IEEE ICNP 2010 Oct. 5 - 8

Assigning AS Relationships to Satisfy the Gao-Rexford Conditions

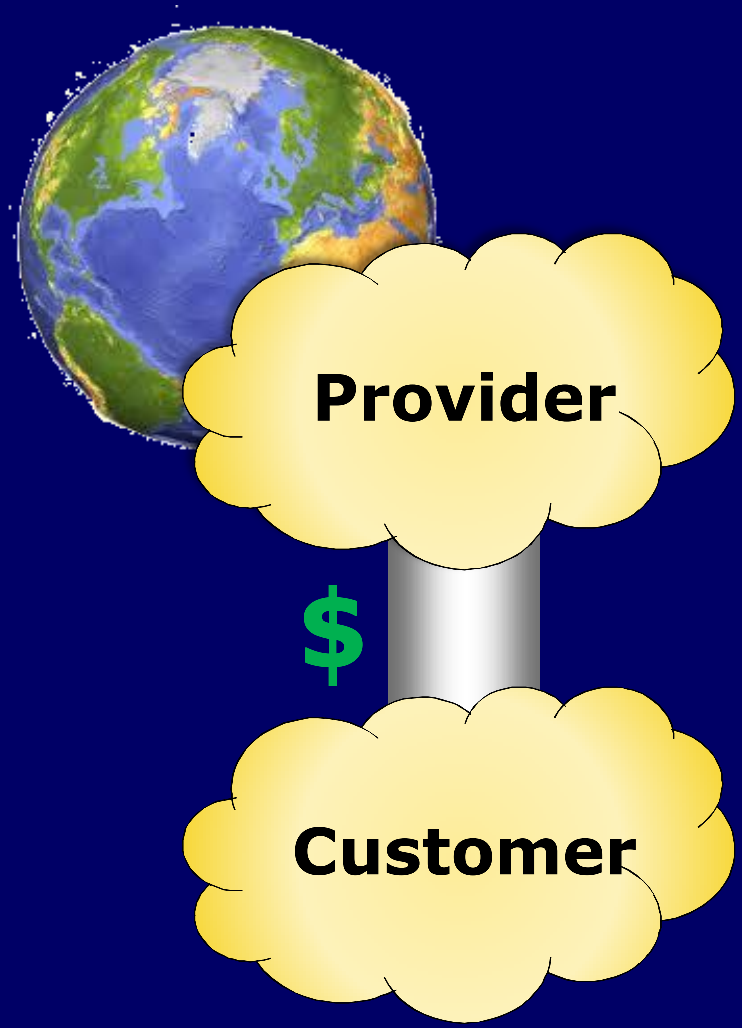
Luca Cittadini (Roma Tre University)
Giuseppe Di Battista (Roma Tre University)
Thomas Erlebach (University of Leicester)
Maurizio Patrignani (Roma Tre University)
Massimo Rimondini (Roma Tre University)

AS Relationships

AS Relationships

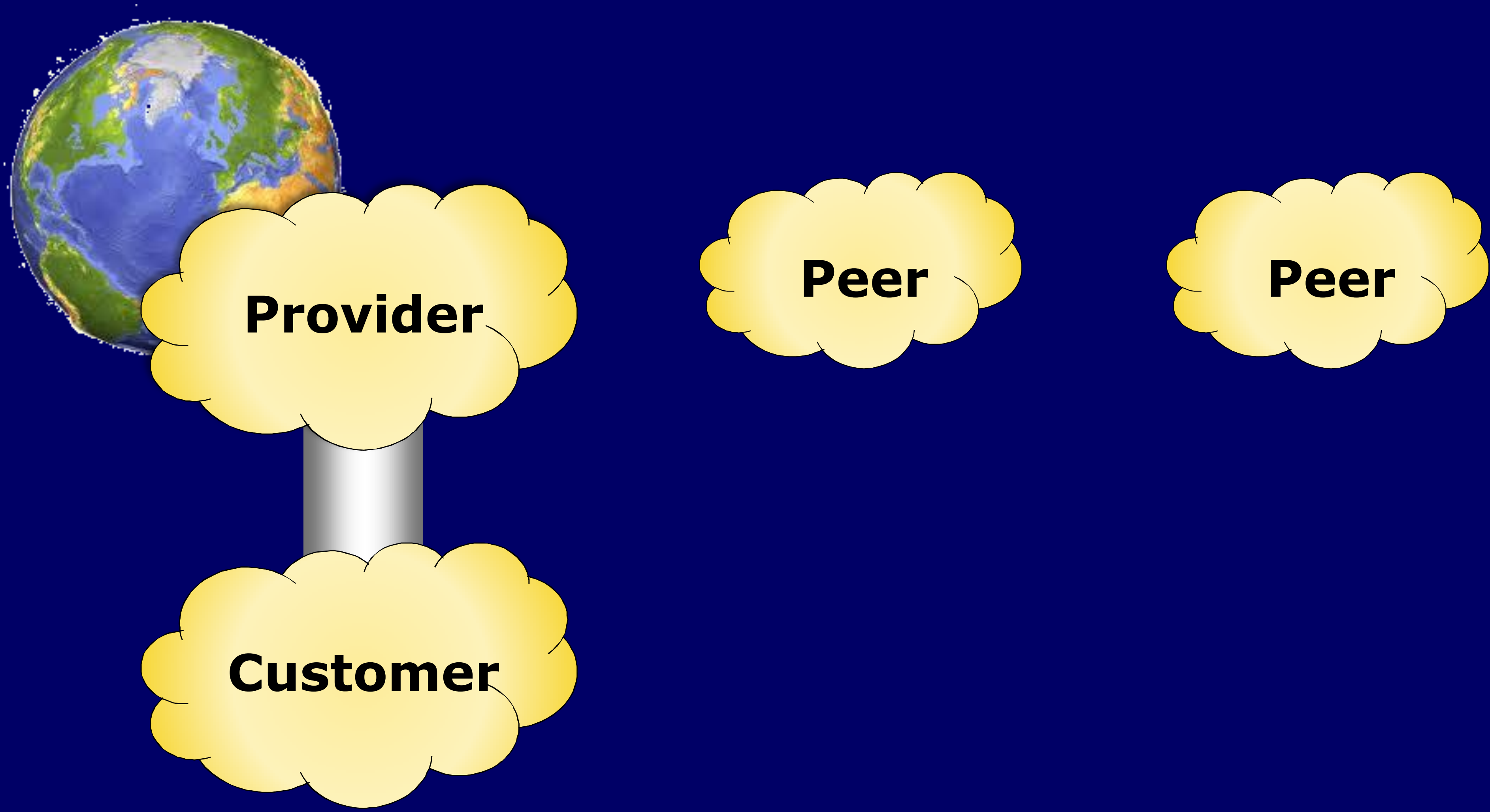
AS Relationships

AS Relationships

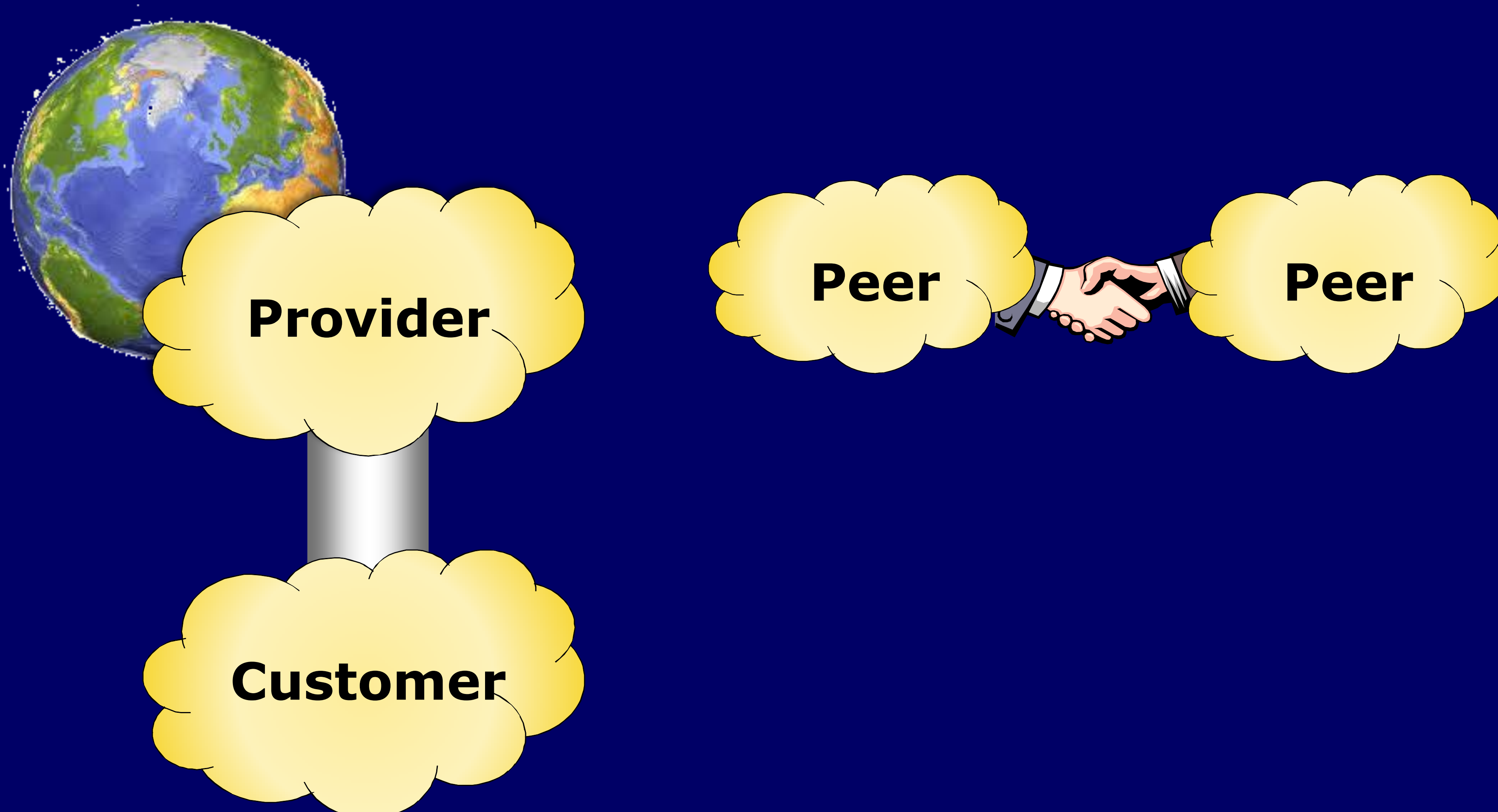


AS Relationships

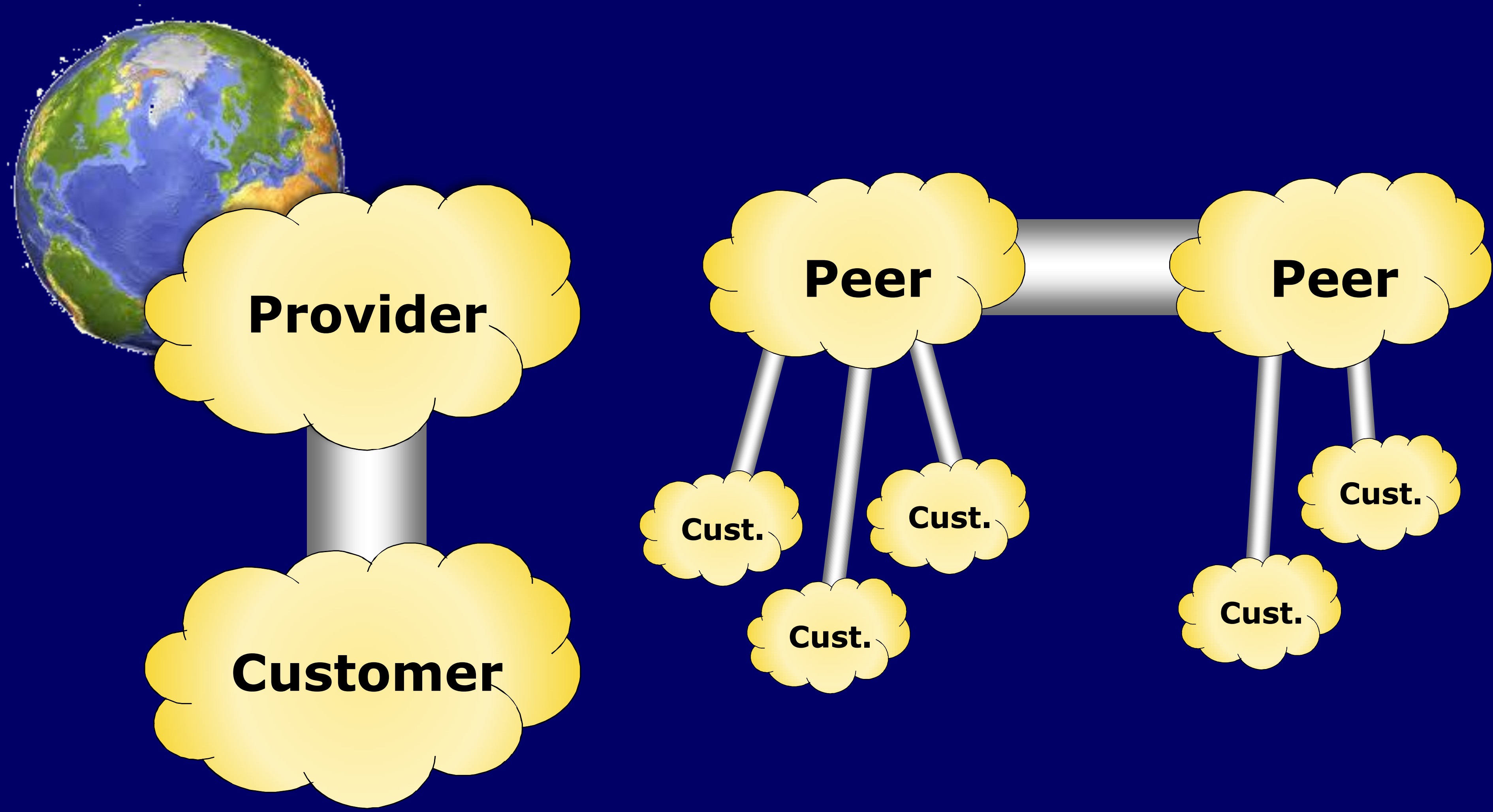
AS Relationships



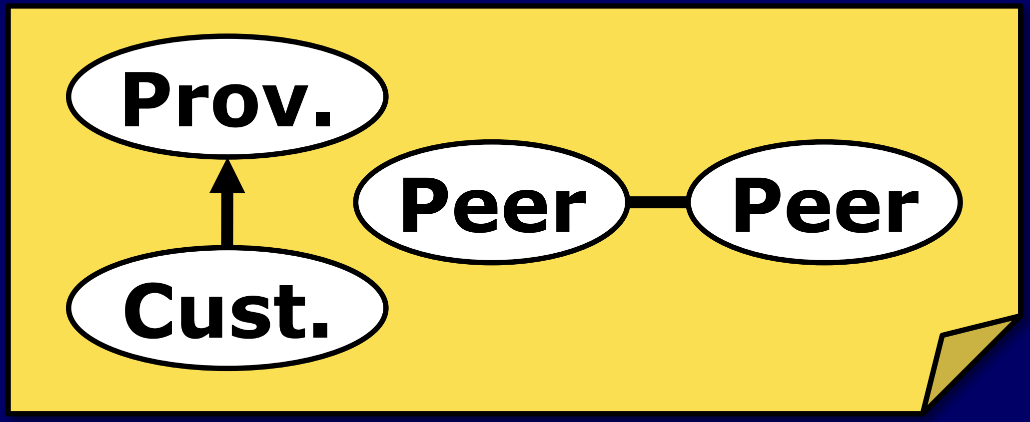
AS Relationships



AS Relationships

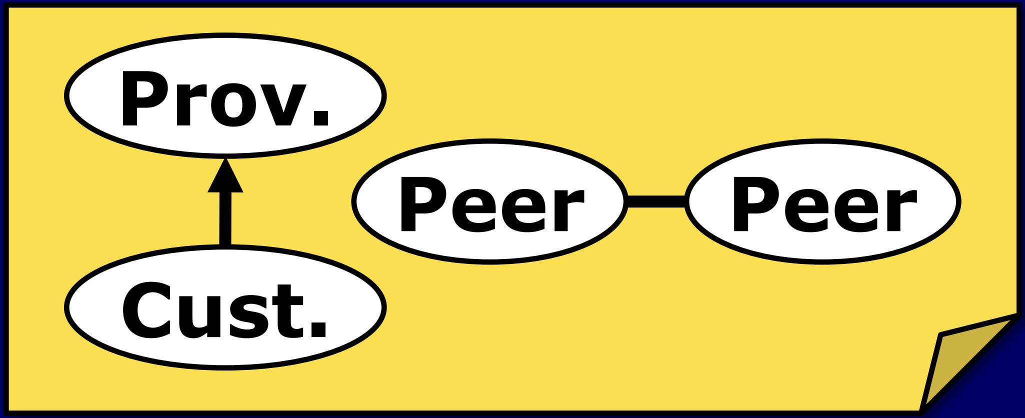


The Gao-Rexford[1] Conditions



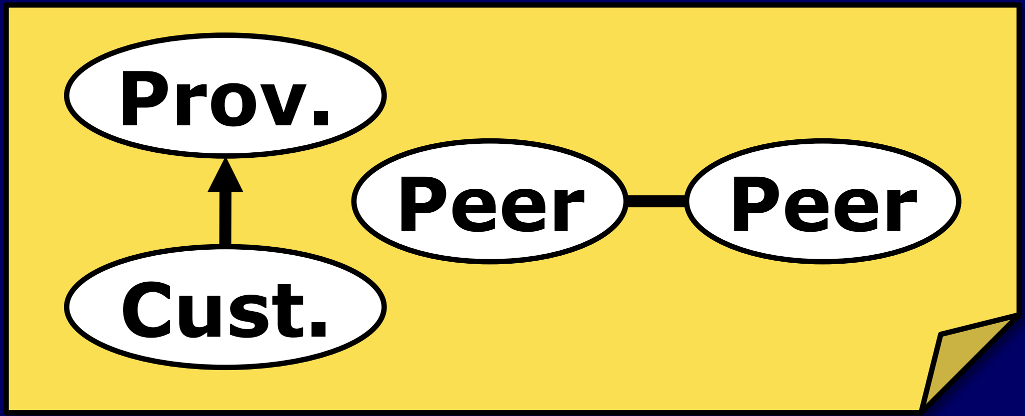
[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

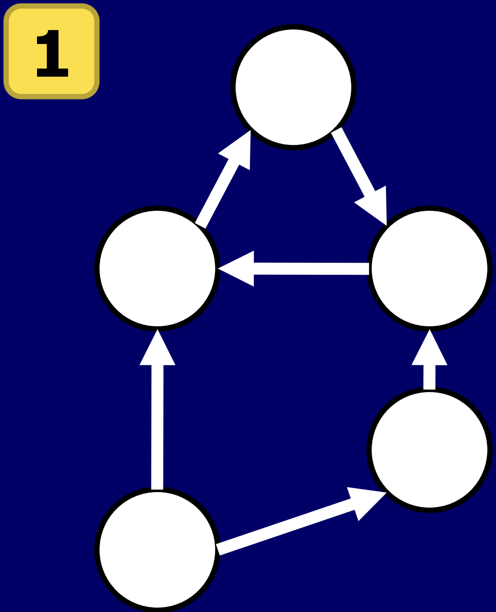
The Gao-Rexford[1] Conditions



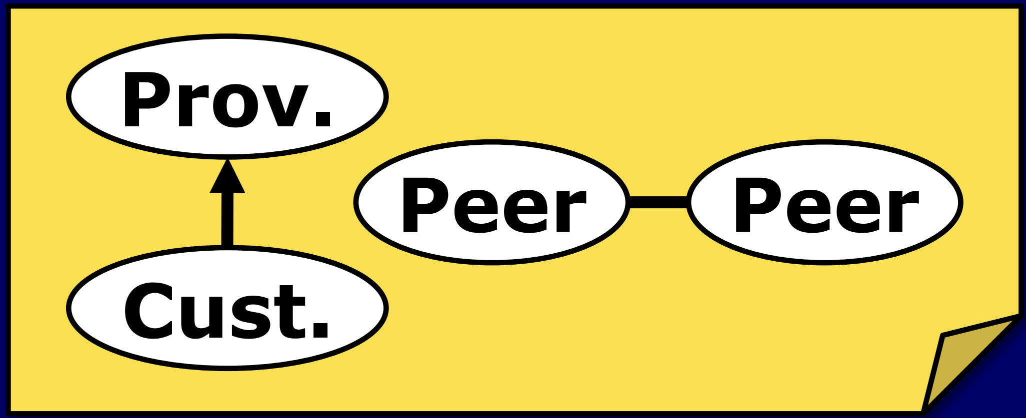
1

The Gao-Rexford[1] Conditions

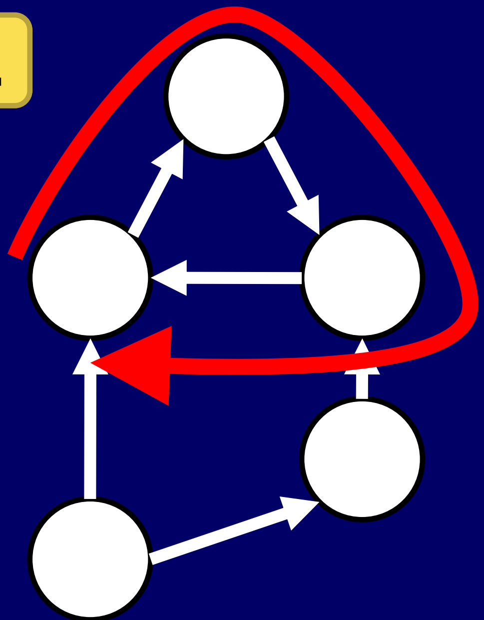




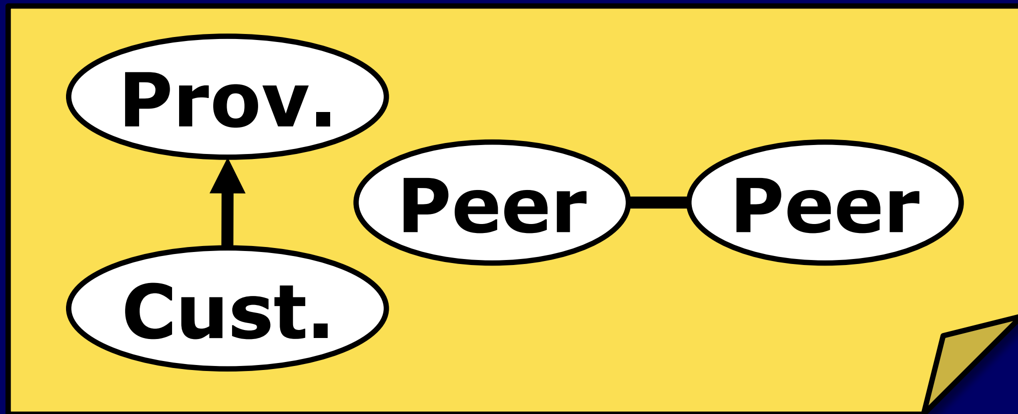
The Gao-Rexford[1] Conditions

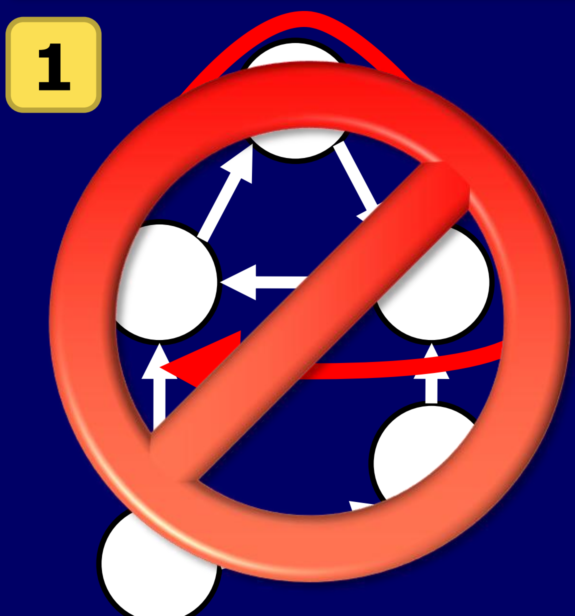


1

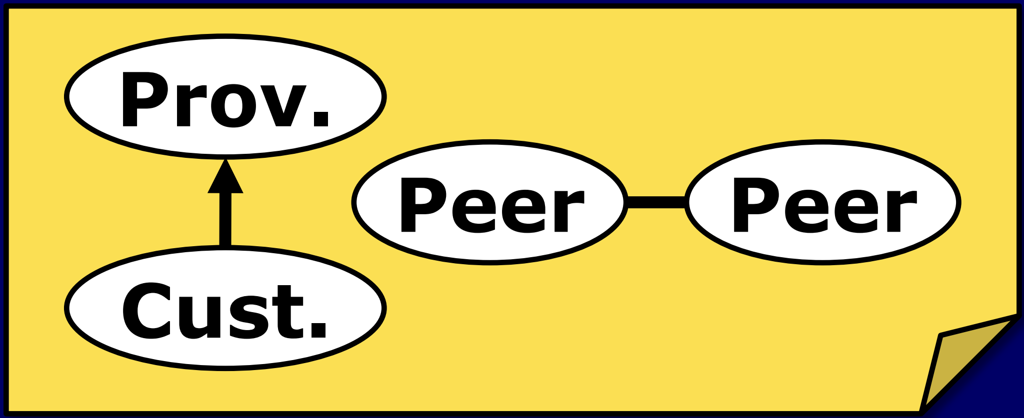


The Gao-Rexford[1] Conditions





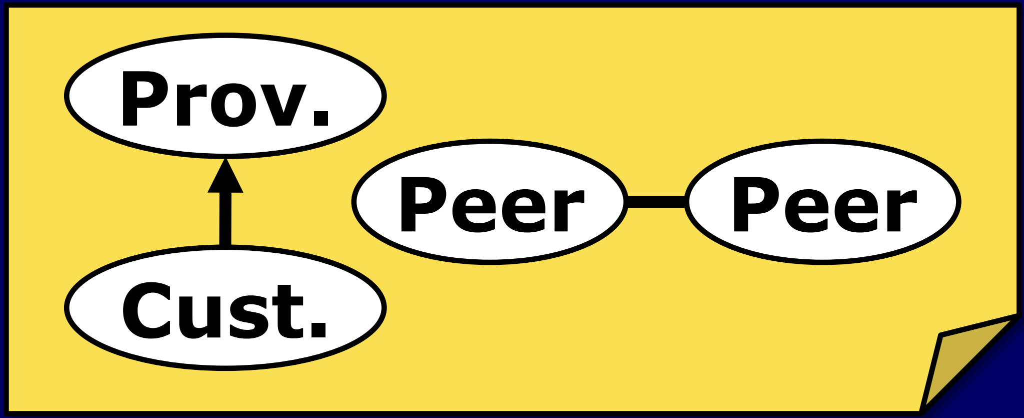
The Gao-Rexford[1] Conditions



1

[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

The Gao-Rexford[1] Conditions

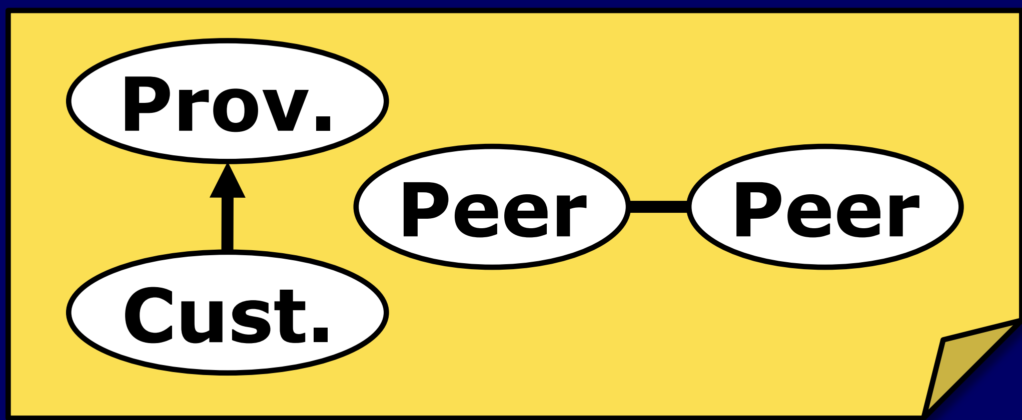


1

2

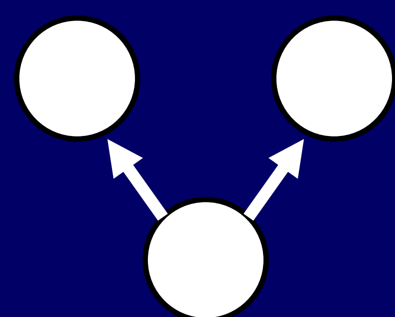
[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

The Gao-Rexford[1] Conditions



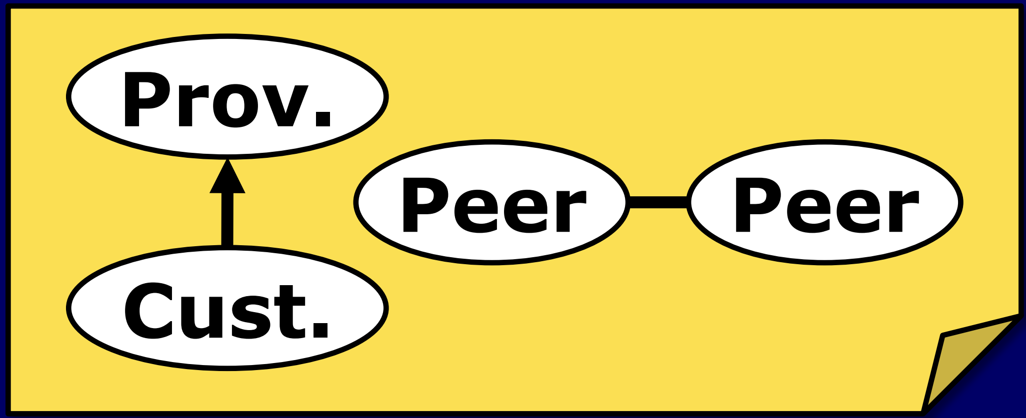
1

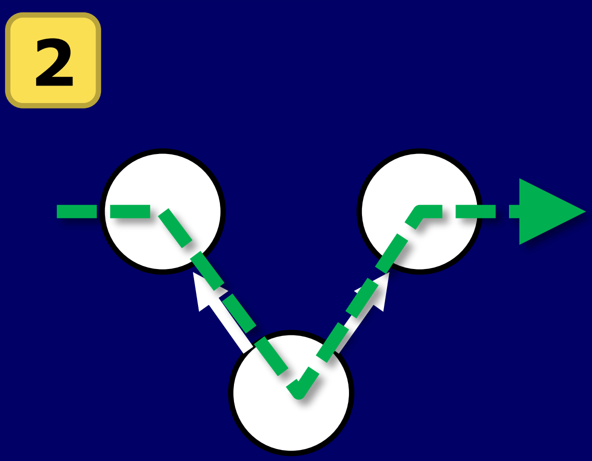
2



[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

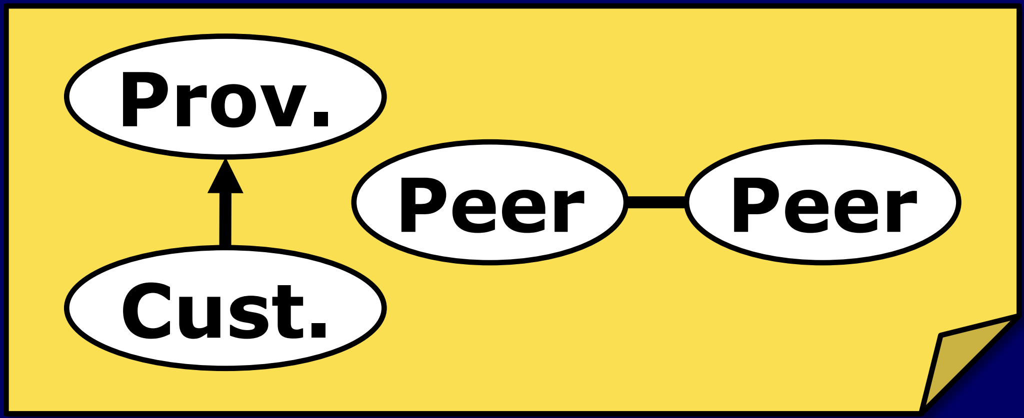
The Gao-Rexford[1] Conditions





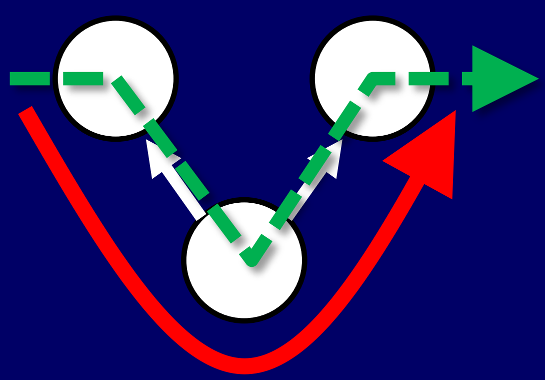
[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

The Gao-Rexford[1] Conditions



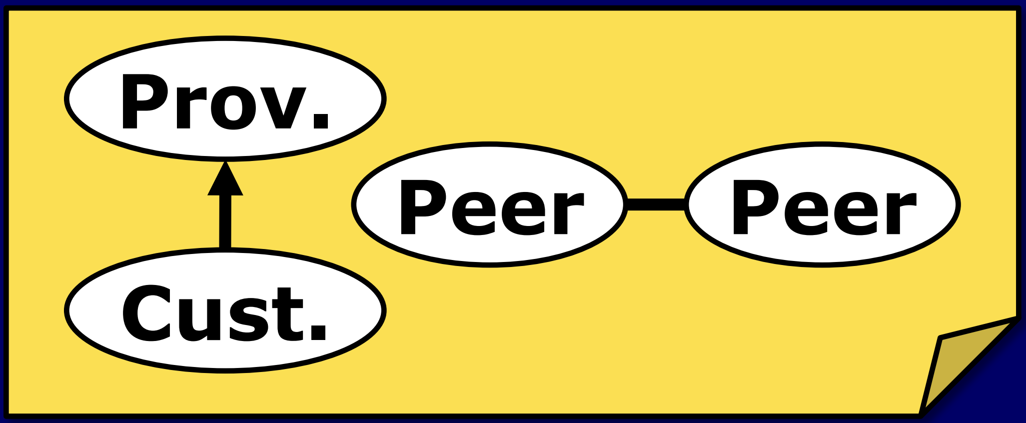
1

2



[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

The Gao-Rexford[1] Conditions

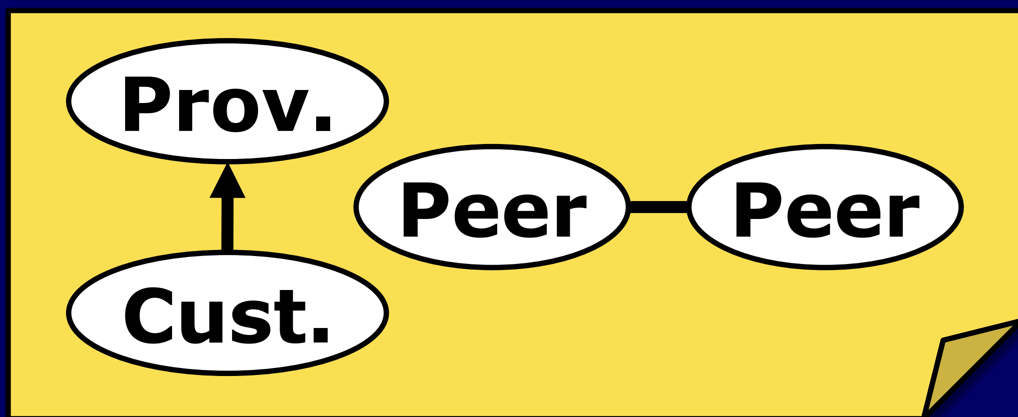


1

2

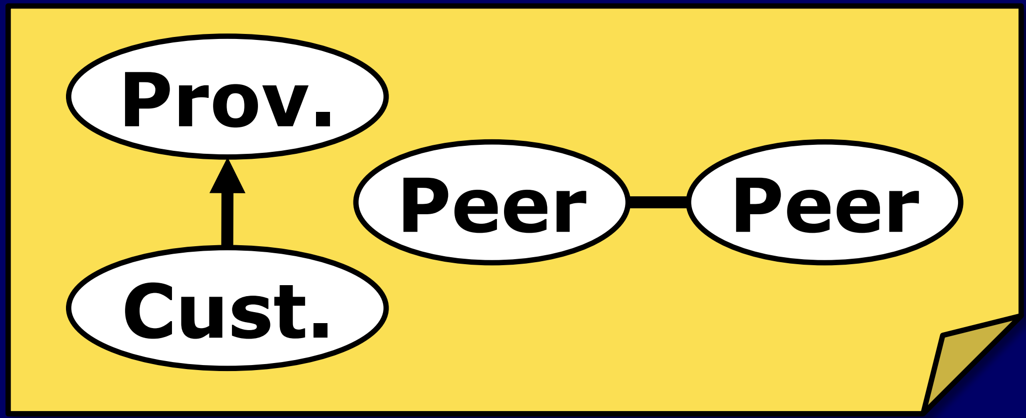
[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

The Gao-Rexford[1] Conditions



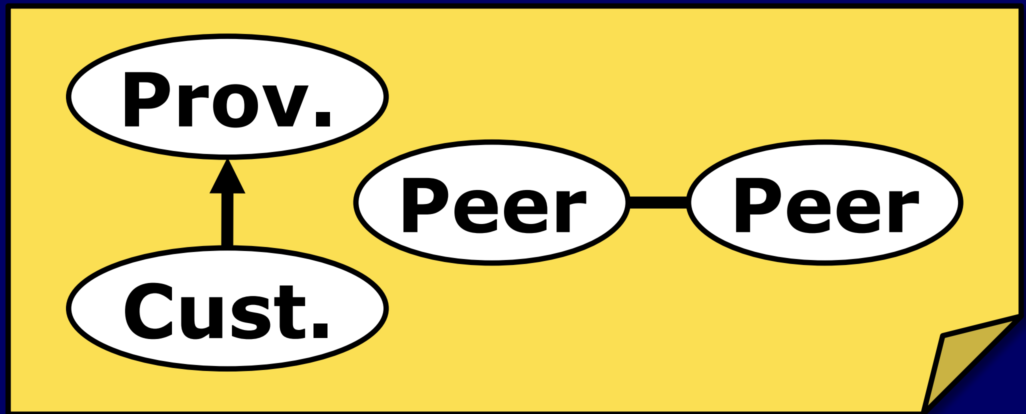
[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

The Gao-Rexford[1] Conditions



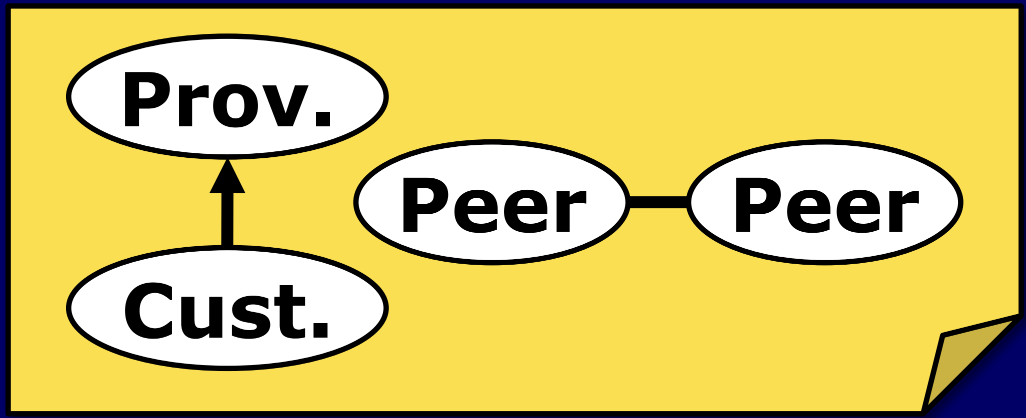
[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

The Gao-Rexford[1] Conditions



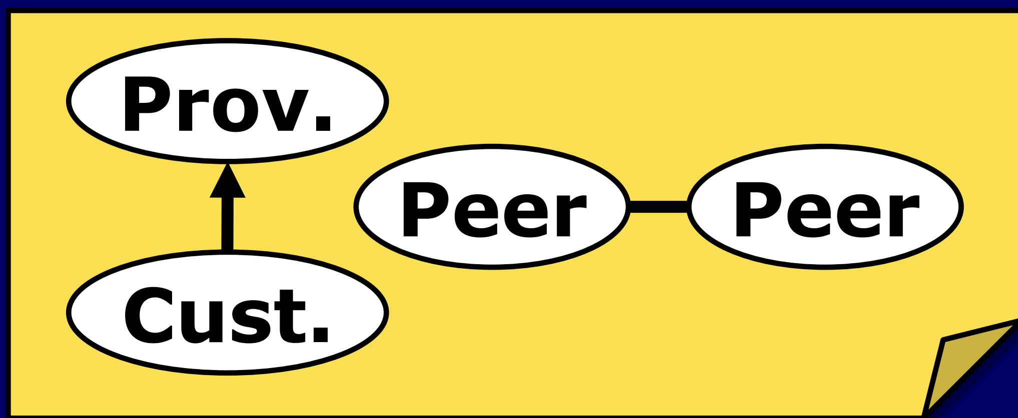
[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

The Gao-Rexford[1] Conditions



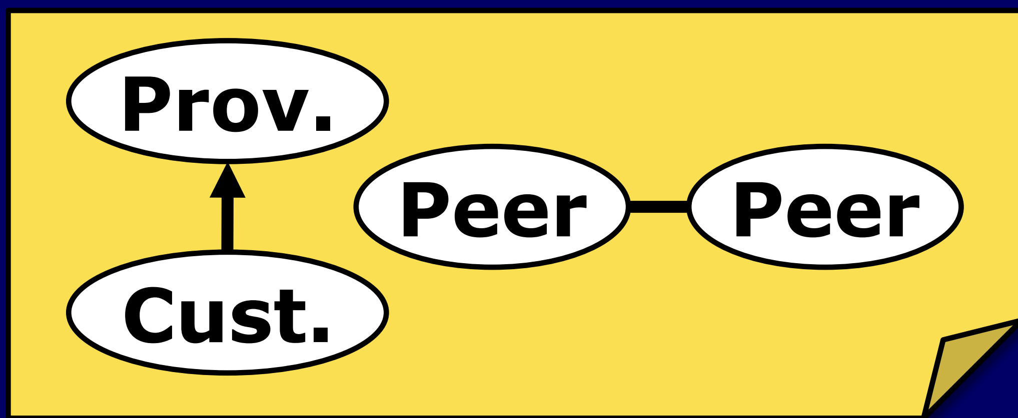
[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

The Gao-Rexford[1] Conditions



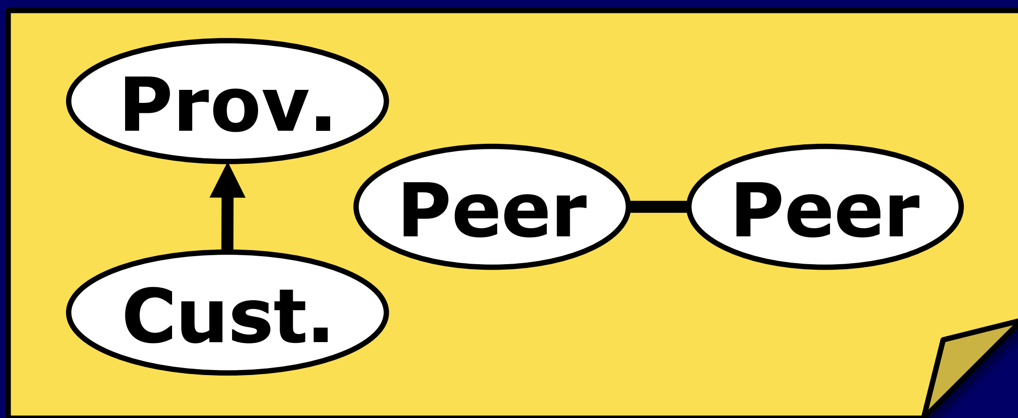
[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

The Gao-Rexford[1] Conditions



[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

The Gao-Rexford[1] Conditions



[1] L. Gao, J. Rexford. Stable Internet Routing without Global Coordination. SIGMETRICS, 2000

Literature
(and a bit of motivation)

- Safety [2] is important...

