

The following are equivalent:

- *Relational Color Refinement distinguishes $\mathcal{A}$ and $\mathcal{B}$.*
- *There is an acyclic $\mathcal{C}$ s.t. $\hom(\mathcal{C}, \mathcal{A}) \neq \hom(\mathcal{C}, \mathcal{B})$.*
- *There is a $\varphi \in \mathsf{GF}(\mathsf{C})$ s.t. $\mathcal{A} \models \varphi$, and $\mathcal{B} \not\models \varphi$.*
- *Spoiler has a win. strategy for the Guarded-Game on $\mathcal{A}$, $\mathcal{B}$.*

RCR can be implemented to run in time $\mathcal{O}(\|\mathcal{A}\| \cdot \log \|\mathcal{A}\|)$.

Our Result

Theorem (Scheidt, Schweikardt 2024)

The following are equivalent:

- *Relational Color Refinement distinguishes $\mathcal{A}$ and $\mathcal{B}$.*
- *There is an acyclic $\mathcal{C}$ s.t. $\hom(\mathcal{C}, \mathcal{A}) \neq \hom(\mathcal{C}, \mathcal{B})$.*
- *There is a $\varphi \in \mathsf{GF}(\mathsf{C})$ s.t. $\mathcal{A} \models \varphi$, and $\mathcal{B} \not\models \varphi$.*
- *Spoiler has a win. strategy for the Guarded-Game on $\mathcal{A}$, $\mathcal{B}$.*

RCR can be implemented to run in time $\mathcal{O}(\|\mathcal{A}\| \cdot \log \|\mathcal{A}\|)$.

On graphs, RCR is equivalent to CR.

Our Result

Theorem (Scheidt, Schweikardt 2024)

The following are equivalent:

- *Relational Color Refinement distinguishes* $\mathcal{A}$ *and* $\mathcal{B}$*.*
- *There is an acyclic* $\mathcal{C}$ *s.t.* $\hom(\mathcal{C}, \mathcal{A}) \neq \hom(\mathcal{C}, \mathcal{B})$*.*
- *There is a* $\varphi \in \mathsf{GF}(\mathsf{C})$ *s.t.* $\mathcal{A} \models \varphi$*, and* $\mathcal{B} \not\models \varphi$*.*
- *Spoiler has a win. strategy for the Guarded-Game on* $\mathcal{A}$*,* $\mathcal{B}$*.*

RCR can be implemented to run in time $\mathcal{O}(\|\mathcal{A}\| \cdot \log \|\mathcal{A}\|)$*.*

On graphs, RCR is equivalent to CR.

Our Result

Theorem (Scheidt, Schweikardt 2024)

The following are equivalent:

- *Relational Color Refinement distinguishes $\mathcal{A}$ and $\mathcal{B}$.*
- *There is an acyclic $\mathcal{C}$ s.t. $\hom(\mathcal{C}, \mathcal{A}) \neq \hom(\mathcal{C}, \mathcal{B})$.*
- *There is a $\varphi \in \mathsf{GF}(\mathsf{C})$ s.t. $\mathcal{A} \models \varphi$, and $\mathcal{B} \not\models \varphi$.*
- *Spoiler has a win. strategy for the Guarded-Game on $\mathcal{A}, \mathcal{B}$.*

RCR can be implemented to run in time $\mathcal{O}(\|\mathcal{A}\| \cdot \log \|\mathcal{A}\|)$.

On graphs, RCR is equivalent to CR.

Acyclic Structures — Join Trees!

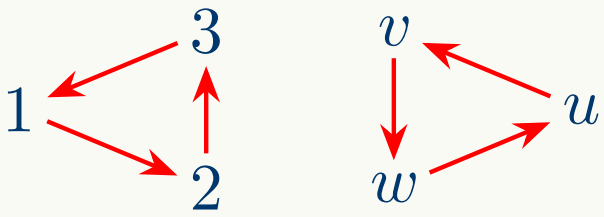
- We call $\mathcal{A}$ acyclic, if there is a join tree for $\mathcal{A}$.

Acyclic Structures — Join Trees!

- We call $\mathcal{A}$ acyclic, if there is a join tree for $\mathcal{A}$.
- A join tree J of $\mathcal{A}$ is a tree whose vertex set consists of the tuples in $\mathcal{A}$ s.t. all elements of $\mathcal{A}$ induce a subtree in J.

Acyclic Structures — Join Trees!

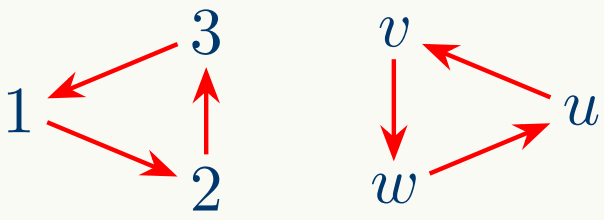
- We call $\mathcal{A}$ acyclic, if there is a join tree for $\mathcal{A}$.
- A join tree J of $\mathcal{A}$ is a tree whose vertex set consists of the tuples in $\mathcal{A}$ s.t. all elements of $\mathcal{A}$ induce a subtree in J.

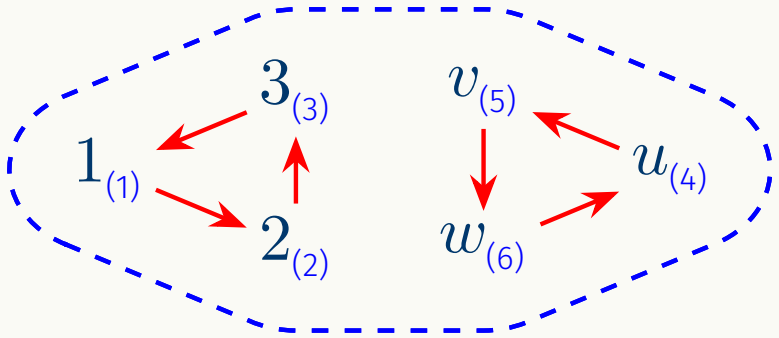


not acyclic.

Acyclic Structures — Join Trees!

- We call $\mathcal{A}$ acyclic, if there is a join tree for $\mathcal{A}$.
- A join tree J of $\mathcal{A}$ is a tree whose vertex set consists of the tuples in $\mathcal{A}$ s.t. all elements of $\mathcal{A}$ induce a subtree in J.

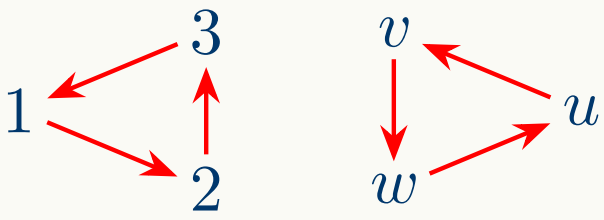




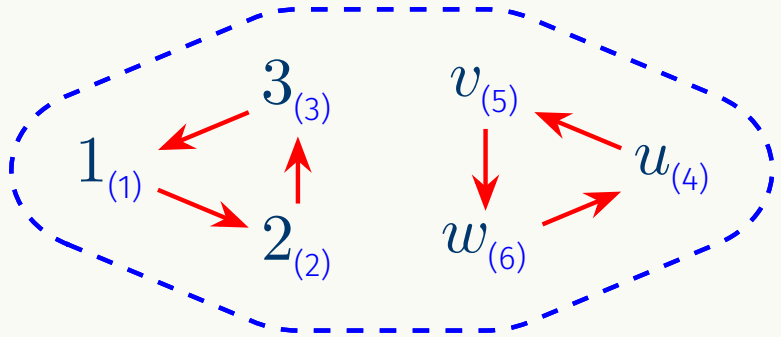
not acyclic.

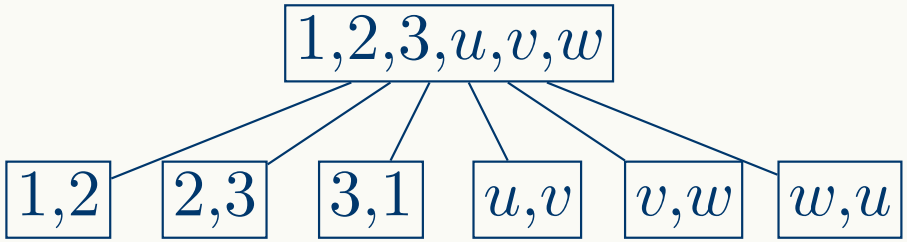
Acyclic Structures — Join Trees!

- We call $\mathcal{A}$ acyclic, if there is a join tree for $\mathcal{A}$.
- A join tree J of $\mathcal{A}$ is a tree whose vertex set consists of the tuples in $\mathcal{A}$ s.t. all elements of $\mathcal{A}$ induce a subtree in J.



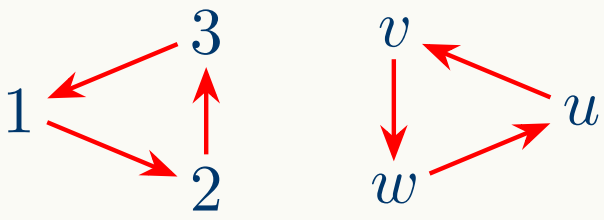
not acyclic.

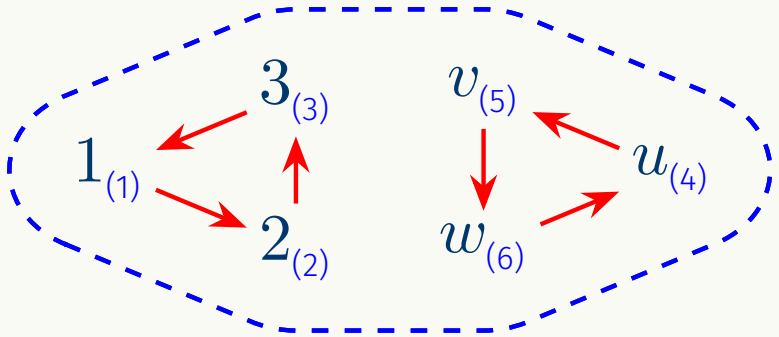




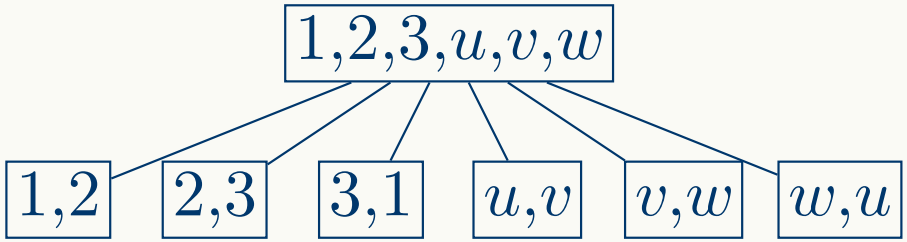
Acyclic Structures — Join Trees!

- We call $\mathcal{A}$ acyclic, if there is a join tree for $\mathcal{A}$.
- A join tree J of $\mathcal{A}$ is a tree whose vertex set consists of the tuples in $\mathcal{A}$ s.t. all elements of $\mathcal{A}$ induce a subtree in J.





not acyclic.



- Standard, most general notion from DB theory, (Abiteboul, Hull, Vianu 1995).

The Guarded Fragment of FO with Counting Quantifiers

Atomic formulas and Boolean operators as usual:

- $x = y$, $R(x_1, \dots, x_\ell)$ for all $R \in \sigma$
- $\neg\varphi$, $(\varphi \wedge \psi)$, ...

The Guarded Fragment of FO with Counting Quantifiers

Atomic formulas and Boolean operators as usual:

- $x = y,\ R(x_1, \dots, x_\ell)$ for all $R \in \sigma$
- $\neg\varphi,\ (\varphi \wedge \psi),\ \dots$

Guarded Fragments: restricted, "local" quantification:

- $\exists^{\geq n}\,(y_1, \dots, y_k).(\,R(x_1, \dots, x_\ell) \wedge \varphi\,)$ with

The Guarded Fragment of FO with Counting Quantifiers

Atomic formulas and Boolean operators as usual:

- $x = y,\ R(x_1, \dots, x_\ell)$ for all $R \in \sigma$
- $\neg\varphi,\ (\varphi \wedge \psi),\ \dots$

Guarded Fragments: restricted, "local" quantification:

- $\exists^{\geq n}\,(y_1, \dots, y_k).(\,R(x_1, \dots, x_\ell) \wedge \varphi\,)$ with
  - $\mathsf{free}(\varphi) \subseteq \{\,x_1, \dots, x_\ell\,\}$

The Guarded Fragment of FO with Counting Quantifiers

Atomic formulas and Boolean operators as usual:

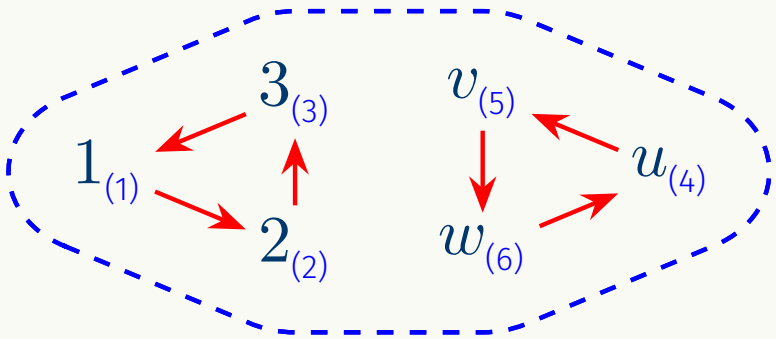
- $x = y$, $R(x_1, \dots, x_\ell)$ for all $R \in \sigma$
- $\neg\varphi$, $(\varphi \wedge \psi)$, ...