[2] T. Griffin, F. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

Literature (and a bit of motivation)

- Safety [2] is important...
- ...but hard to check [3], [4]

[2] T. Griffin, F. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.
[3] A. Fabrikant, C. Papadimitriou. The Complexity of Game Dynamics: BGP Oscillations, Sink Equilibria, and beyond. SODA 2008.
[4] T. Griffin, G. Wilfong. An Analysis of BGP Convergence Properties. SIGCOMM 1999.

Literature (and a bit of motivation)

- Safety [2] is important...
- ...but hard to check [3], [4]
- Achieved by different approaches

[2] T. Griffin, F. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.
[3] A. Fabrikant, C. Papadimitriou. The Complexity of Game Dynamics: BGP Oscillations, Sink Equilibria, and beyond. SODA 2008.
[4] T. Griffin, G. Wilfong. An Analysis of BGP Convergence Properties. SIGCOMM 1999.

Literature (and a bit of motivation)

- Safety [2] is important...
- ...but hard to check [3], [4]
- Achieved by different approaches
  - let oscillations occur, but dynamically resolve them [5], [6]

[2] T. Griffin, F. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

[3] A. Fabrikant, C. Papadimitriou. The Complexity of Game Dynamics: BGP Oscillations, Sink Equilibria, and beyond. SODA 2008.

[4] T. Griffin, G. Wilfong. An Analysis of BGP Convergence Properties. SIGCOMM 1999.

[5] T. Griffin, G. Wilfong. A Safe Path Vector Protocol. INFOCOM 2000.

[6] C. Ee, V. Ramachandran, B.-G. Chun, K. Lakshminarayanan, S. Shenker. Resolving Inter-domain Policy Disputes. SIGCOMM 2007.

Literature
(and a bit of motivation)

- Safety [2] is important...
- ...but hard to check [3], [4]
- Achieved by different approaches
  - let oscillations occur, but dynamically resolve them [5], [6]
  - limit policy expressiveness [7], [2]

[2] T. Griffin, F. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

[3] A. Fabrikant, C. Papadimitriou. The Complexity of Game Dynamics: BGP Oscillations, Sink Equilibria, and beyond. SODA 2008.

[4] T. Griffin, G. Wilfong. An Analysis of BGP Convergence Properties. SIGCOMM 1999.

[5] T. Griffin, G. Wilfong. A Safe Path Vector Protocol. INFOCOM 2000.

[6] C. Ee, V. Ramachandran, B.-G. Chun, K. Lakshminarayanan, S. Shenker. Resolving Inter-domain Policy Disputes. SIGCOMM 2007.

[7] N. Feamster, R. Johari, H. Balakrishnan. Implications of Autonomy for the Expressiveness of Policy Routing. ToN, 2007.

Literature (and a bit of motivation)

- Safety [2] is important...
- ...but hard to check [3], [4]
- Achieved by different approaches
  - let oscillations occur, but dynamically resolve them [5], [6]
  - limit policy expressiveness [7], [2]
  - GR

[2] T. Griffin, F. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

[3] A. Fabrikant, C. Papadimitriou. The Complexity of Game Dynamics: BGP Oscillations, Sink Equilibria, and beyond. SODA 2008.

[4] T. Griffin, G. Wilfong. An Analysis of BGP Convergence Properties. SIGCOMM 1999.

[5] T. Griffin, G. Wilfong. A Safe Path Vector Protocol. INFOCOM 2000.

[6] C. Ee, V. Ramachandran, B.-G. Chun, K. Lakshminarayanan, S. Shenker. Resolving Inter-domain Policy Disputes. SIGCOMM 2007.

[7] N. Feamster, R. Johari, H. Balakrishnan. Implications of Autonomy for the Expressiveness of Policy Routing. ToN, 2007.

Literature (and a bit of motivation)

- Safety [2] is important...
- ...but hard to check [3], [4]
- Achieved by different approaches
  - ~~let oscillations occur~~, but dynamically resolve them [5], [6]
  - limit policy expressiveness [7], [2]
  - GR

[2] T. Griffin, F. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

[3] A. Fabrikant, C. Papadimitriou. The Complexity of Game Dynamics: BGP Oscillations, Sink Equilibria, and beyond. SODA 2008.

[4] T. Griffin, G. Wilfong. An Analysis of BGP Convergence Properties. SIGCOMM 1999.

[5] T. Griffin, G. Wilfong. A Safe Path Vector Protocol. INFOCOM 2000.

[6] C. Ee, V. Ramachandran, B.-G. Chun, K. Lakshminarayanan, S. Shenker. Resolving Inter-domain Policy Disputes. SIGCOMM 2007.

[7] N. Feamster, R. Johari, H. Balakrishnan. Implications of Autonomy for the Expressiveness of Policy Routing. ToN, 2007.

Literature (and a bit of motivation)

- Safety [2] is important...
- ...but hard to check [3], [4]
- Achieved by different approaches
  - ~~let oscillations occur~~, but dynamically resolve them [5], [6]
  - ~~limit policy expressiveness~~ [7], [2]
  - GR

[2] T. Griffin, F. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.
[3] A. Fabrikant, C. Papadimitriou. The Complexity of Game Dynamics: BGP Oscillations, Sink Equilibria, and beyond. SODA 2008.
[4] T. Griffin, G. Wilfong. An Analysis of BGP Convergence Properties. SIGCOMM 1999.
[5] T. Griffin, G. Wilfong. A Safe Path Vector Protocol. INFOCOM 2000.
[6] C. Ee, V. Ramachandran, B.-G. Chun, K. Lakshminarayanan, S. Shenker. Resolving Inter-domain Policy Disputes. SIGCOMM 2007.
[7] N. Feamster, R. Johari, H. Balakrishnan. Implications of Autonomy for the Expressiveness of Policy Routing. ToN, 2007.

Motivation (and a bit of literature)

- A GR-compliant network...
  - ...preserves autonomy of each AS in configuring local policies

Motivation (and a bit of literature)

- A GR-compliant network...
  - ...preserves autonomy of each AS in configuring local policies
  - ...is *safe* and *robust* [8]

[8] L. Gao, T. Griffin, J. Rexford. Inherently Safe Backup Routing with BGP. INFOCOM 2001

Motivation (and a bit of literature)

- A GR-compliant network...
  - ...preserves autonomy of each AS in configuring local policies
  - ...is *safe* and *robust* [8]
  - ...has a convergence time that is roughly bounded by a constant [9]

[8] L. Gao, T. Griffin, J. Rexford. Inherently Safe Backup Routing with BGP. INFOCOM 2001
[9] R. Sami, M. Schapira, A. Zohar. Searching for Stability in Interdomain Routing. INFOCOM 2009

Motivation (and a bit of literature)

- A GR-compliant network...
  - ...preserves autonomy of each AS in configuring local policies
  - ...is *safe* and *robust* [8]
  - ...has a convergence time that is roughly bounded by a constant [9]
- Remark:
  GR compliance is regarded as a possible explanation for Internet stability [2]

[8] L. Gao, T. Griffin, J. Rexford. Inherently Safe Backup Routing with BGP. INFOCOM 2001
[9] R. Sami, M. Schapira, A. Zohar. Searching for Stability in Interdomain Routing. INFOCOM 2009

Other "Grail Seekers"

- [10]: relationship inference heuristic

[10] L. Gao. On Inferring Autonomous System Relationships in the Internet. ToN, 2001.

Other "Grail Seekers"

- [10]: relationship inference heuristic
- [11]: a valley-free assignment can be achieved efficiently

[10] L. Gao. On Inferring Autonomous System Relationships in the Internet. ToN, 2001.
[11] G. Di Battista, T. Erlebach, A. Hall, M. Patrignani, M. Pizzonia, T. Schank. Computing the Types of the Relationships between Autonomous Systems. ToN, 2007.

Other “Grail Seekers”

- [10]: relationship inference heuristic
- [11]: a valley-free assignment can be achieved efficiently
- [12]: a valley-free+acyclic assignment can be achieved efficiently

[10] L. Gao. On Inferring Autonomous System Relationships in the Internet. ToN, 2001.
[11] G. Di Battista, T. Erlebach, A. Hall, M. Patrignani, M. Pizzonia, T. Schank. Computing the Types of the Relationships between Autonomous Systems. ToN, 2007.
[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006.

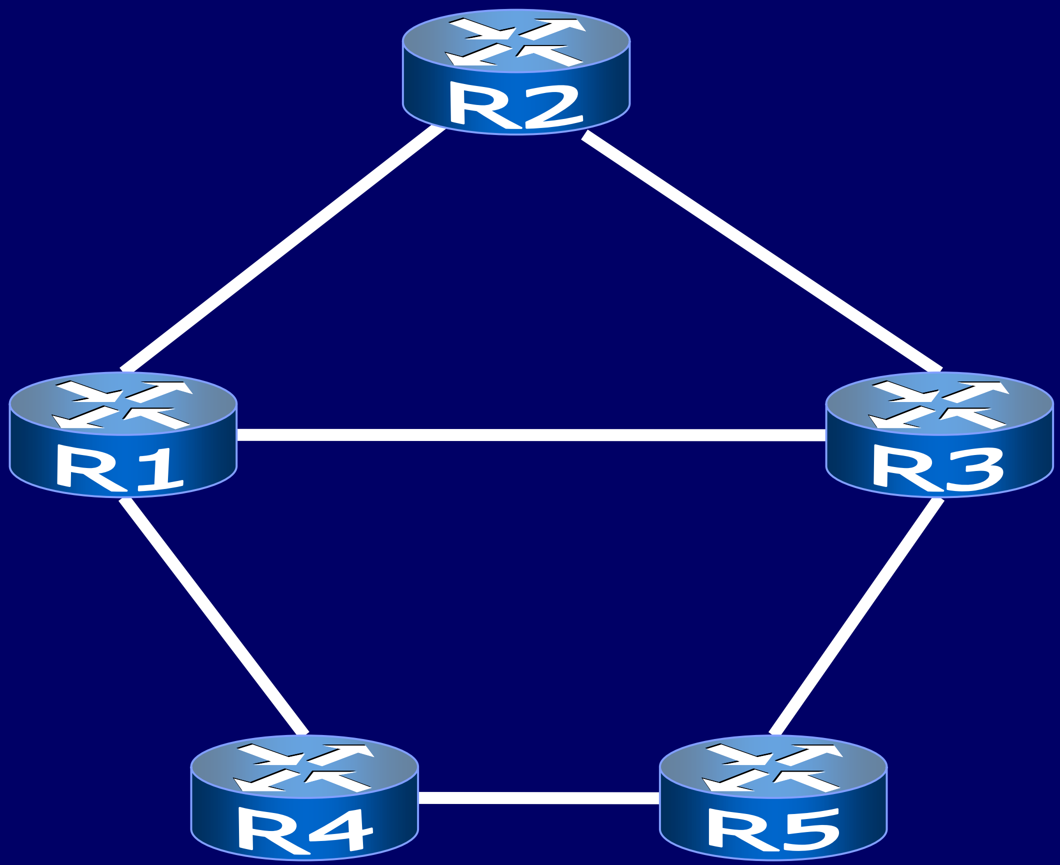
Other "Grail Seekers"

- [10]: relationship inference heuristic
- [11]: a valley-free assignment can be achieved efficiently
- [12]: a valley-free+acyclic assignment can be achieved efficiently
- [13]: distributed detection of the GR conditions (with known relationships)

[10] L. Gao. On Inferring Autonomous System Relationships in the Internet. ToN, 2001.
[11] G. Di Battista, T. Erlebach, A. Hall, M. Patrignani, M. Pizzonia, T. Schank. Computing the Types of the Relationships between Autonomous Systems. ToN, 2007.
[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006.
[13] S. Epstein, K. Mattar, I. Matta. Principles of Safe Policy Routing Dynamics. ICNP 2009.

Problem
GAO-REXFORD-CHECK

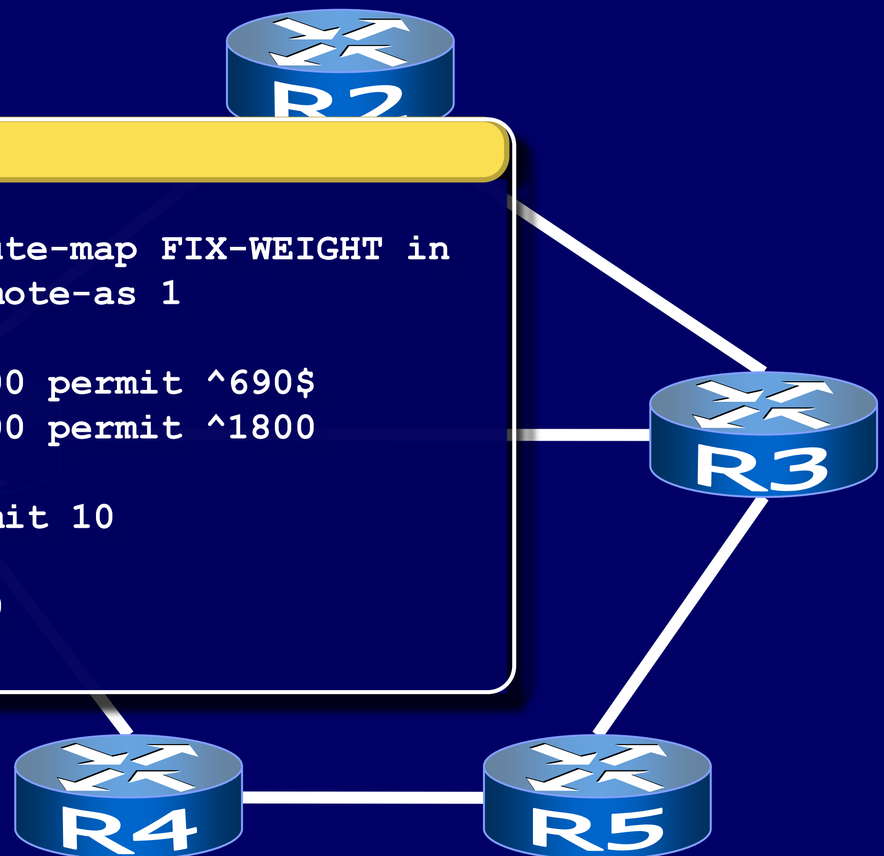
Instance: (model of) a BGP configuration



Problem
GAO-REXFORD-CHECK

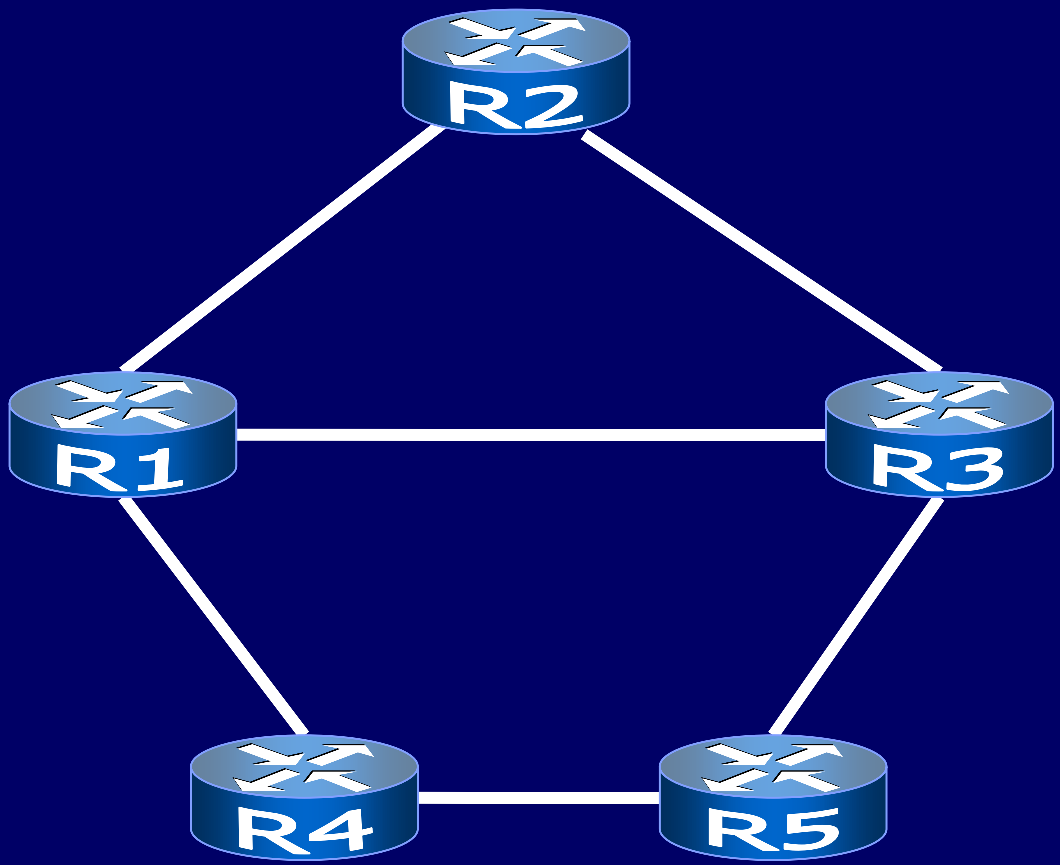
Instance: (model of) a BGP configuration

```
router bgp 100
!
 neighbor 140.222.1.1 route-map FIX-WEIGHT in
 neighbor 140.222.1.1 remote-as 1
!
ip as-path access-list 200 permit ^690$
ip as-path access-list 200 permit ^1800
!
route-map FIX-WEIGHT permit 10
 match as-path 200
 set local-preference 250
 set weight200
```



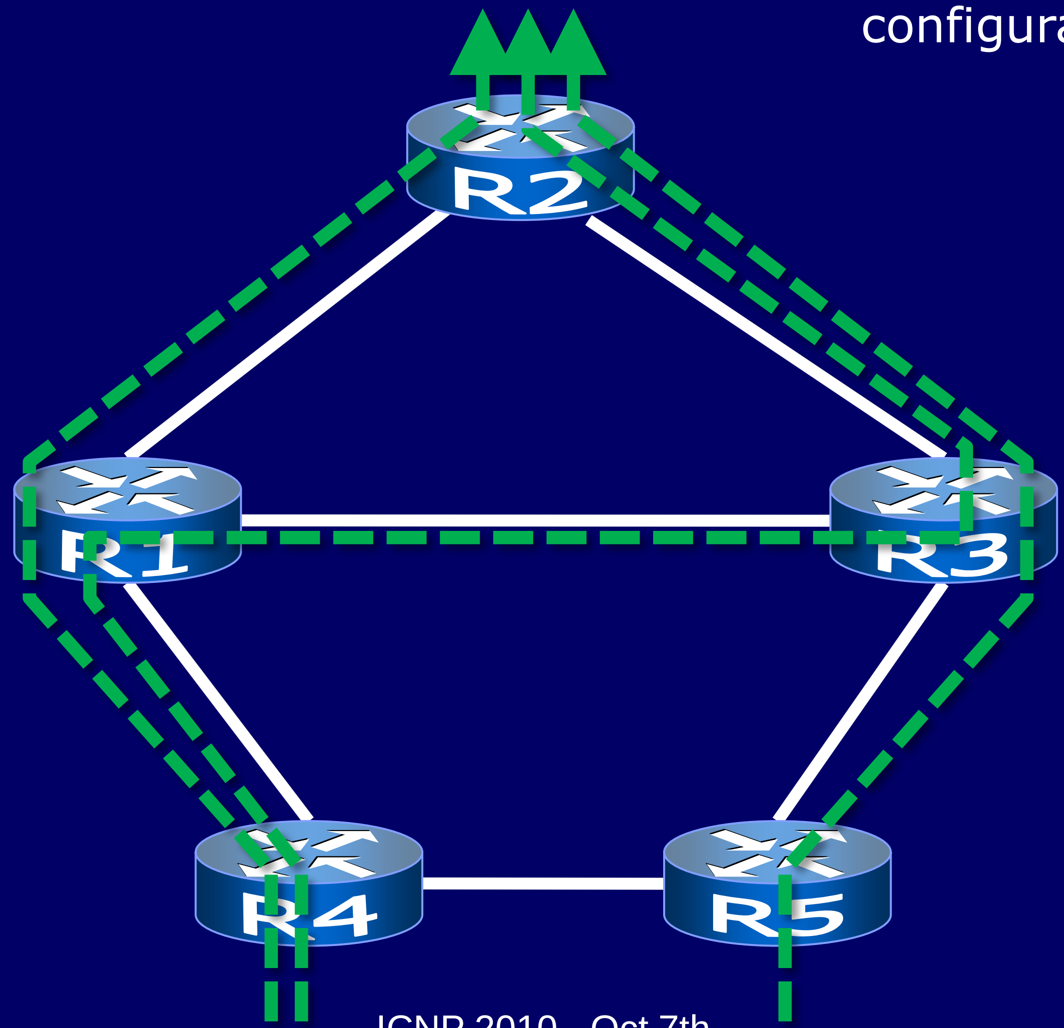
Problem
GAO-REXFORD-CHECK

Instance: (model of) a BGP configuration



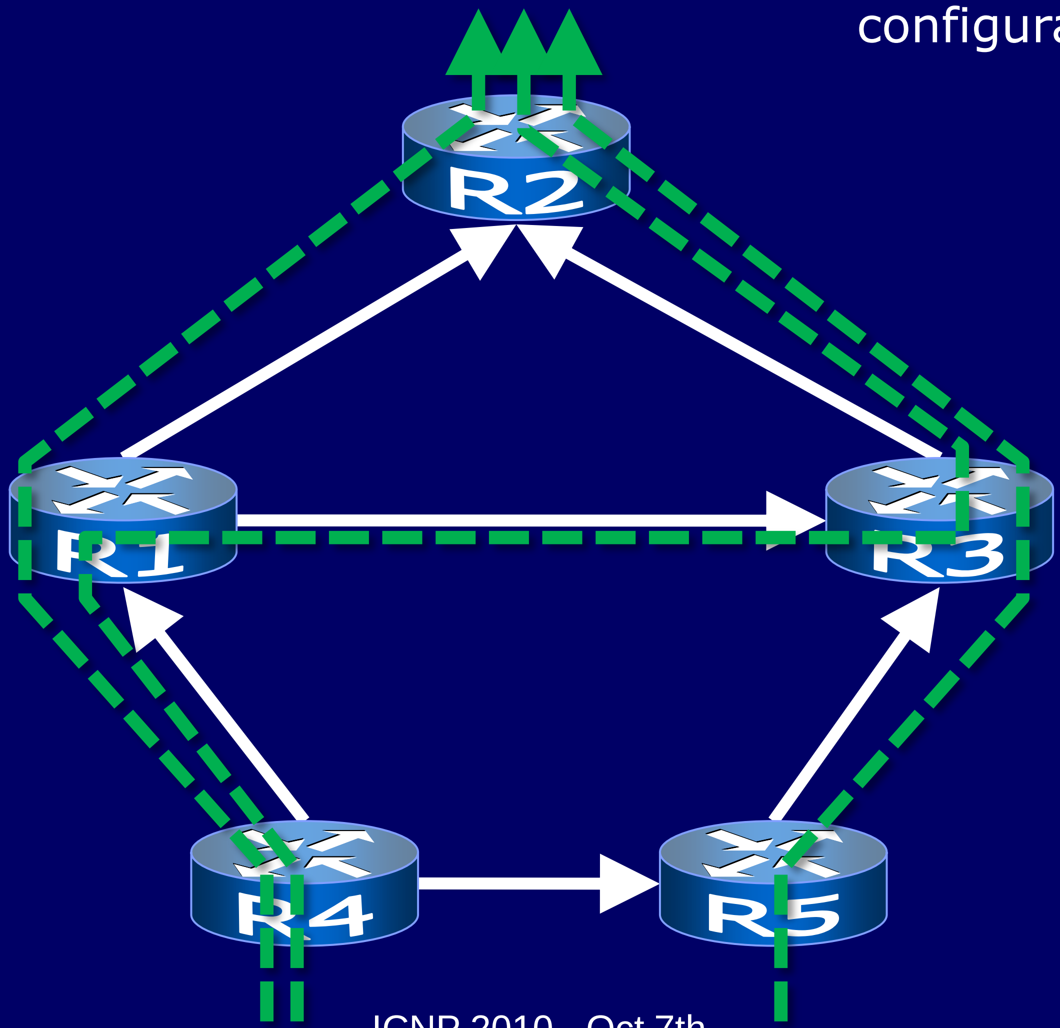
Problem
GAO-REXFORD-CHECK

Instance: (model of) a BGP configuration



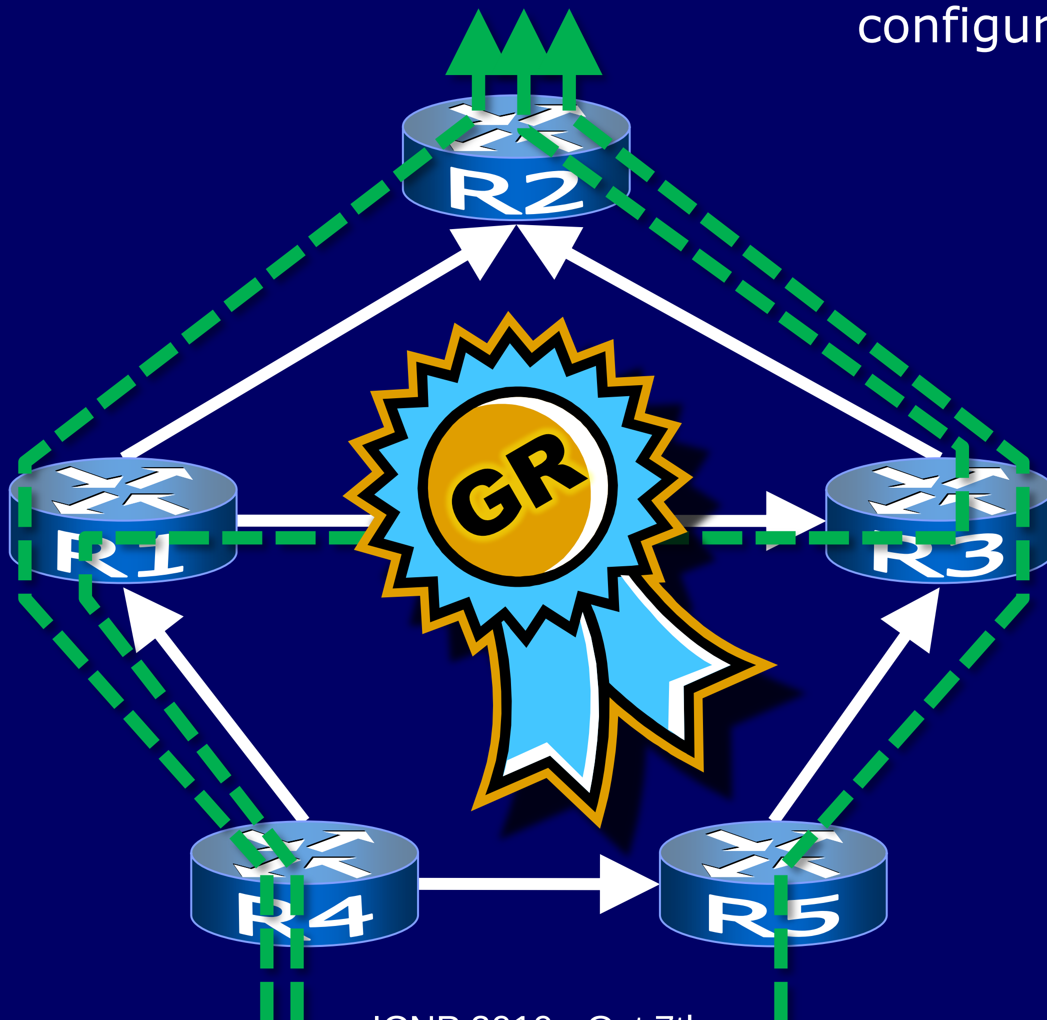
Problem
GAO-REXFORD-CHECK

Instance: (model of) a BGP configuration



Problem
GAO-REXFORD-CHECK

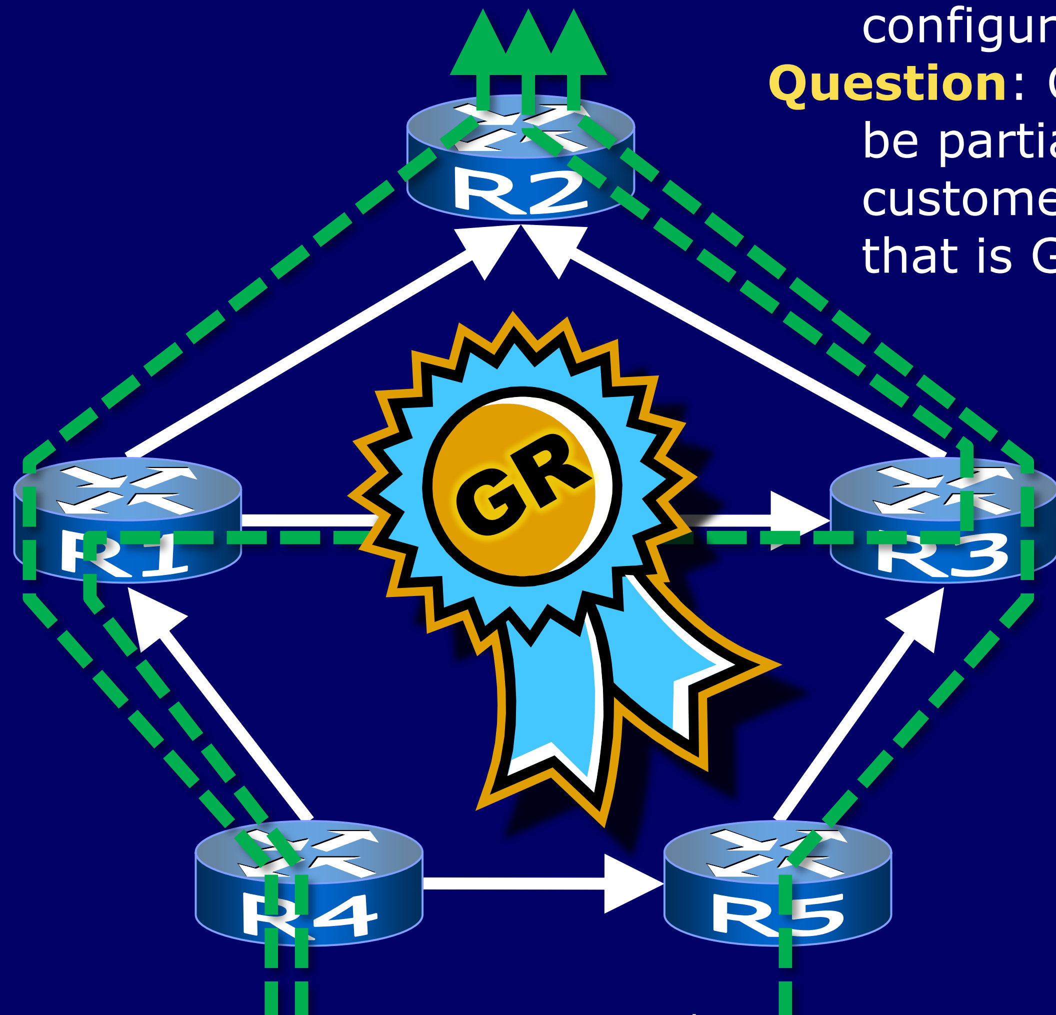
Instance: (model of) a BGP configuration



Problem
GAO-REXFORD-CHECK

Instance: (model of) a BGP configuration

Question: Can the network be partially oriented to a customer-provider graph that is GR-compliant?



Results

Results

1. Polynomial algorithm for GAO-REXFORD-CHECK

Results

1. Polynomial algorithm for GAO-REXFORD-CHECK

GAO-REXFORD-STRICT-CHECK:

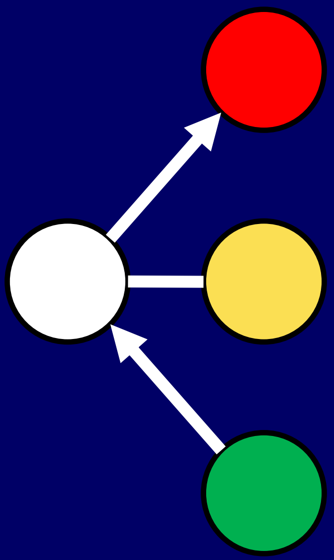
same as GAO-REXFORD-CHECK, but peers are preferred to providers

Results

1. Polynomial algorithm for GAO-REXFORD-CHECK

GAO-REXFORD-STRICT-CHECK:

same as GAO-REXFORD-CHECK, but peers are preferred to providers

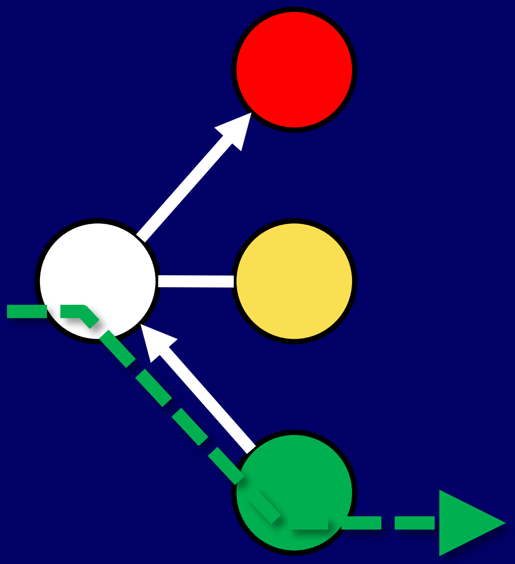


Results

1. Polynomial algorithm for GAO-REXFORD-CHECK

GAO-REXFORD-STRICT-CHECK:

same as GAO-REXFORD-CHECK, but peers are preferred to providers

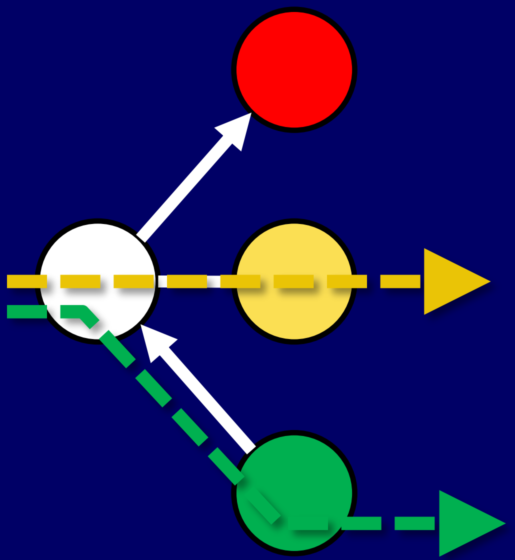


Results

1. Polynomial algorithm for GAO-REXFORD-CHECK

GAO-REXFORD-STRICT-CHECK:

same as GAO-REXFORD-CHECK, but peers are preferred to providers

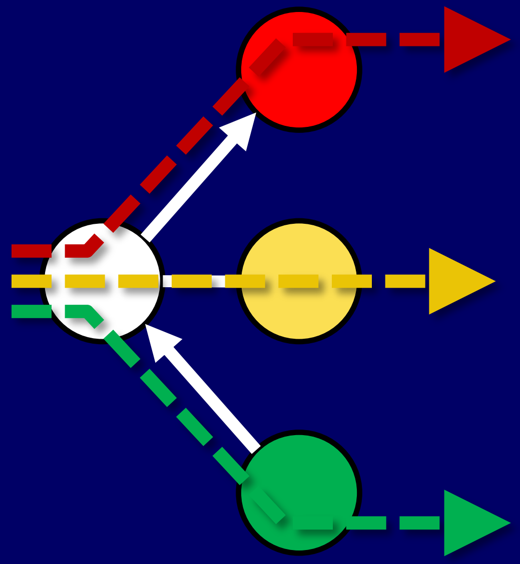


Results

1. Polynomial algorithm for GAO-REXFORD-CHECK

GAO-REXFORD-STRICT-CHECK:

same as GAO-REXFORD-CHECK, but peers are preferred to providers

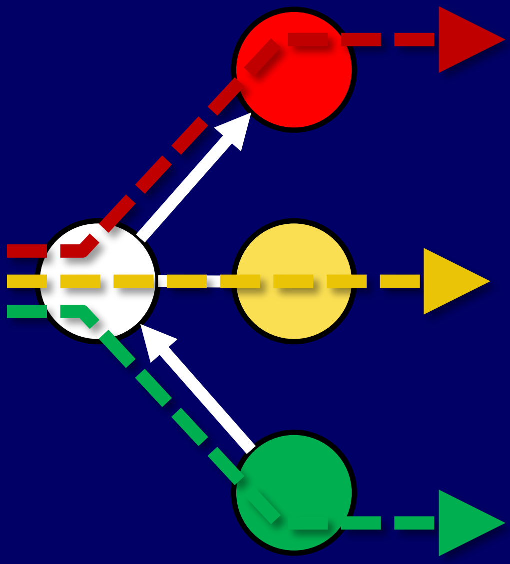


Results

1. Polynomial algorithm for GAO-REXFORD-CHECK

   GAO-REXFORD-STRICT-CHECK:

   same as GAO-REXFORD-CHECK, but peers are preferred to providers



2. NP-hardness of GAO-REXFORD-STRICT-CHECK

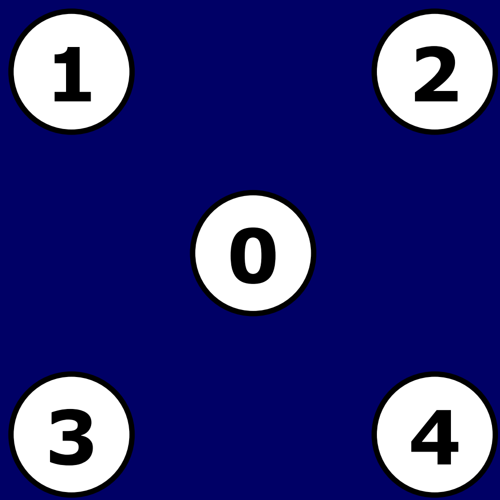
Models (briefly)

1

2

Models (briefly)

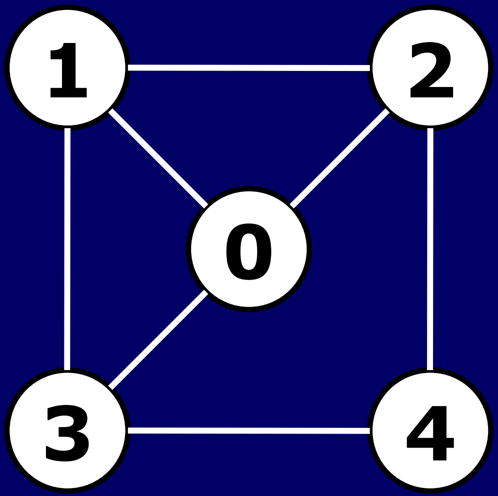
1



2

Models (briefly)

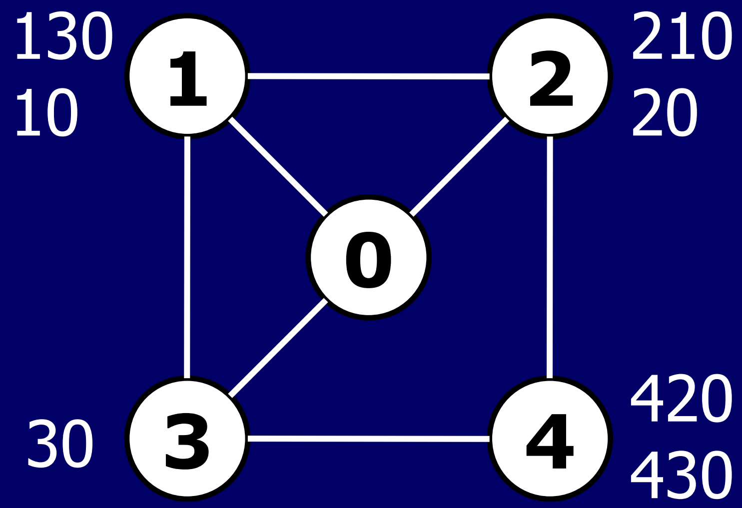
1



2

Models (briefly)

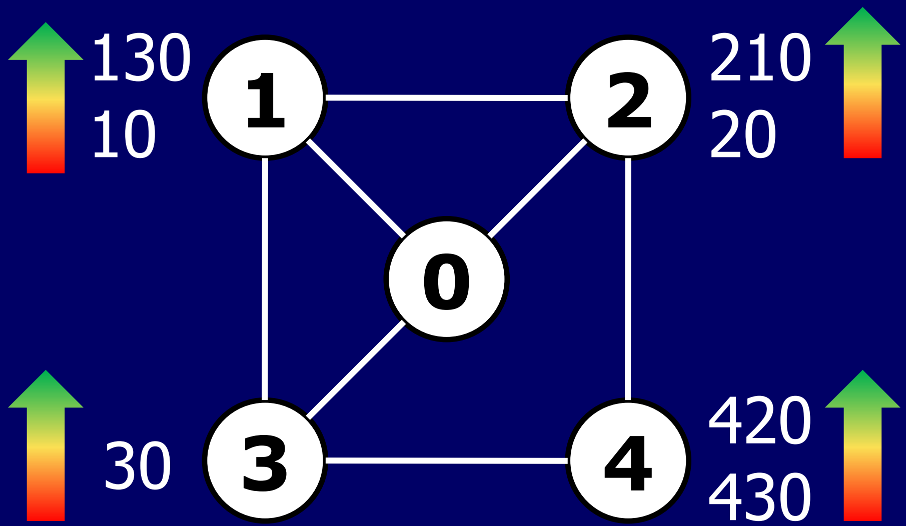
1



2

Models (briefly)

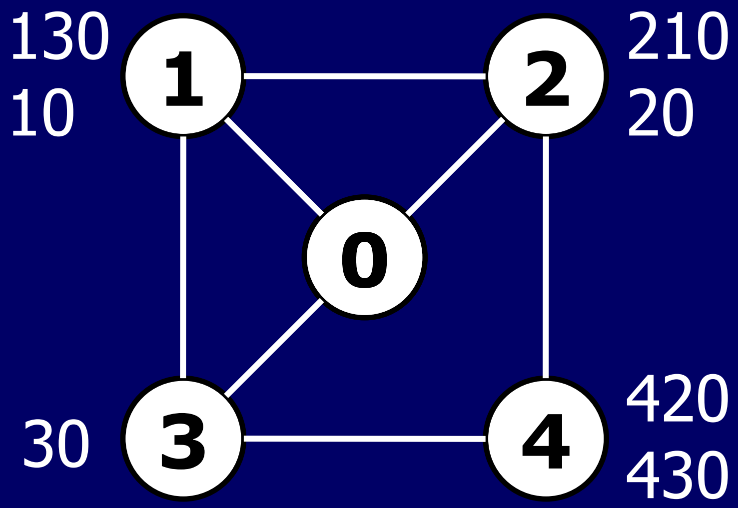
1



2

Models (briefly)

1



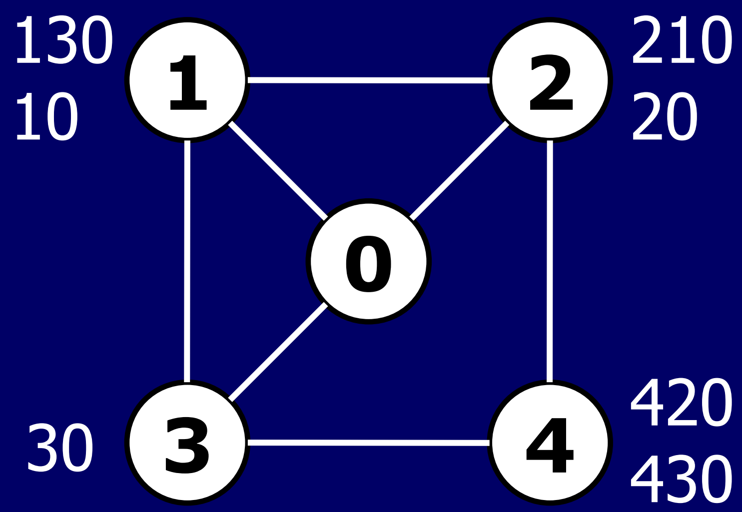
Stable Paths Problem (SPP) [2]

2

[2] T. G. Griffin, F. B. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

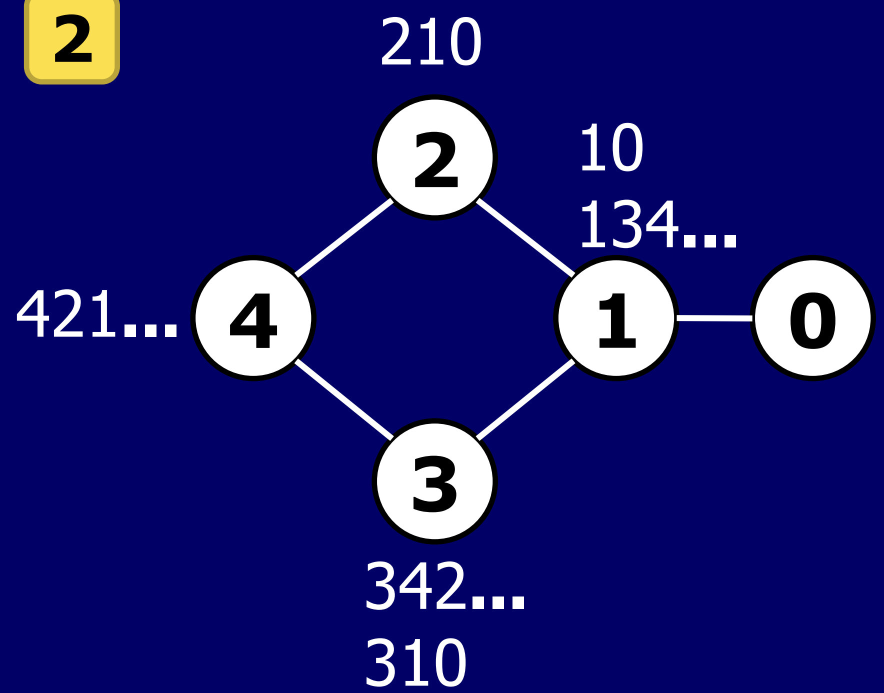
Models (briefly)

1



Stable Paths Problem
(SPP) [2]

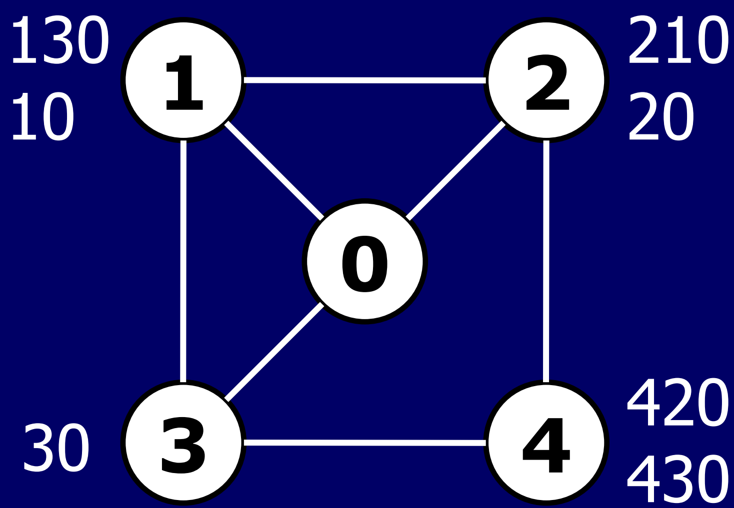
2



[2] T. G. Griffin, F. B. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

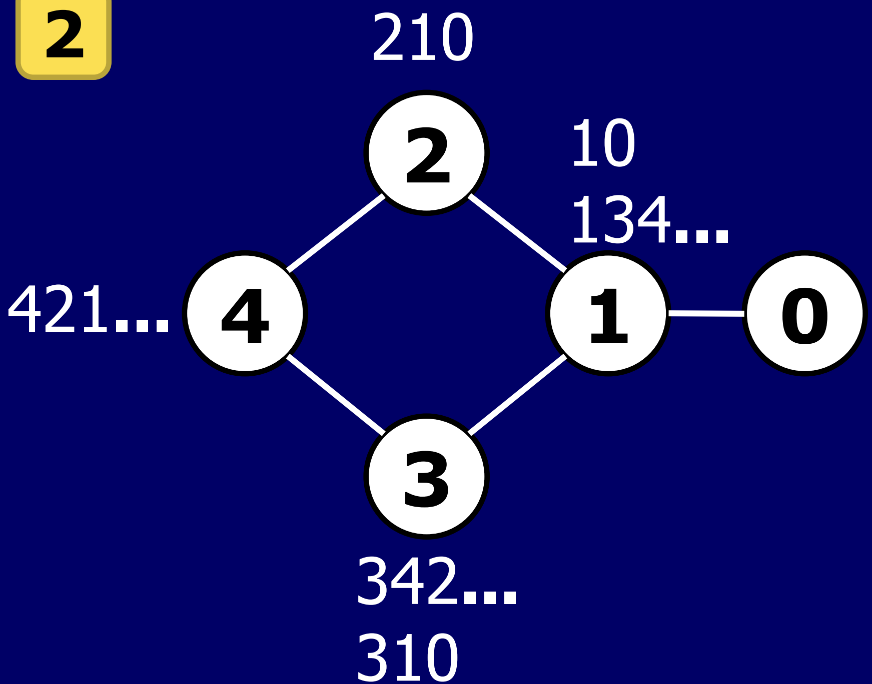
Models (briefly)

1



Stable Paths Problem (SPP) [2]

2

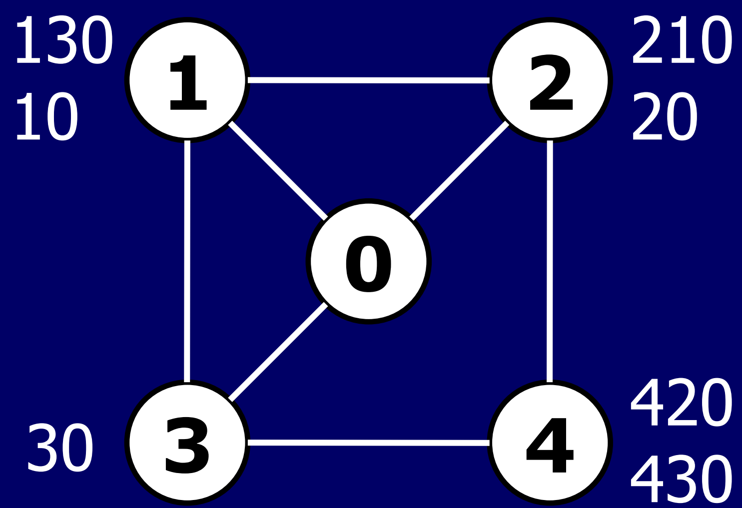


Succinct SPP (SSPP)

[2] T. G. Griffin, F. B. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

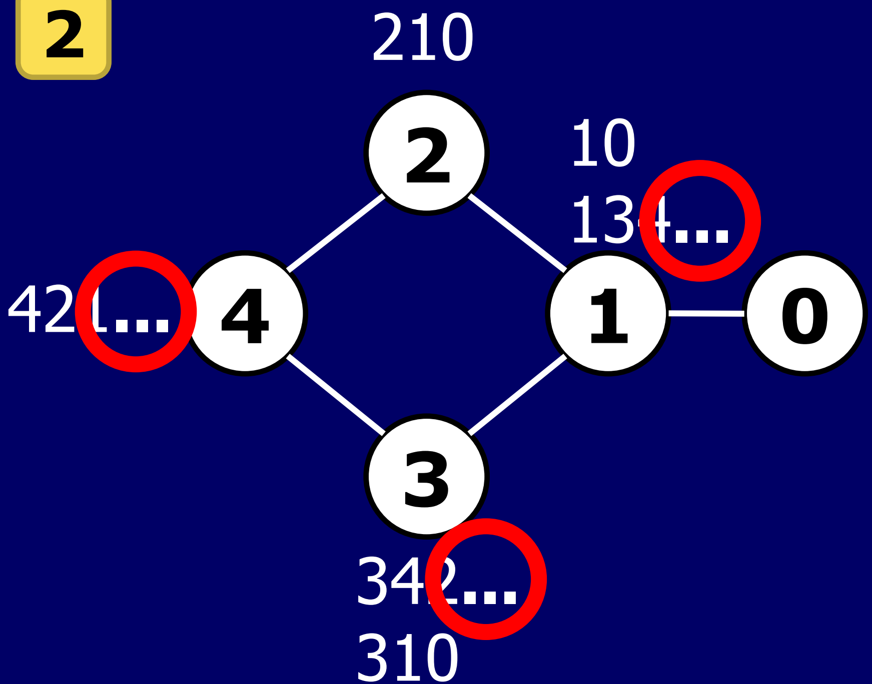
Models (briefly)

1



Stable Paths Problem
(SPP) [2]

2

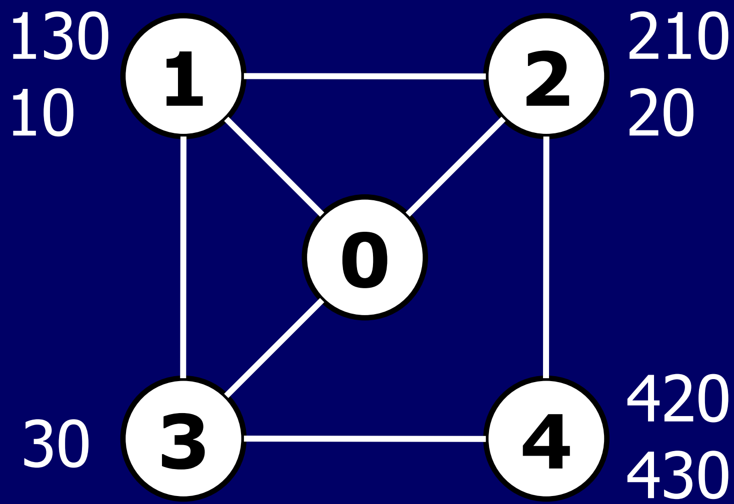


Succinct SPP
(SSPP)

[2] T. G. Griffin, F. B. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

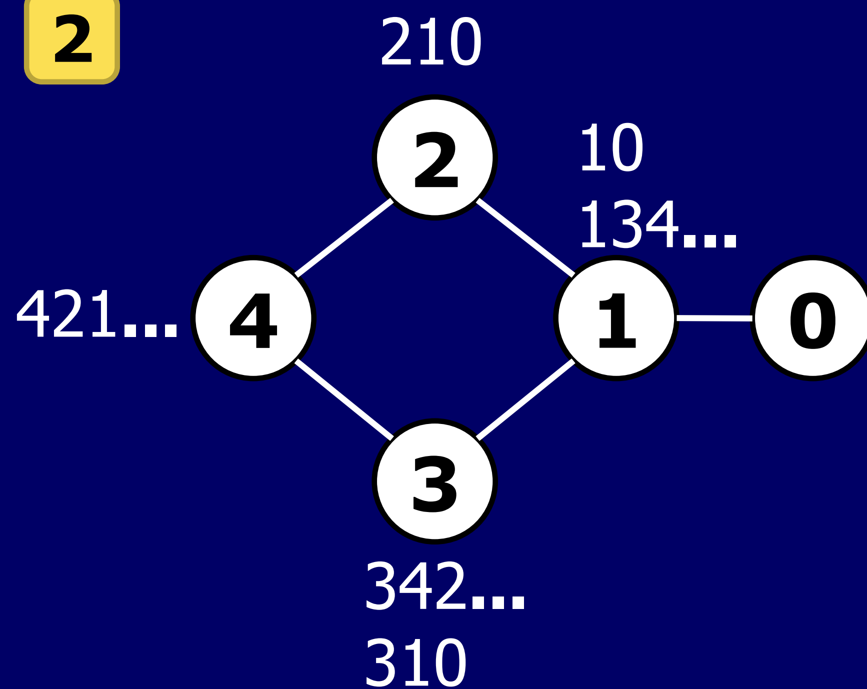
Models (briefly)

1



Stable Paths Problem (SPP) [2]

2

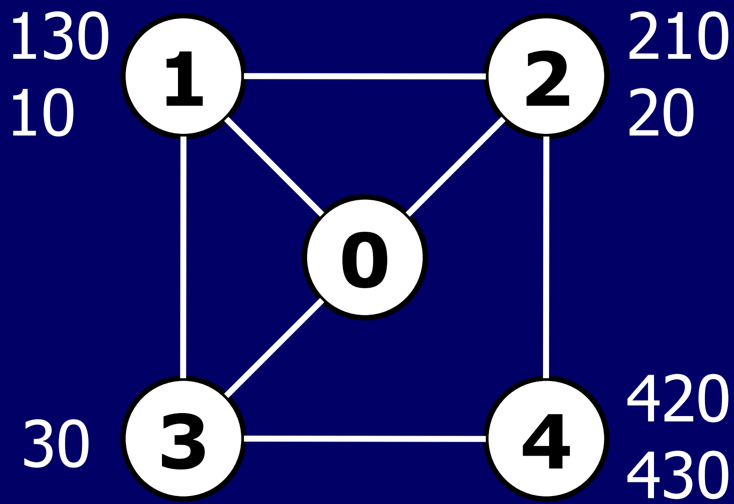


Succinct SPP (SSPP)

[2] T. G. Griffin, F. B. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

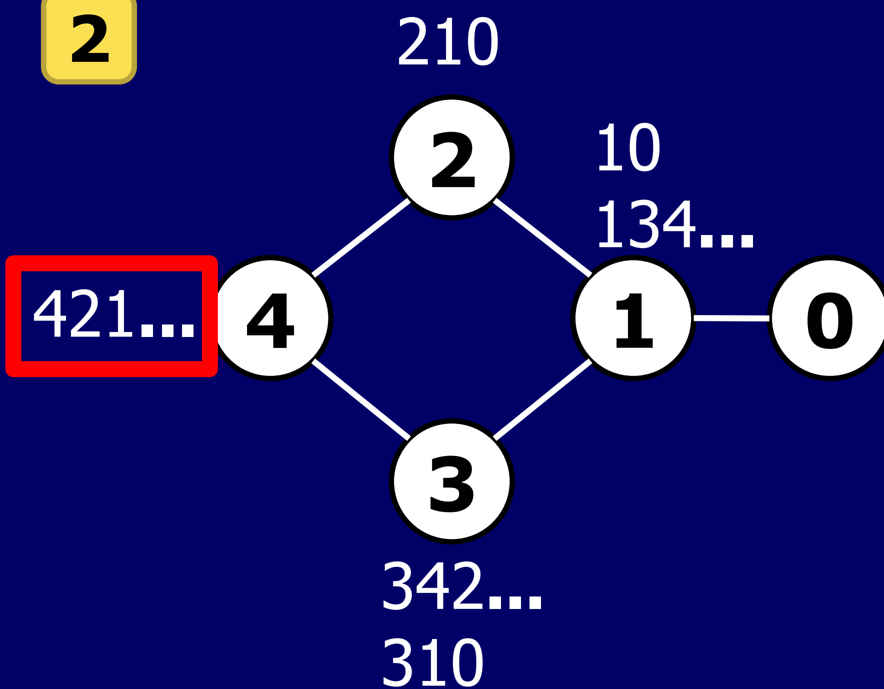
Models (briefly)

1



Stable Paths Problem (SPP) [2]

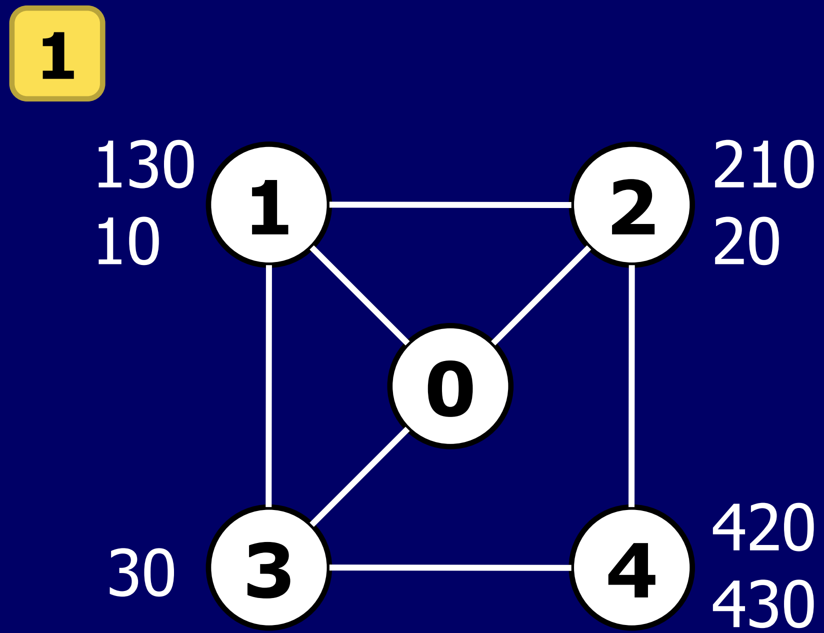
2



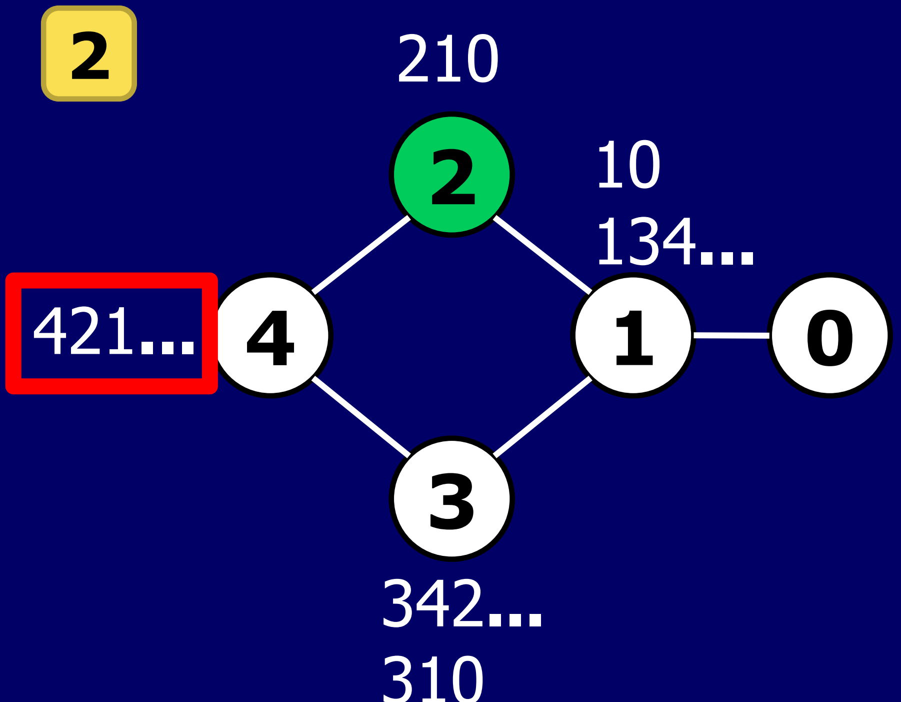
Succinct SPP (SSPP)

[2] T. G. Griffin, F. B. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

Models (briefly)



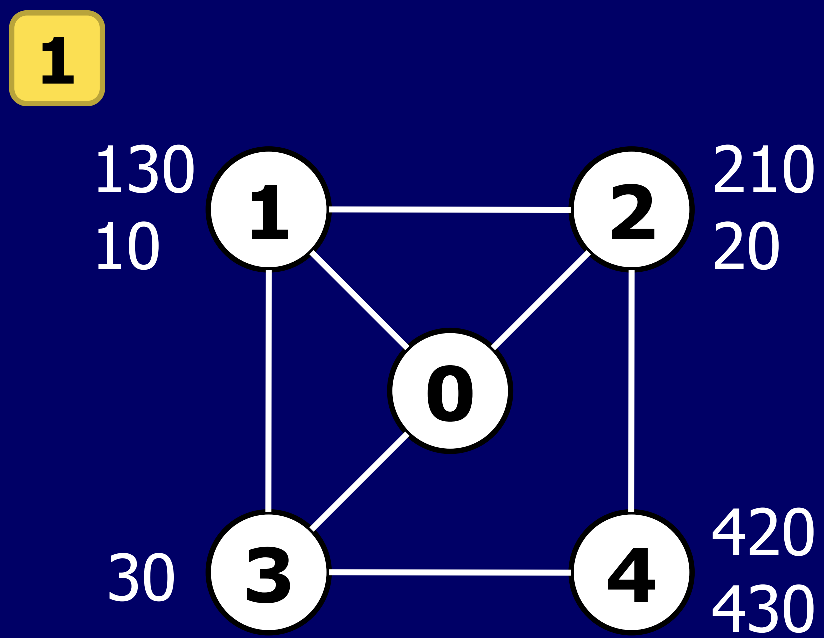
Stable Paths Problem (SPP) [2]



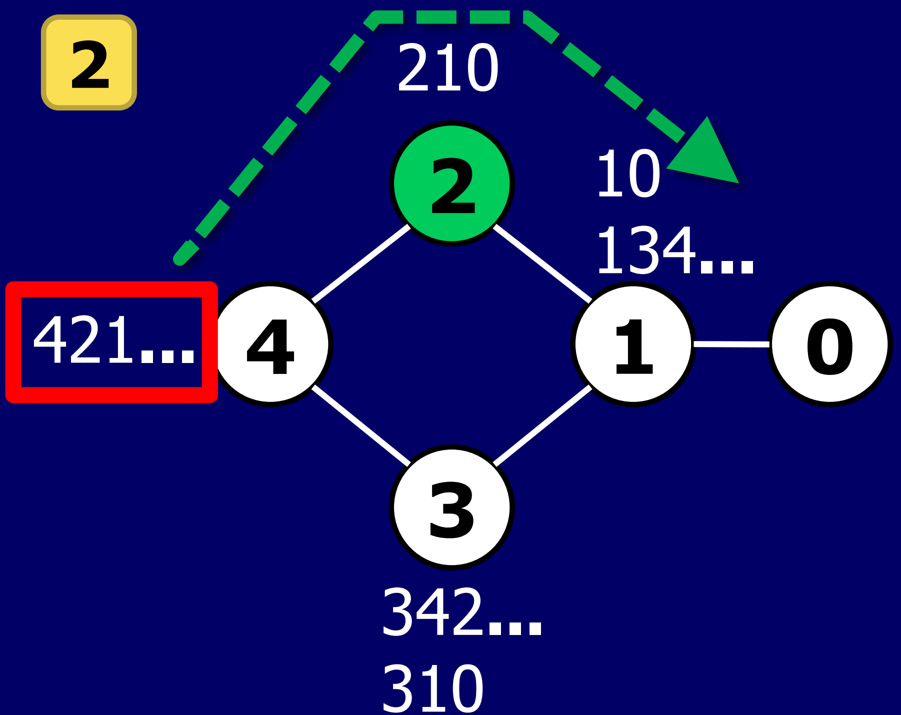
Succinct SPP (SSPP)

[2] T. G. Griffin, F. B. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

Models (briefly)



Stable Paths Problem (SPP) [2]

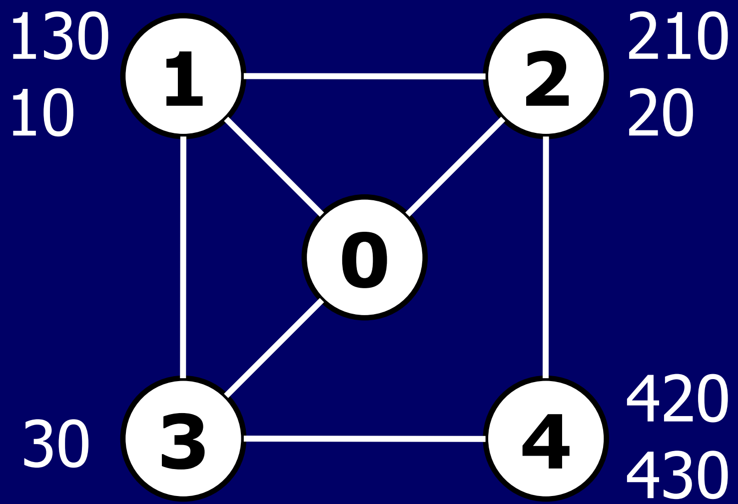


Succinct SPP (SSPP)

[2] T. G. Griffin, F. B. Shepherd, G. Wilfong. The Stable Paths Problem and Interdomain Routing. ToN, 2002.

Models (briefly)

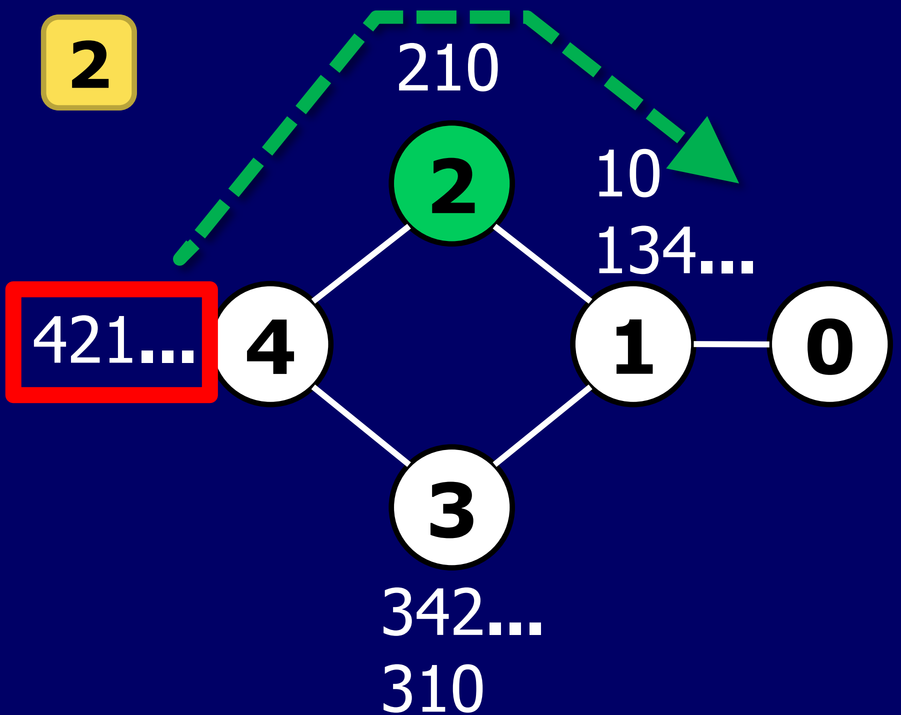
1



Stable Paths Problem (SPP) [2]

- Size: exponential in |V|
- Highly expressive

2



Succinct SPP (SSPP)

- Size: polynomial in |V|
- Close to real configurations

Models (briefly)

1

Stable Paths Problem (SPP) [2]

- Size: exponential in |V|
- Highly expressive

2

Succinct SPP (SSPP)

- Size: polynomial in |V|
- Close to real configurations

Models (briefly)

1

Stable Paths Problem (SPP) [2]

- Size: exponential in |V|
- Highly expressive

2

Succinct SPP (SSPP)

- Size: polynomial in |V|
- Close to real configurations

unique mapping

by *chaining* path fragments

Models (briefly)

1

Stable Paths Problem (SPP) [2]

- Size: exponential in |V|
- Highly expressive

2

Succinct SPP (SSPP)

unique mapping

by *chaining* path fragments

- Size: polynomial in |V|
- Close to real configurations

Our results hold in both models

A Polynomial Time Algorithm for GAO-REXFORD-CHECK

Approach

- Input: instance of (S)SPP

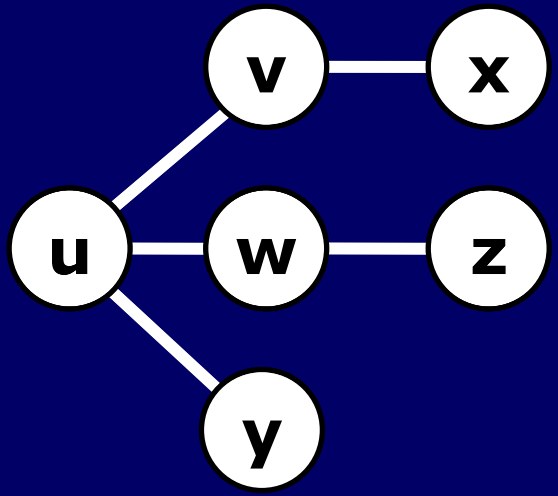
Approach

- Input: instance of (S)SPP
- Consider relation $\prec$
  - $(u, v) \prec (u, w)$ iff u prefers some path starting with (u, v) to some path starting with (u, w)
  - take the transitive closure

Approach

- Input: instance of (S)SPP
- Consider relation $\prec$
  - $(u, v) \prec (u, w)$ iff u prefers some path starting with (u, v) to some path starting with (u, w)
  - take the transitive closure

uvx
uy
uwz



$$(u, v) \prec (u, w)$$

Approach

- Input: instance of (S)SPP
- Consider relation $\prec$
  - $(u, v) \prec (u, w)$ iff u prefers some path starting with (u, v) to some path starting with (u, w)
  - take the transitive closure
  - interpretation: $(u, v) \prec (u, w)$ reads $(u \leftarrow w) \Rightarrow (u \leftarrow v)$

Approach

- Input: instance of (S)SPP
- Consider relation $\prec$
  - $(u, v) \prec (u, w)$ iff u prefers some path starting with (u, v) to some path starting with (u, w)
  - take the transitive closure
  - interpretation: $(u, v) \prec (u, w)$ reads $(u \leftarrow w) \Rightarrow (u \leftarrow v)$
- Can the input graph be partially oriented to an acyclic customer-provider graph such that paths are valley-free and $\prec$ constraints are honored?

Approach

- Inspired by [12]

[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

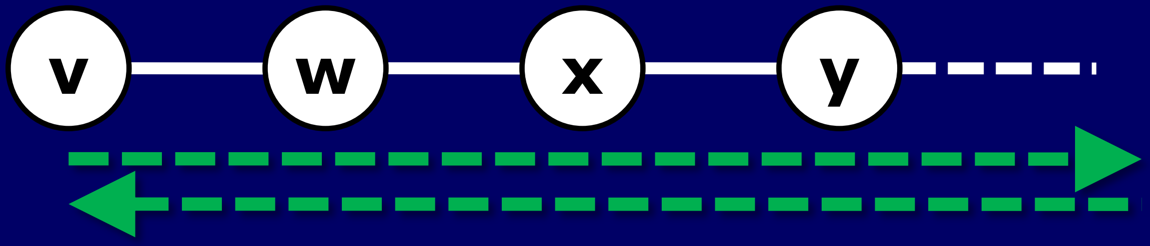
Approach

- Inspired by [12]
  - Find a v that

[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

Approach

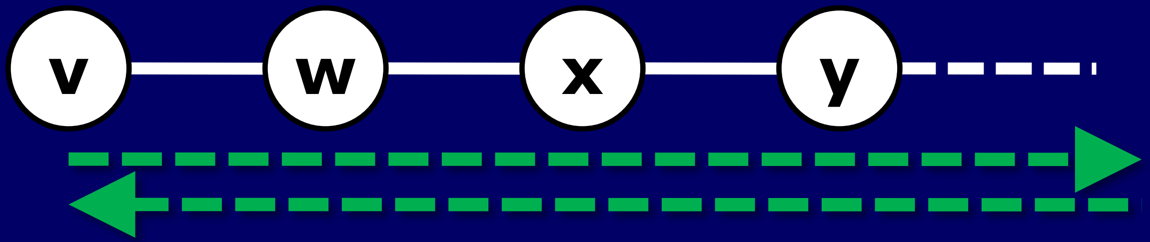
- Inspired by [12]
  - Find a v that
    - never appears as an internal node in any paths



[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

Approach

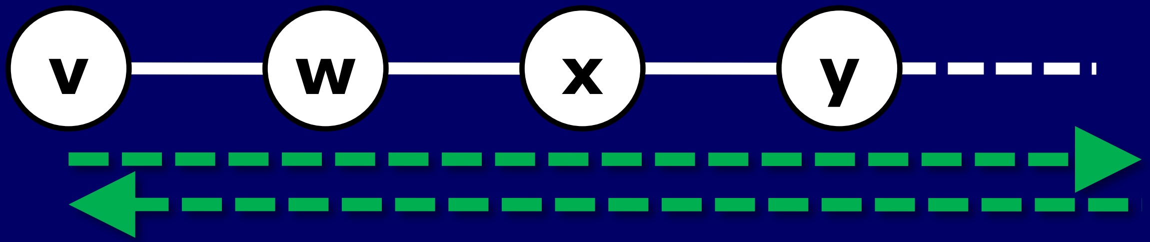
- Inspired by [12]
  - Find a v that
    - never appears as an internal node in any paths
    - does not have incoming edges



[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

Approach

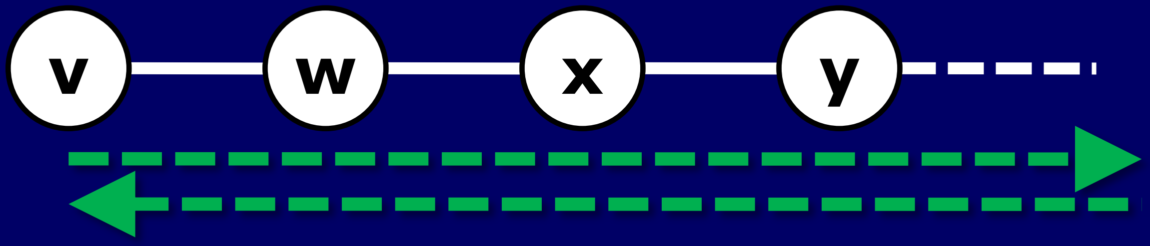
- Inspired by [12]
  - Find a v that
    - never appears as an internal node in any paths
    - does not have incoming edges
      - one must exist in any GR-compliant orientation



[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

Approach

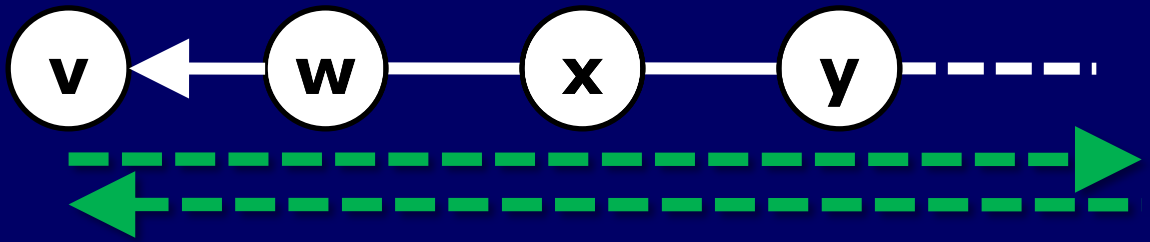
- Inspired by [12]
  - Find a v that
    - never appears as an internal node in any paths
    - does not have incoming edges
      - one must exist in any GR-compliant orientation
  - Orient edges away from v



[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

Approach

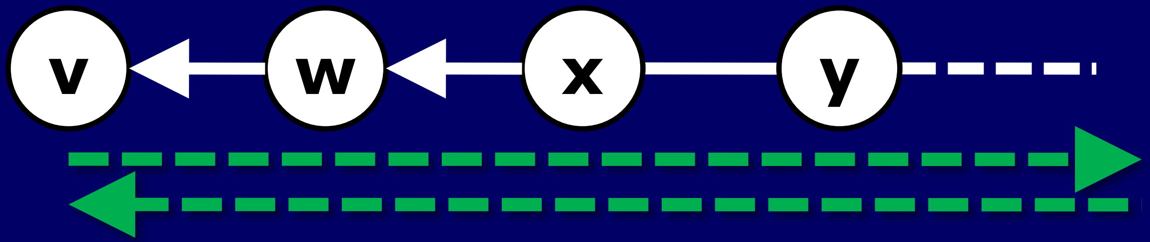
- Inspired by [12]
  - Find a v that
    - never appears as an internal node in any paths
    - does not have incoming edges
      - one must exist in any GR-compliant orientation
  - Orient edges away from v



[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

Approach

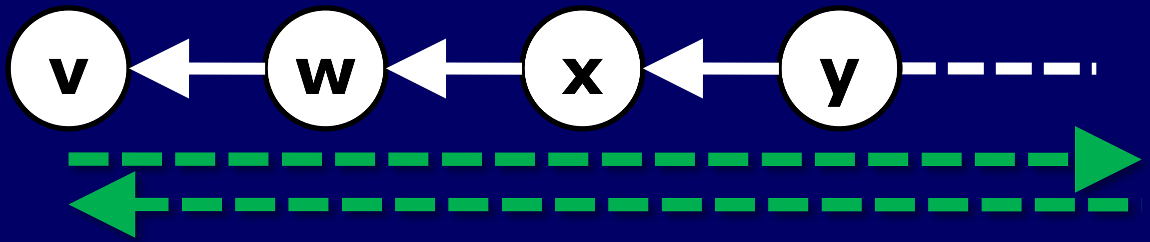
- Inspired by [12]
  - Find a v that
    - never appears as an internal node in any paths
    - does not have incoming edges
      - one must exist in any GR-compliant orientation
  - Orient edges away from v



[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

Approach

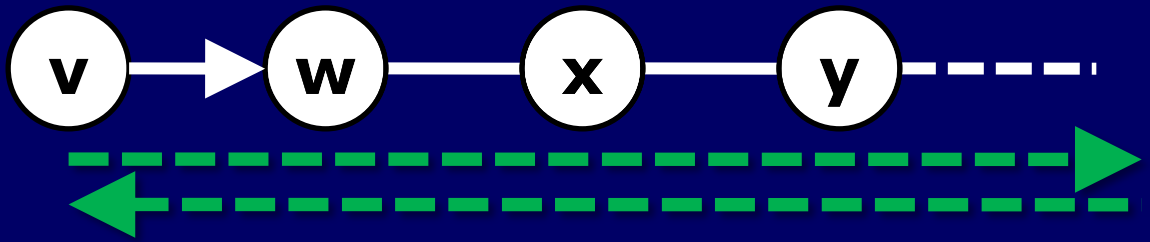
- Inspired by [12]
  - Find a v that
    - never appears as an internal node in any paths
    - does not have incoming edges
      - one must exist in any GR-compliant orientation
  - Orient edges away from v



[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

Approach

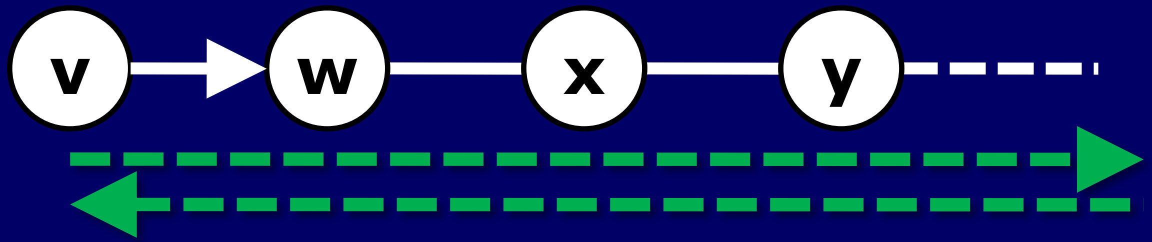
- Inspired by [12]
  - Find a v that
    - never appears as an internal node in any paths
    - does not have incoming edges
      - one must exist in any GR-compliant orientation
  - Orient edges away from v



[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

Approach

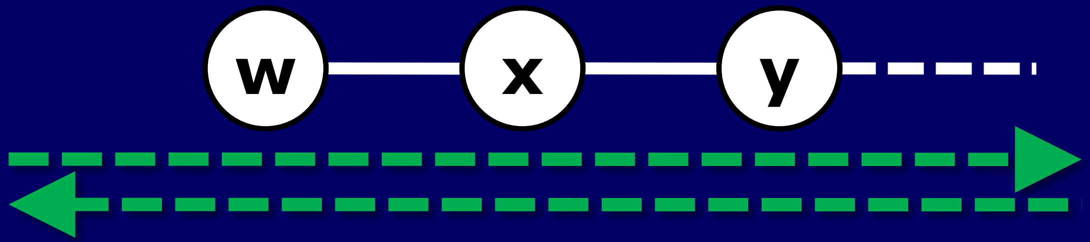
- Inspired by [12]
  - Find a v that
    - never appears as an internal node in any paths
    - does not have incoming edges
      - one must exist in any GR-compliant orientation
  - Orient edges away from v
  - Recursive call



[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

Approach

- Inspired by [12]
  - Find a v that
    - never appears as an internal node in any paths
    - does not have incoming edges
      - – one must exist in any GR-compliant orientation
  - Orient edges away from v
  - Recursive call



[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

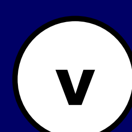
Approach

- Inspired by [12]
  - Find a v that
    - never appears as an internal node in any paths
    - does not have incoming edges
      - one must exist in any GR-compliant orientation
  - Orient edges away from v
  - Recursive call

- Not that easy due to $\prec$ constraints...