Guarded Fragments: restricted, "local" quantification:

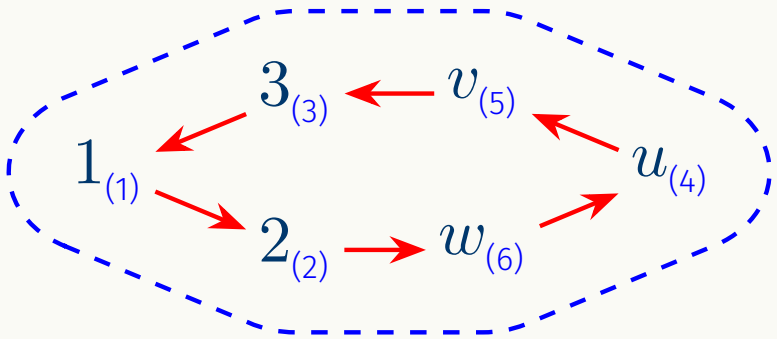
- $\exists^{\geq n}(y_1, \dots, y_k).(R(x_1, \dots, x_\ell) \wedge \varphi)$ with
  - $\text{free}(\varphi) \subseteq \{x_1, \dots, x_\ell\}$ and
  - $\{y_1, \dots, y_k\} \subseteq \{x_1, \dots, x_\ell\}$.

GF(C) — Example

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

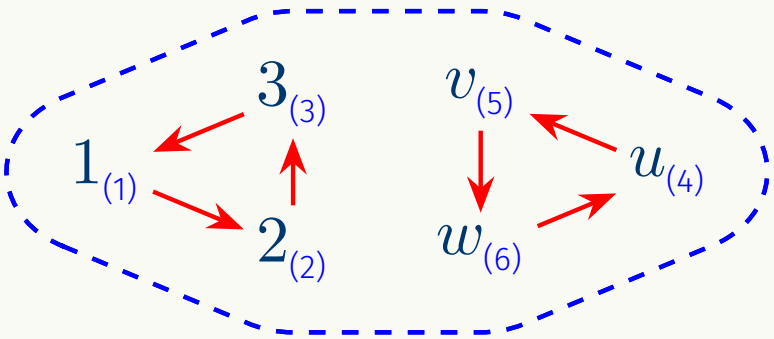


$R^{\mathcal{B}}$	$1,2,3,u,v,w$		
$E^{\mathcal{B}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

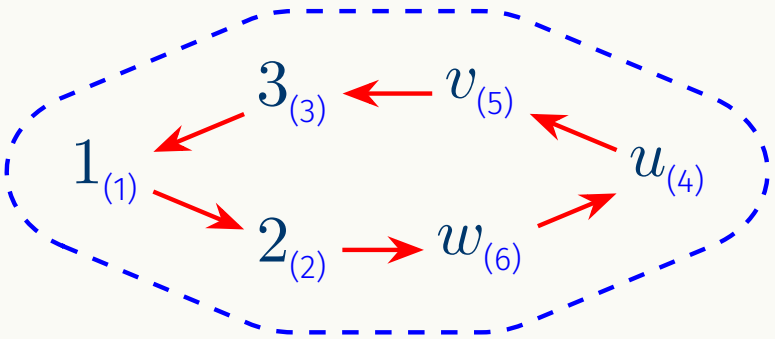


GF(C) — Example

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u



$R^{\mathcal{B}}$	$1,2,3,u,v,w$		
$E^{\mathcal{B}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

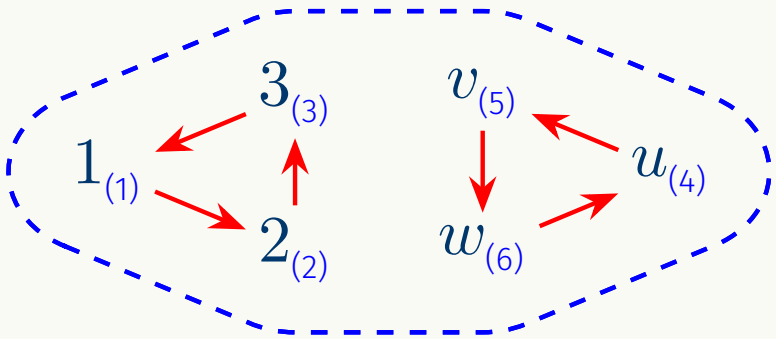


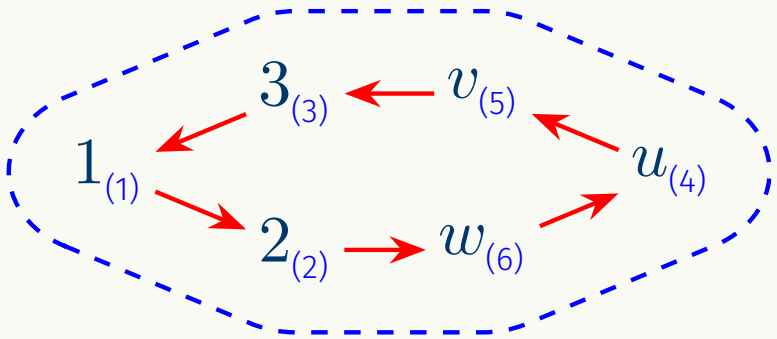
$$\hom(\mathcal{A},\mathcal{B}) = 0 \quad \text{and} \quad \hom(\mathcal{A},\mathcal{A}) > 0$$

GF(C) — Example

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

$R^{\mathcal{B}}$	$1,2,3,u,v,w$		
$E^{\mathcal{B}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u





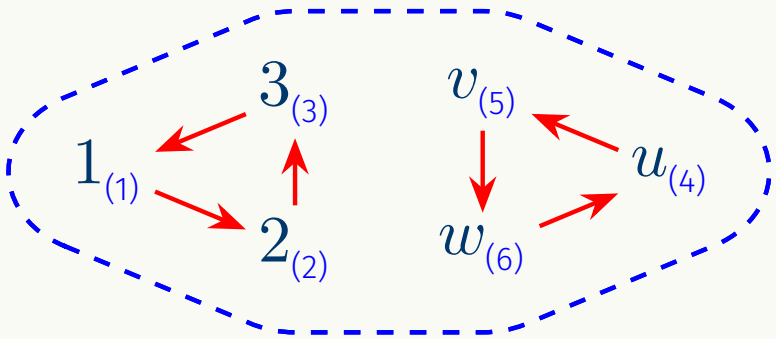
$$\hom(\mathcal{A},\mathcal{B}) = 0 \quad \text{and} \quad \hom(\mathcal{A},\mathcal{A}) > 0$$

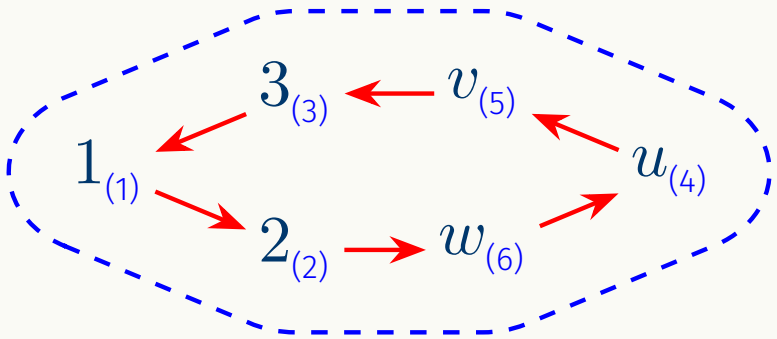
$$\exists^{\geq 1}(x_1,\dots,x_6).\Big(R(x_1,\dots,x_6) \wedge \bigwedge_{1\leq i\leq 5} E(x_i,x_{i+1}) \wedge E(x_6,x_1)\Big)$$

GF(C) — Example

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

$R^{\mathcal{B}}$	$1,2,3,u,v,w$		
$E^{\mathcal{B}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u





$$\hom(\mathcal{A},\mathcal{B}) = 0 \quad \text{and} \quad \hom(\mathcal{A},\mathcal{A}) > 0$$

$$\exists^{\geq 1}(x_1,\dots,x_6).\left(R(x_1,\dots,x_6) \wedge \bigwedge_{1\leq i\leq 5} E(x_i,x_{i+1}) \wedge E(x_6,x_1)\right)$$

RCR should also distinguish $\mathcal{A}$ and $\mathcal{B}$!

Relational Color Refinement

Idea: Color *tuples* instead of *elements*:

Relational Color Refinement

Idea: Color *tuples* instead of *elements*:

$$\varrho_{i+1}(\mathbf{a}) \;:=\; \big(\varrho_i(\mathbf{a}), \{\!\{\, (stp(\mathbf{a},\mathbf{b}), \varrho_i(\mathbf{b})) \;:\; \mathbf{a} \cap \mathbf{b} \neq \emptyset \,\}\!\}\big)$$

Relational Color Refinement

Idea: Color *tuples* instead of *elements*:

$$\varrho_{i+1}(\mathbf{a}) \;:=\; \big(\varrho_i(\mathbf{a}), \{\!\{\, (stp(\mathbf{a},\mathbf{b}), \varrho_i(\mathbf{b})) \;:\; \mathbf{a} \cap \mathbf{b} \neq \emptyset \,\}\!\}\big)$$

Still based on "message passing": should be able to encode $\mathcal{A}$ as a graph $G_{\mathcal{A}}$ and use Color Refinement on $G_{\mathcal{A}}$.

Relational Color Refinement

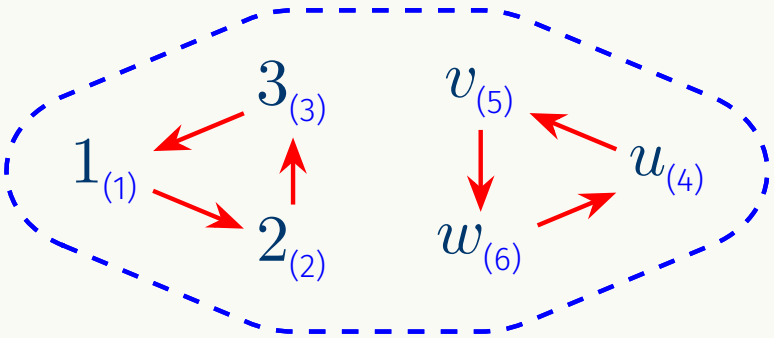
Idea: Color *tuples* instead of *elements*:

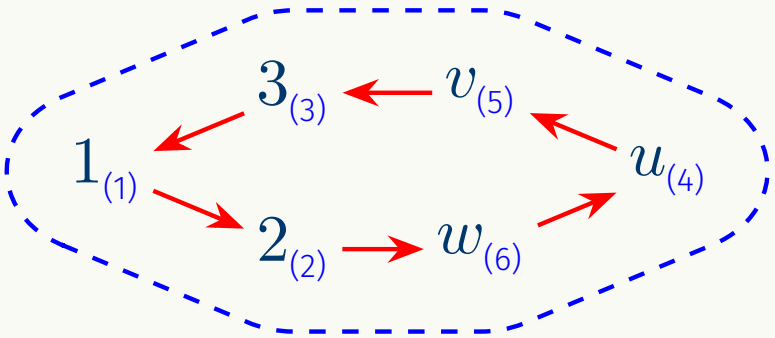
$$\varrho_{i+1}(\mathbf{a}) \;:=\; \big(\varrho_i(\mathbf{a}), \{\!\{\, (stp(\mathbf{a},\mathbf{b}), \varrho_i(\mathbf{b})) \;:\; \mathbf{a}\cap\mathbf{b}\neq\emptyset \,\}\!\}\big)$$

Still based on "message passing": should be able to encode $\mathcal{A}$ as a graph $G_{\mathcal{A}}$ and use Color Refinement on $G_{\mathcal{A}}$.

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

$R^{\mathcal{B}}$	$1,2,3,u,v,w$		
$E^{\mathcal{B}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u