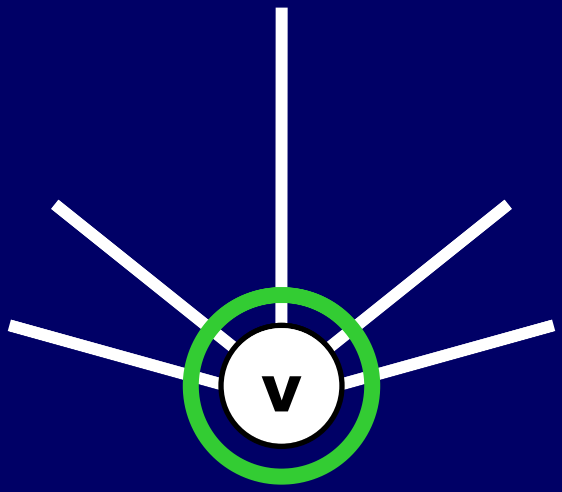
[12] S. Kosub, M. G. Maaß, H. Täubig. Acyclic Type-of-Relationship Problems on the Internet. CAAN 2006

The Algorithm

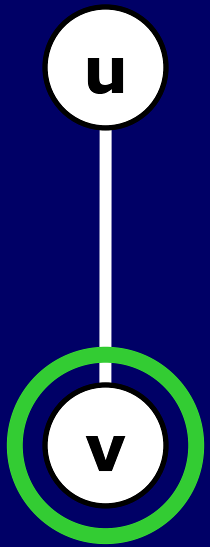


The Algorithm

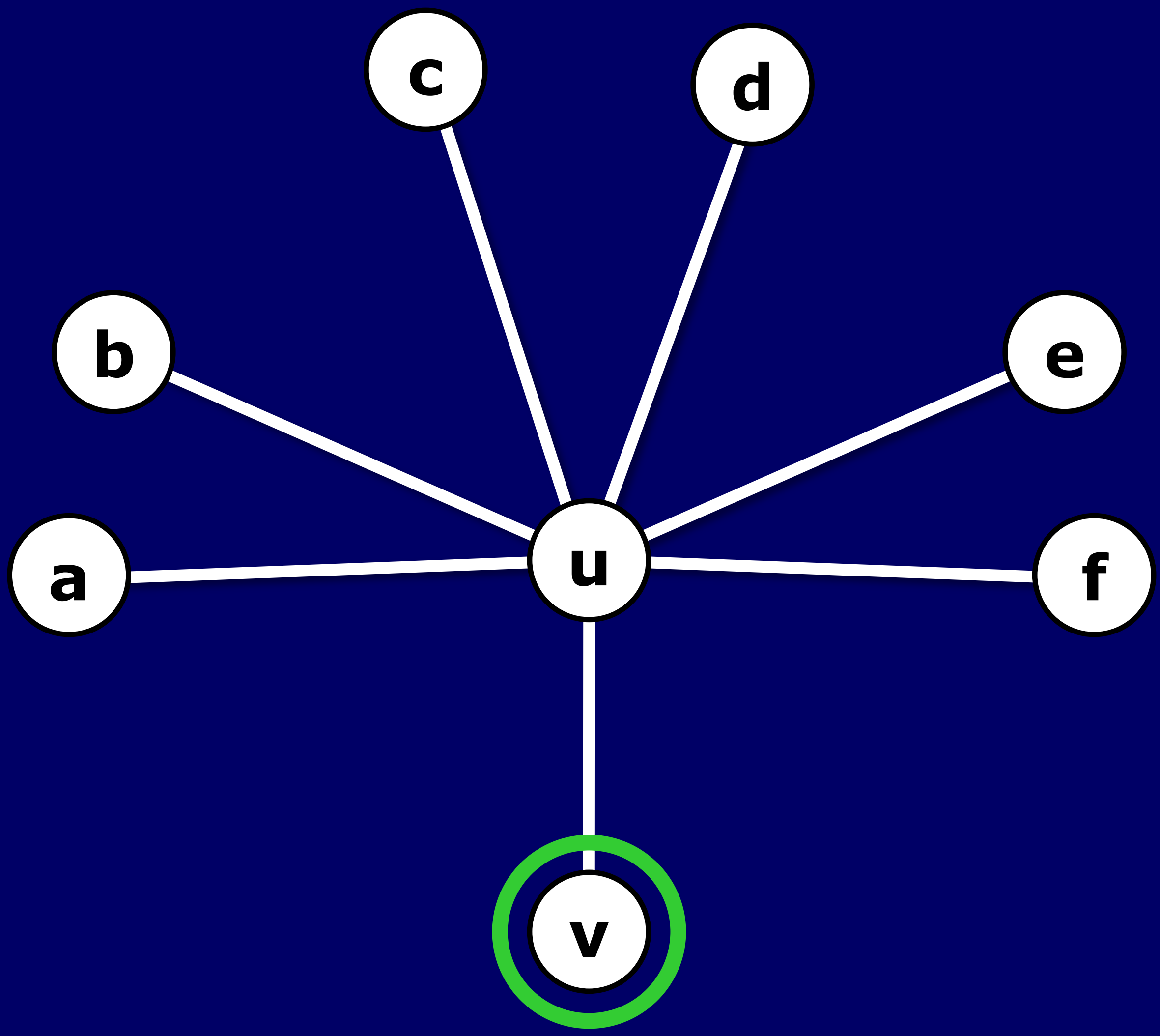
The Algorithm



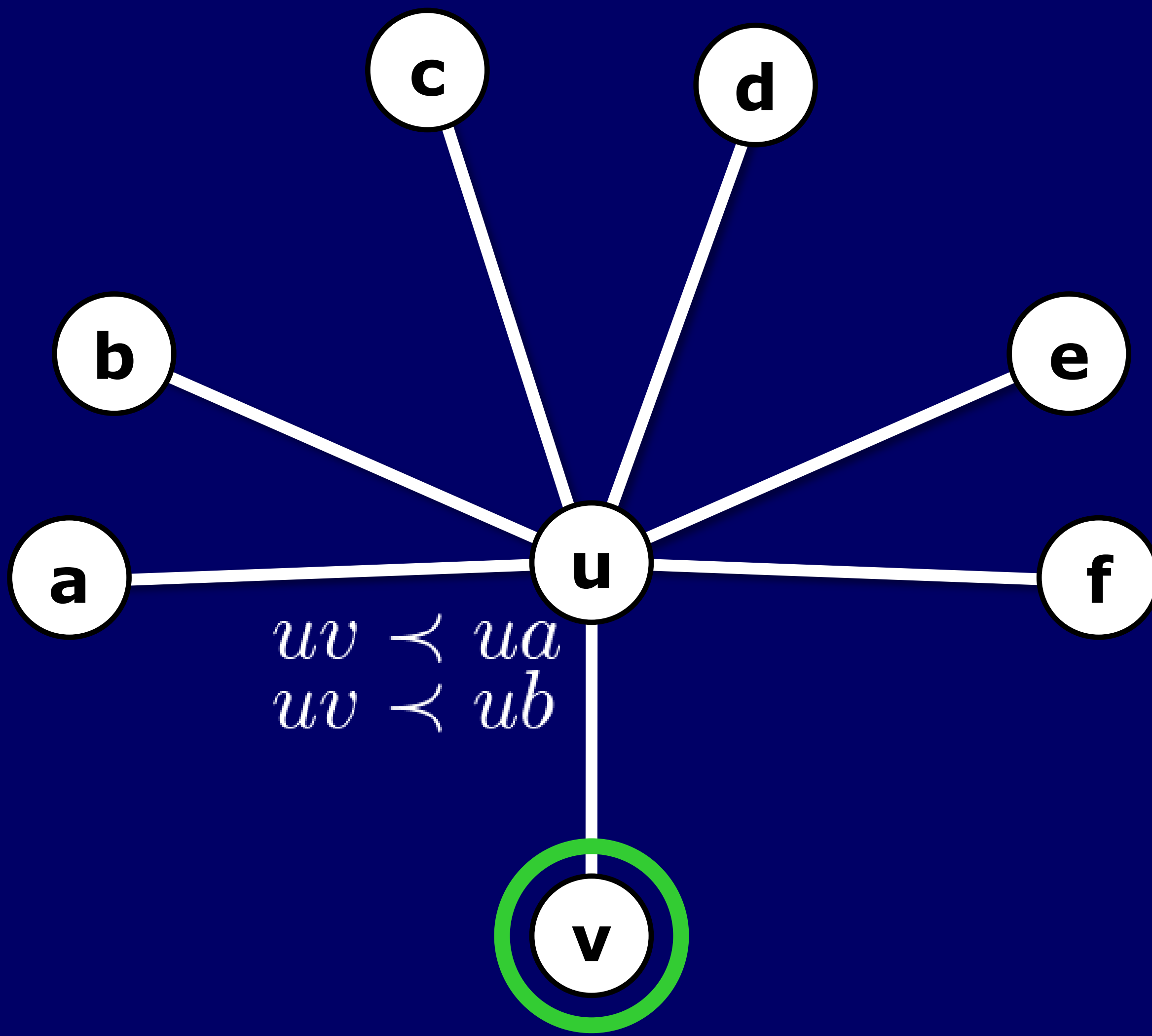
The Algorithm



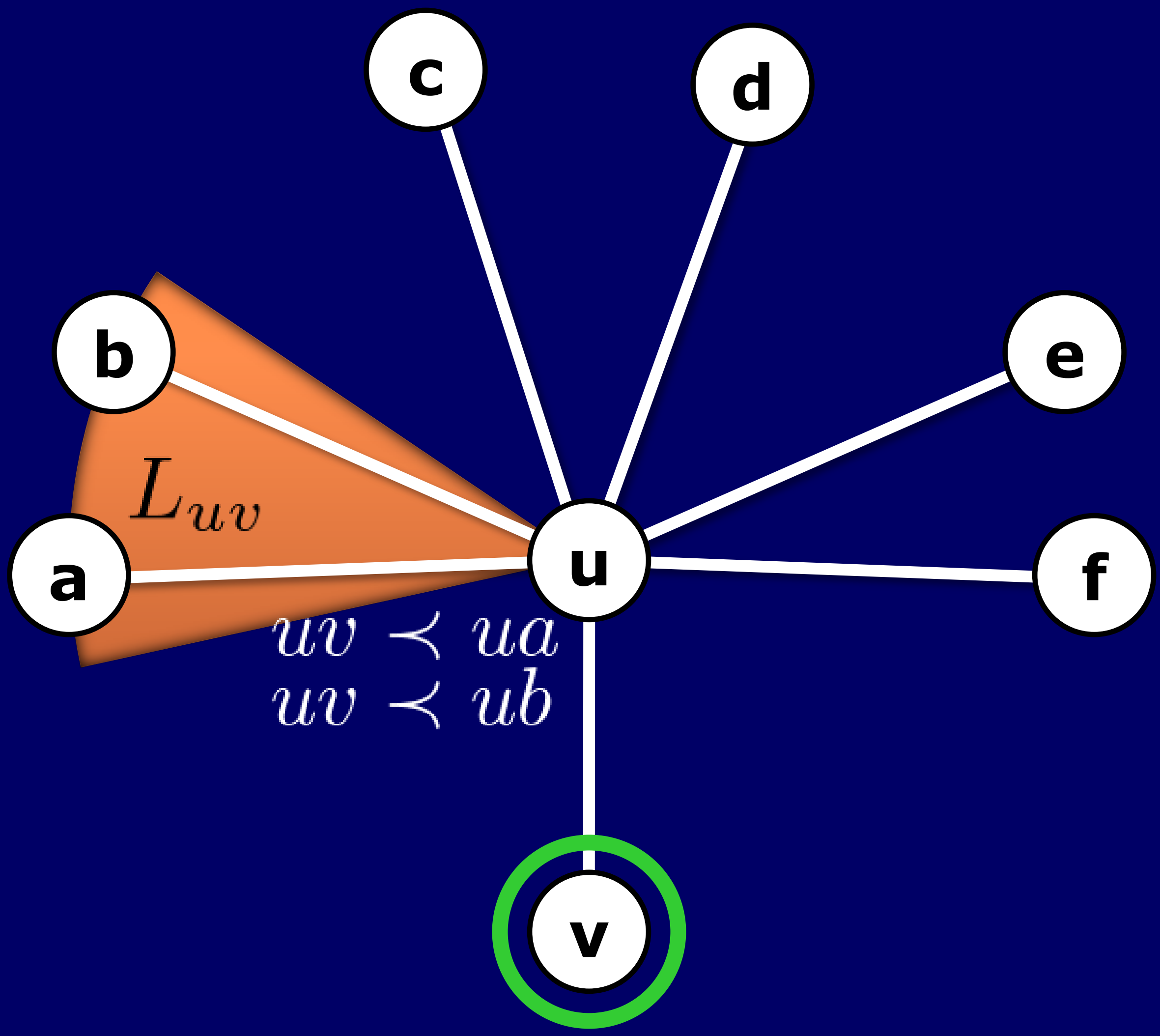
The Algorithm



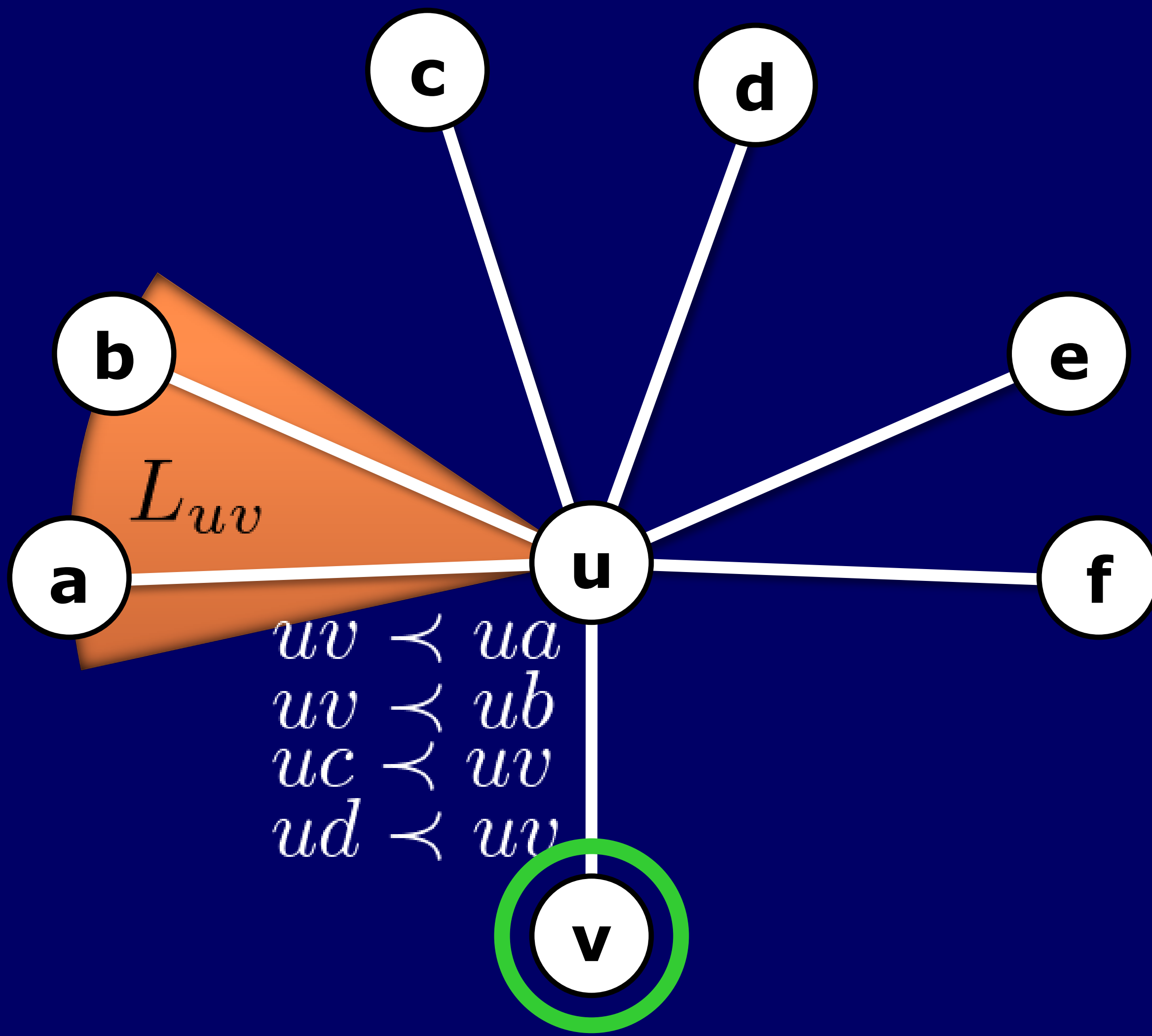
The Algorithm



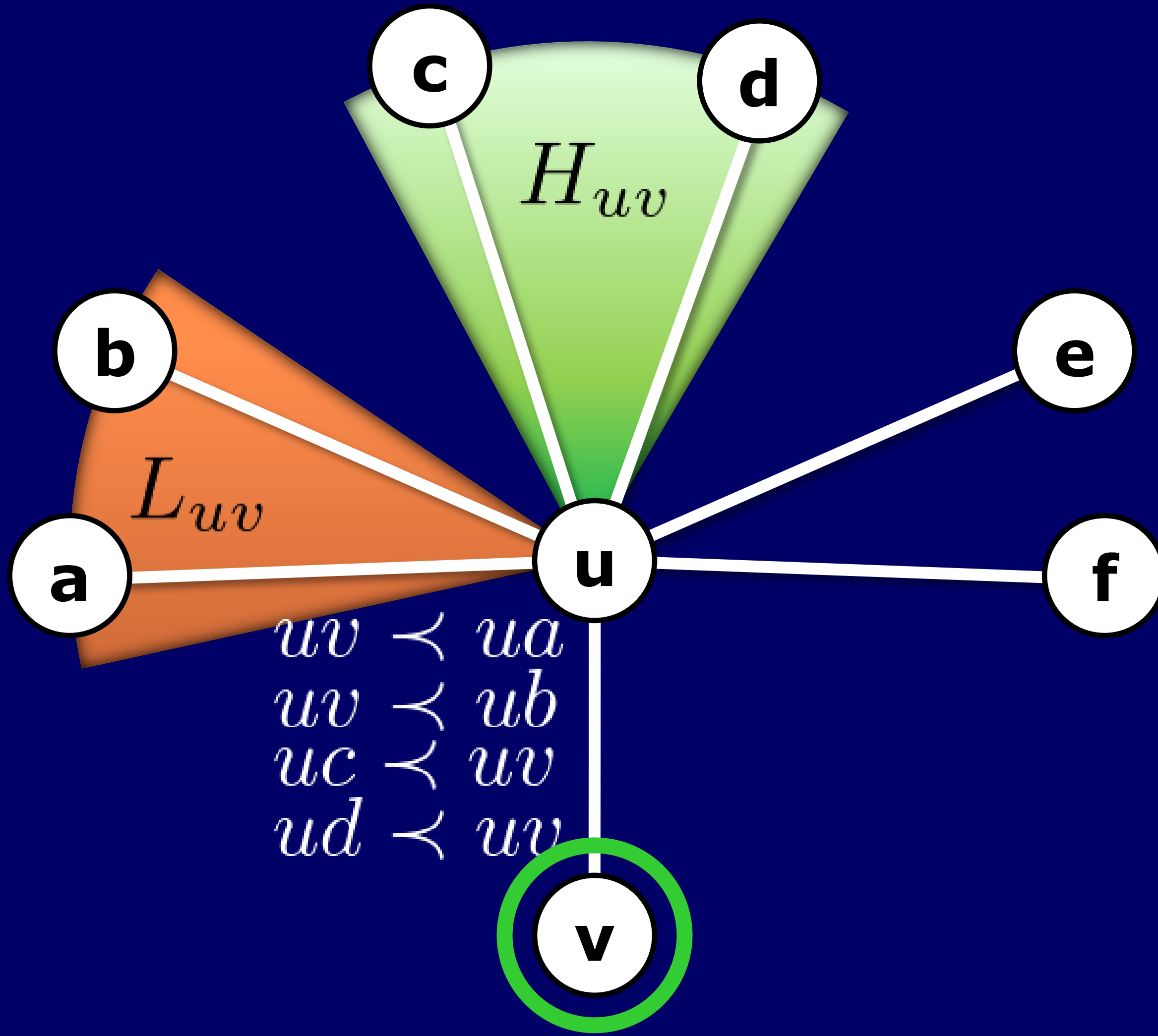
The Algorithm



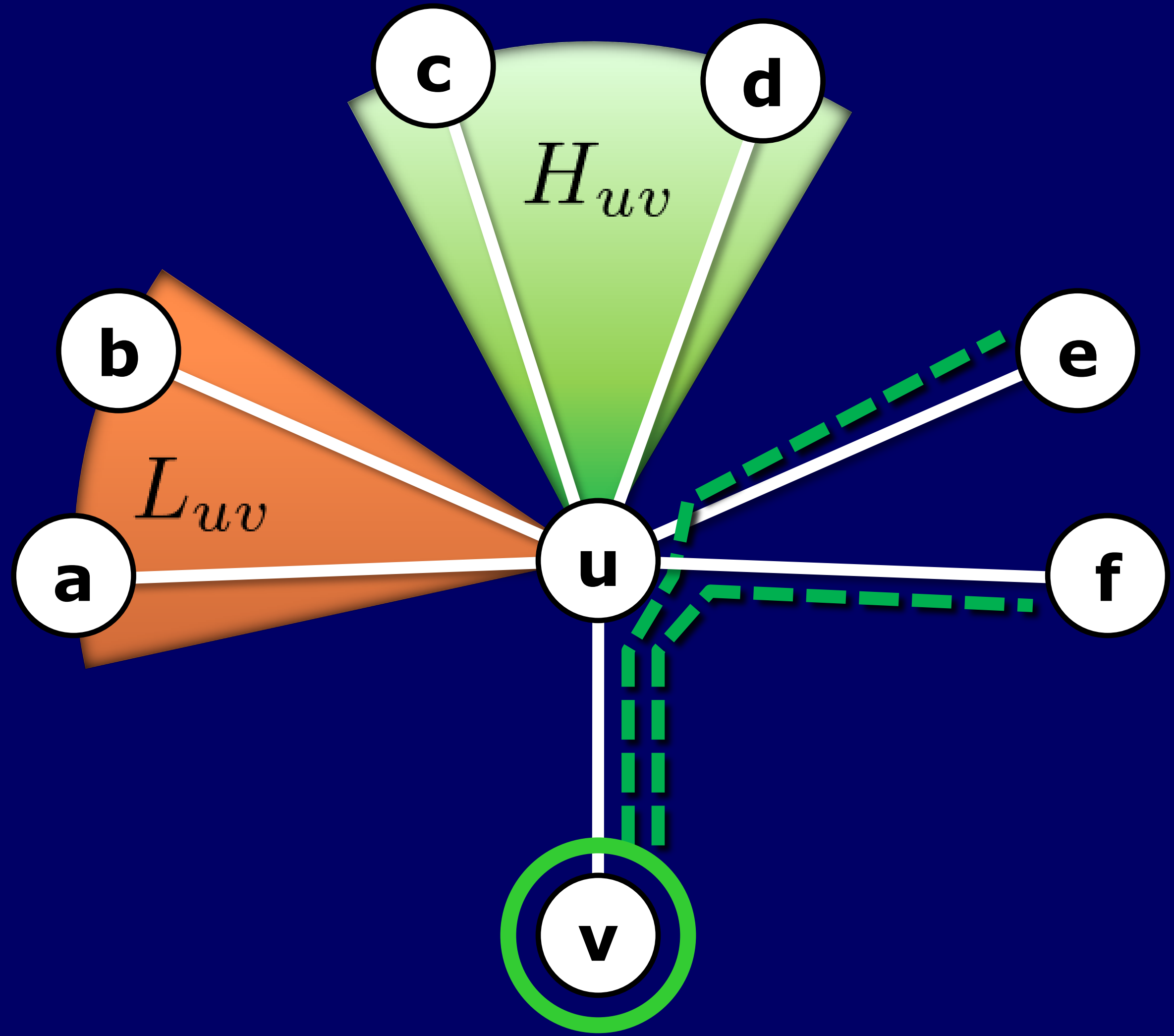
The Algorithm



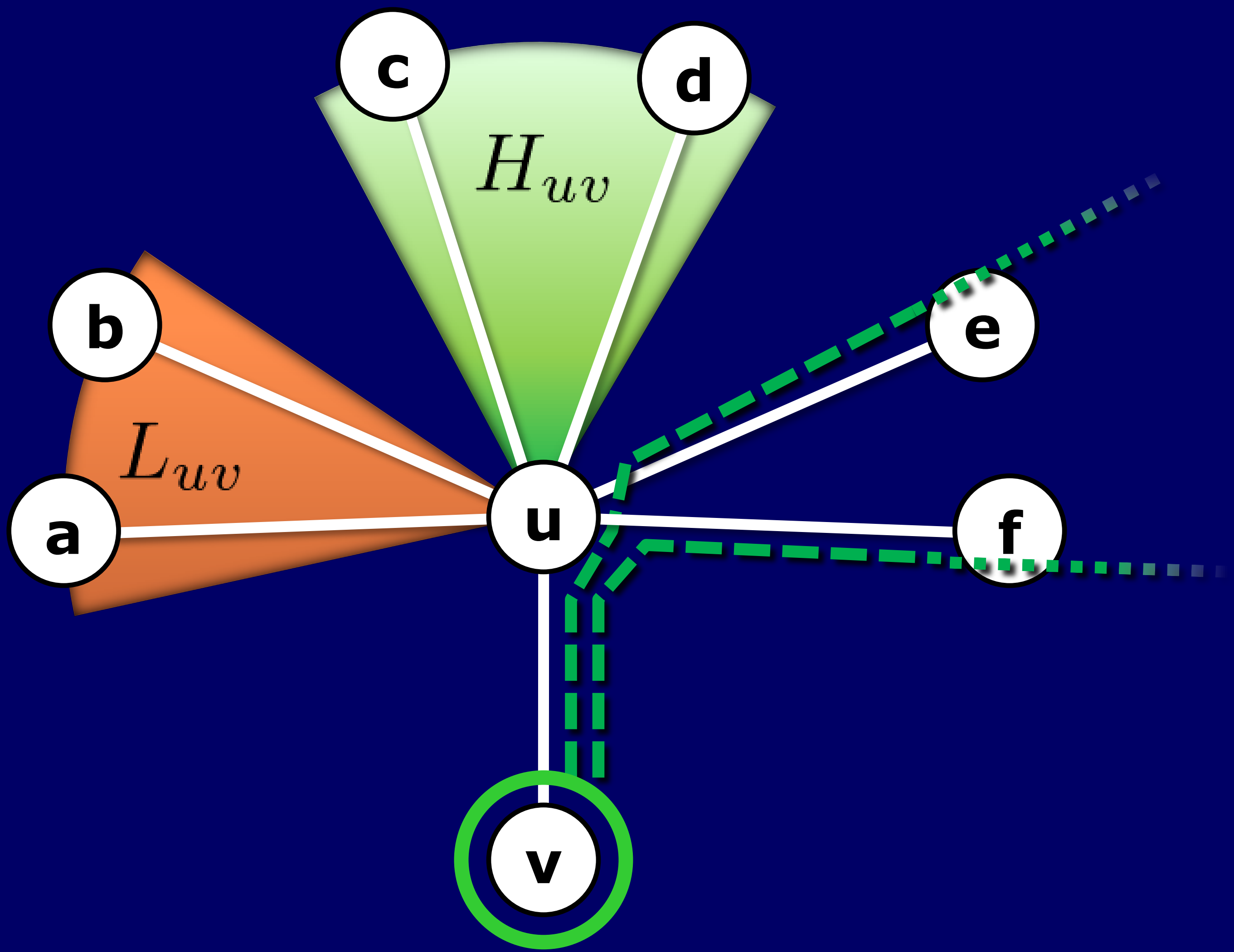
The Algorithm



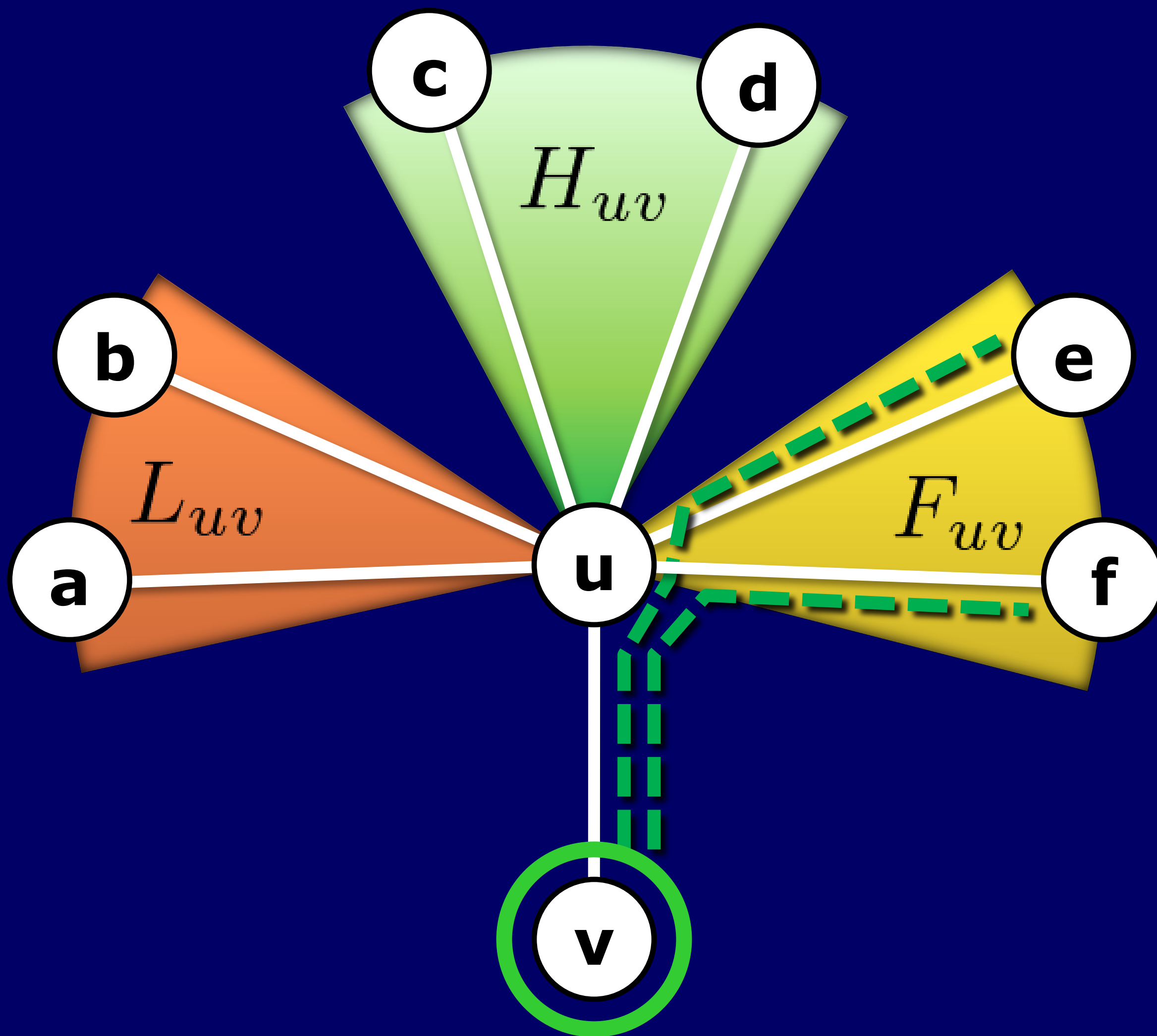
The Algorithm



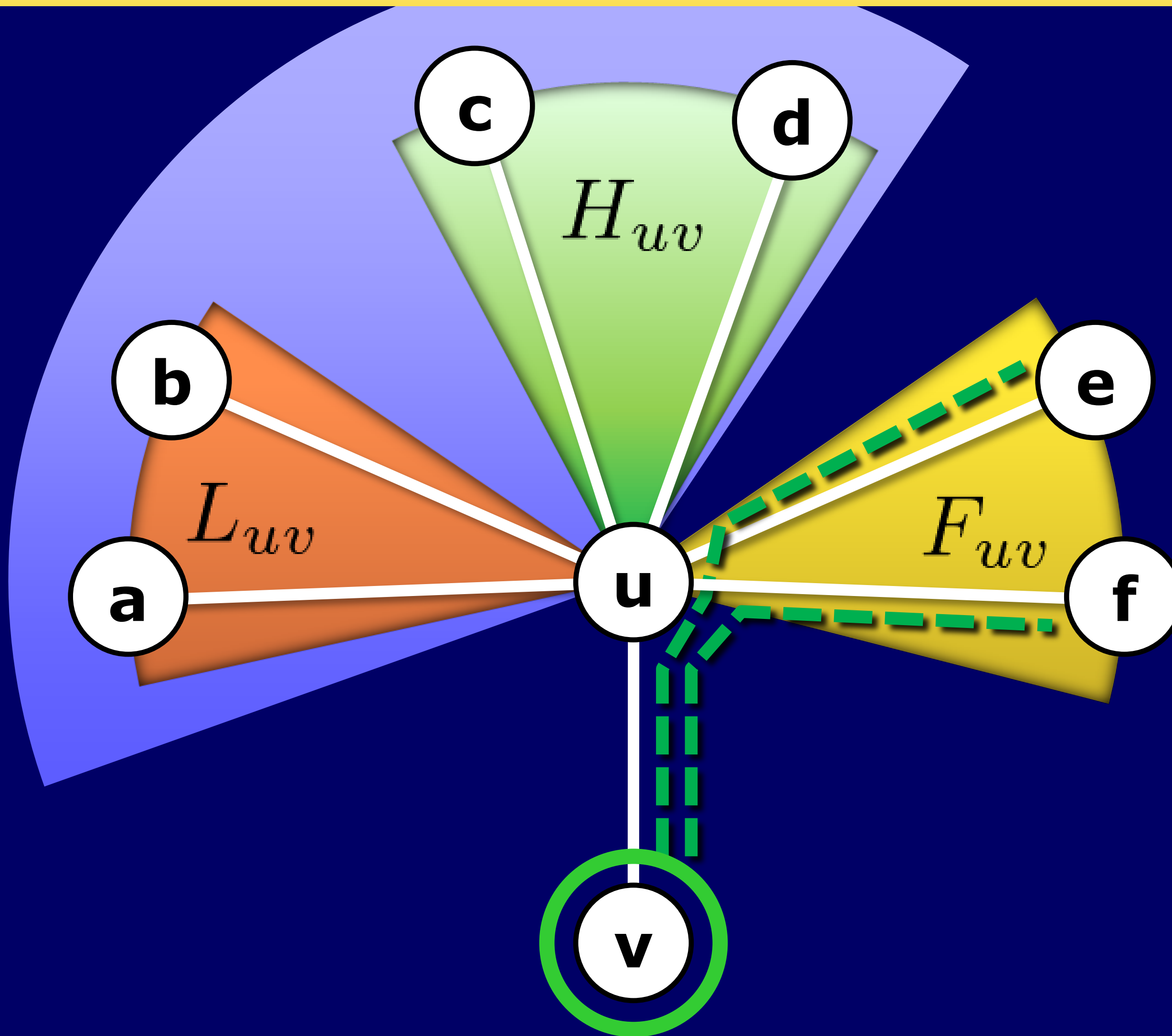
The Algorithm



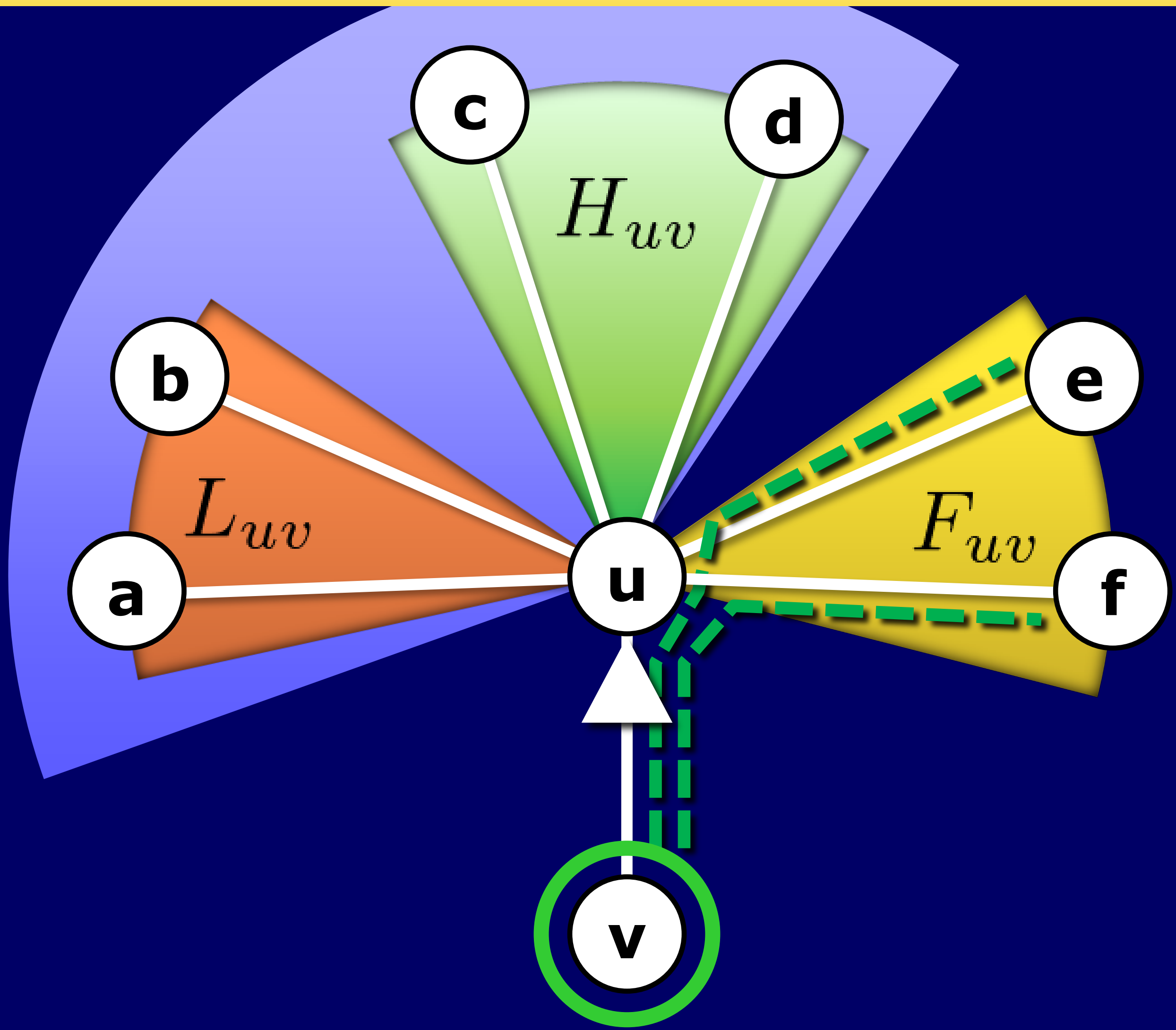
The Algorithm



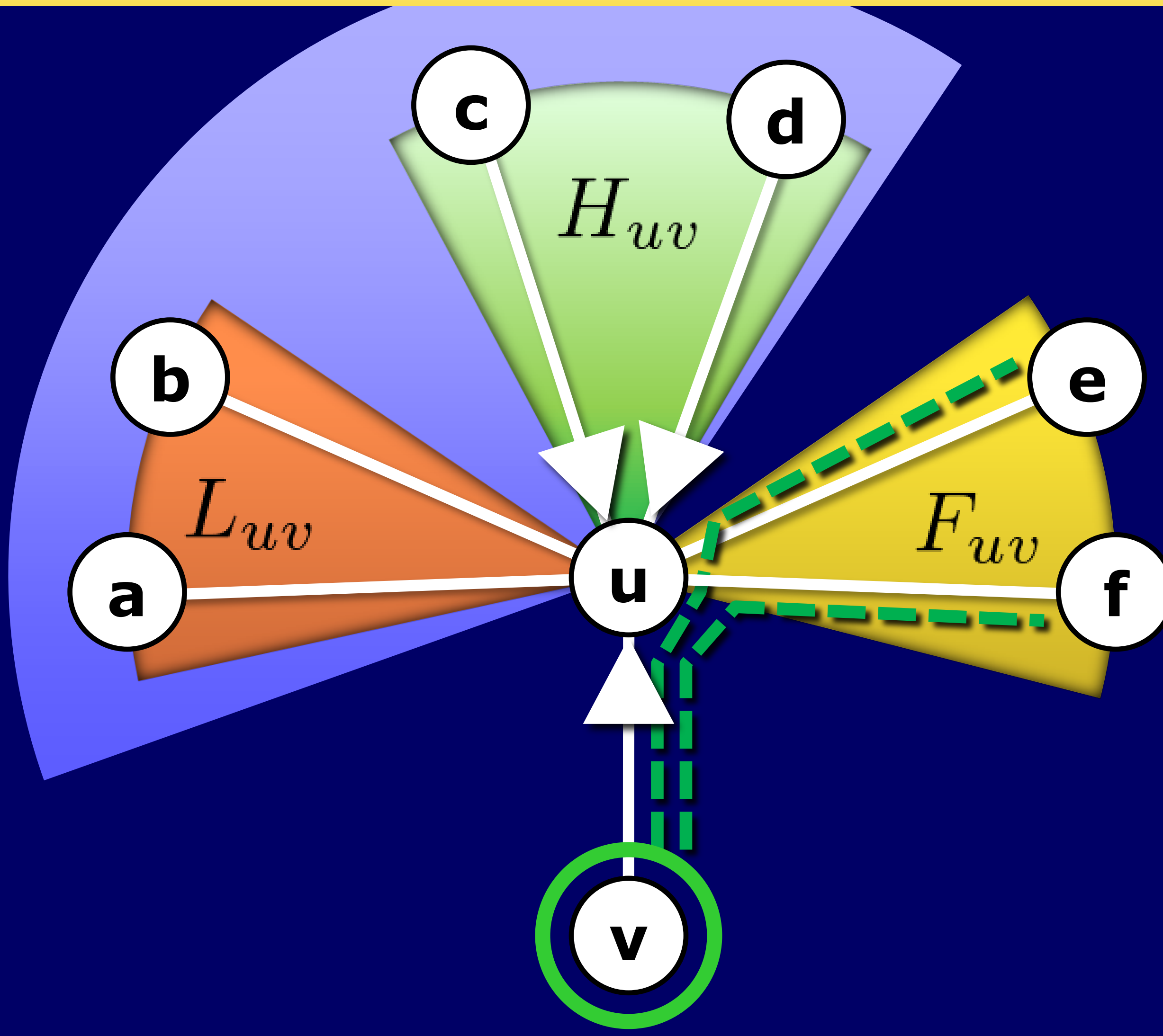
The Algorithm



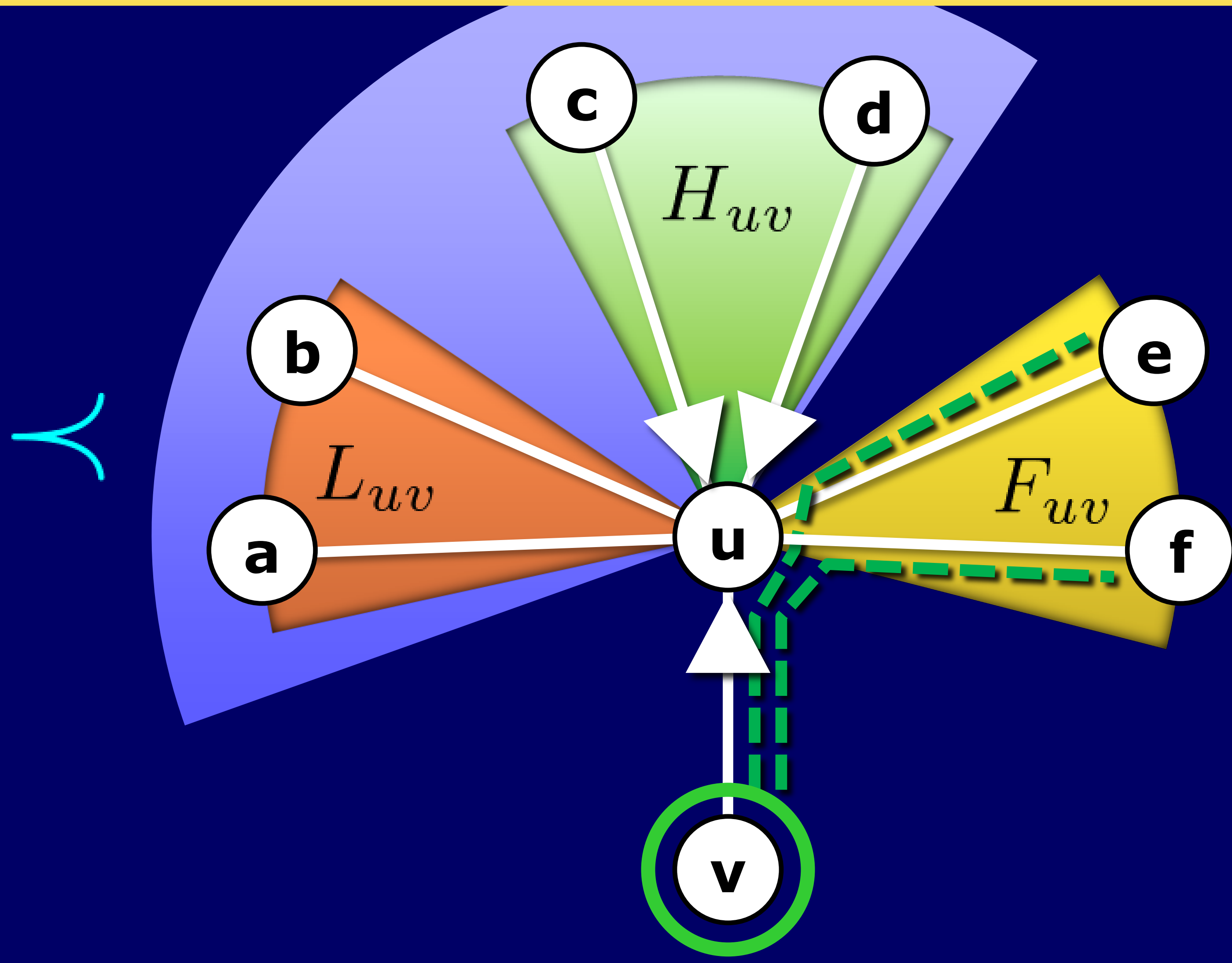
The Algorithm



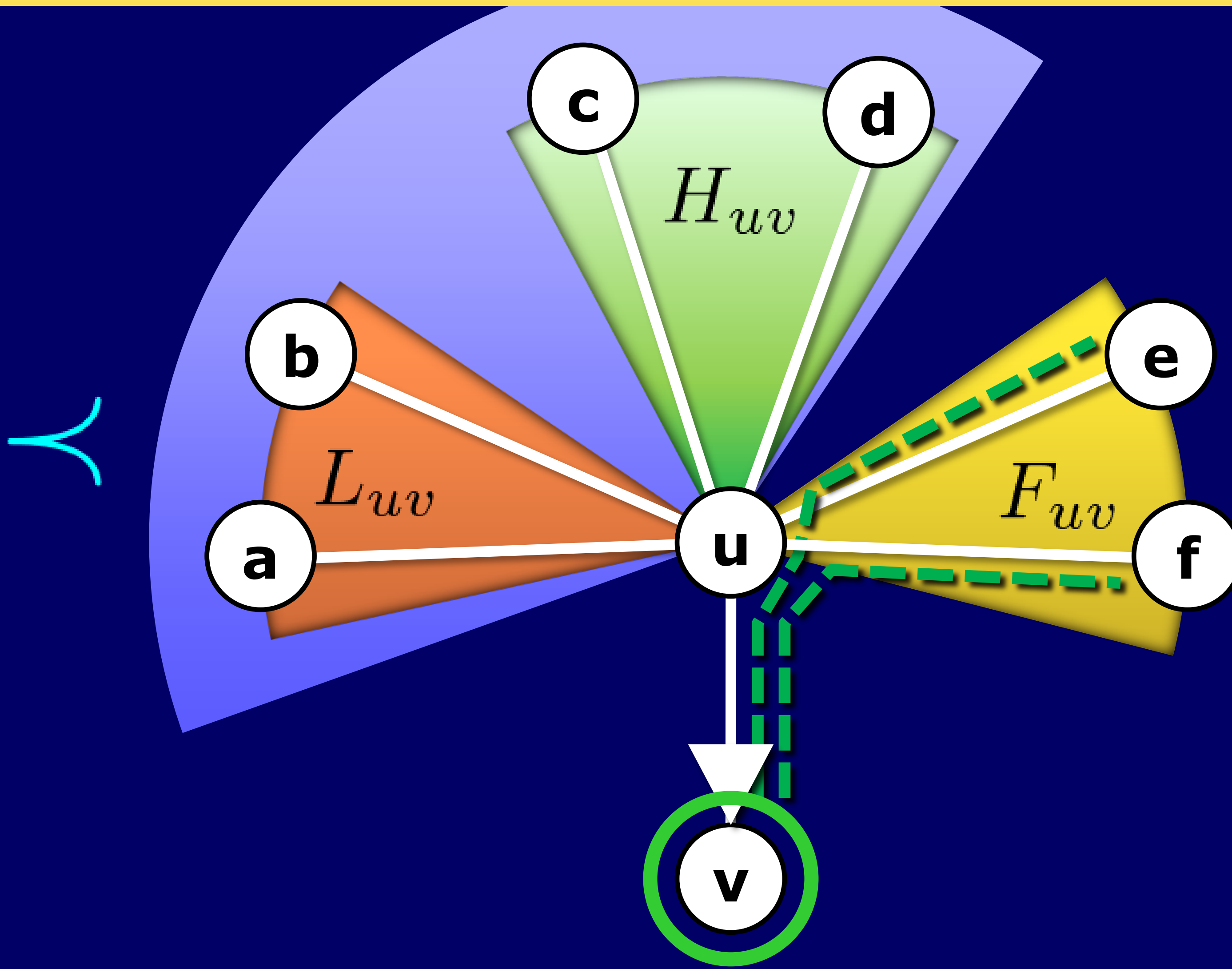
The Algorithm



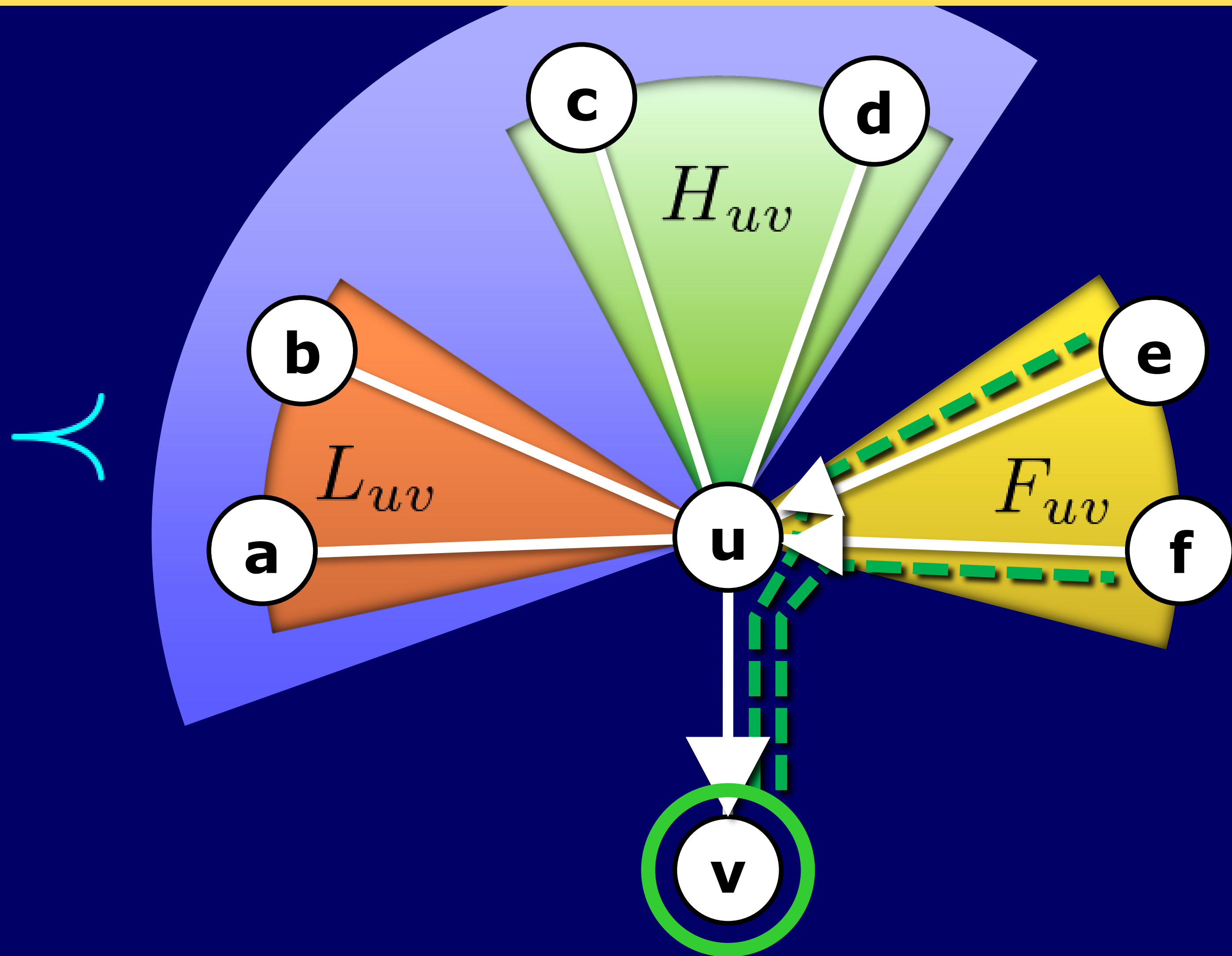
The Algorithm



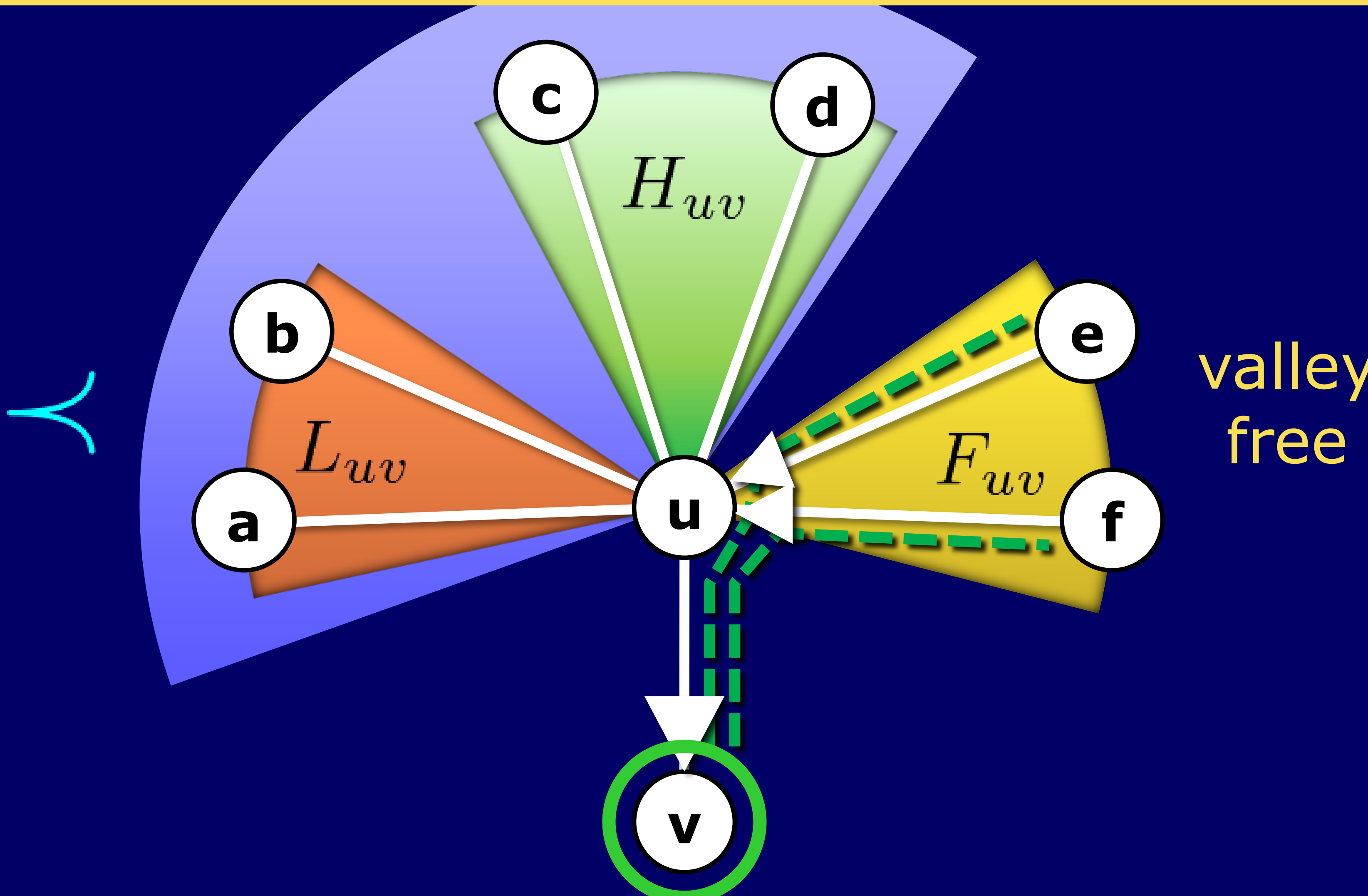
The Algorithm



The Algorithm



The Algorithm



Forced Orientations

- All edges in $H_{uv} \cap F_{uv}$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $L_{uv} \cap F_{uv} \neq \emptyset$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $\exists (u \leftarrow w) \in L_{uv}$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $\exists (u \rightarrow w) \in F_{uv}$

Forced Orientations

- All edges in $H_{uv} \cap F_{uv}$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $L_{uv} \cap F_{uv} \neq \emptyset$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $\exists (u \leftarrow w) \in L_{uv}$

by $\prec$

- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $\exists (u \rightarrow w) \in F_{uv}$

Forced Orientations

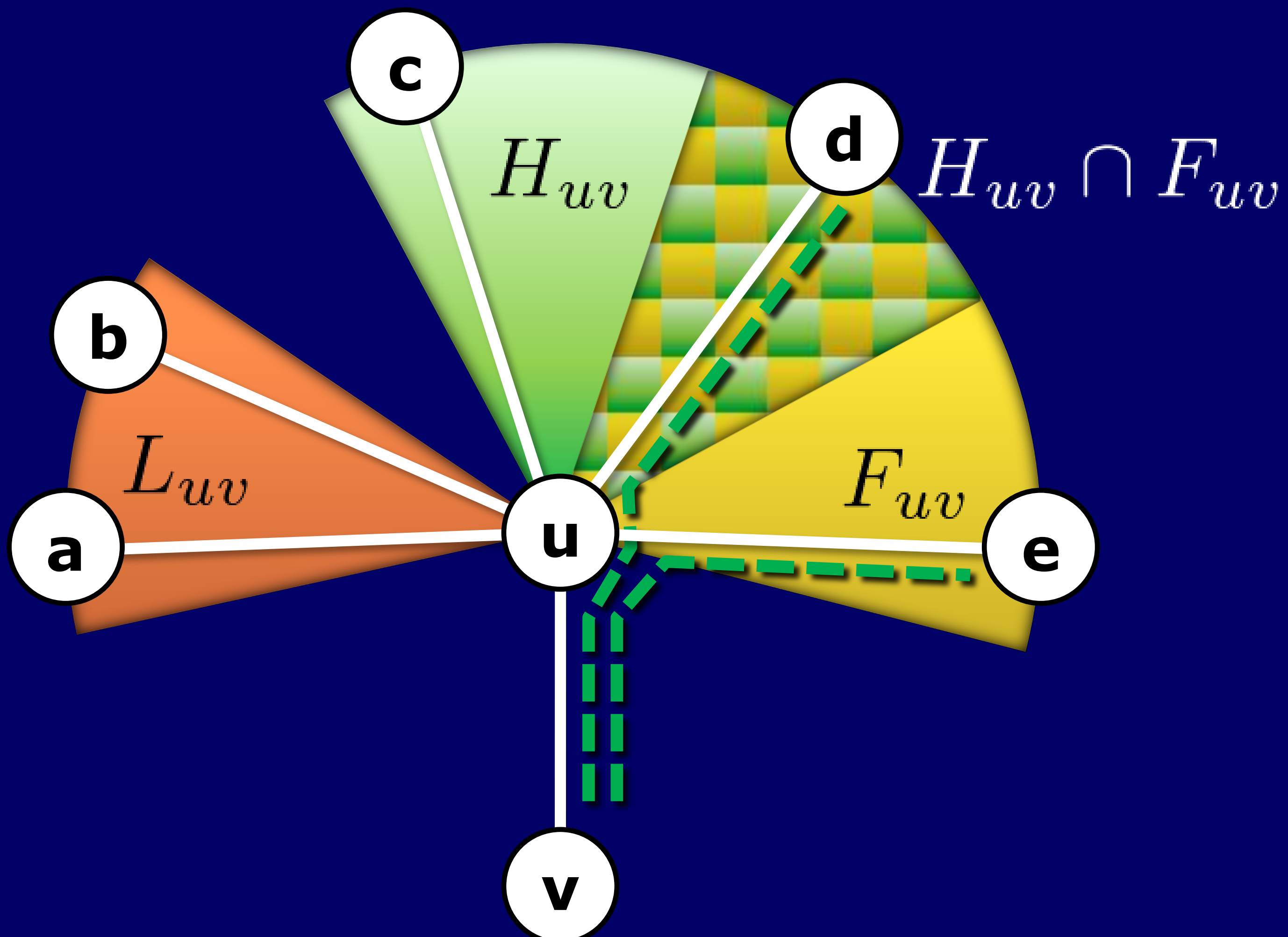
- All edges in $H_{uv} \cap F_{uv}$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $L_{uv} \cap F_{uv} \neq \emptyset$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $\exists (u \leftarrow w) \in L_{uv}$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $\exists (u \rightarrow w) \in F_{uv}$

by valley freeness

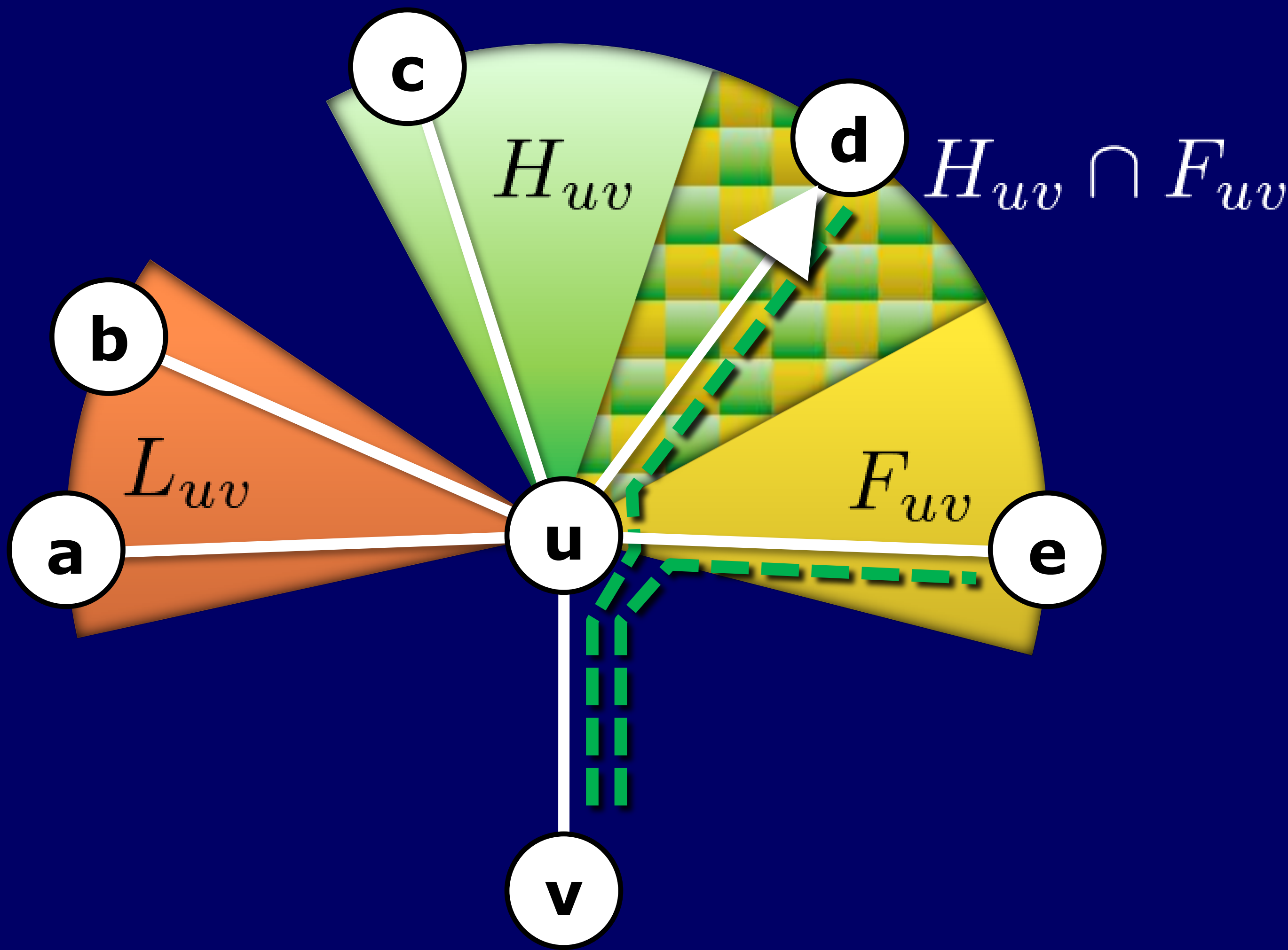
Forced Orientations

- All edges in $H_{uv} \cap F_{uv}$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $L_{uv} \cap F_{uv} \neq \emptyset$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $\exists (u \leftarrow w) \in L_{uv}$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $\exists (u \rightarrow w) \in F_{uv}$