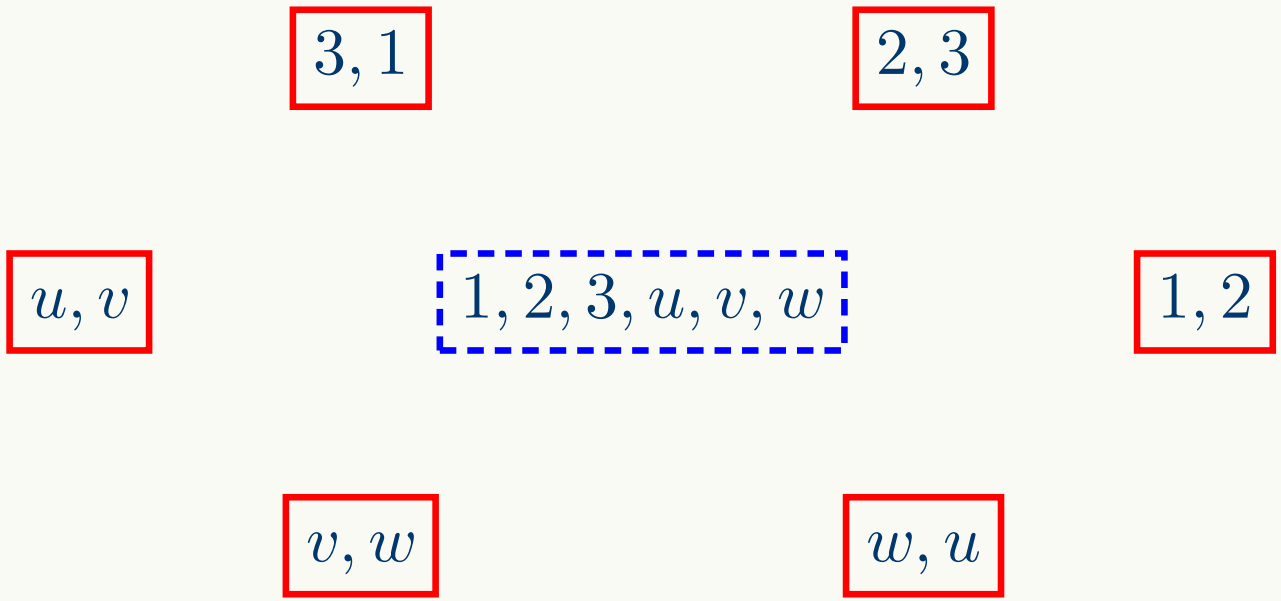


Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

Representing a Structure as a Graph — $G_{\mathcal{A}}$

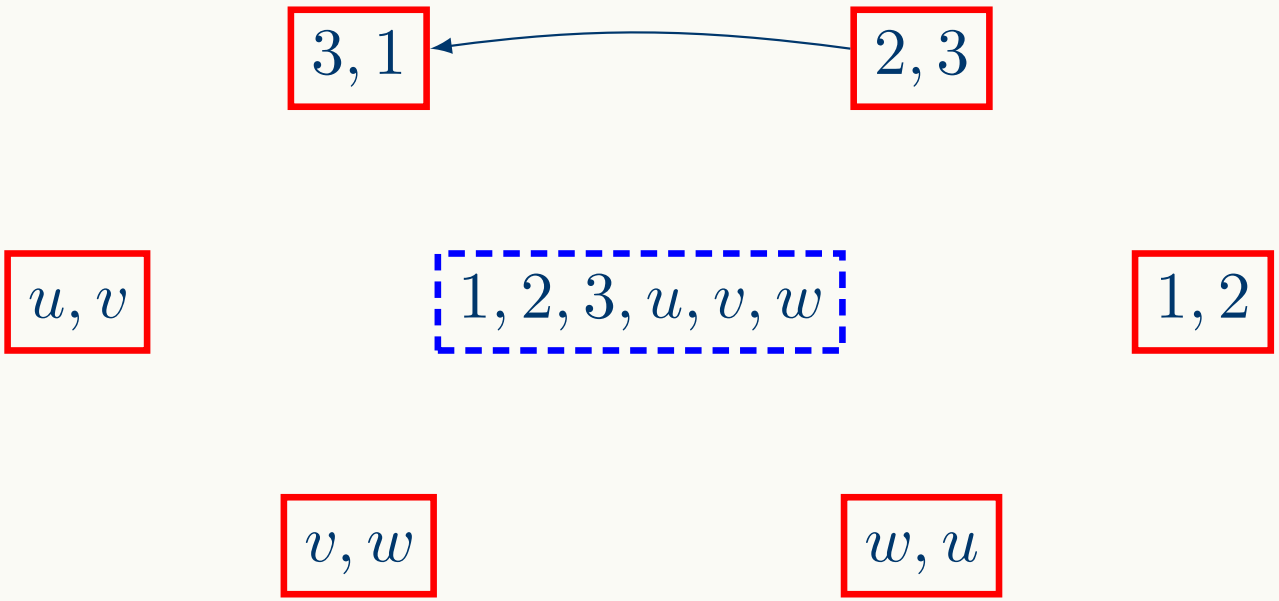
$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u



Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

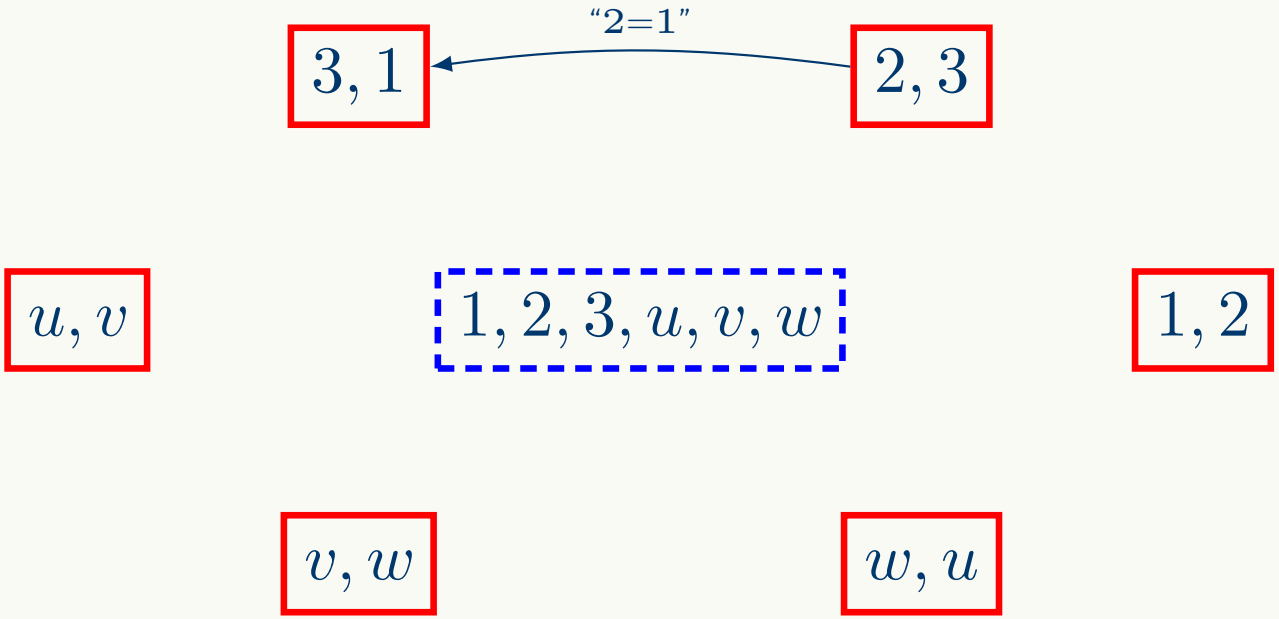
Draw edge if tuples “intersect”



Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

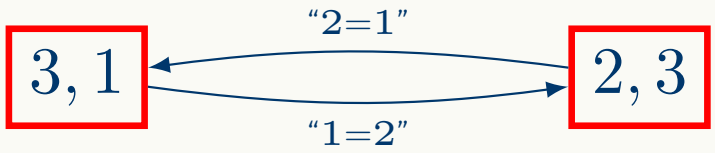
Draw edge if tuples “intersect”



Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

Draw edge if tuples “intersect”



u, v

$1, 2, 3, u, v, w$

$1, 2$

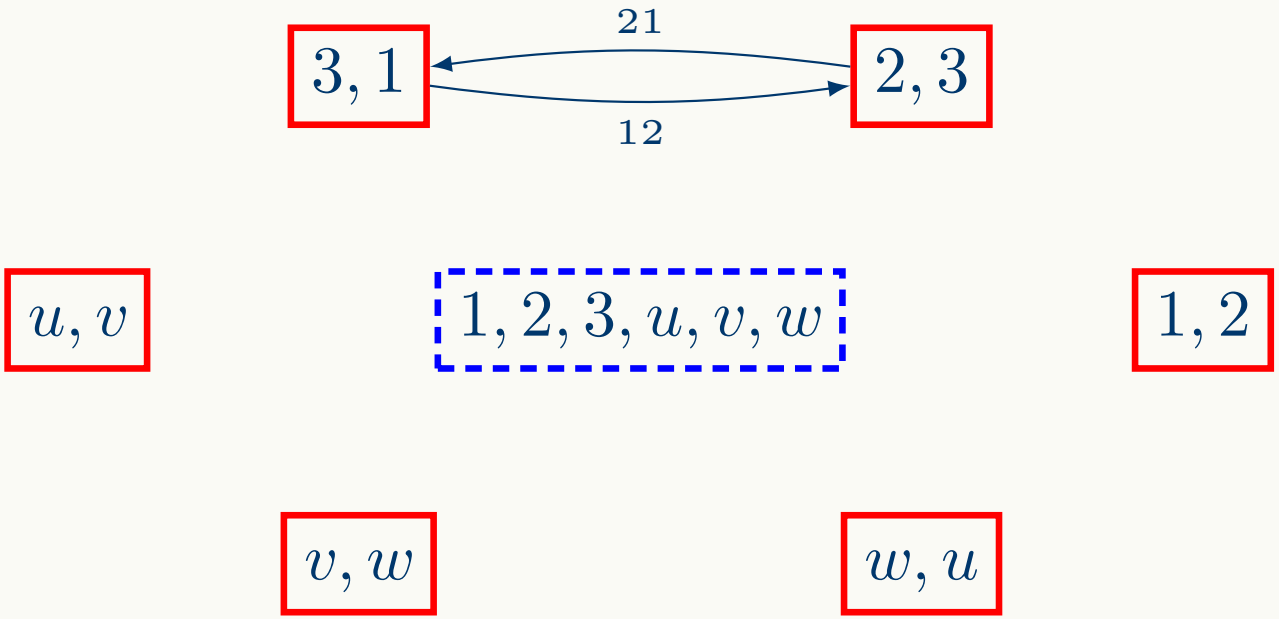
v, w

w, u

Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

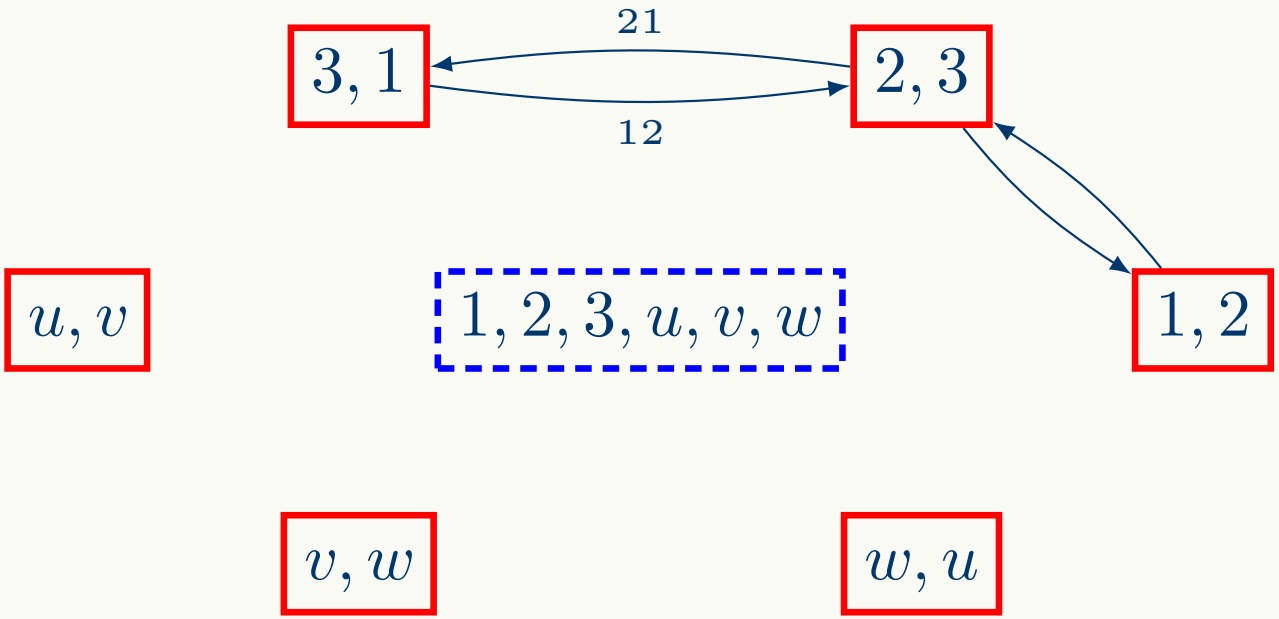
Draw edge if tuples “intersect”



Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

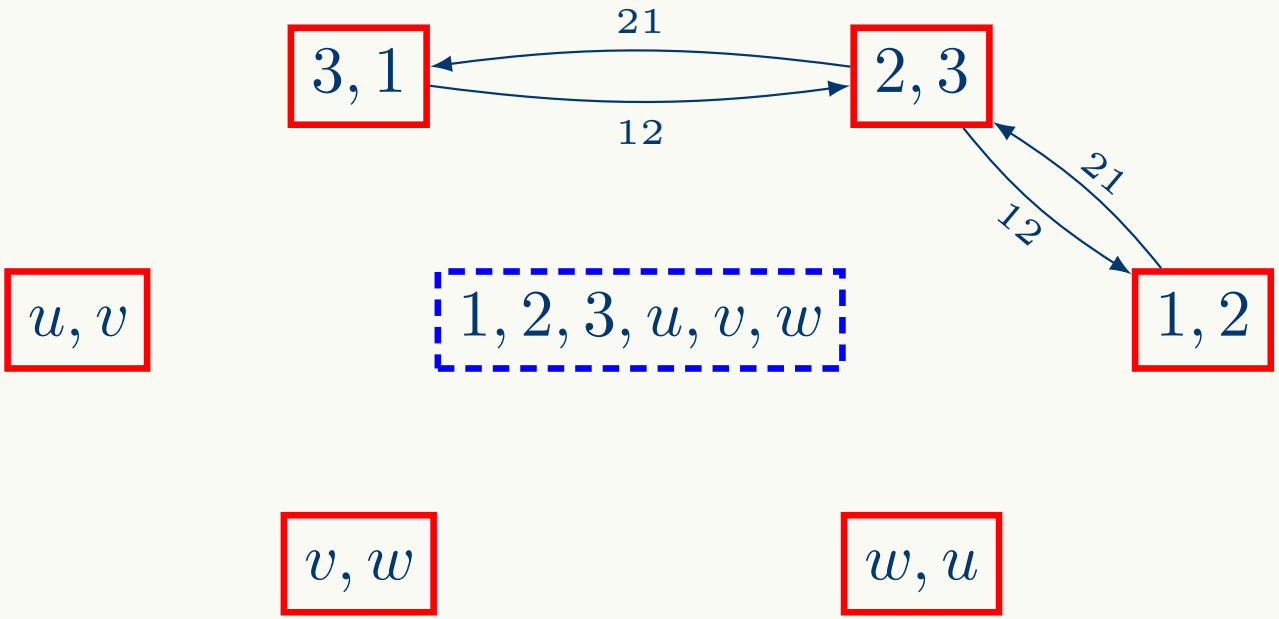
Draw edge if tuples “intersect”



Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

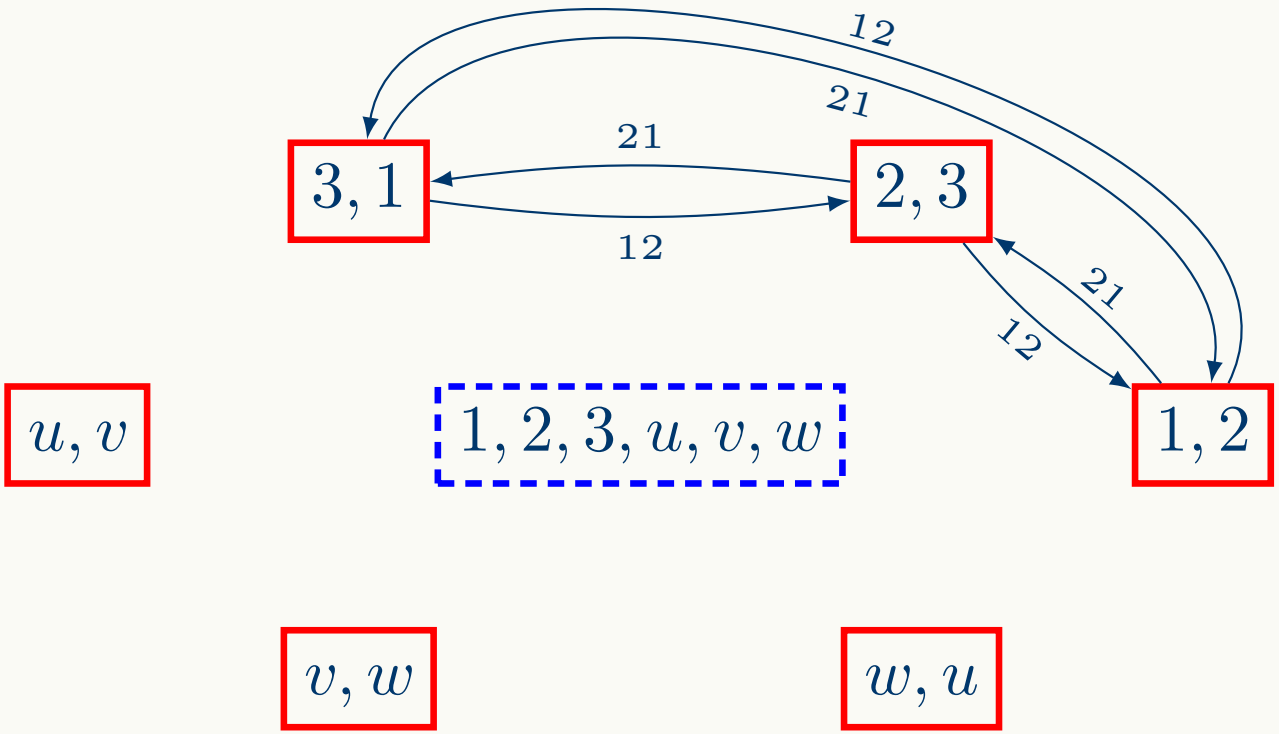
Draw edge if tuples “intersect”



Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

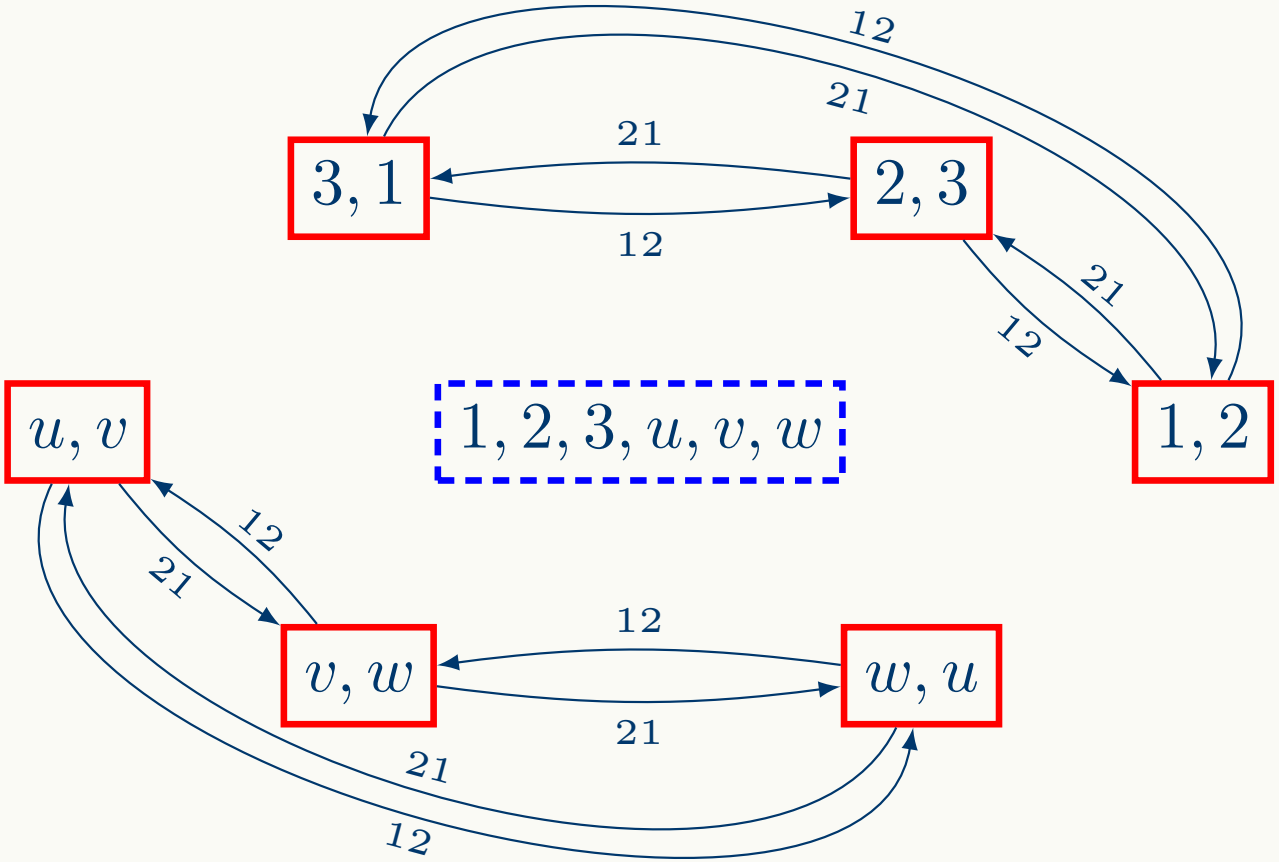
Draw edge if tuples “intersect”



Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

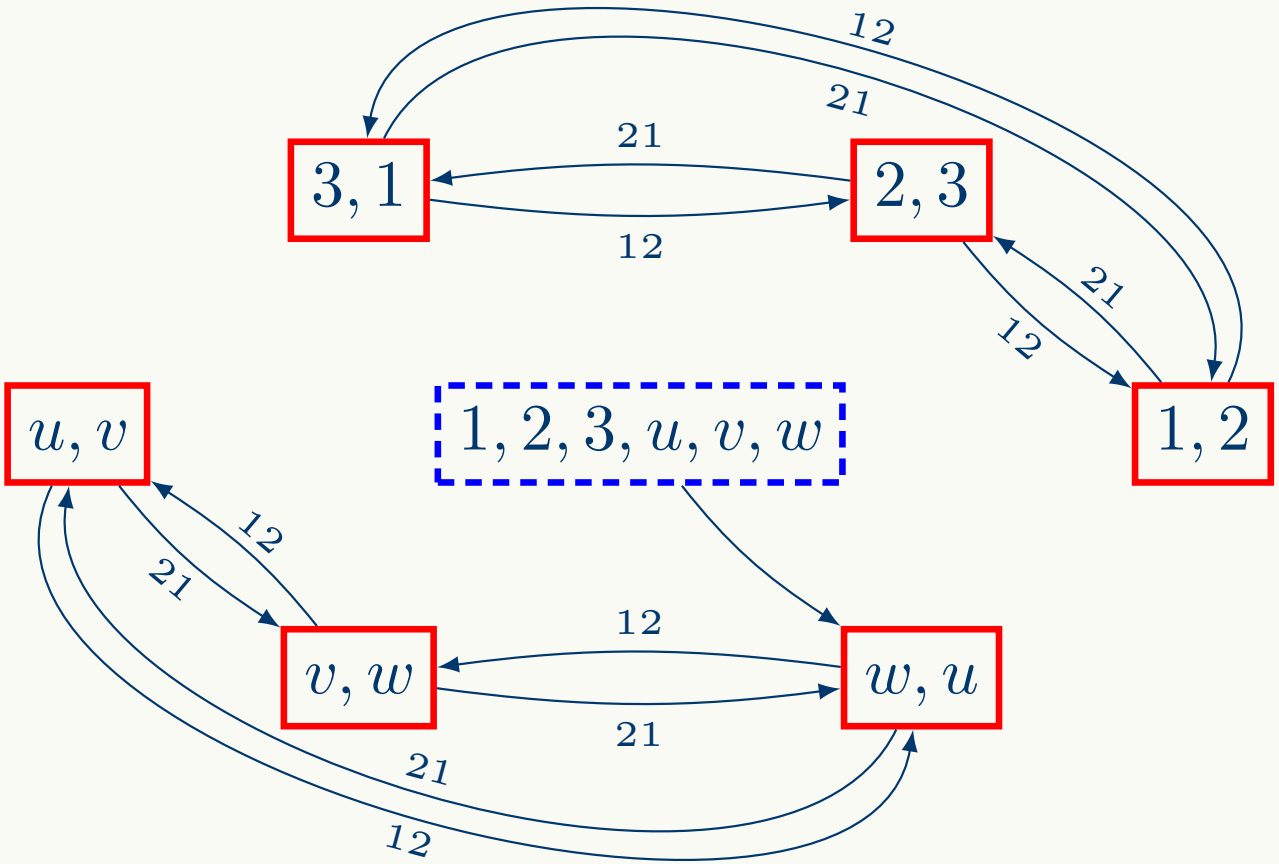
Draw edge if tuples “intersect”



Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

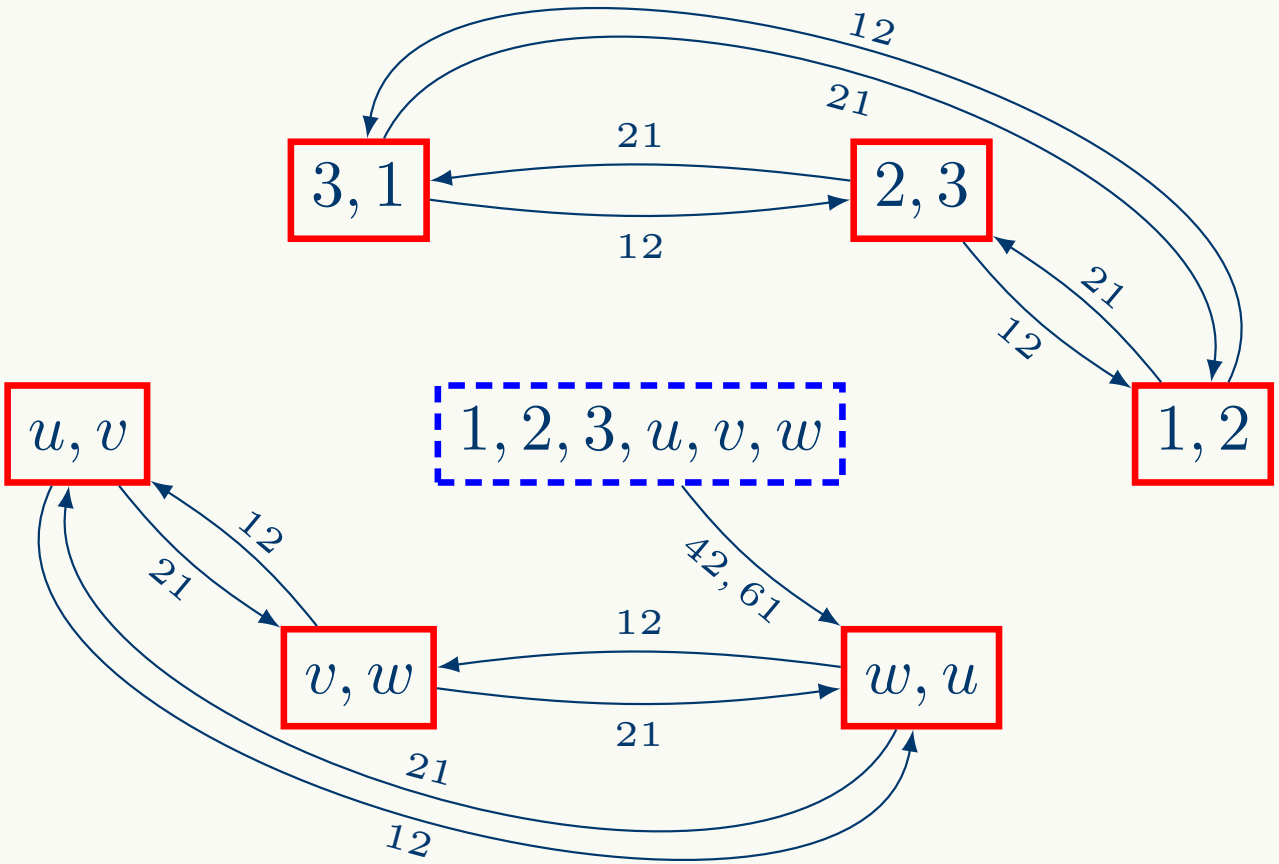
Draw edge if tuples “intersect”



Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

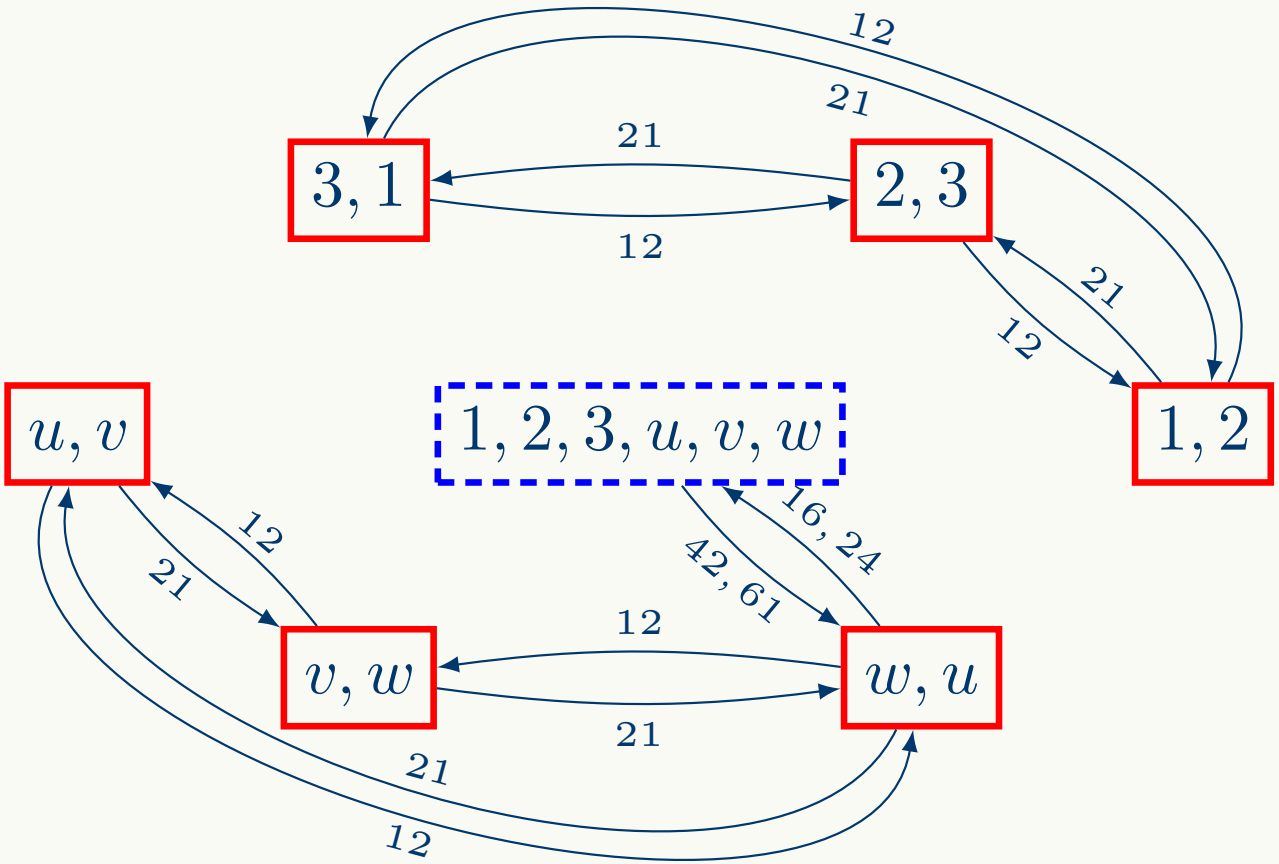
Draw edge if tuples “intersect”



Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

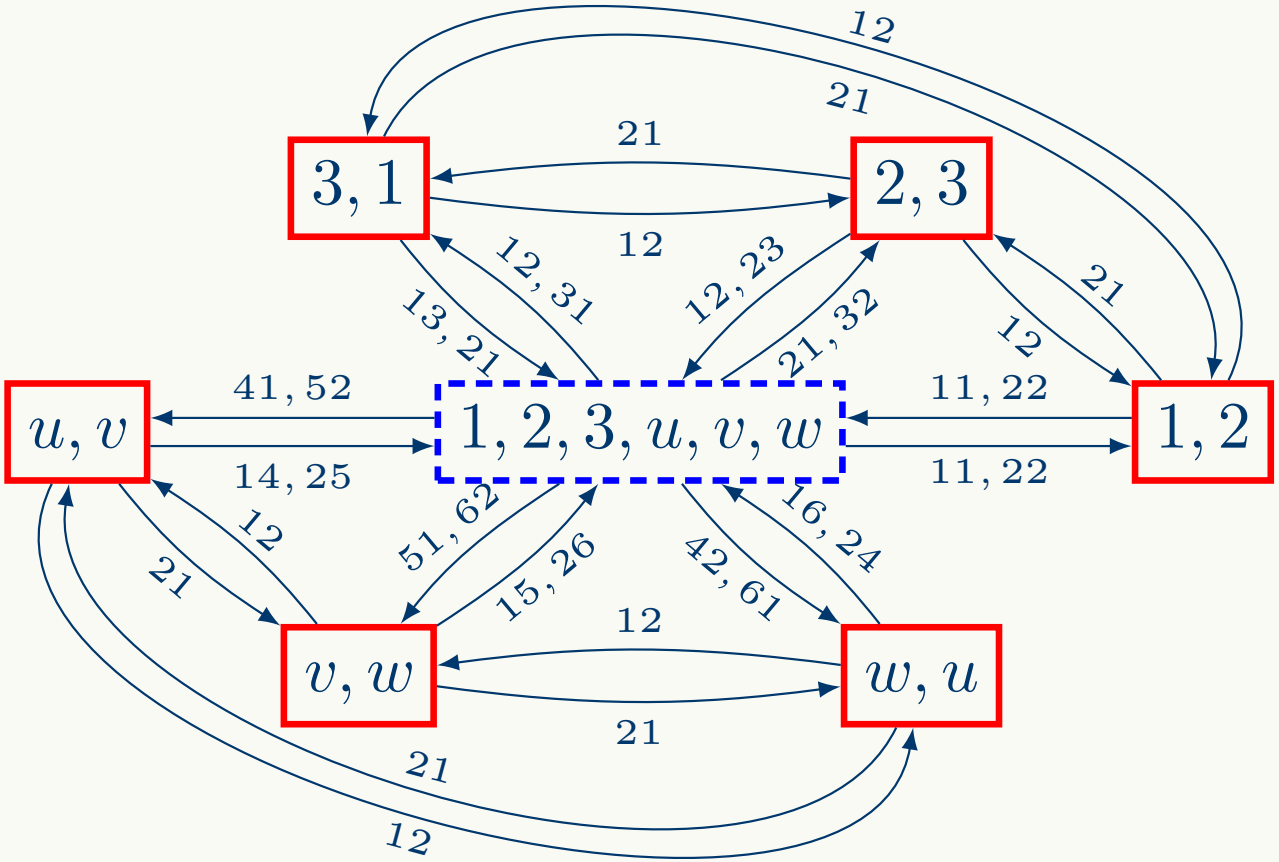
Draw edge if tuples “intersect”



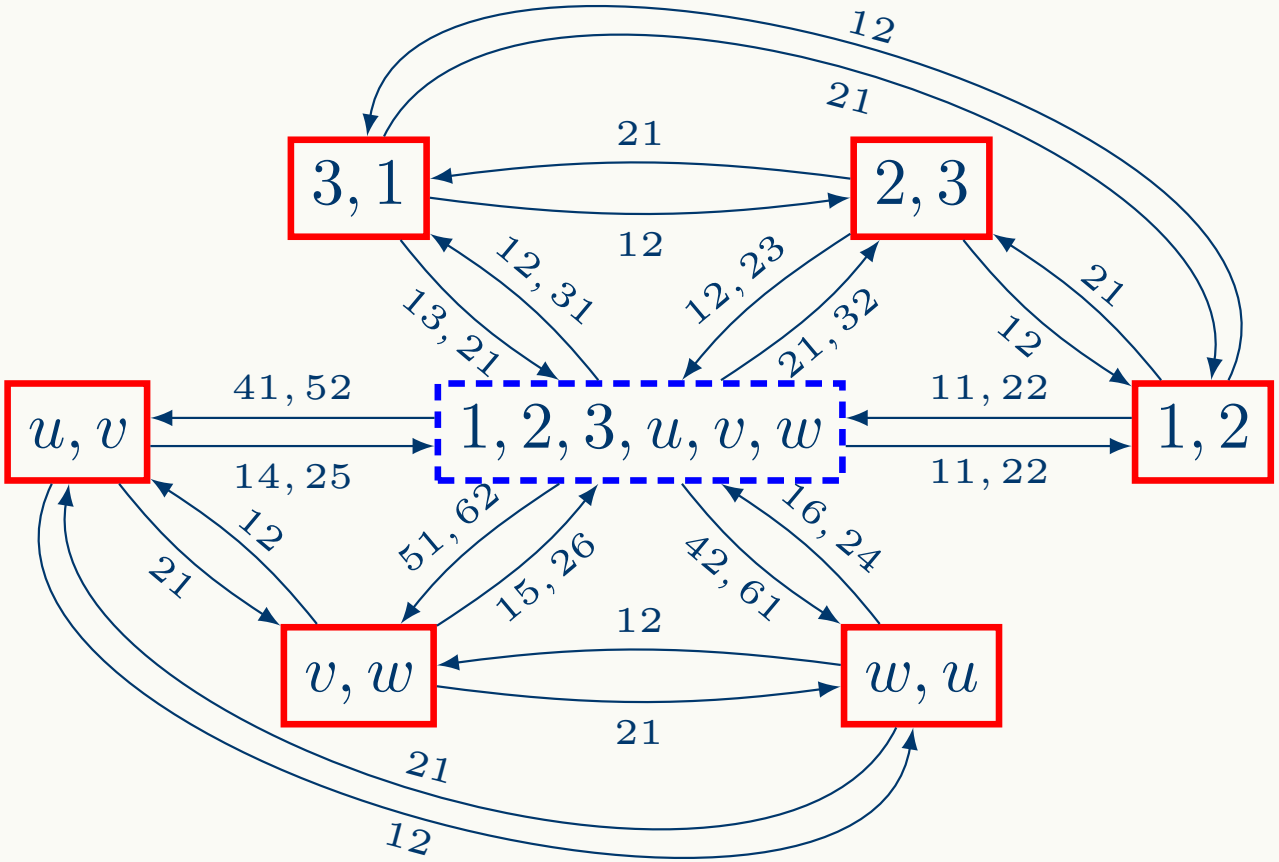
Representing a Structure as a Graph — $G_{\mathcal{A}}$

$R^{\mathcal{A}}$	$1,2,3,u,v,w$		
$E^{\mathcal{A}}$	$1,2$	$2,3$	$3,1$
	u,v	v,w	w,u

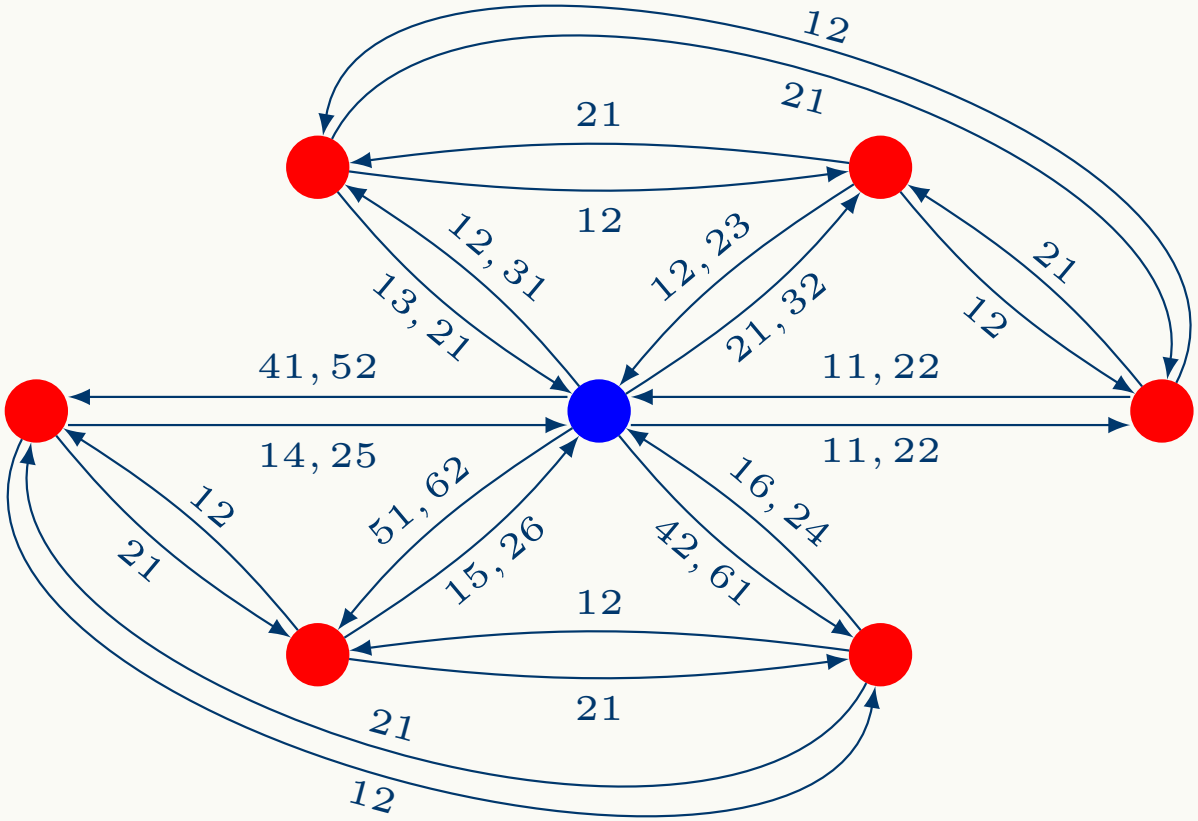
Draw edge if tuples “intersect”