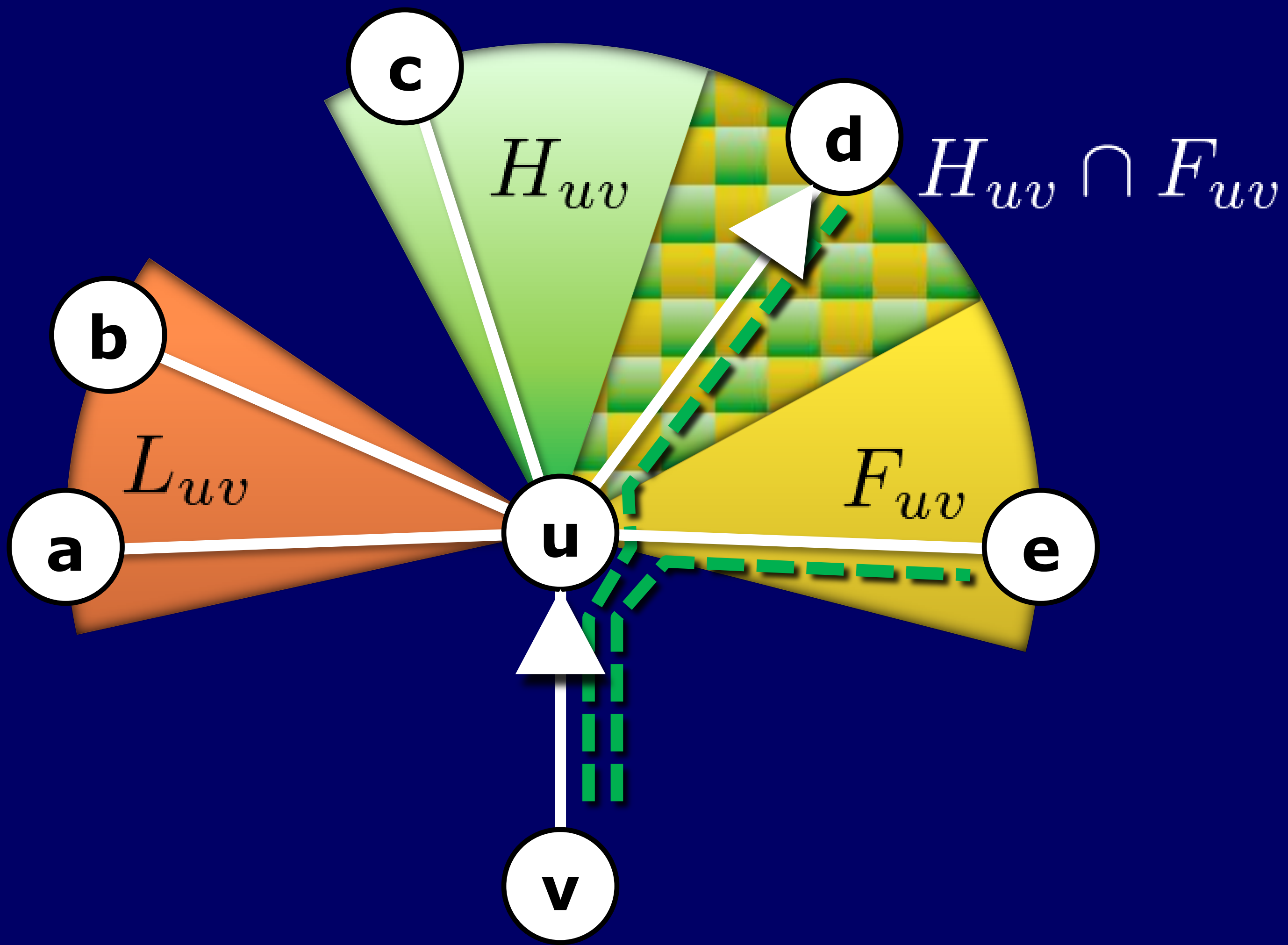
Forced Orientations



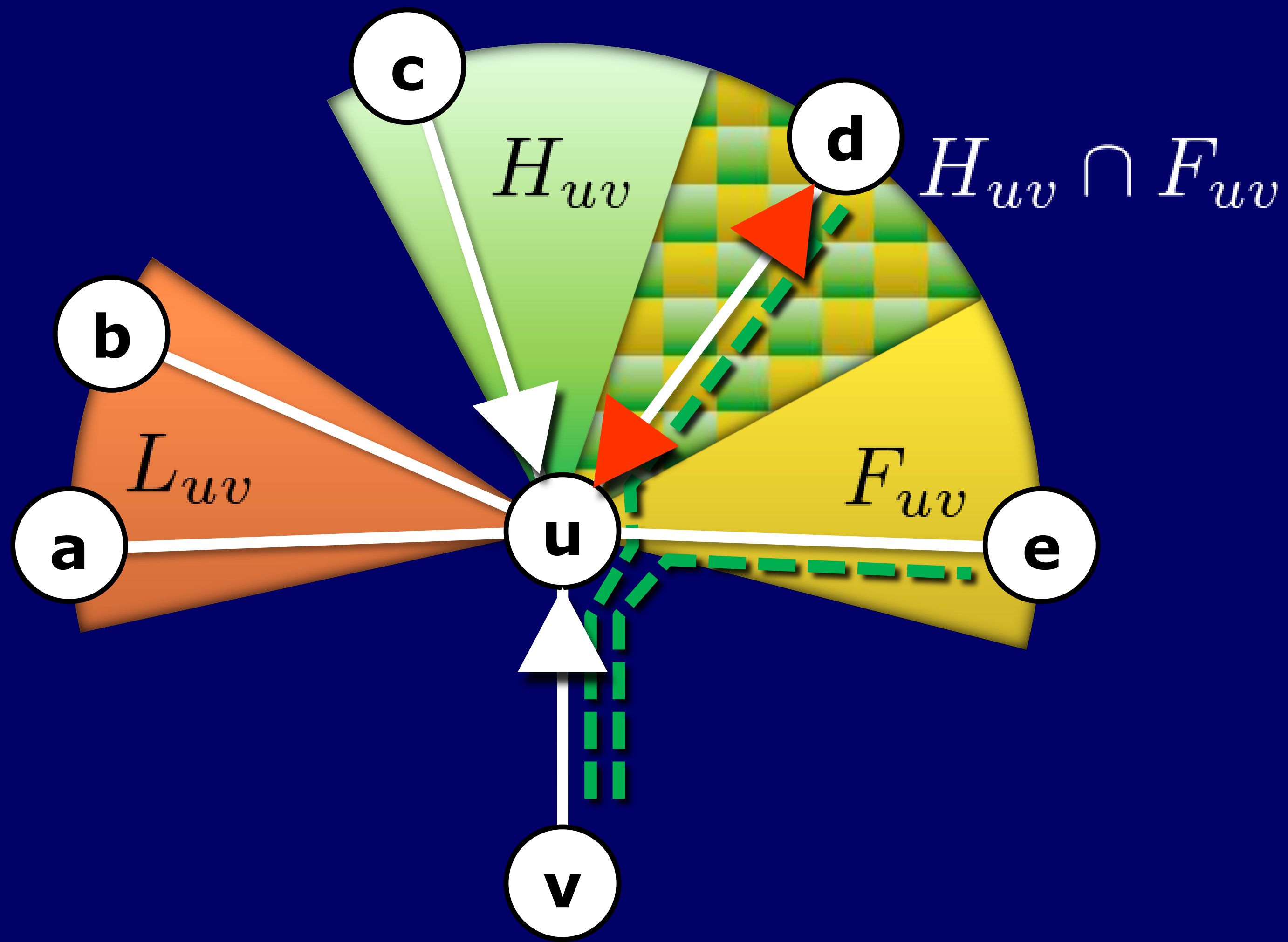
Forced Orientations



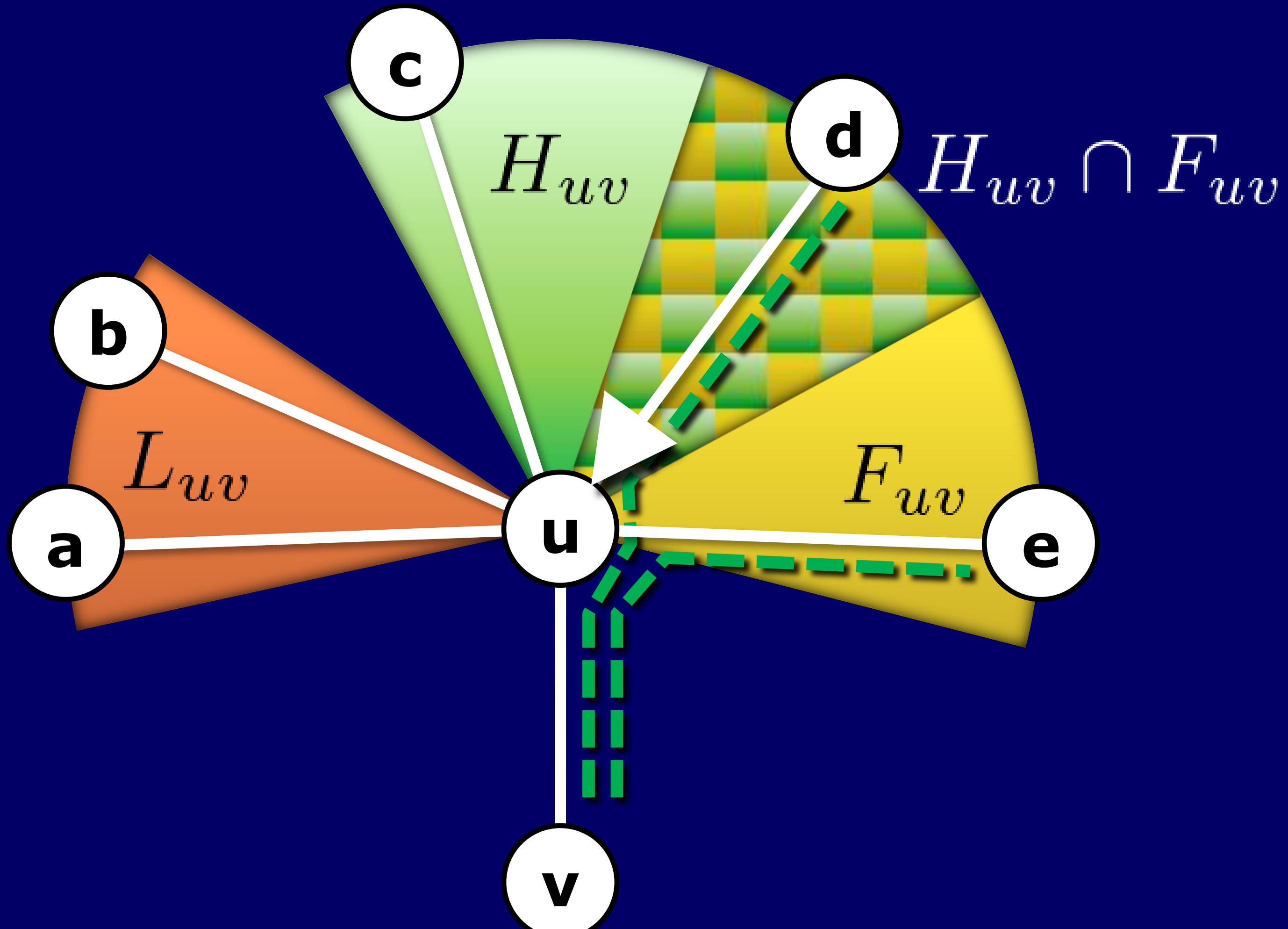
Forced Orientations



Forced Orientations



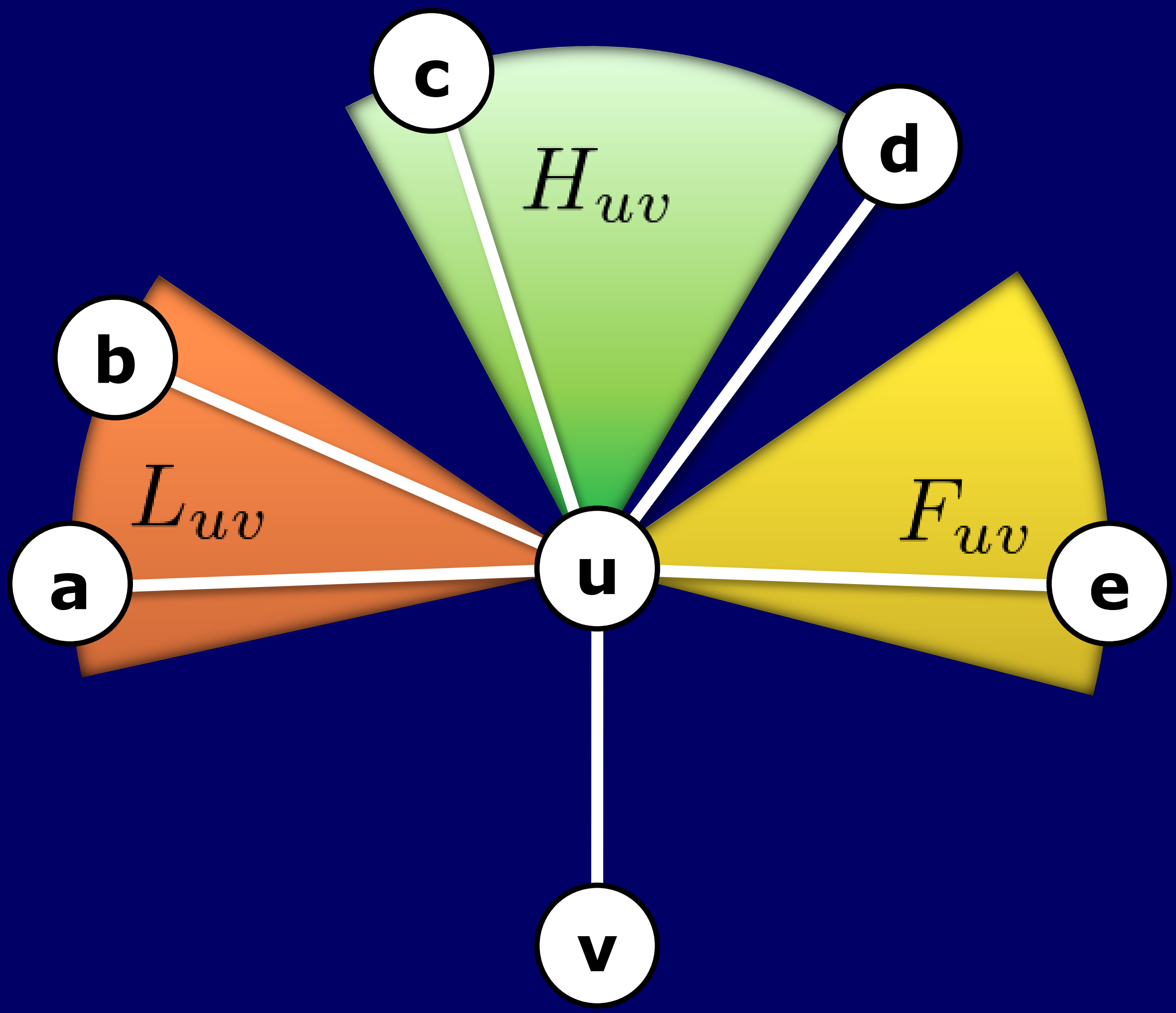
Forced Orientations



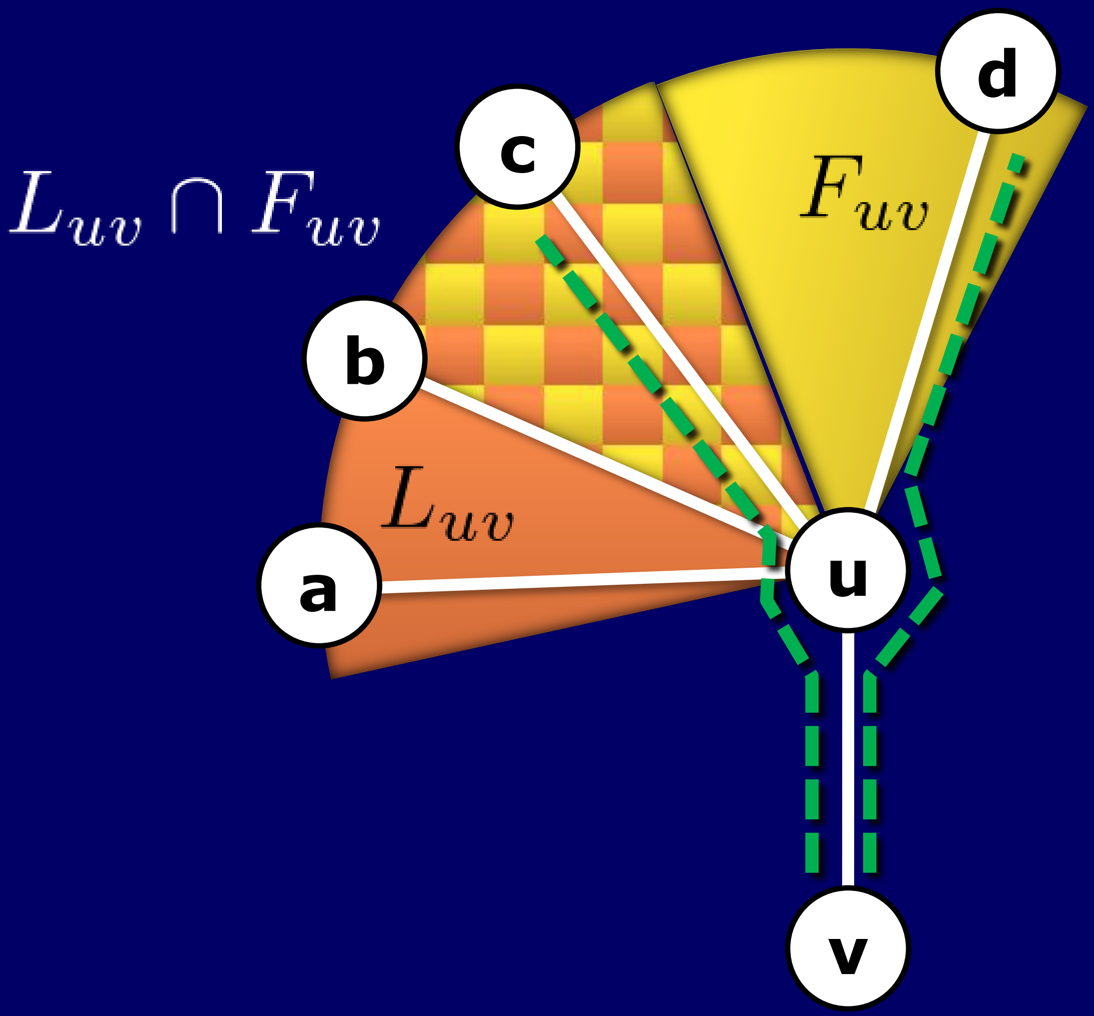
Forced Orientations

- All edges in $H_{uv} \cap F_{uv}$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $L_{uv} \cap F_{uv} \neq \emptyset$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $\exists (u \leftarrow w) \in L_{uv}$
- $(v \rightarrow u)$ and $(u \leftarrow x) \in H_{uv}$ if $\exists (u \rightarrow w) \in F_{uv}$

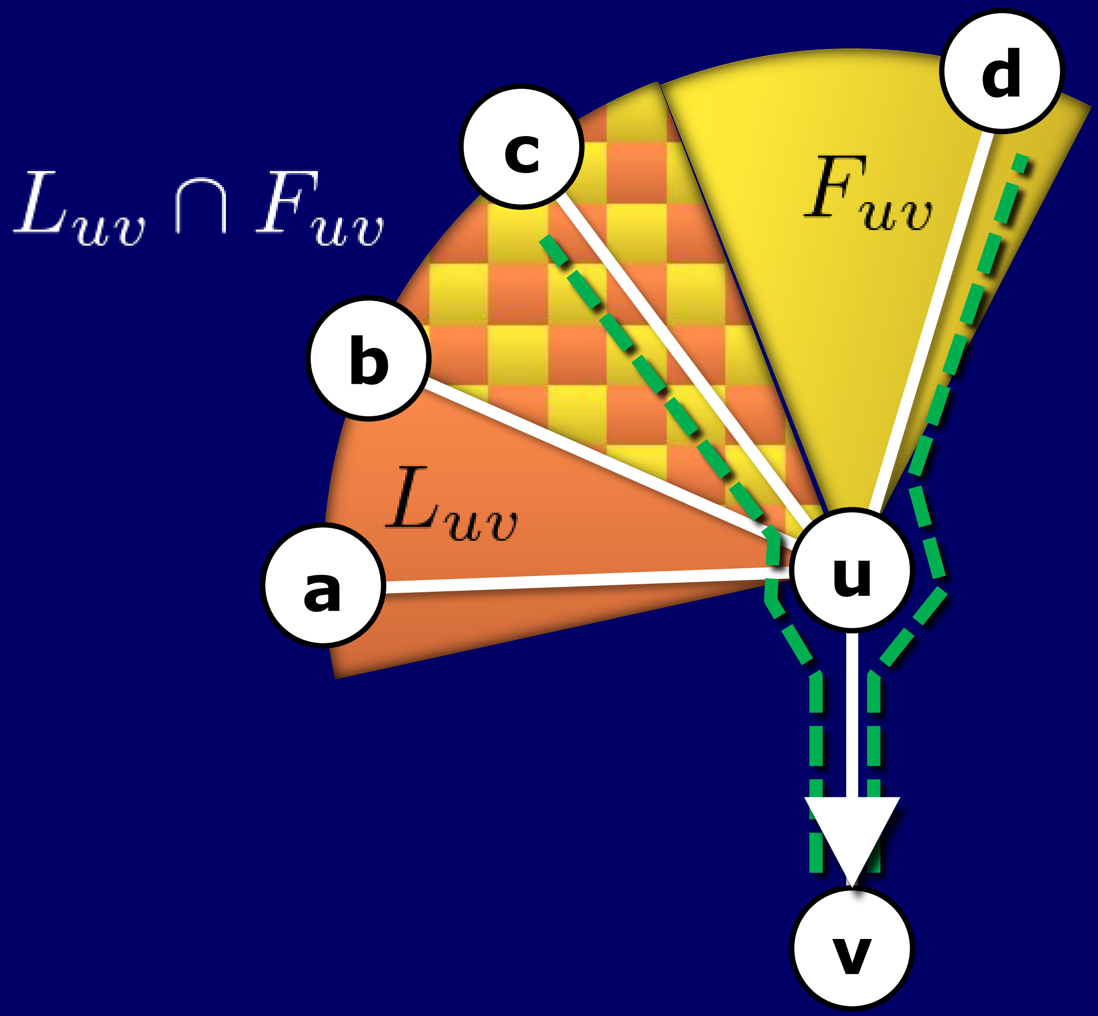
Forced Orientations



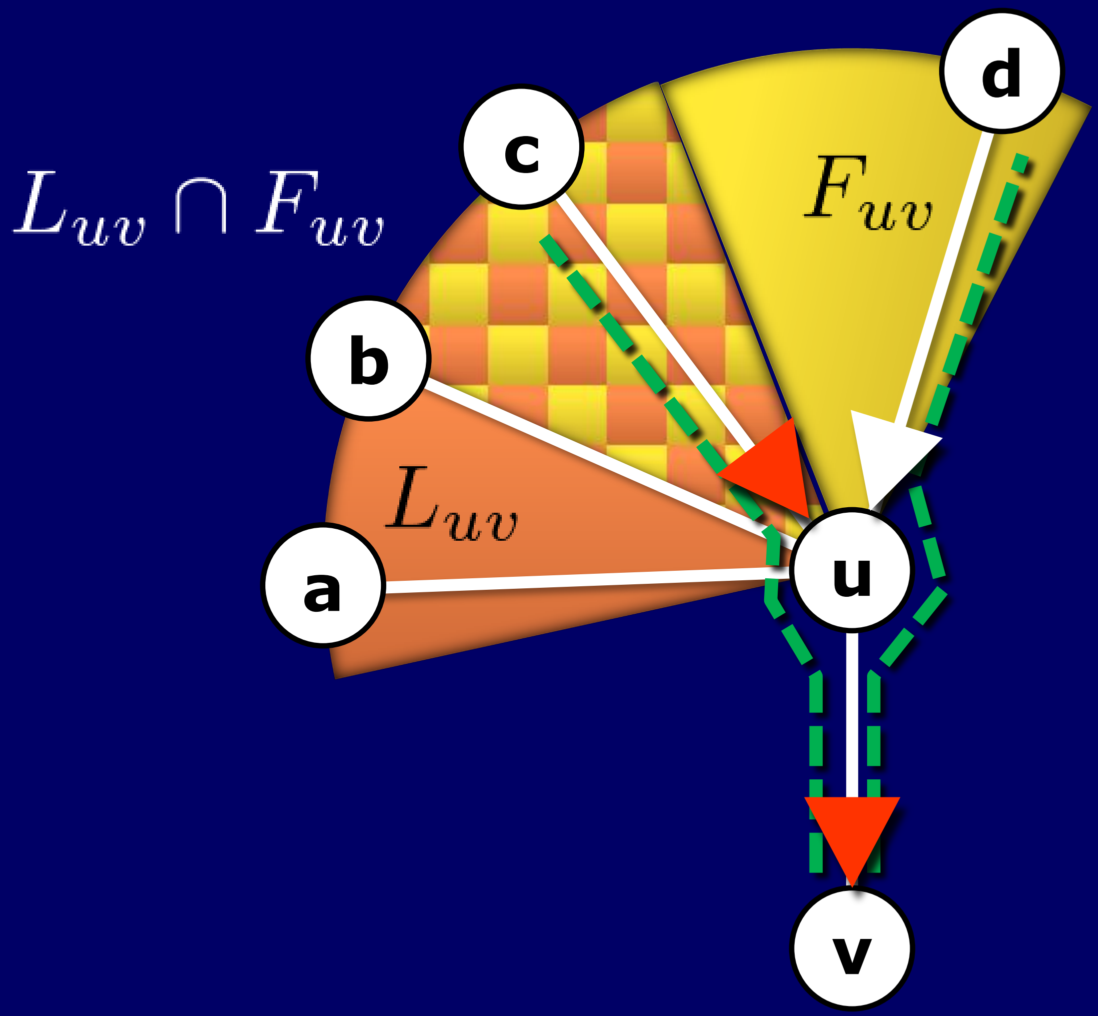
Forced Orientations



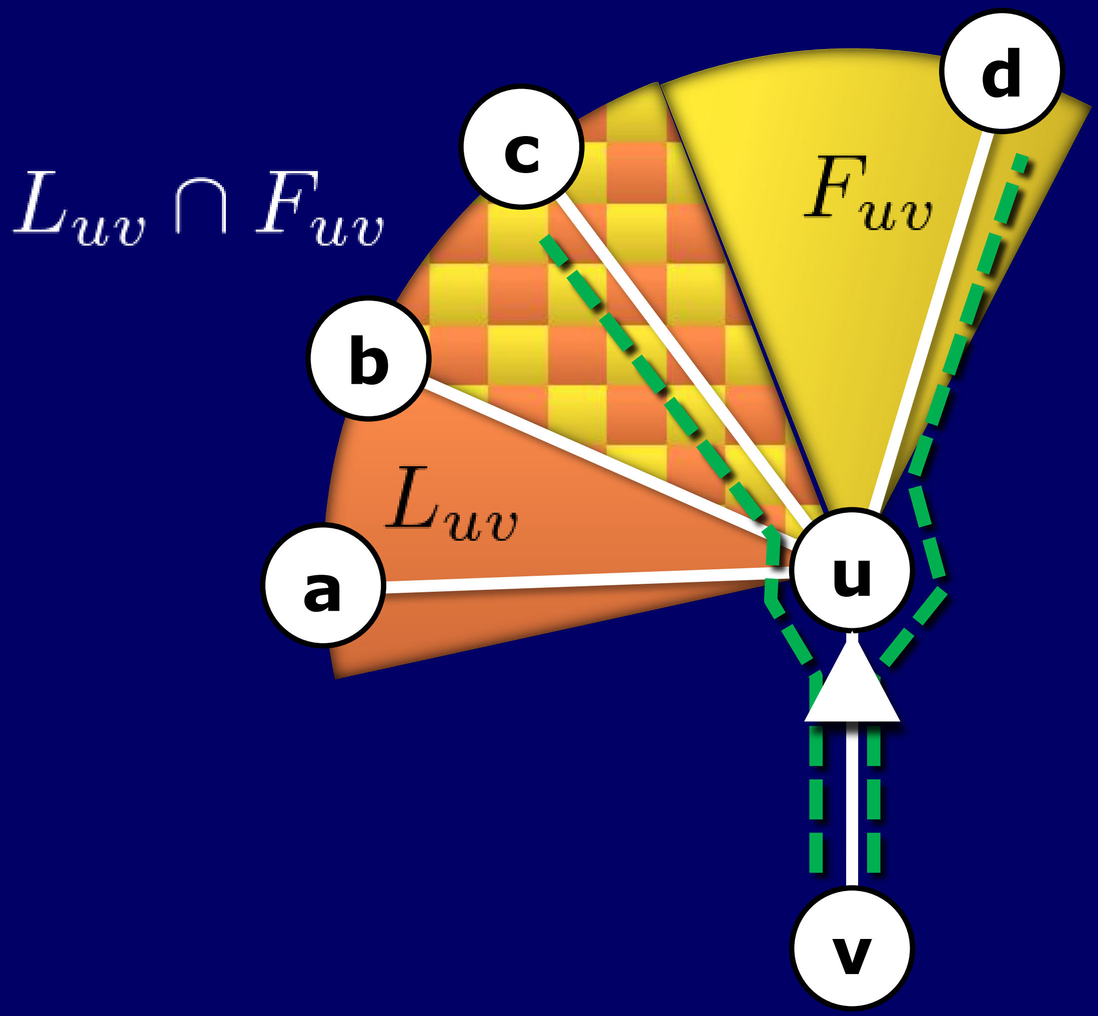
Forced Orientations



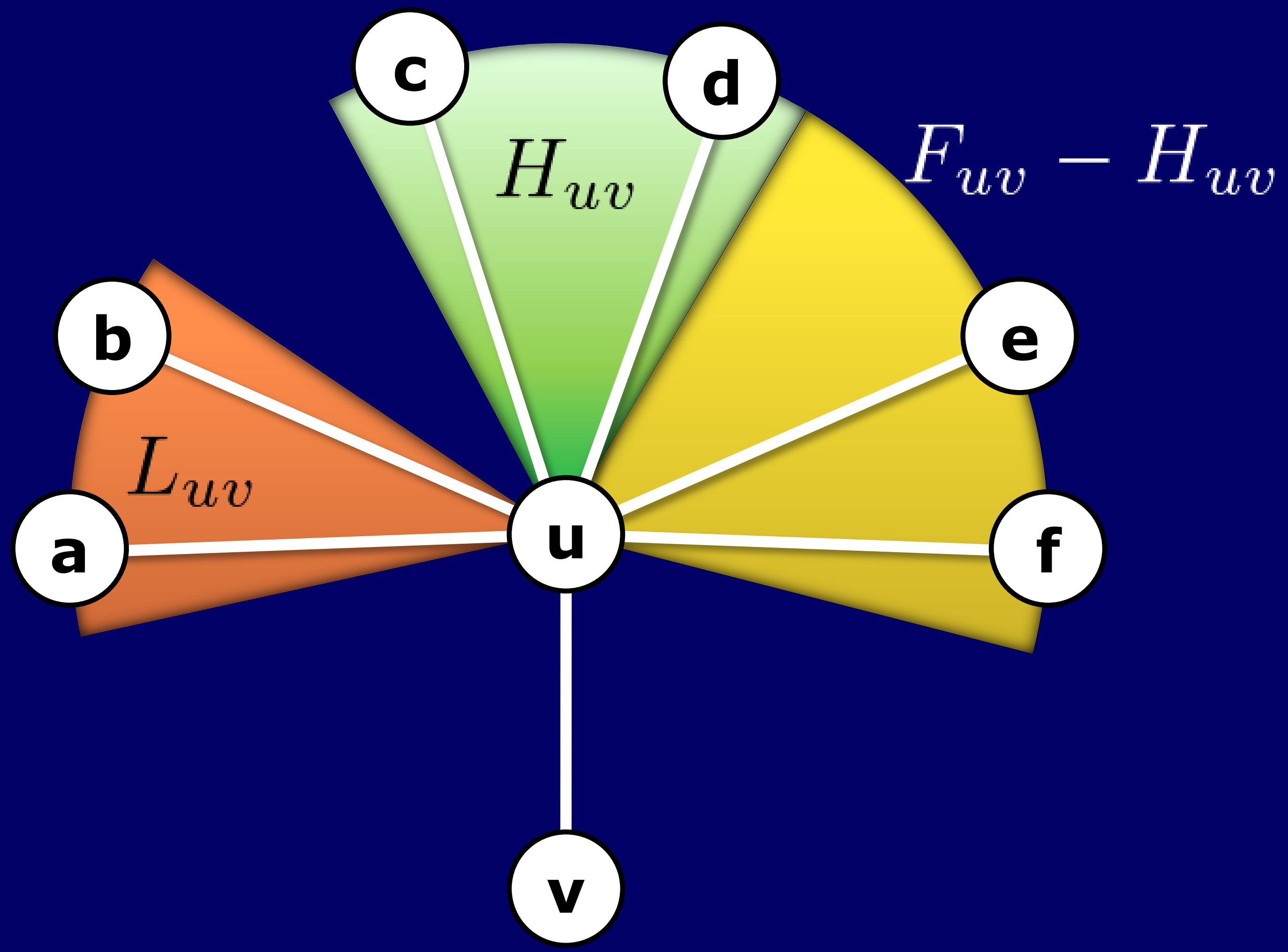
Forced Orientations



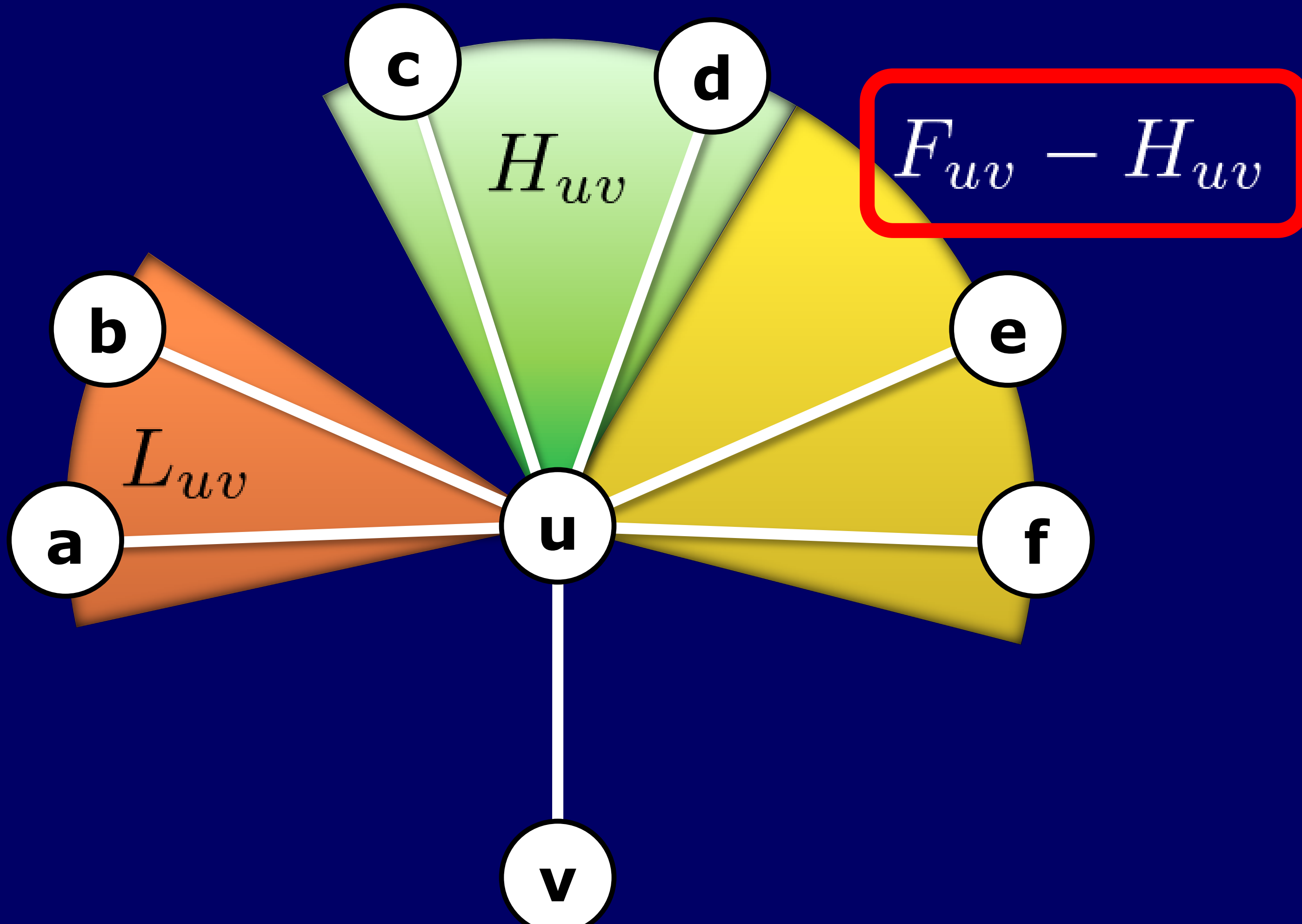
Forced Orientations



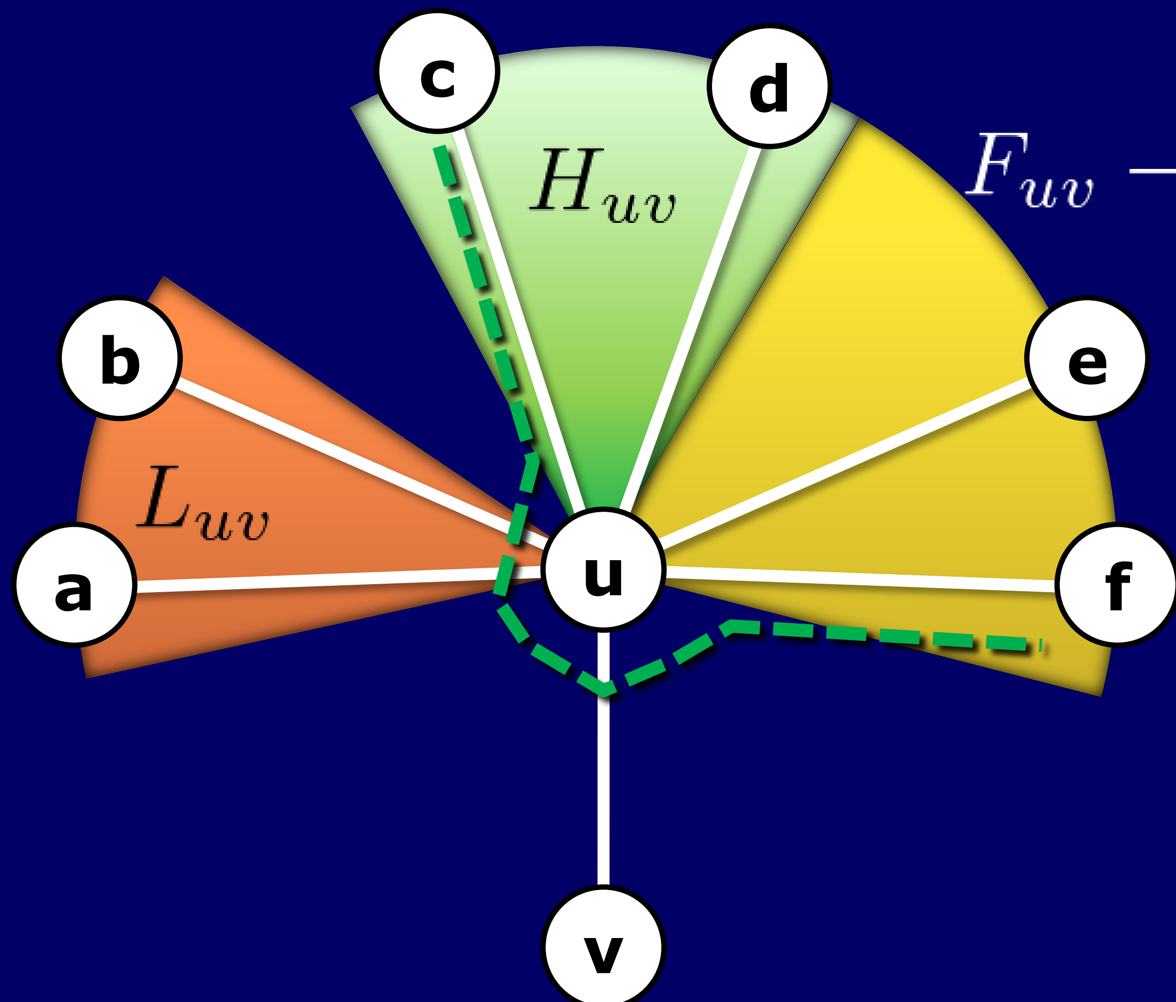
Unconstrained Orientations



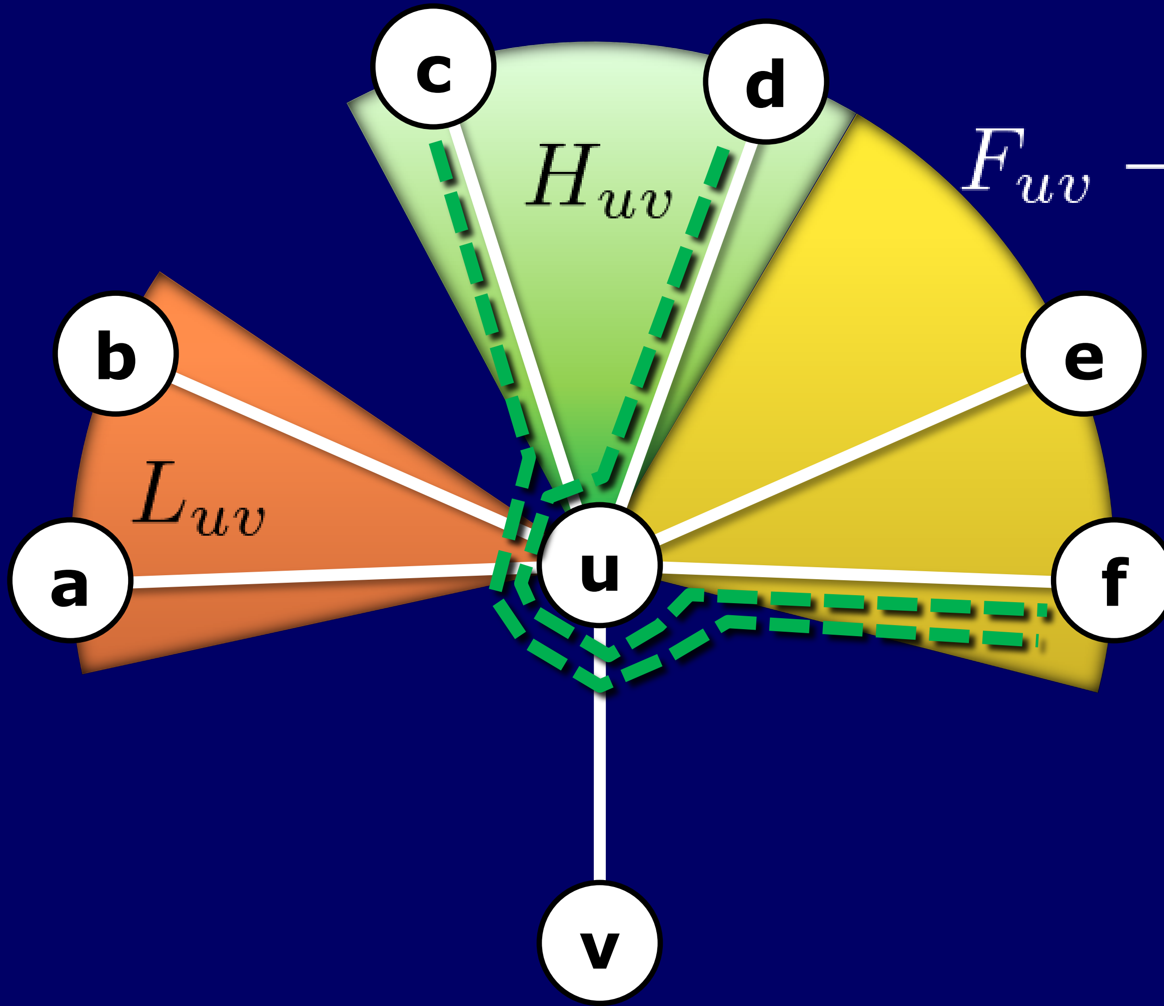
Unconstrained Orientations



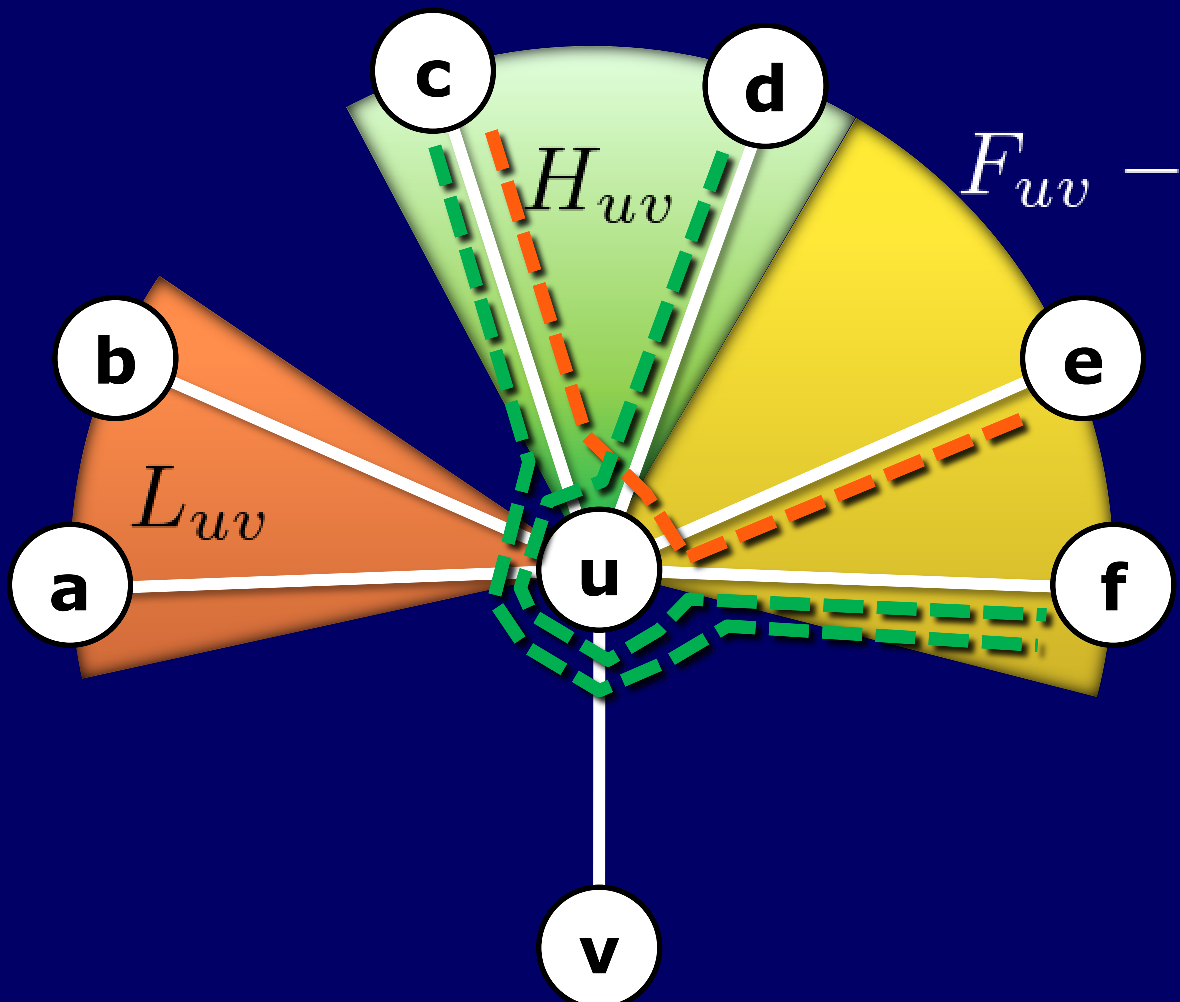
Unconstrained Orientations



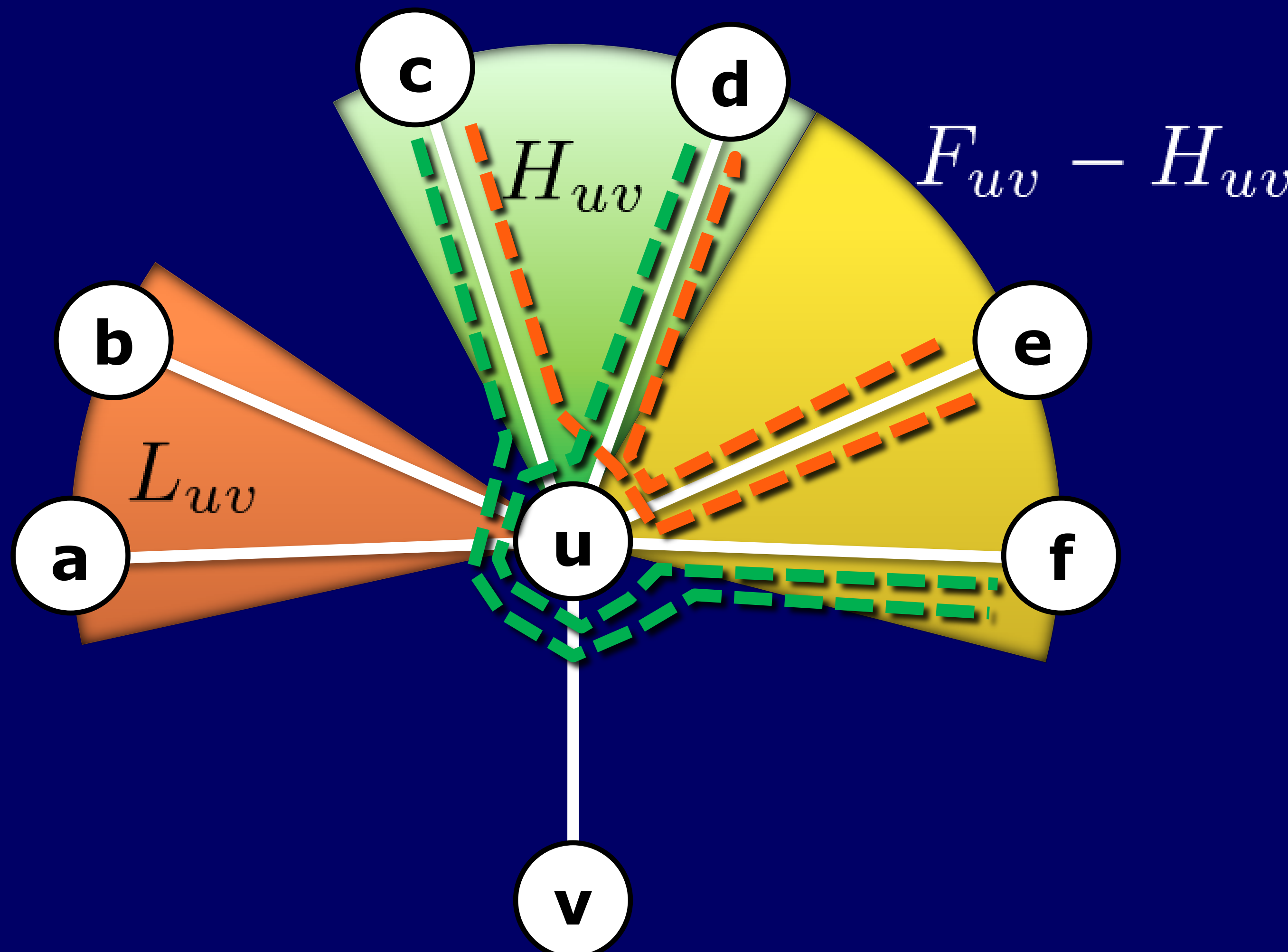
Unconstrained Orientations



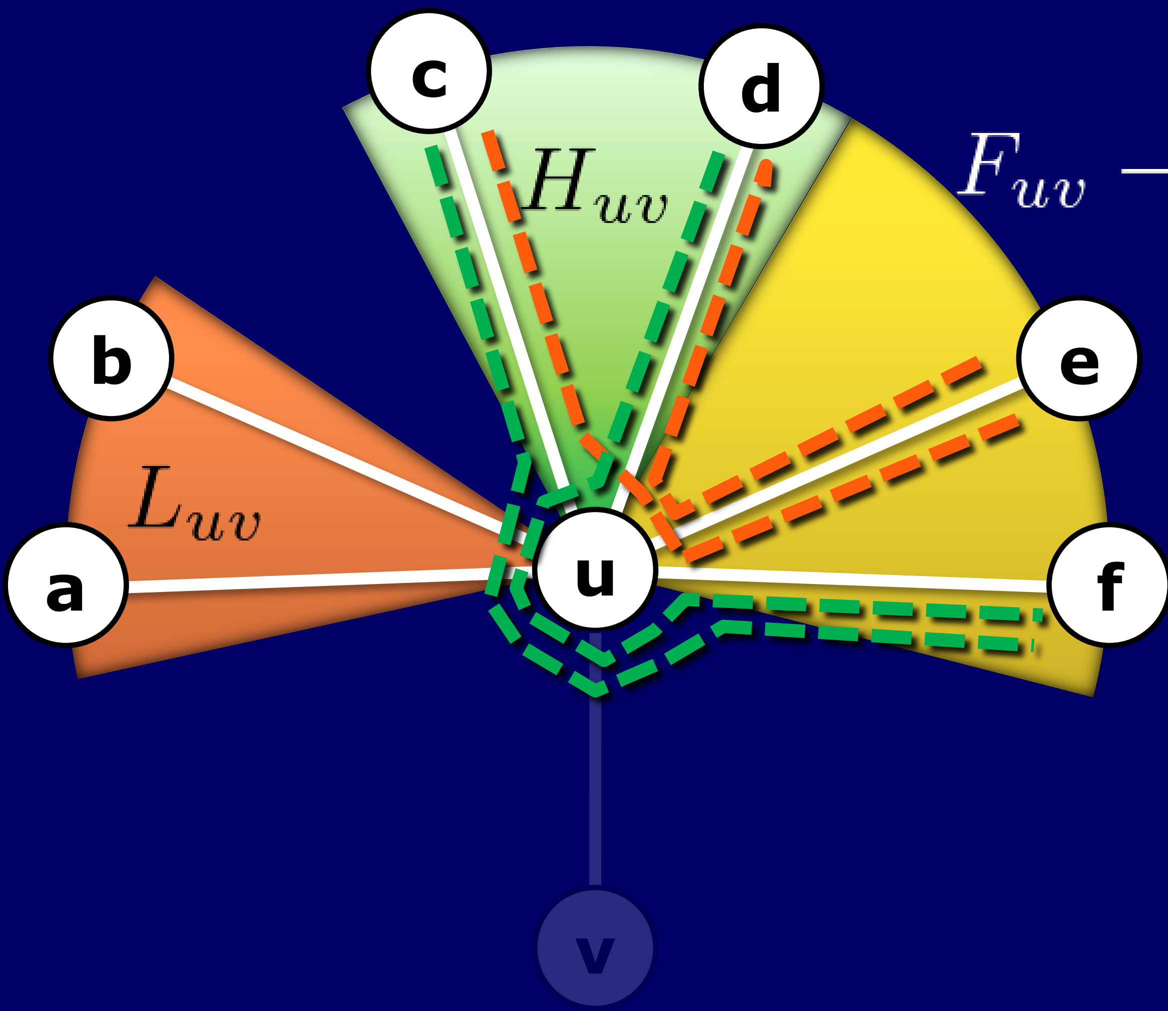
Unconstrained Orientations



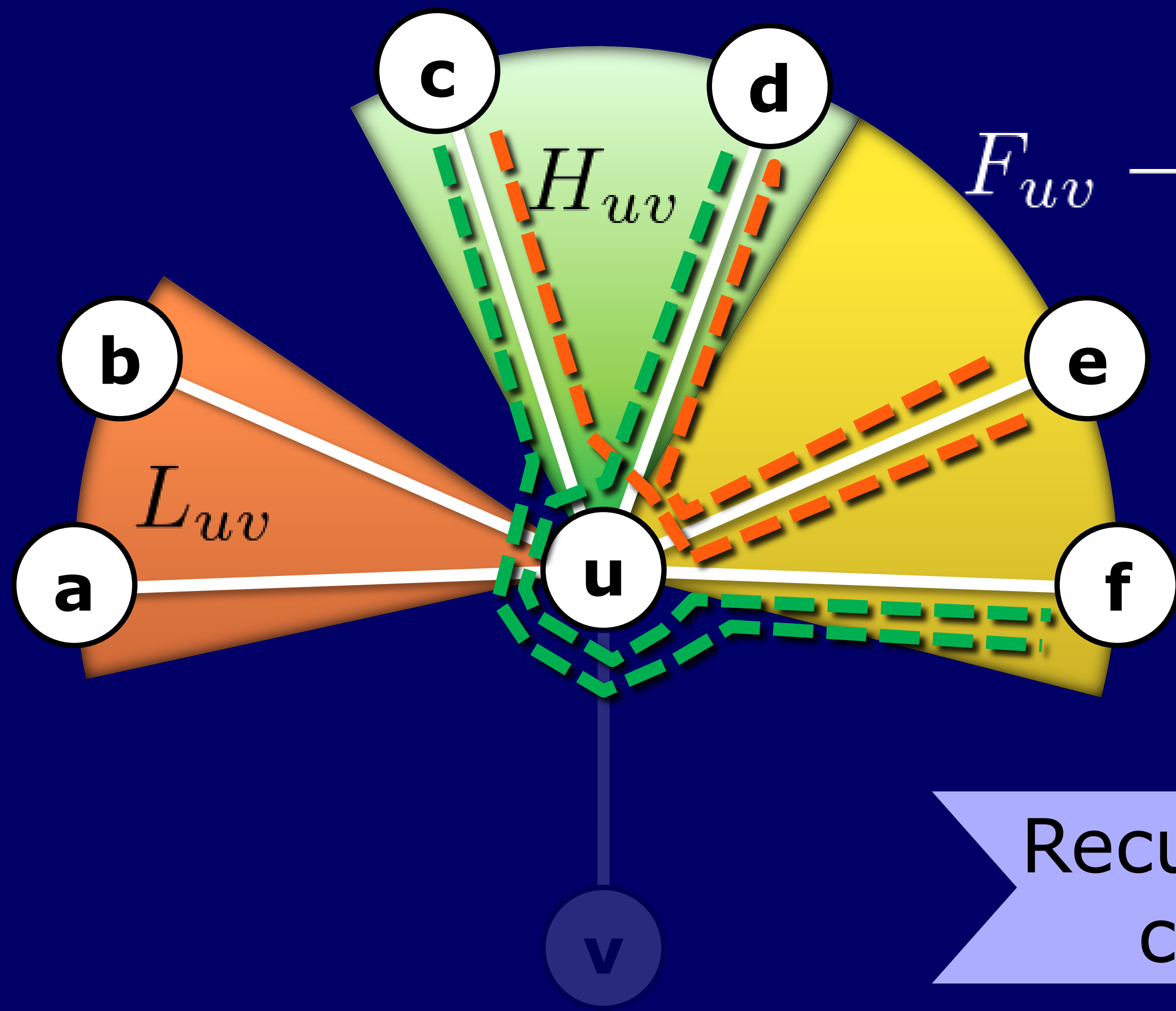
Unconstrained Orientations



Unconstrained Orientations



Unconstrained Orientations



Recursive call

After Recursion

Recursive call

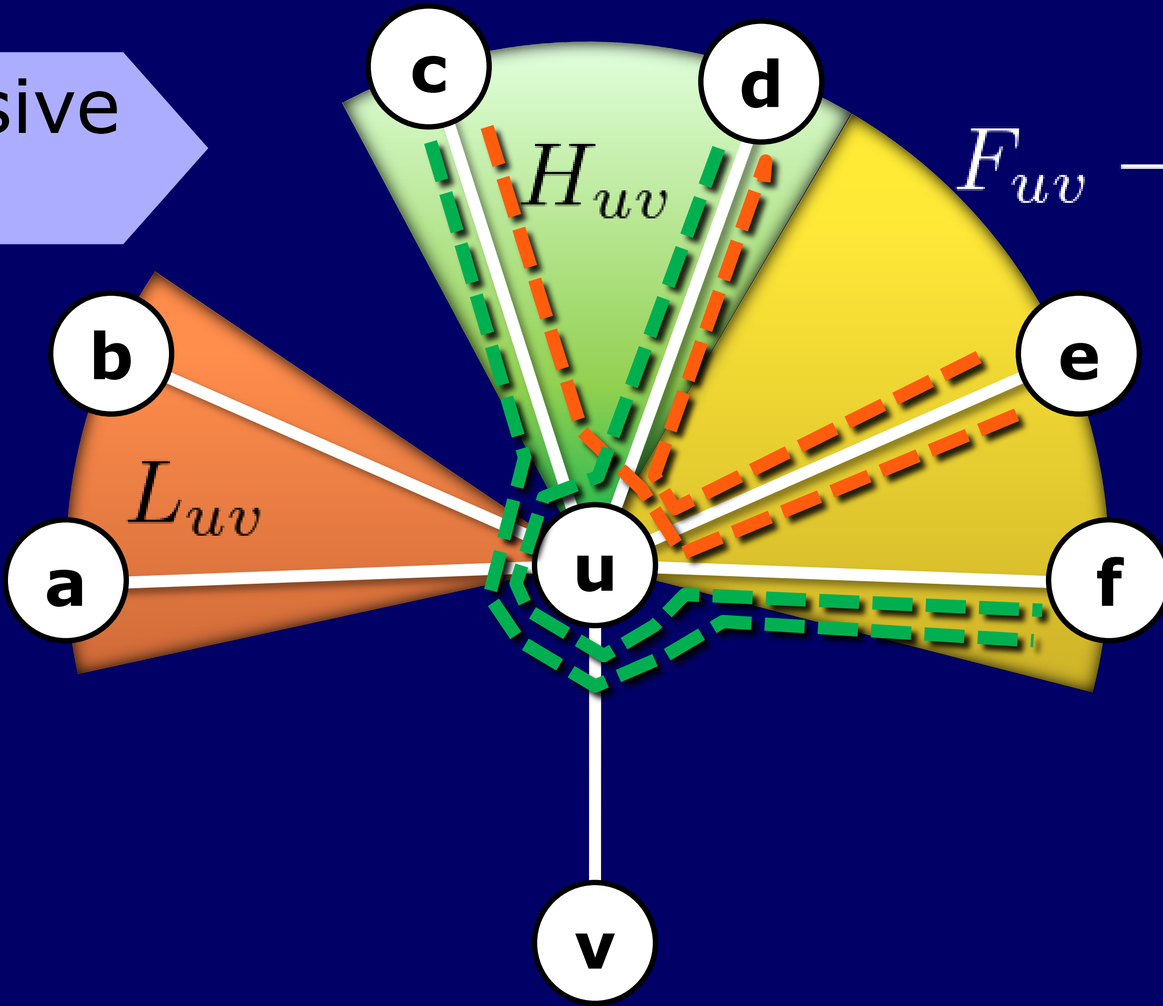
After Recursion

Recursive call

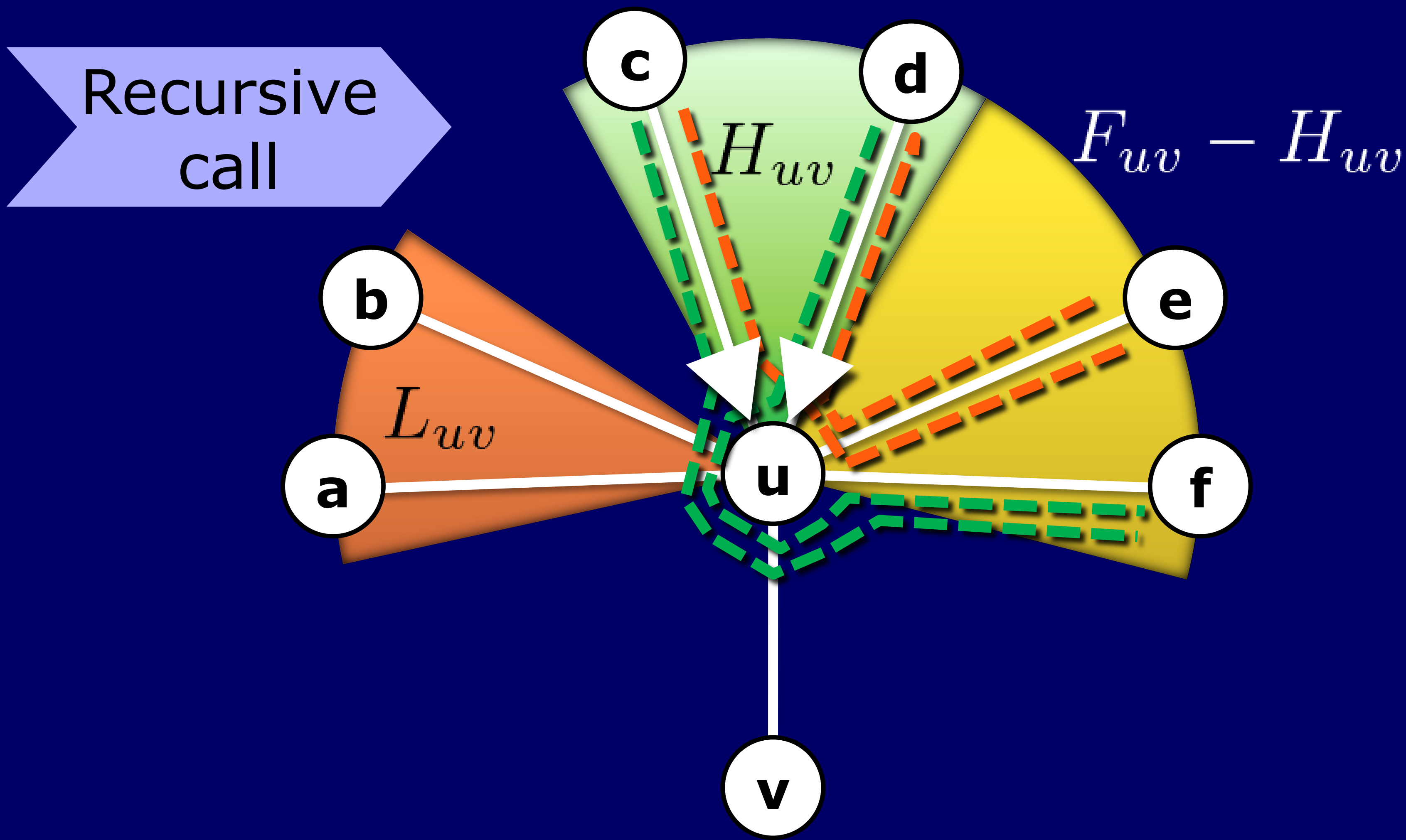
- No valid orientation?
  - Return “no valid orientation”
- Otherwise...

After Recursion

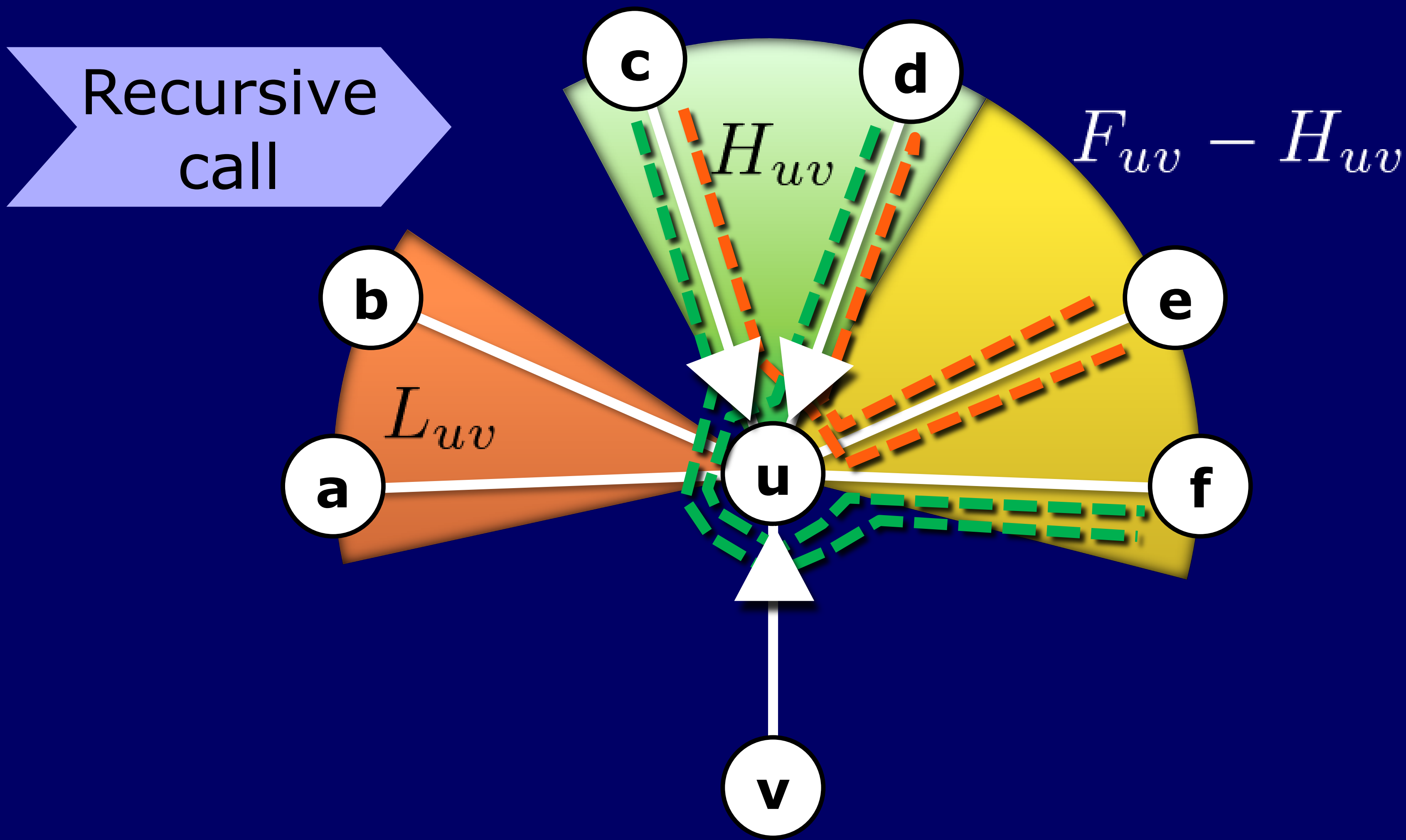
Recursive call



After Recursion

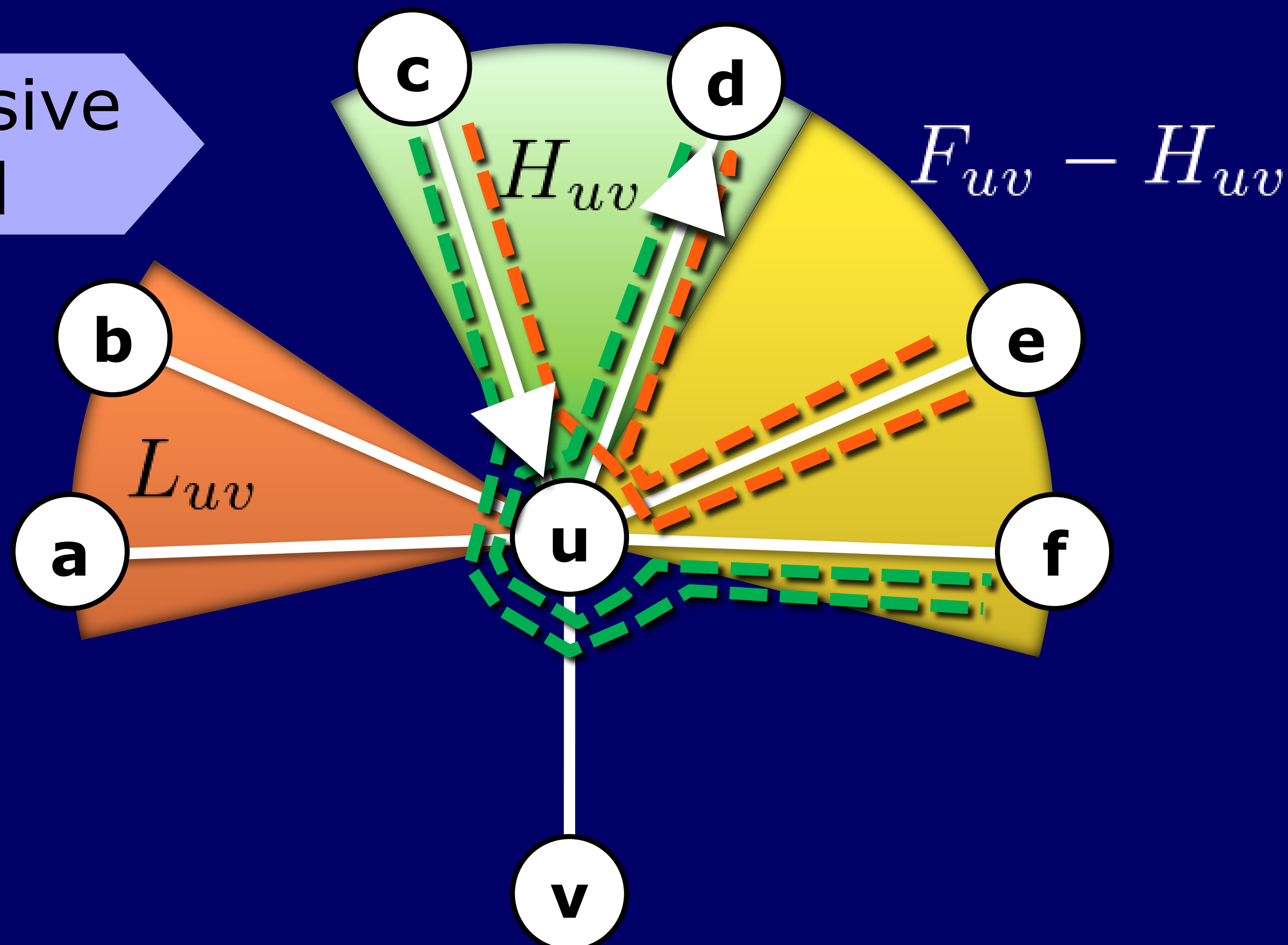


After Recursion



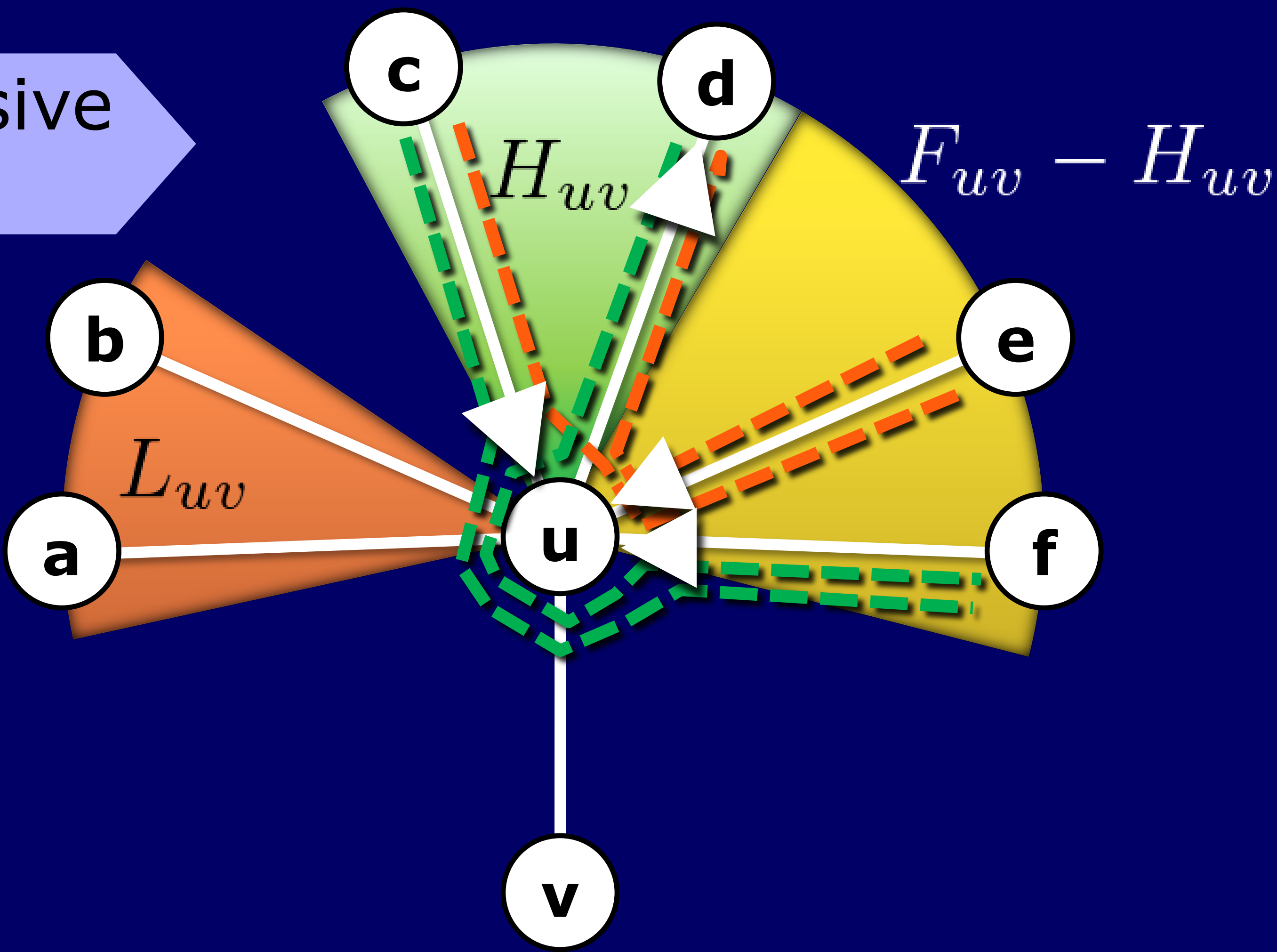
After Recursion

Recursive call

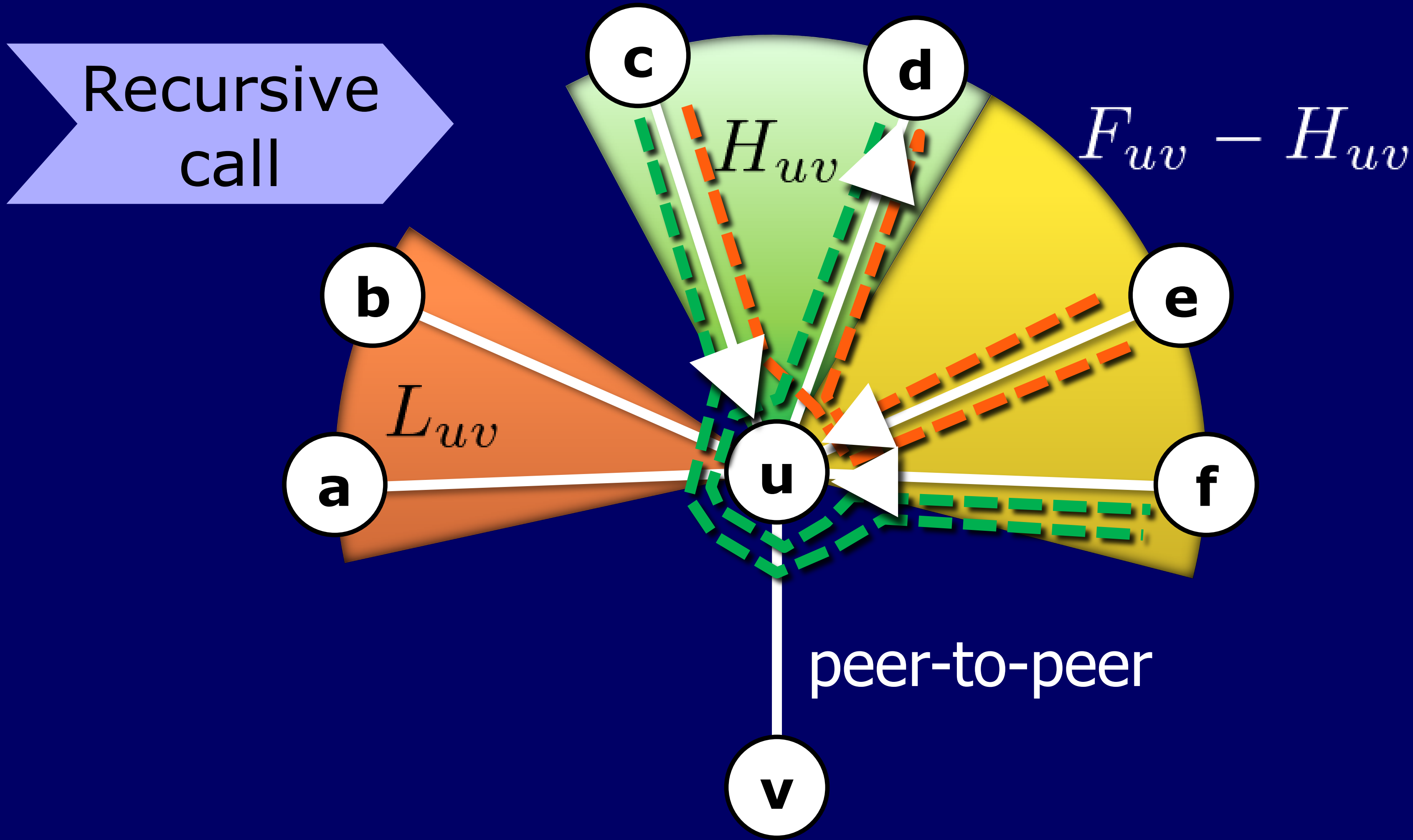


After Recursion

Recursive call



After Recursion



About the Algorithm

- Solves GAO-REXFORD-CHECK with $\prec$ constraints

About the Algorithm

- Solves GAO-REXFORD-CHECK with $\prec$ constraints
  - edges are oriented only if...

About the Algorithm

- Solves GAO-REXFORD-CHECK with $\prec$ constraints
  - edges are oriented only if...
    - ...constrained

About the Algorithm

- Solves GAO-REXFORD-CHECK with $\prec$ constraints
  - edges are oriented only if...
    - ...constrained
    - ...this does not introduce conflicts

About the Algorithm

- Solves GAO-REXFORD-CHECK with $\prec$ constraints
  - edges are oriented only if...