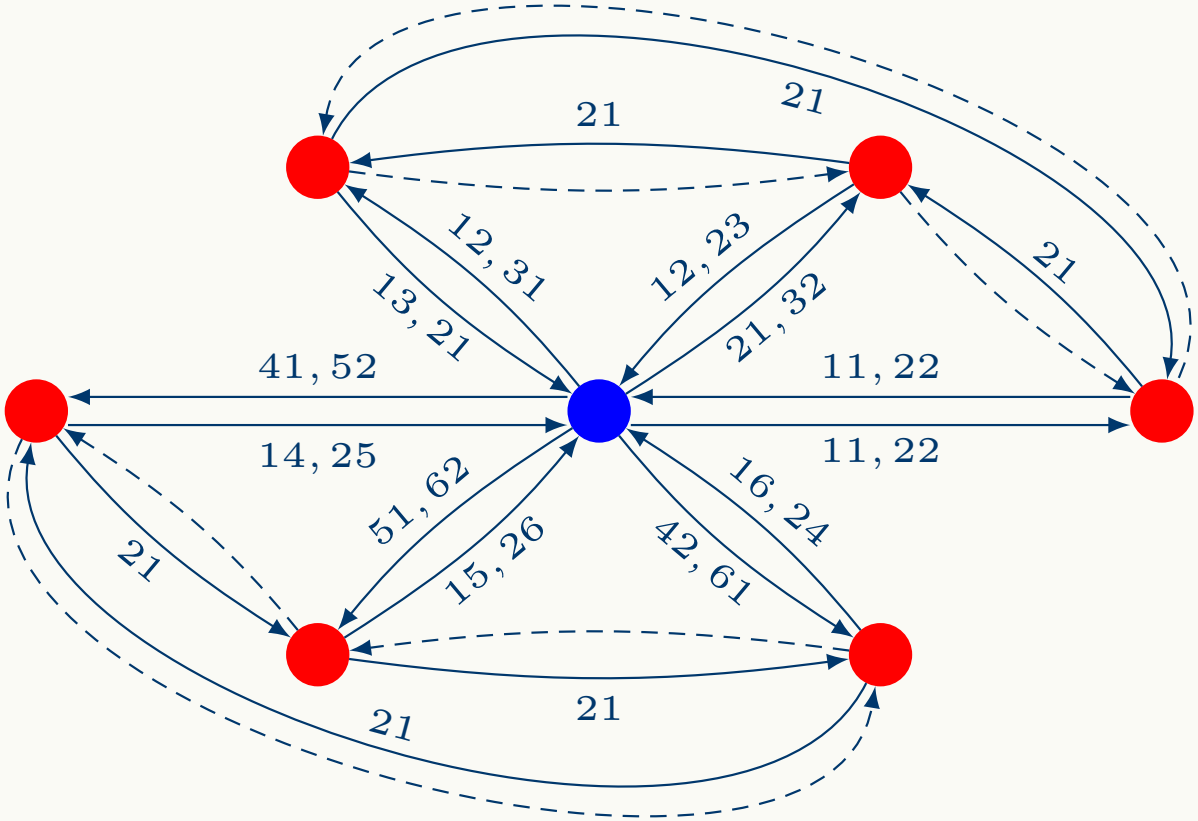
Representing a Structure as a Graph — $G_{\mathcal{A}}$



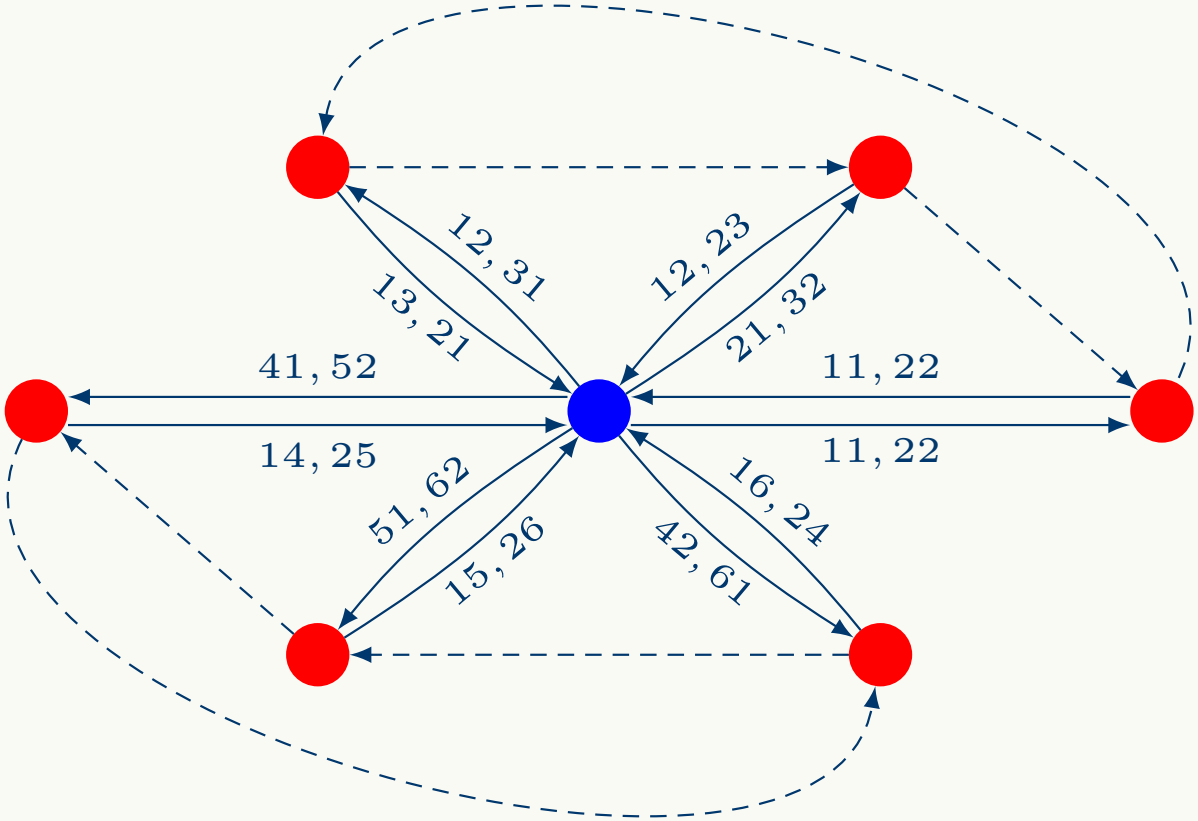
Representing a Structure as a Graph — $G_{\mathcal{A}}$



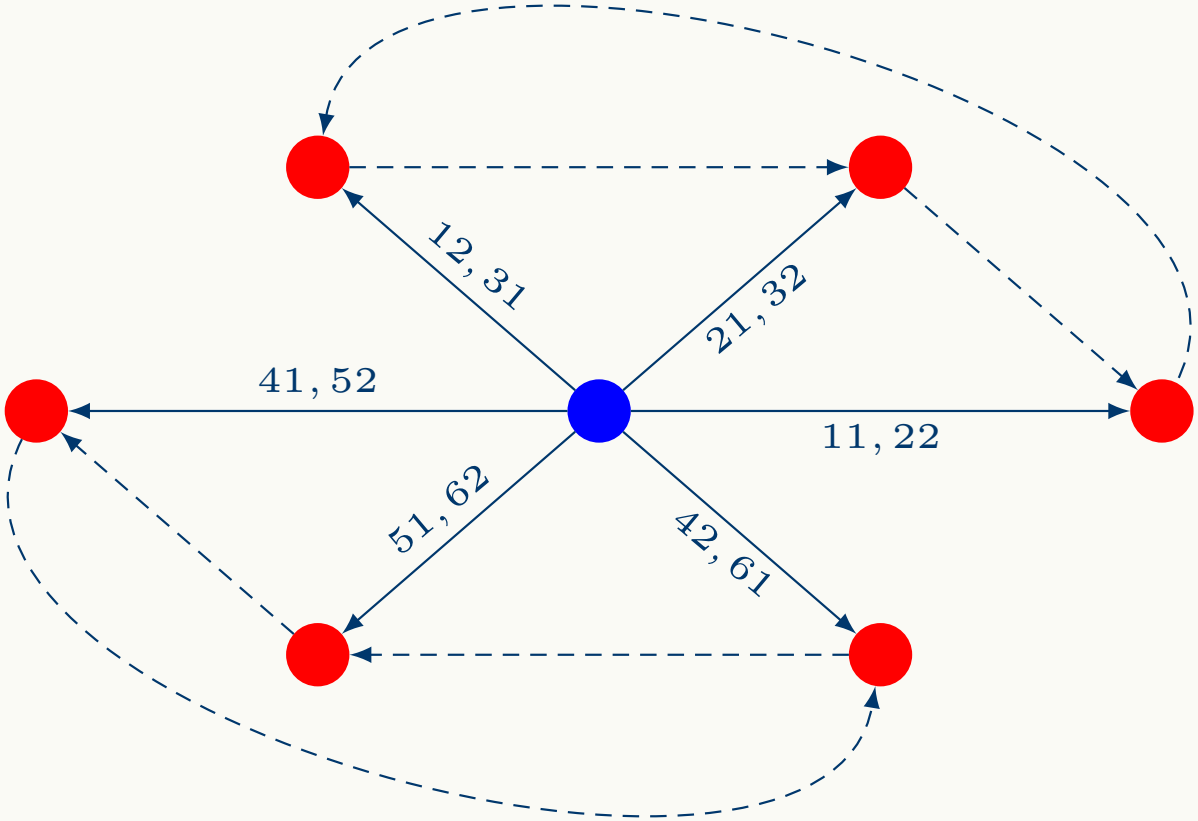
Representing a Structure as a Graph — $G_{\mathcal{A}}$



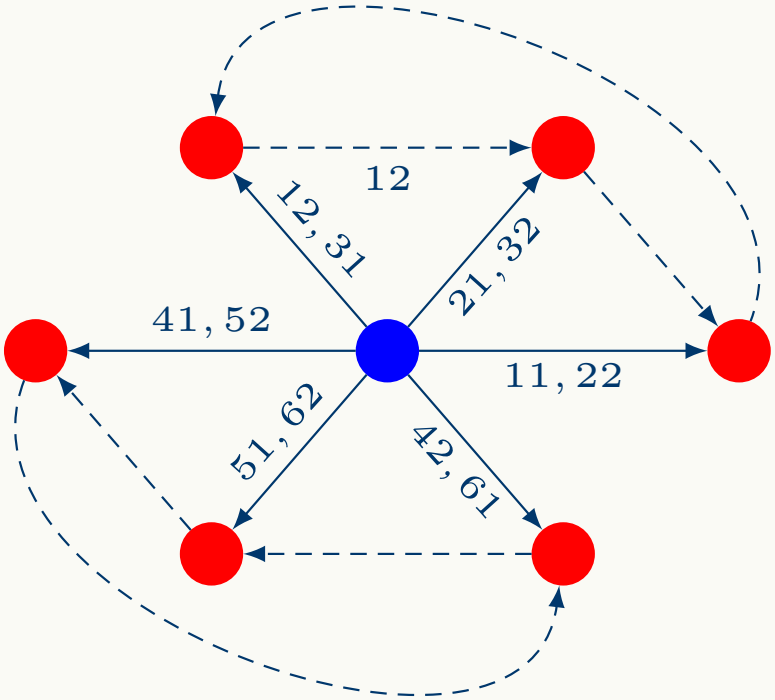
Representing a Structure as a Graph — $G_{\mathcal{A}}$

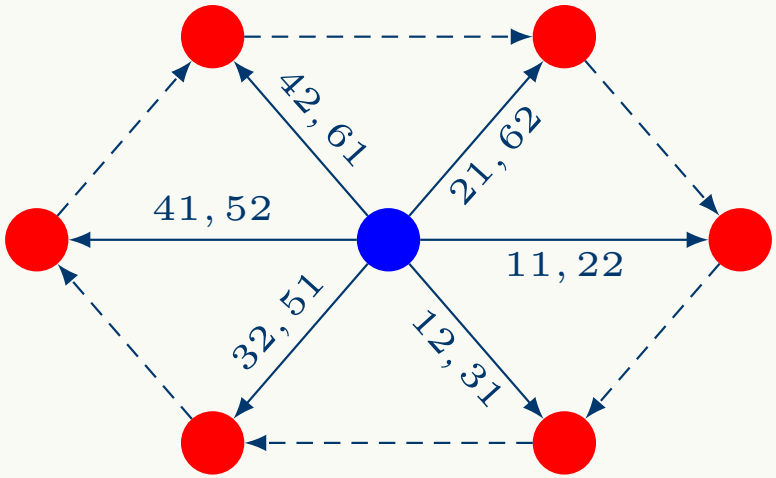


Representing a Structure as a Graph — $G_{\mathcal{A}}$



RCR on $\mathcal{A}, \mathcal{B}$ is equivalent to CR on $G_{\mathcal{A}}, G_{\mathcal{B}}$





This might get large

- RCR can be implemented to run in time

$$\mathcal{O}(\|G_{\mathcal{A}}\| \cdot \log \|G_{\mathcal{A}}\|).$$

This might get large

- RCR can be implemented to run in time

$$\mathcal{O}(\|G_{\mathcal{A}}\| \cdot \log \|G_{\mathcal{A}}\|).$$

- How big can $\|G_{\mathcal{A}}\|$ get?

This might get large

- RCR can be implemented to run in time

$$\mathcal{O}(\|G_{\mathcal{A}}\| \cdot \log \|G_{\mathcal{A}}\|).$$

- How big can $\|G_{\mathcal{A}}\|$ get?
  - Append $\bot$ to every tuple of $\mathcal{A}$ and $G_{\mathcal{A}}$ will be a complete graph.

This might get large

- RCR can be implemented to run in time

$$\mathcal{O}(\|G_{\mathcal{A}}\| \cdot \log \|G_{\mathcal{A}}\|).$$

- How big can $\|G_{\mathcal{A}}\|$ get?
  - Append $\perp$ to every tuple of $\mathcal{A}$ and $G_{\mathcal{A}}$ will be a complete graph.
  - $\mathcal{O}(\|\mathcal{A}\|^2)$ edges in the worst case.

This might get large

- RCR can be implemented to run in time

$$\mathcal{O}(\|G_{\mathcal{A}}\| \cdot \log \|G_{\mathcal{A}}\|).$$

- How big can $\|G_{\mathcal{A}}\|$ get?
  - Append $\perp$ to every tuple of $\mathcal{A}$ and $G_{\mathcal{A}}$ will be a complete graph.
  - $\mathcal{O}(\|\mathcal{A}\|^2)$ edges in the worst case.
- So RCR runs in time $\mathcal{O}(\|\mathcal{A}\|^2 \cdot \log \|\mathcal{A}\|)$ in the worst case?

This might get large

- RCR can be implemented to run in time

$$\mathcal{O}(\|G_{\mathcal{A}}\| \cdot \log \|G_{\mathcal{A}}\|).$$

- How big can $\|G_{\mathcal{A}}\|$ get?
  - Append $\perp$ to every tuple of $\mathcal{A}$ and $G_{\mathcal{A}}$ will be a complete graph.
  - $\mathcal{O}(\|\mathcal{A}\|^2)$ edges in the worst case.
- So RCR runs in time $\mathcal{O}(\|\mathcal{A}\|^2 \cdot \log \|\mathcal{A}\|)$ in the worst case?
- Didn't I promise $\mathcal{O}(\|\mathcal{A}\| \cdot \log \|\mathcal{A}\|)$?!

Runtime of Relational Color Refinement

Use a different encoding $H_{\mathcal{A}}$ of $\mathcal{A}$ instead of $G_{\mathcal{A}}$ where

$$\|H_{\mathcal{A}}\| \in \mathcal{O}(\|\mathcal{A}\|).$$

Runtime of Relational Color Refinement

Use a different encoding $H_{\mathcal{A}}$ of $\mathcal{A}$ instead of $G_{\mathcal{A}}$ where

$$\|H_{\mathcal{A}}\| \in \mathcal{O}(\|\mathcal{A}\|).$$

But how?

Runtime of Relational Color Refinement

Use a different encoding $H_{\mathcal{A}}$ of $\mathcal{A}$ instead of $G_{\mathcal{A}}$ where

$$\|H_{\mathcal{A}}\| \in \mathcal{O}(\|\mathcal{A}\|).$$

But how?

- Problem with $G_{\mathcal{A}}$: tuples sharing an element form a clique.

Runtime of Relational Color Refinement

Use a different encoding $H_{\mathcal{A}}$ of $\mathcal{A}$ instead of $G_{\mathcal{A}}$ where

$$\|H_{\mathcal{A}}\| \in \mathcal{O}(\|\mathcal{A}\|).$$

But how?

- Problem with $G_{\mathcal{A}}$: tuples sharing an element form a clique.
- Solution: Resolve cliques via “intermediate” vertices.

Runtime of Relational Color Refinement

Use a different encoding $H_{\mathcal{A}}$ of $\mathcal{A}$ instead of $G_{\mathcal{A}}$ where

$$\|H_{\mathcal{A}}\| \in \mathcal{O}(\|\mathcal{A}\|).$$

But how?

- Problem with $G_{\mathcal{A}}$: tuples sharing an element form a clique.
- Solution: Resolve cliques via “intermediate” vertices.
- The number of cliques a tuple can be involved in depends on its arity, which is a constant.

Runtime of Relational Color Refinement

Use a different encoding $H_{\mathcal{A}}$ of $\mathcal{A}$ instead of $G_{\mathcal{A}}$ where

$$\|H_{\mathcal{A}}\| \in \mathcal{O}(\|\mathcal{A}\|).$$

But how?

- Problem with $G_{\mathcal{A}}$: tuples sharing an element form a clique.
- Solution: Resolve cliques via “intermediate” vertices.
- The number of cliques a tuple can be involved in depends on its arity, which is a constant.
- The proof that CR on $H_{\mathcal{A}}$ is equivalent to RCR on $\mathcal{A}$ is quite involved.

Isn’t this way too complicated?!

Let us look at “straightforward” approaches.

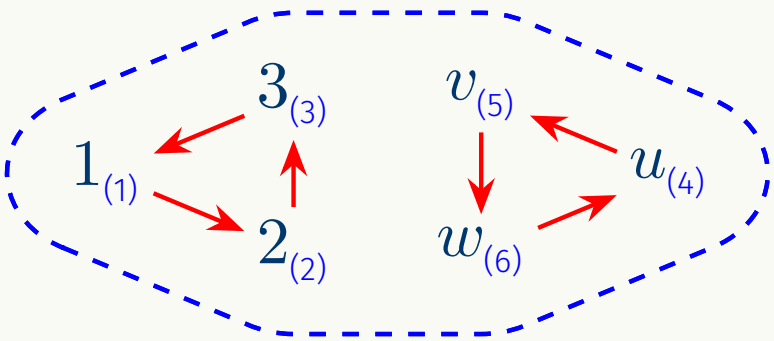
Approach 1:

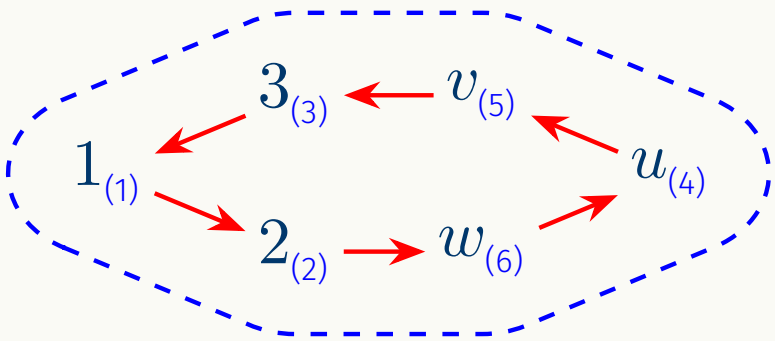
Run Color Refinement on the Gaifman graph.

Approach 2:

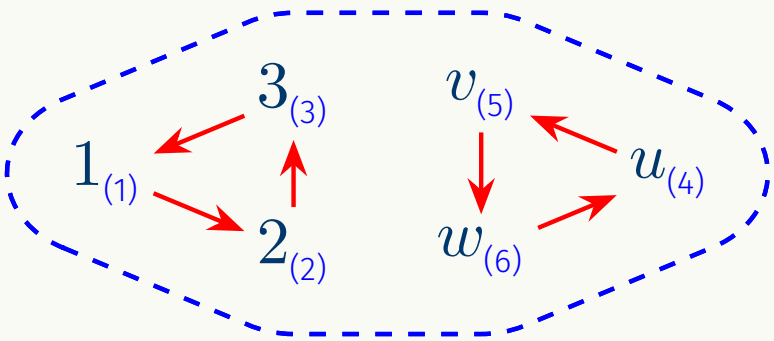
Run Color Refinement on the incidence graph.

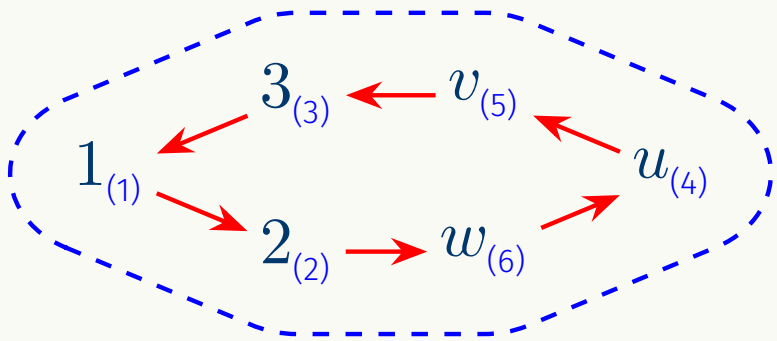
Consider our Examples...

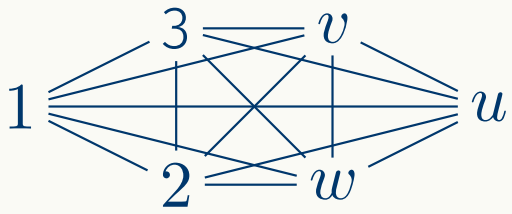


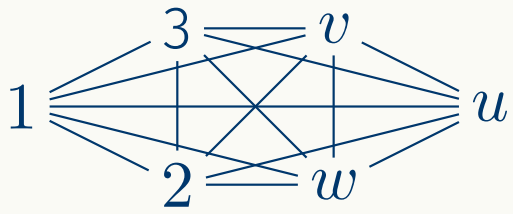


Consider our Examples...

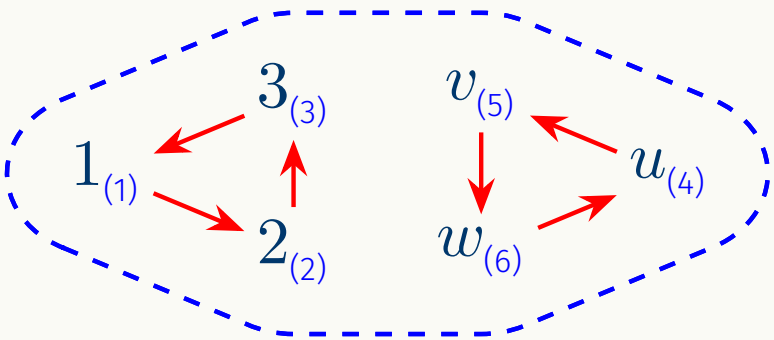


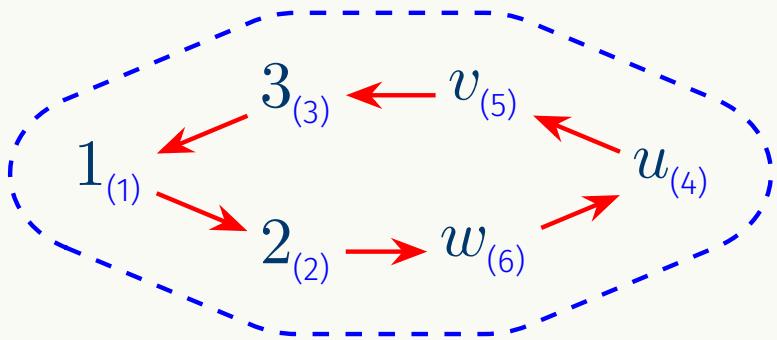


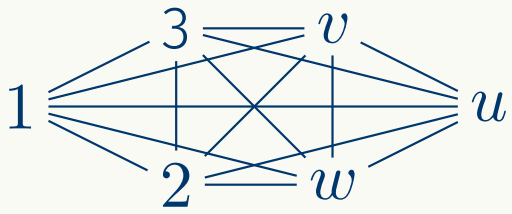


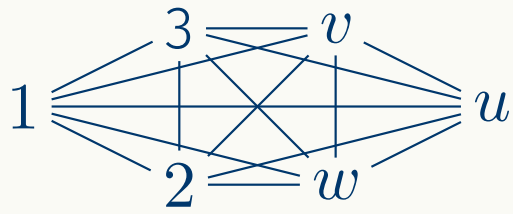


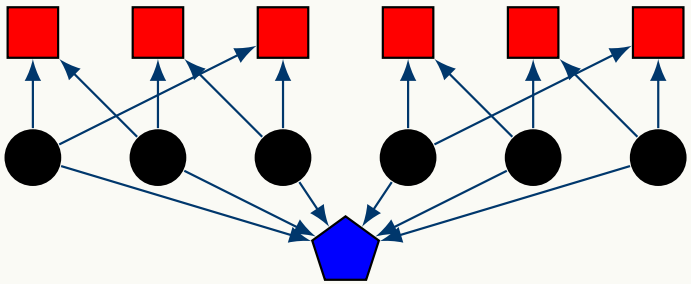
Consider our Examples...