    - ...constrained
    - ...this does not introduce conflicts
- Solves GAO-REXFORD-CHECK

About the Algorithm

- Solves GAO-REXFORD-CHECK with $\prec$ constraints
  - edges are oriented only if...
    - ...constrained
    - ...this does not introduce conflicts
- Solves GAO-REXFORD-CHECK
- Polynomial

About the Algorithm

- Solves GAO-REXFORD-CHECK with $\prec$ constraints
  - edges are oriented only if...
    - ...constrained
    - ...this does not introduce conflicts
- Solves GAO-REXFORD-CHECK
- Polynomial
  - steps before recursion
  - steps after recursion

About the Algorithm

- Solves GAO-REXFORD-CHECK with $\prec$ constraints
  - edges are oriented only if...
    - ...constrained
    - ...this does not introduce conflicts
- Solves GAO-REXFORD-CHECK
- Polynomial
  - steps before recursion
  - one vertex removed at each call
  - steps after recursion

About the Algorithm

- Solves GAO-REXFORD-CHECK with $\prec$ constraints
  - edges are oriented only if...
    - ...constrained
    - ...this does not introduce conflicts
- Solves GAO-REXFORD-CHECK
- Polynomial
  - steps before recursion
  - one vertex removed at each call
  - steps after recursion
- Works pretty much the same in the succinct model

An NP-hardness proof for GAO-REXFORD-STRICT-CHECK

Proof Outline

- 3SAT → GAO-REXFORD-STRICT-CHECK

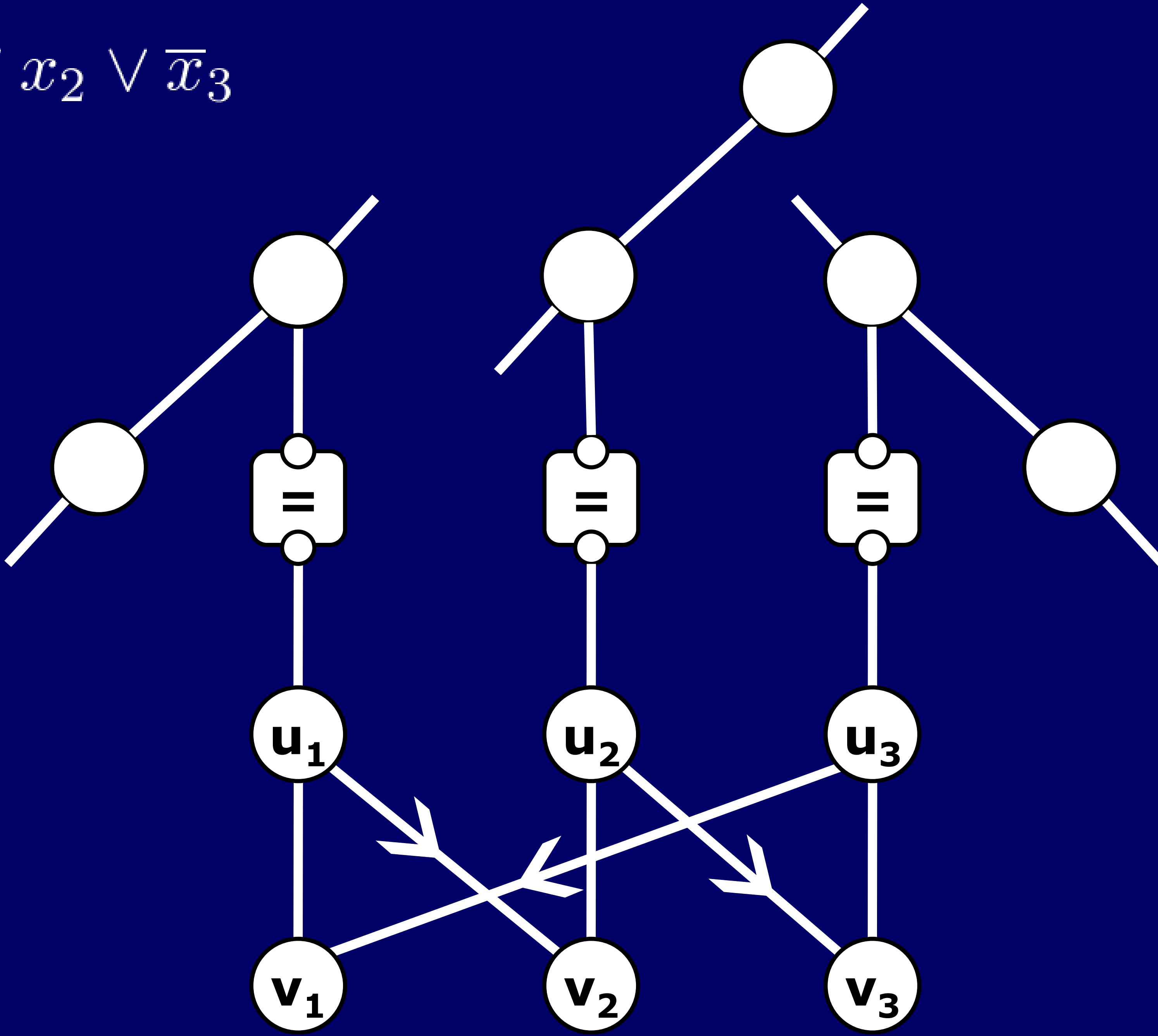
Proof Outline

- 3SAT $\rightarrow$ GAO-REXFORD-STRICT-CHECK
- $\exists$ satisfying assignment $\Leftrightarrow$ $\exists$ Gao-Rexford-Strict-compliant orientation

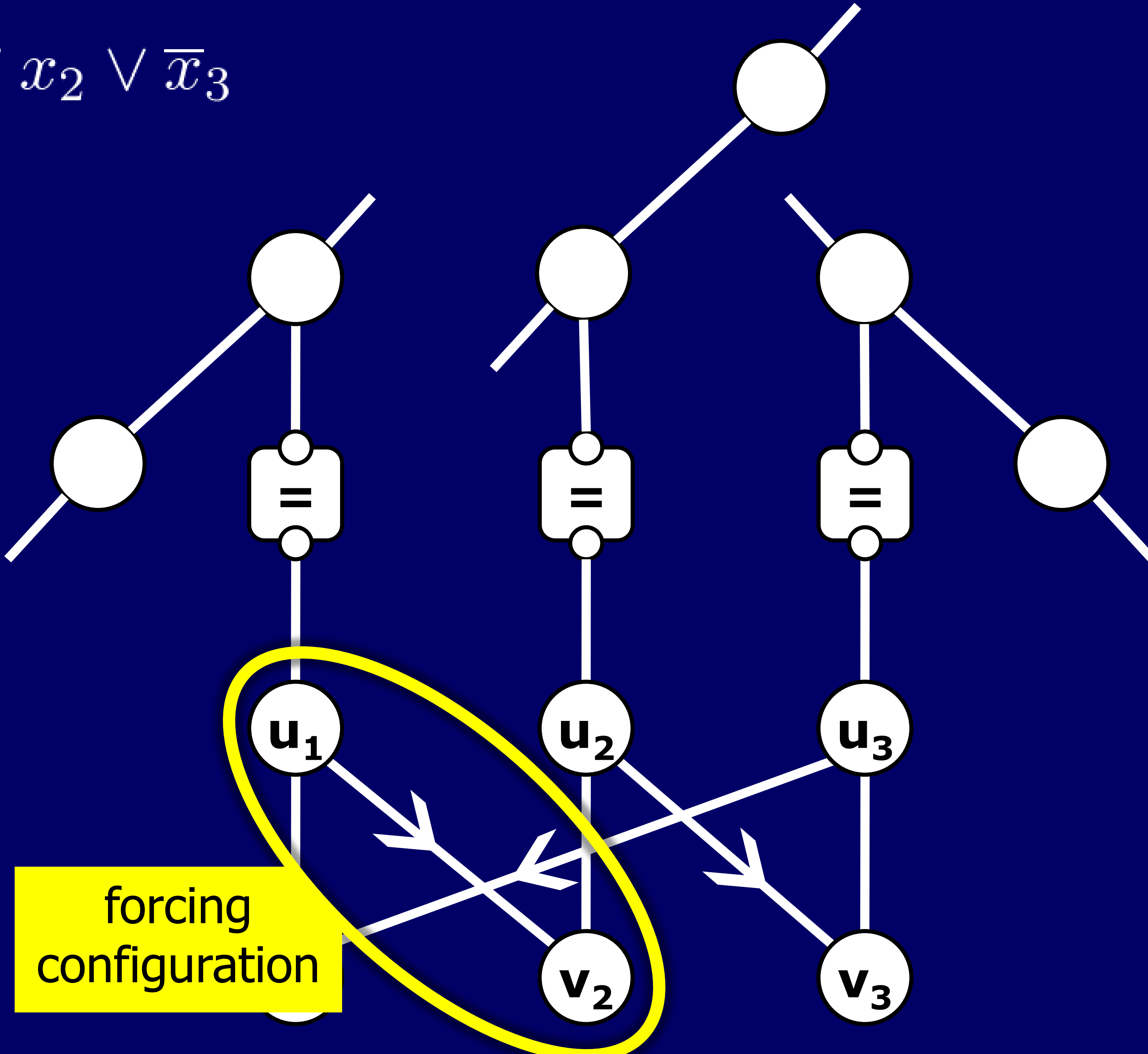
Proof Outline

- 3SAT $\rightarrow$ GAO-REXFORD-STRICT-CHECK
- $\exists$ satisfying assignment $\Leftrightarrow$ $\exists$ Gao-Rexford-Strict-compliant orientation
- $\nexists$ Gao-Rexford-Strict-compliant orientation: obtained by introducing a forced cycle

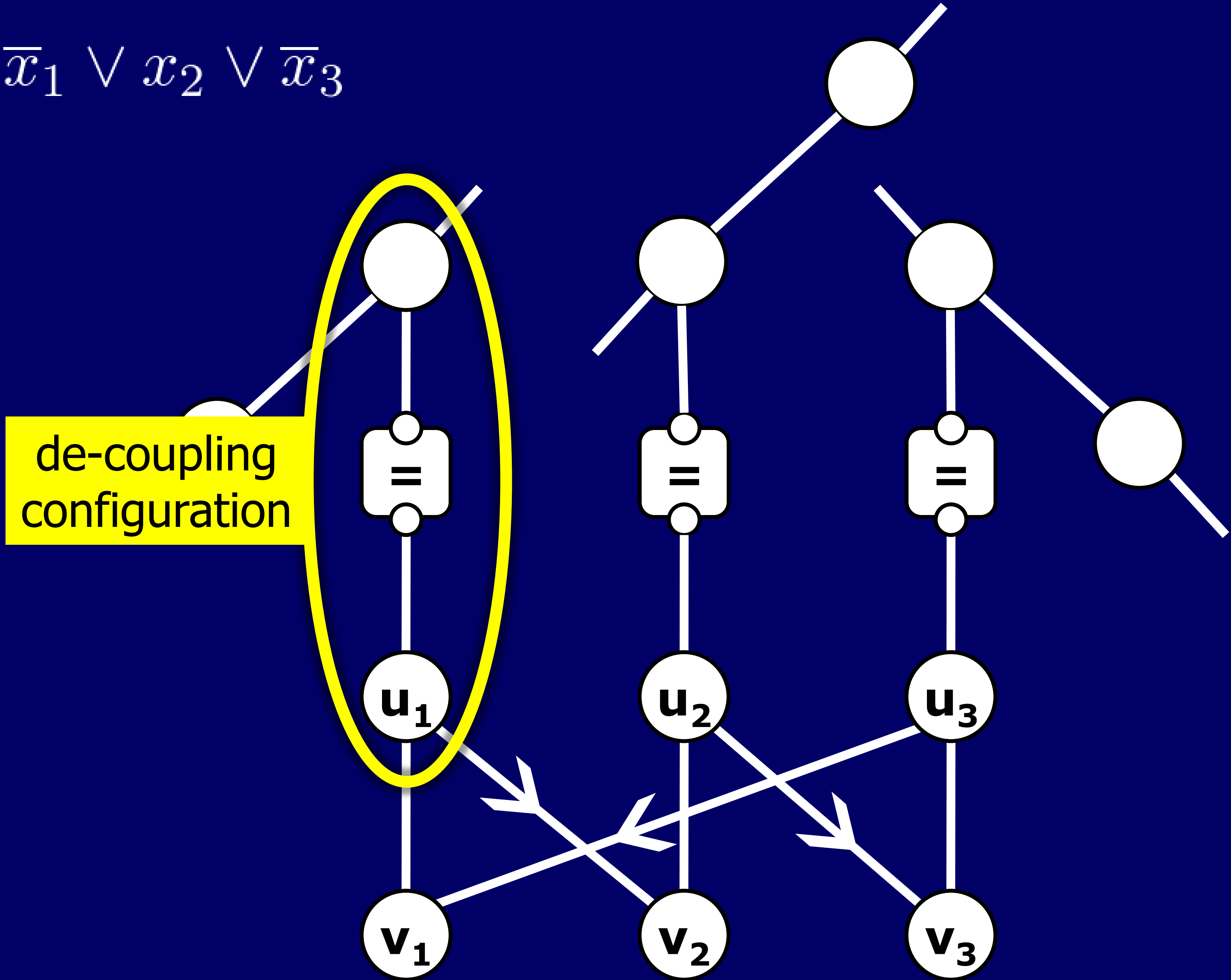
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