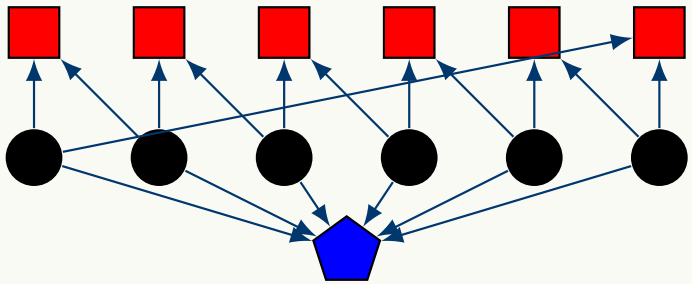




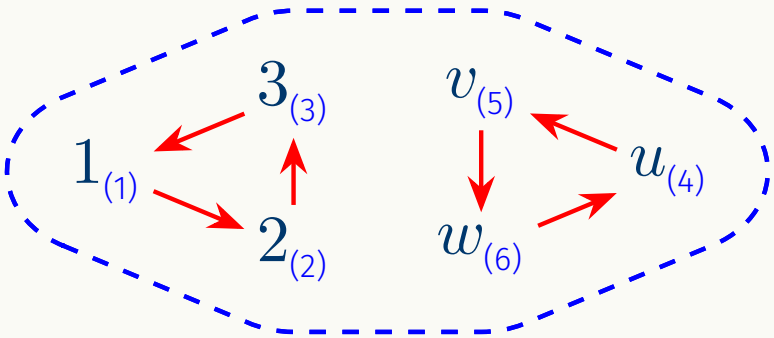


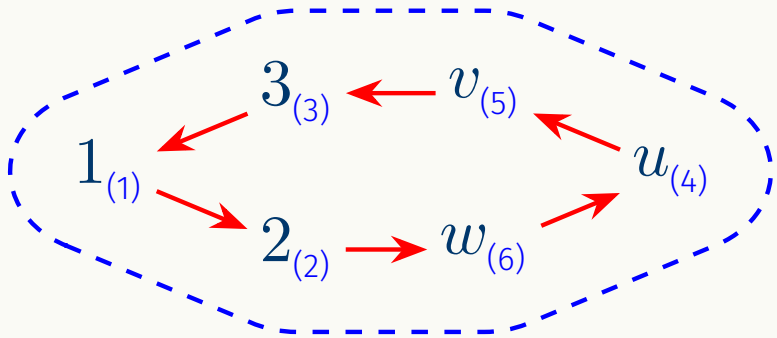


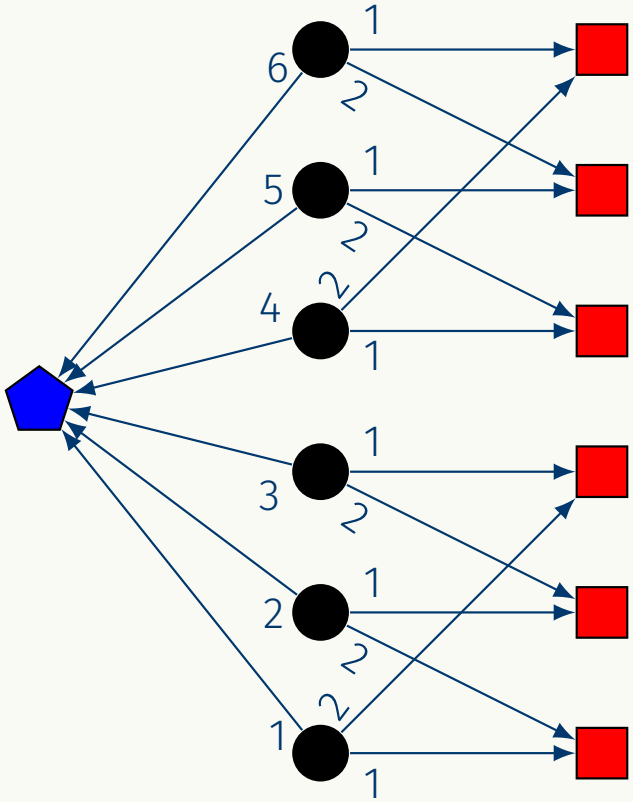


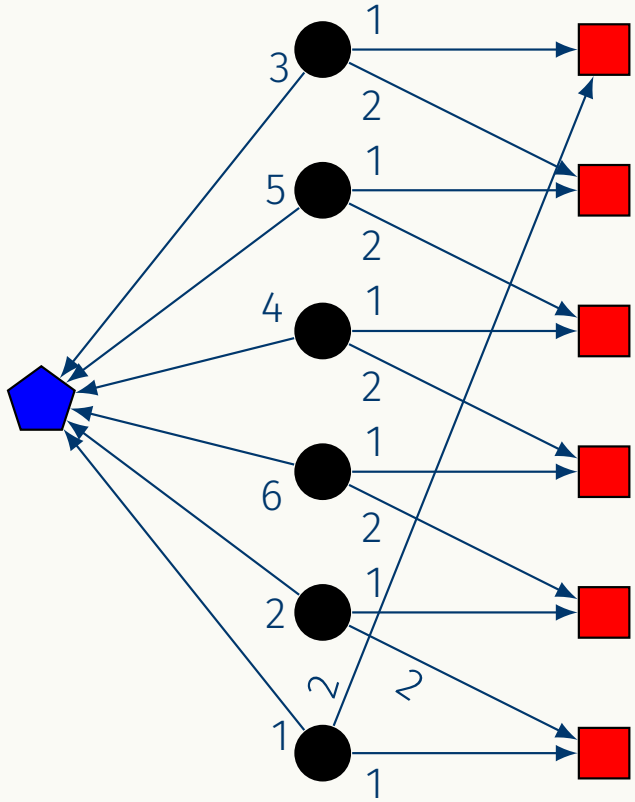


Press more Information into the Representations

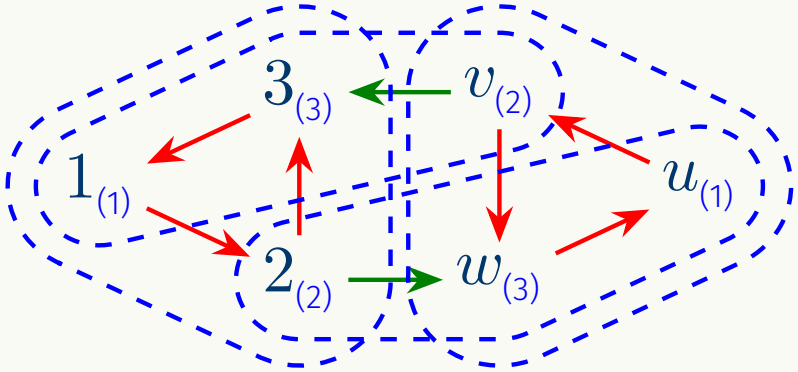


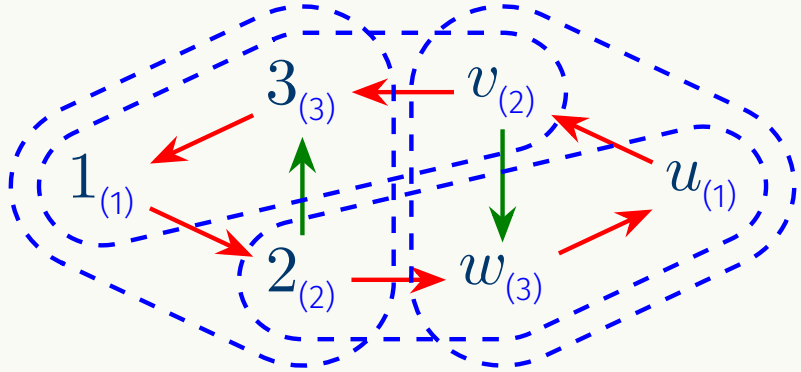




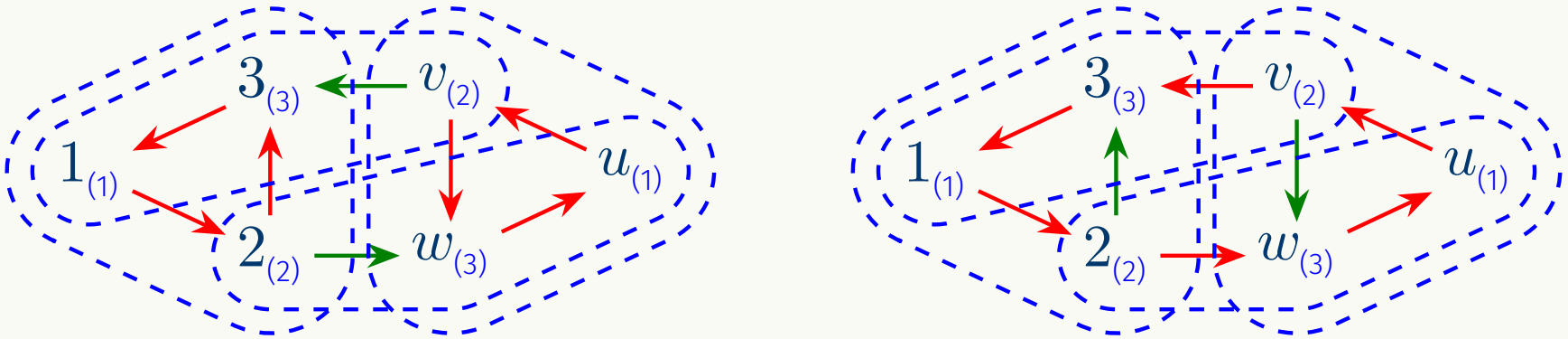


Another Counter-Example



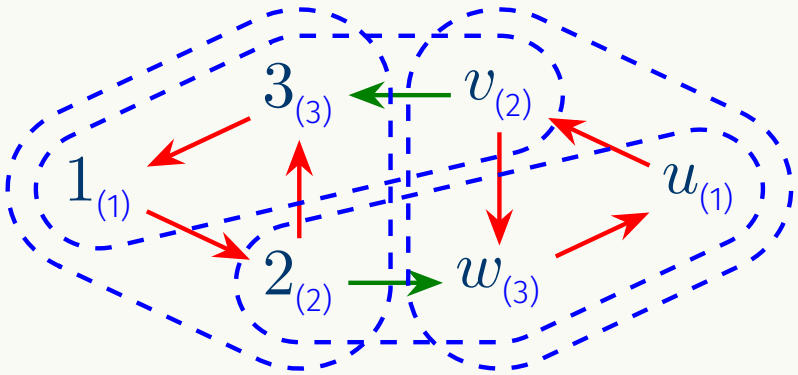


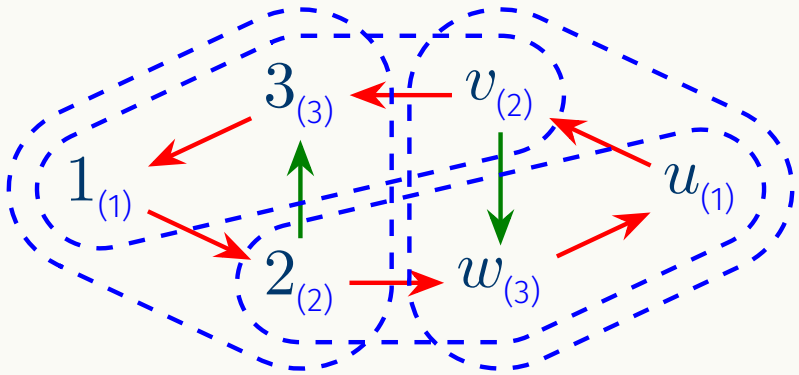
Another Counter-Example



CR does not distinguish their incidence graphs, even with labeled edges.

Another Counter-Example





CR does not distinguish their incidence graphs, even with labeled edges.

$$\exists^{\geq 1}(x_1, x_2, x_3)\,.\,\big(\,R(x_1, x_2, x_3) \wedge E(x_1, x_2) \;\wedge\; F(x_2, x_3) \wedge E(x_3, x_1)\,\big)$$

Bottom line:

These representations do not seem to capture homomorphisms from acyclic structures.

They somehow lose crucial information.

Main Result & Outlook

Theorem (Scheidt, Schweikardt 2024)

The following are equivalent:

- *Relational Color Refinement distinguishes $\mathcal{A}$ and $\mathcal{B}$.*
- *There is an acyclic $\mathcal{C}$ s.t. $\hom(\mathcal{C}, \mathcal{A}) \neq \hom(\mathcal{C}, \mathcal{B})$.*
- *There is a $\varphi \in \mathrm{GF}(\mathrm{C})$ s.t. $\mathcal{A} \models \varphi$, and $\mathcal{B} \not\models \varphi$.*
- *Spoiler has a win. strategy for the Guarded-Game on $\mathcal{A}$, $\mathcal{B}$.*

RCR can be implemented to run in time $\mathcal{O}(\|\mathcal{A}\| \cdot \log \|\mathcal{A}\|)$.

Outlook: lift results on graphs to relational structures?

- CR can be used to speed up the evaluation of fc-ACQs on binary schemas (Riveros, Scheidt, Schweikardt 2024).
- CR is related to Graph Neural Networks.
- k-dimensional Relational Weisfeiler-Leman?

Further Reading

📝 Benjamin Scheidt, Nicole Schweikardt.
“Color Refinement for Relational Structures”.
2024. DOI: 10.48550/ARXIV.2407.16022.

📝 Benjamin Scheidt, Nicole Schweikardt.
“Counting Homomorphisms from Hypergraphs of Bounded Generalised Hypertree Width: A Logical Characterisation”.
2023. DOI: 10.4230/LIPIcs.MFCS.2023.79.

📝 Zdeněk Dvořák.
“On recognizing graphs by numbers of homomorphisms”.
2010. DOI: 10.1002/jgt.20461.

Other References

📝 Cristian Riveros, Benjamin Scheidt, Nicole Schweikardt.
“Using Color Refinement to Boost Enumeration and Counting for Acyclic CQs of Binary Schemas”.
2024.

📝 Holger Dell, Martin Grohe, Gaurav Rattan.
“Lovász Meets Weisfeiler and Leman”.
2018. DOI: 10.4230/LIPIcs.ICALP.2018.40.

📝 Serge Abiteboul, Richard Hull, Victor Vianu.
Foundations of Databases.
1995.

📝 Neil Immerman, Eric Lander.
“Describing Graphs: A First-Order Approach to Graph Canonization”.
1990. DOI: 10.1007/978-1-4612-4478-3_5.