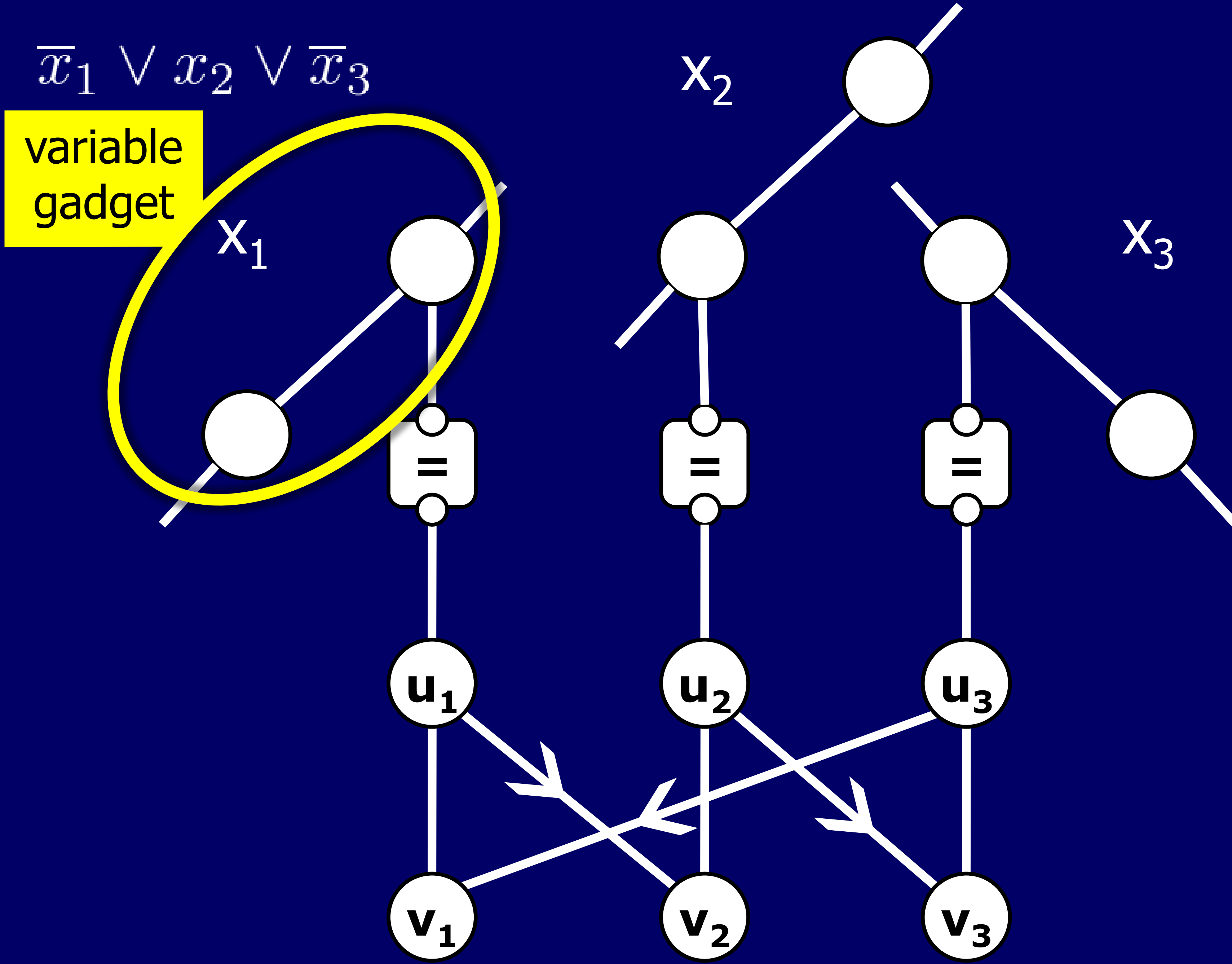
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



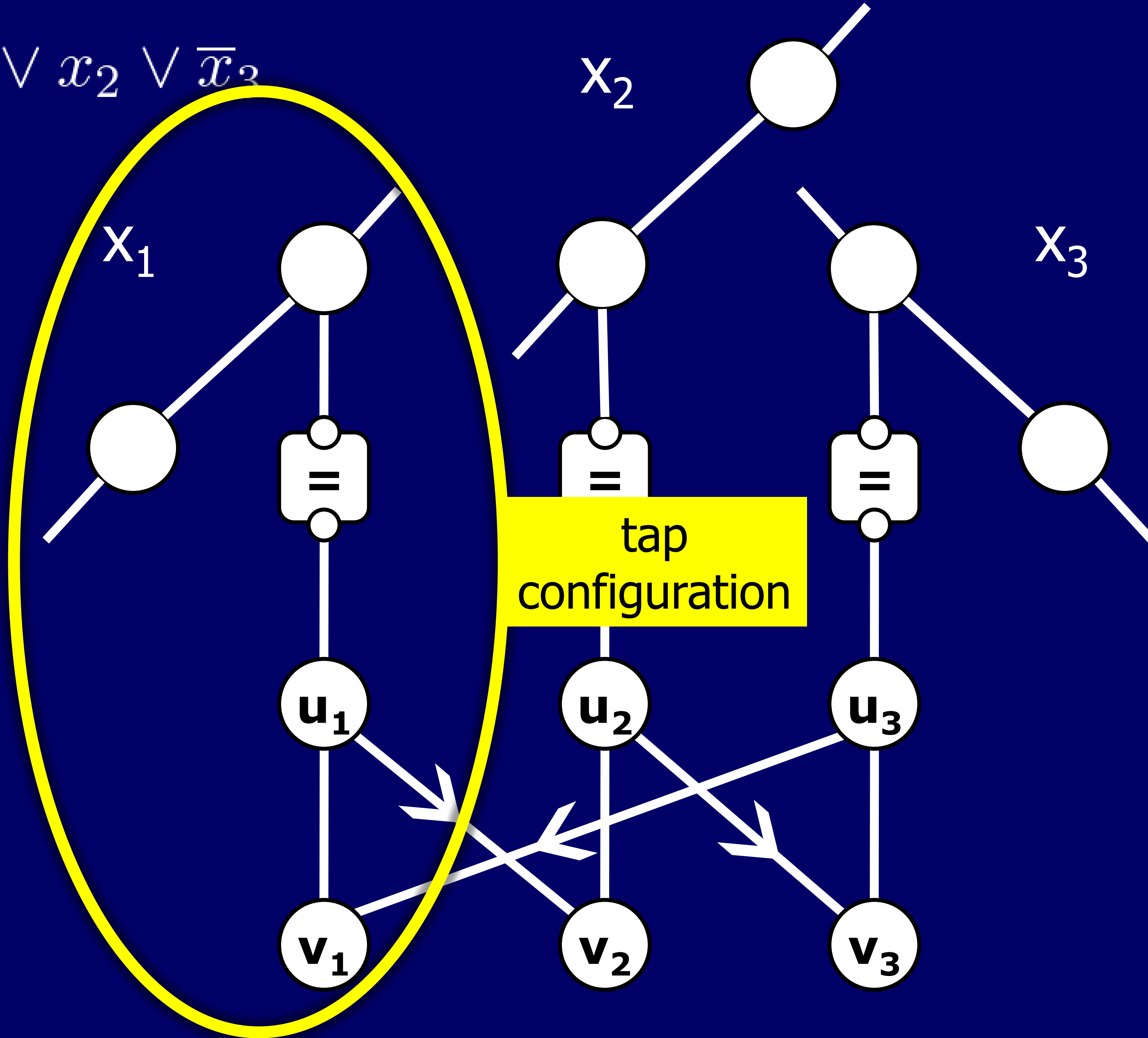
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$

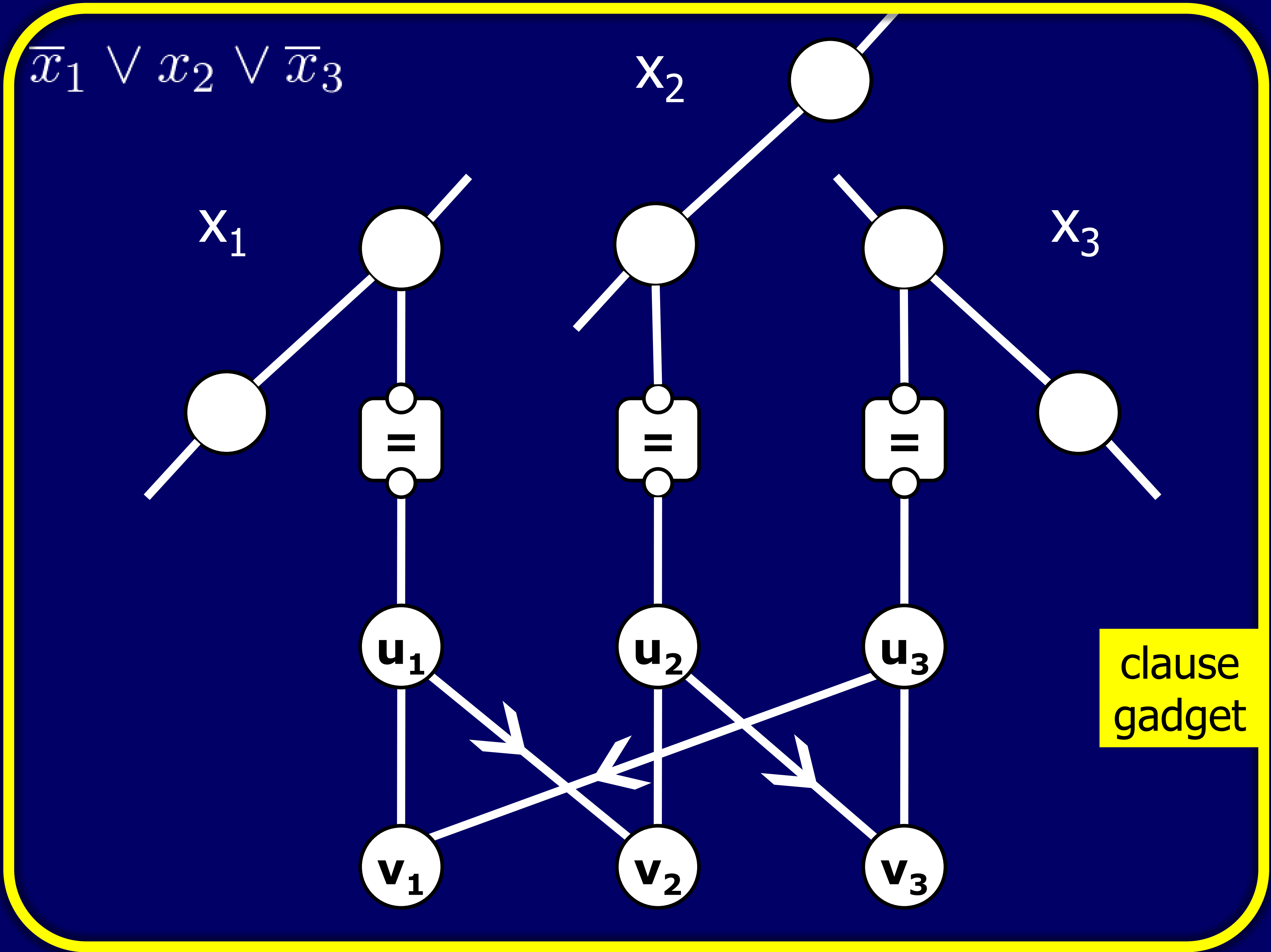


$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



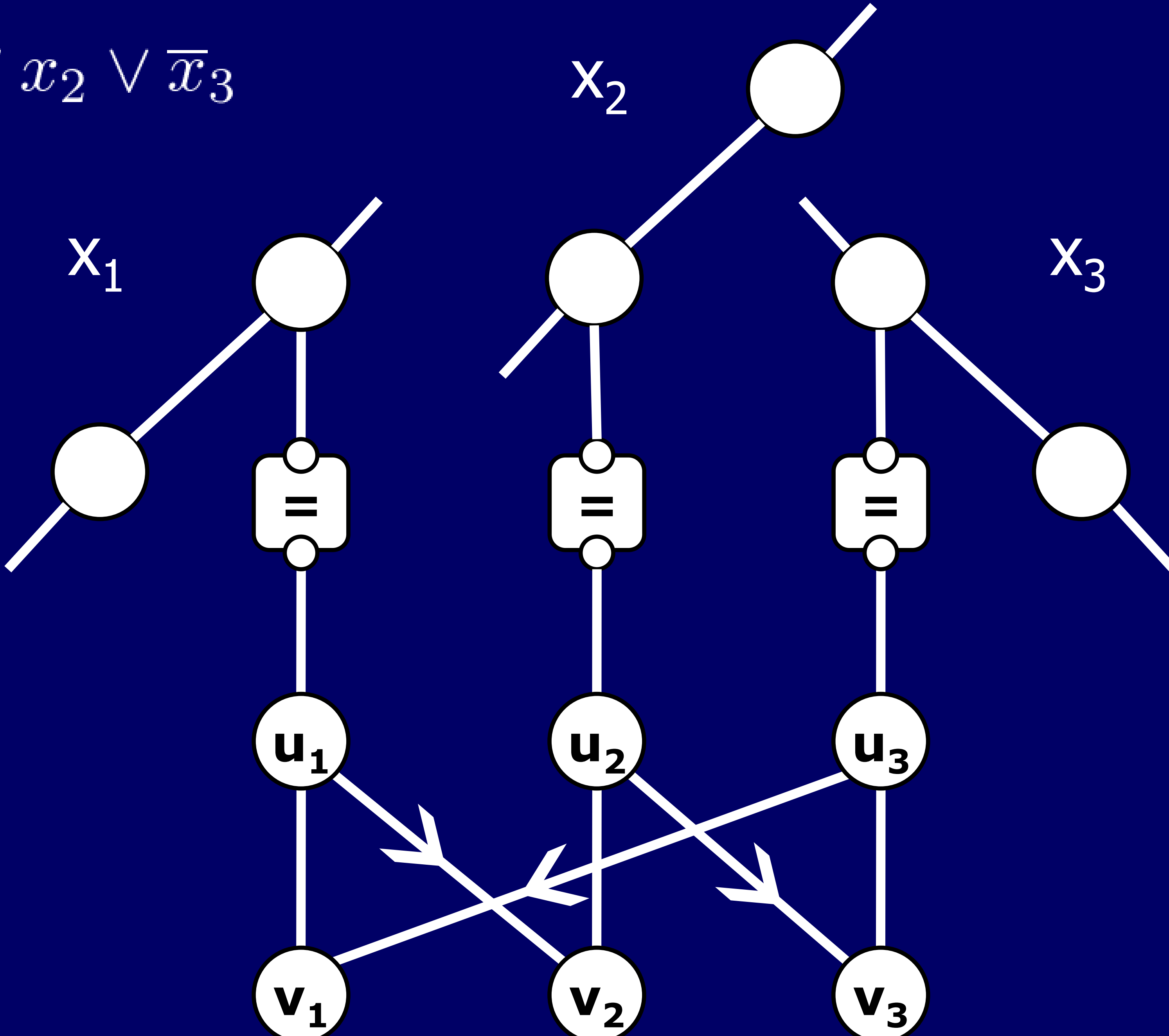
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



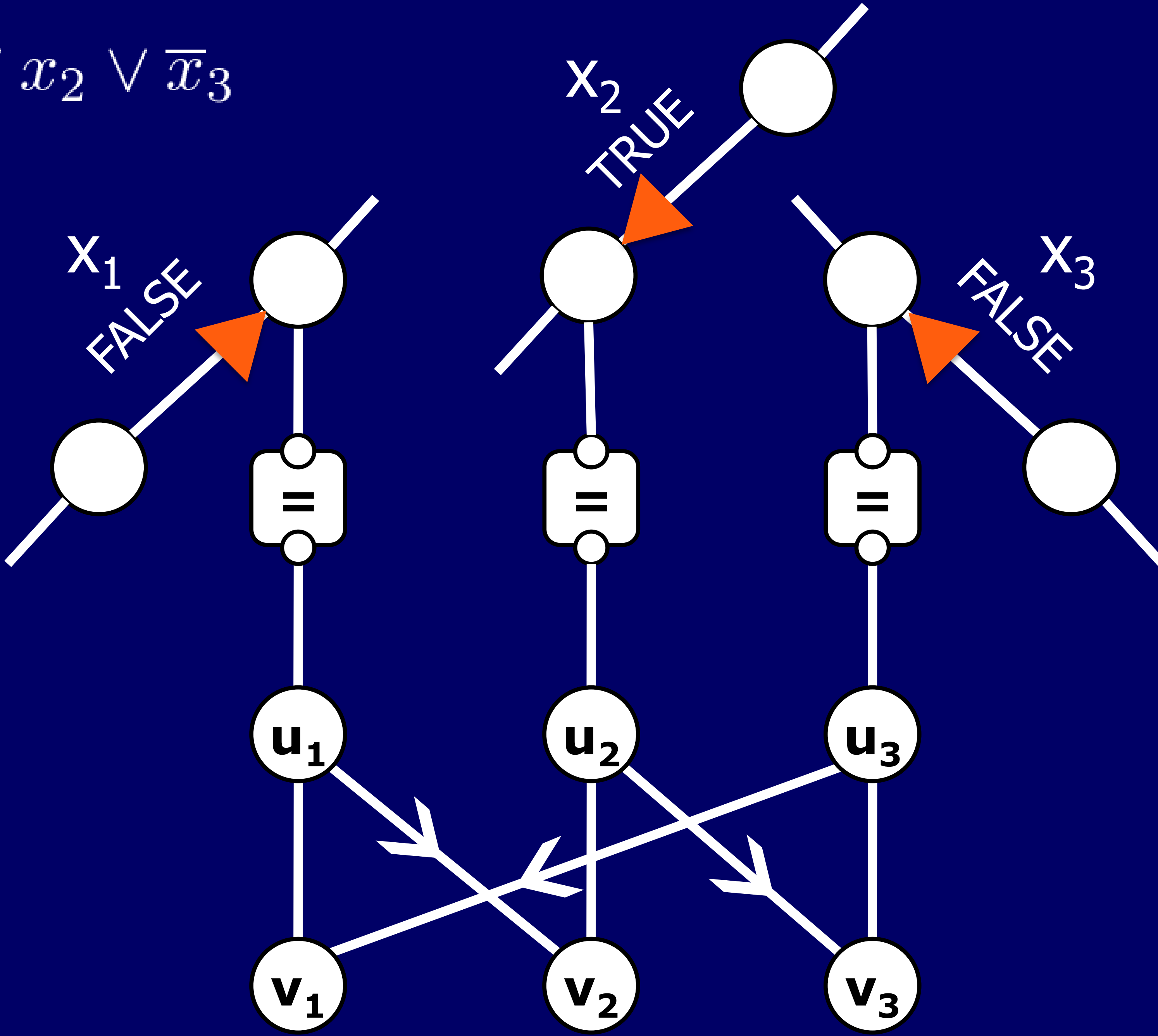


$\overline{x}_1 \vee x_2 \vee \overline{x}_3$
x2
x1
x3
=
=
=
u1
u2
u3
v1
v2
v3
clause gadget

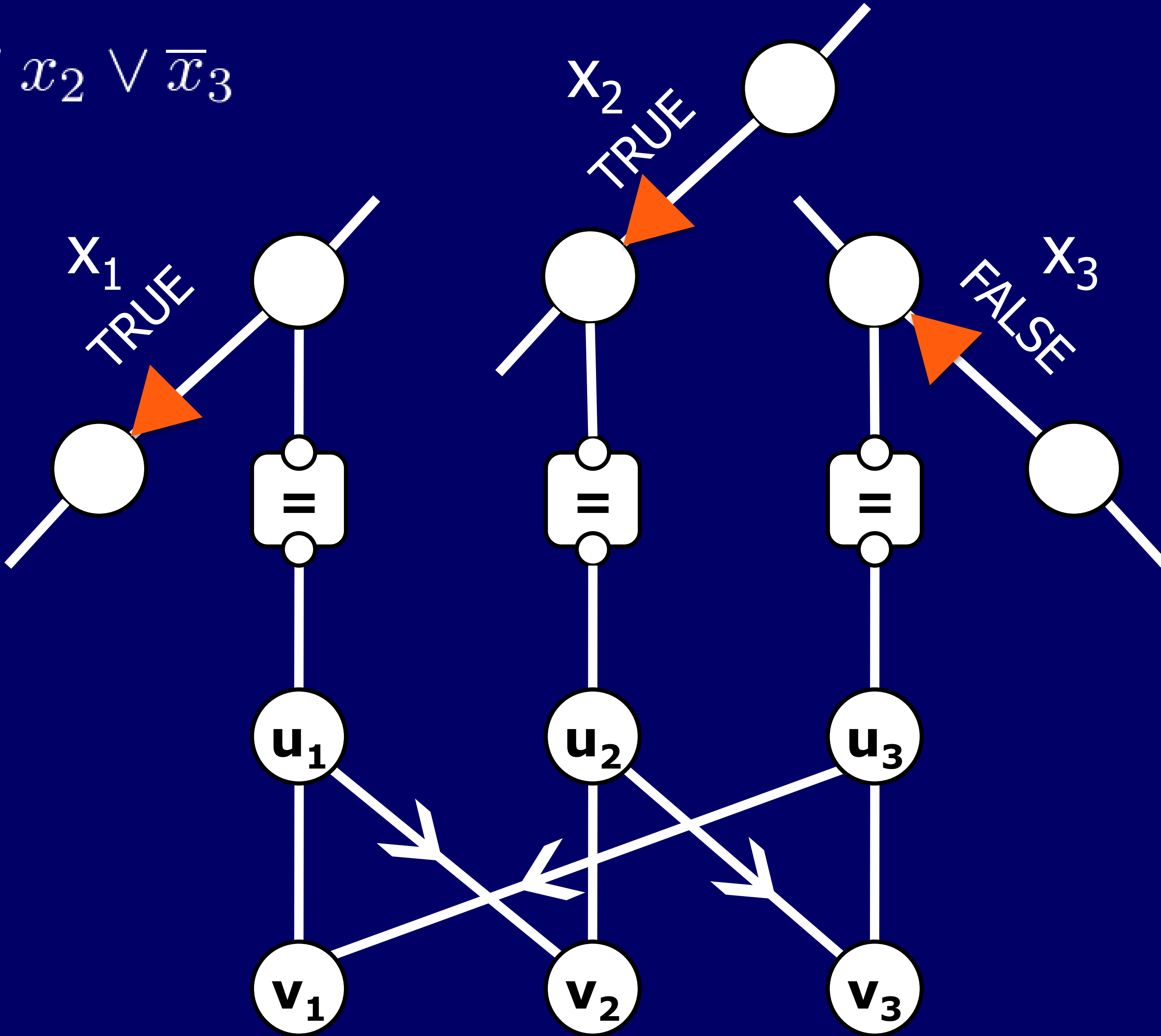
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



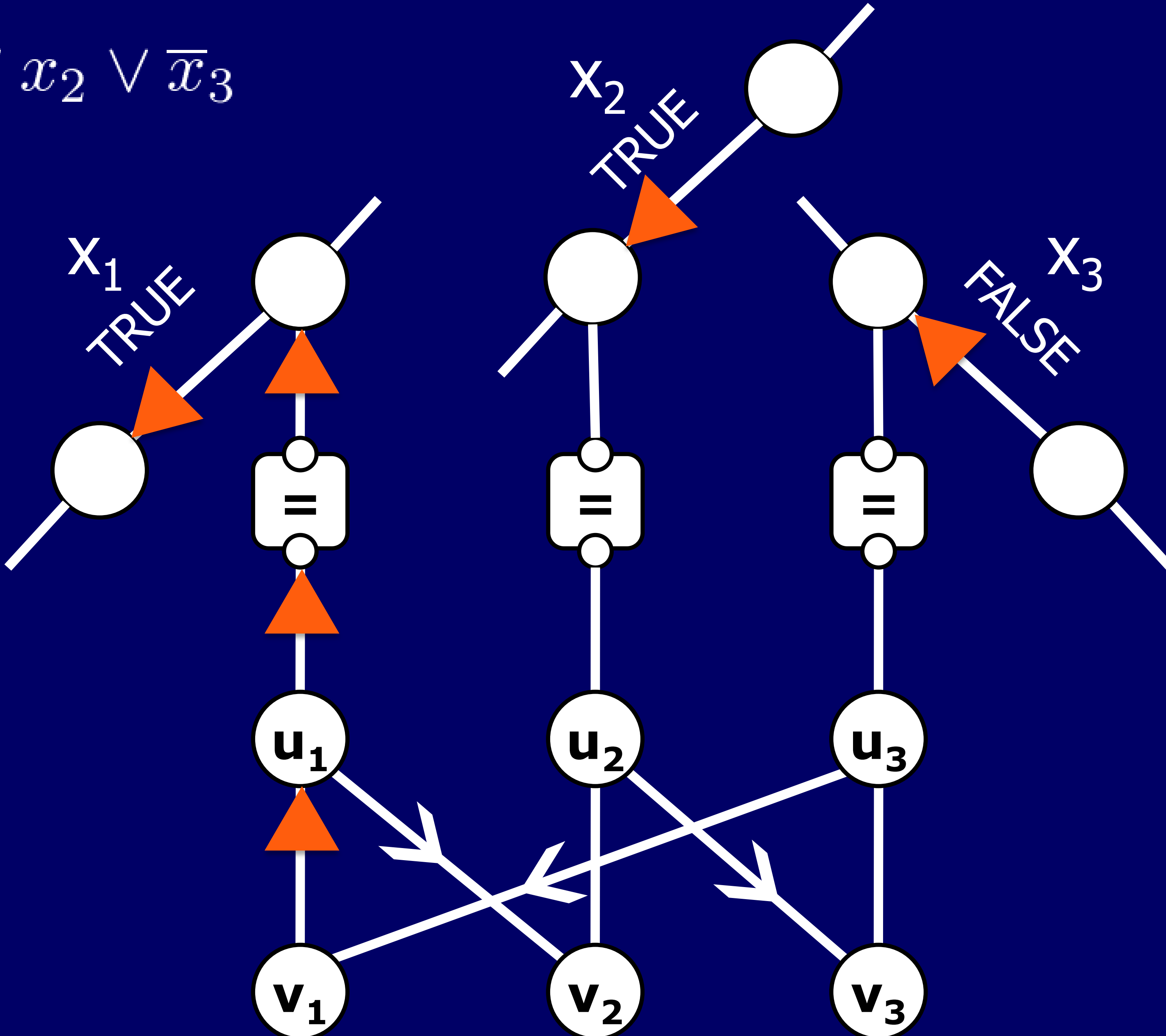
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



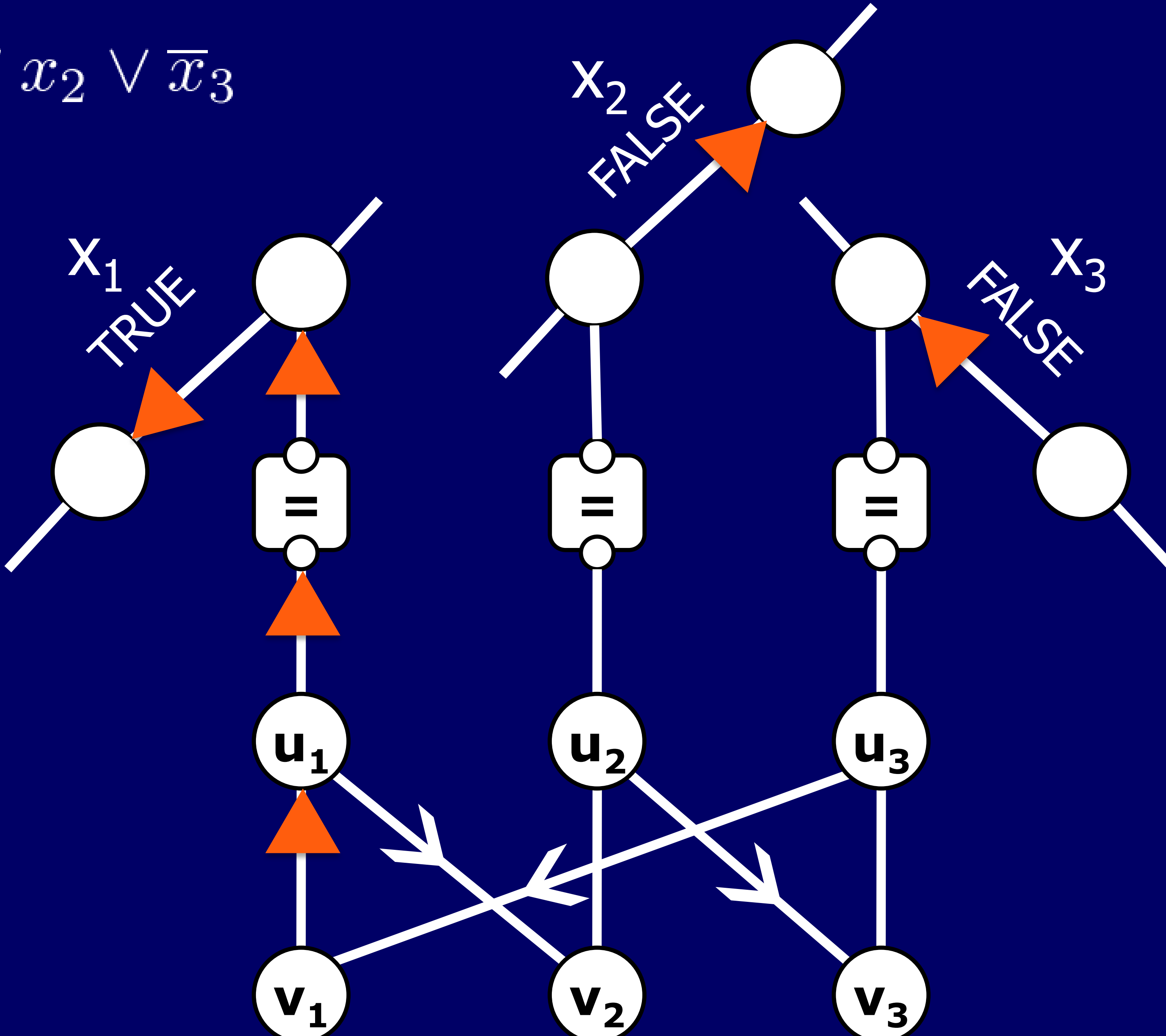
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



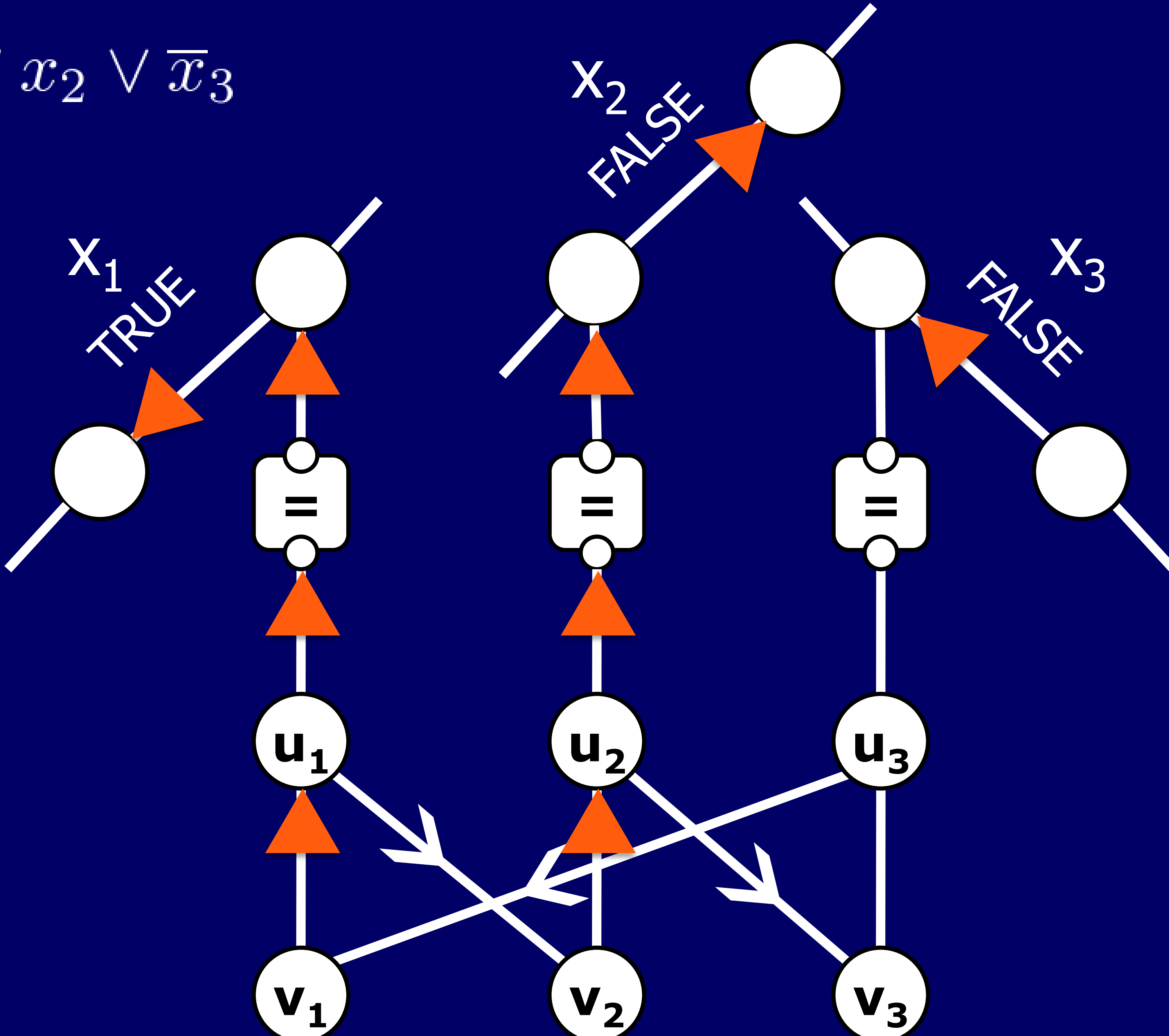
$$\overline{x}_1 \vee x_2 \vee \overline{x}_3$$



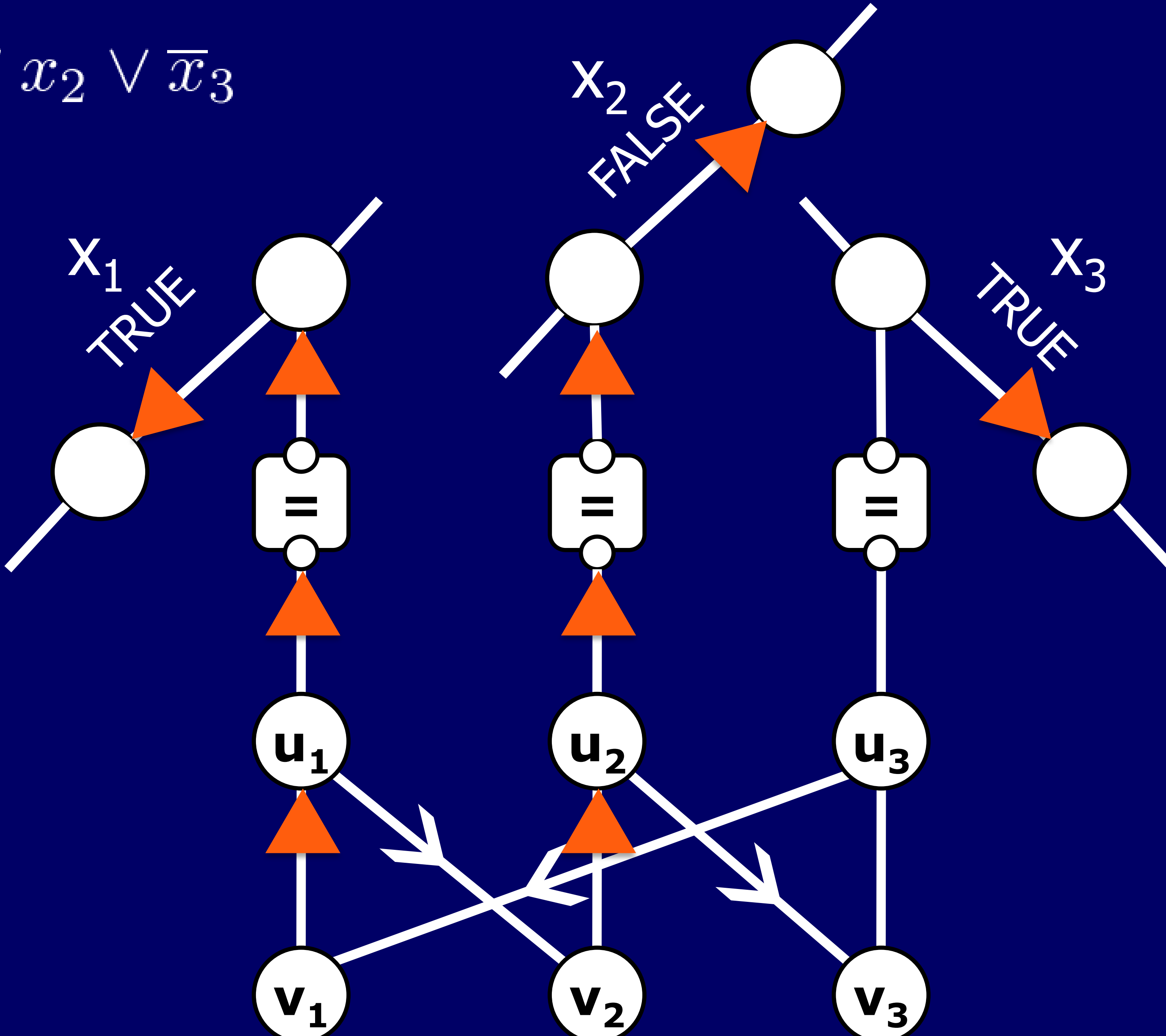
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



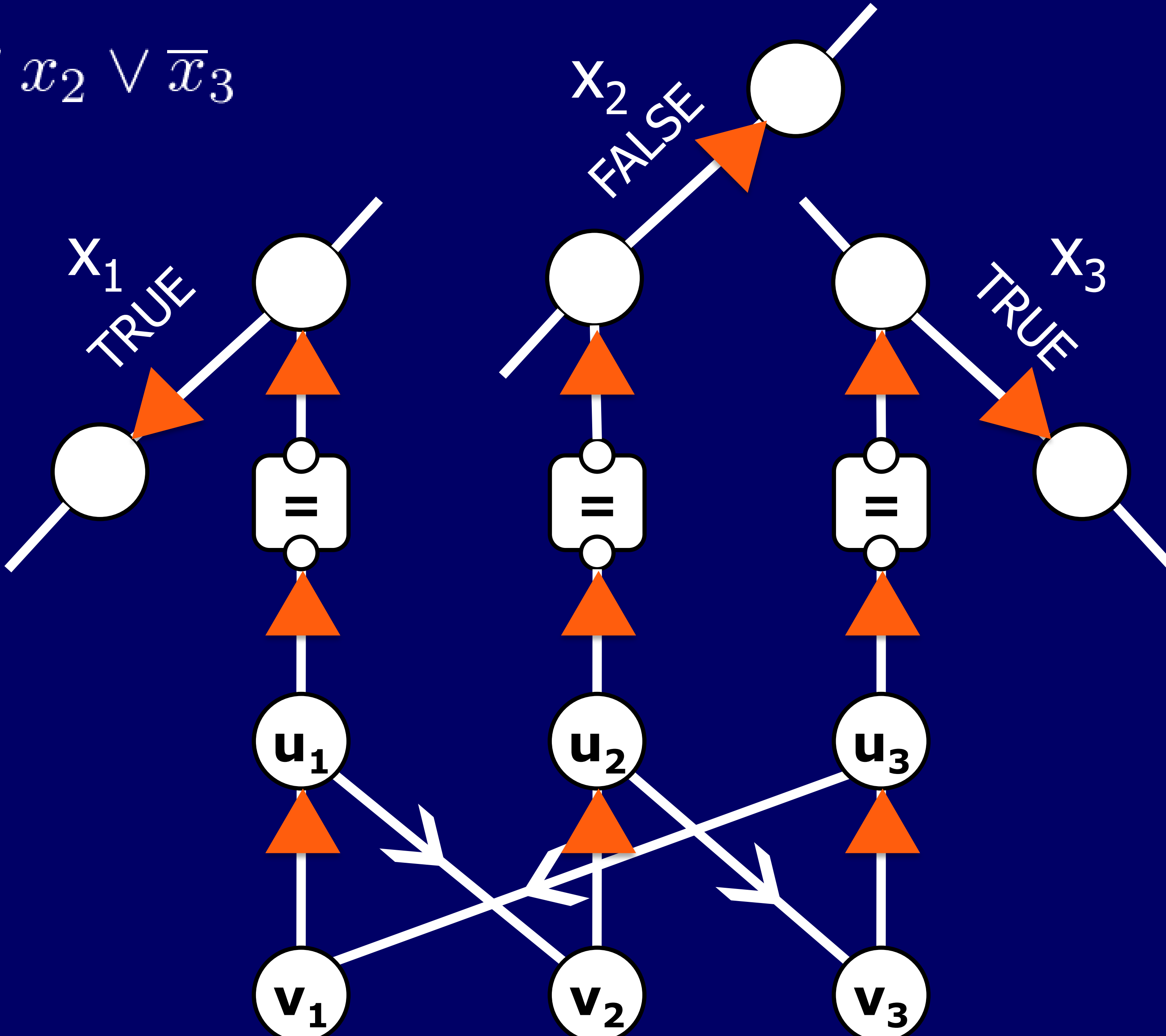
$$\overline{x}_1 \vee x_2 \vee \overline{x}_3$$



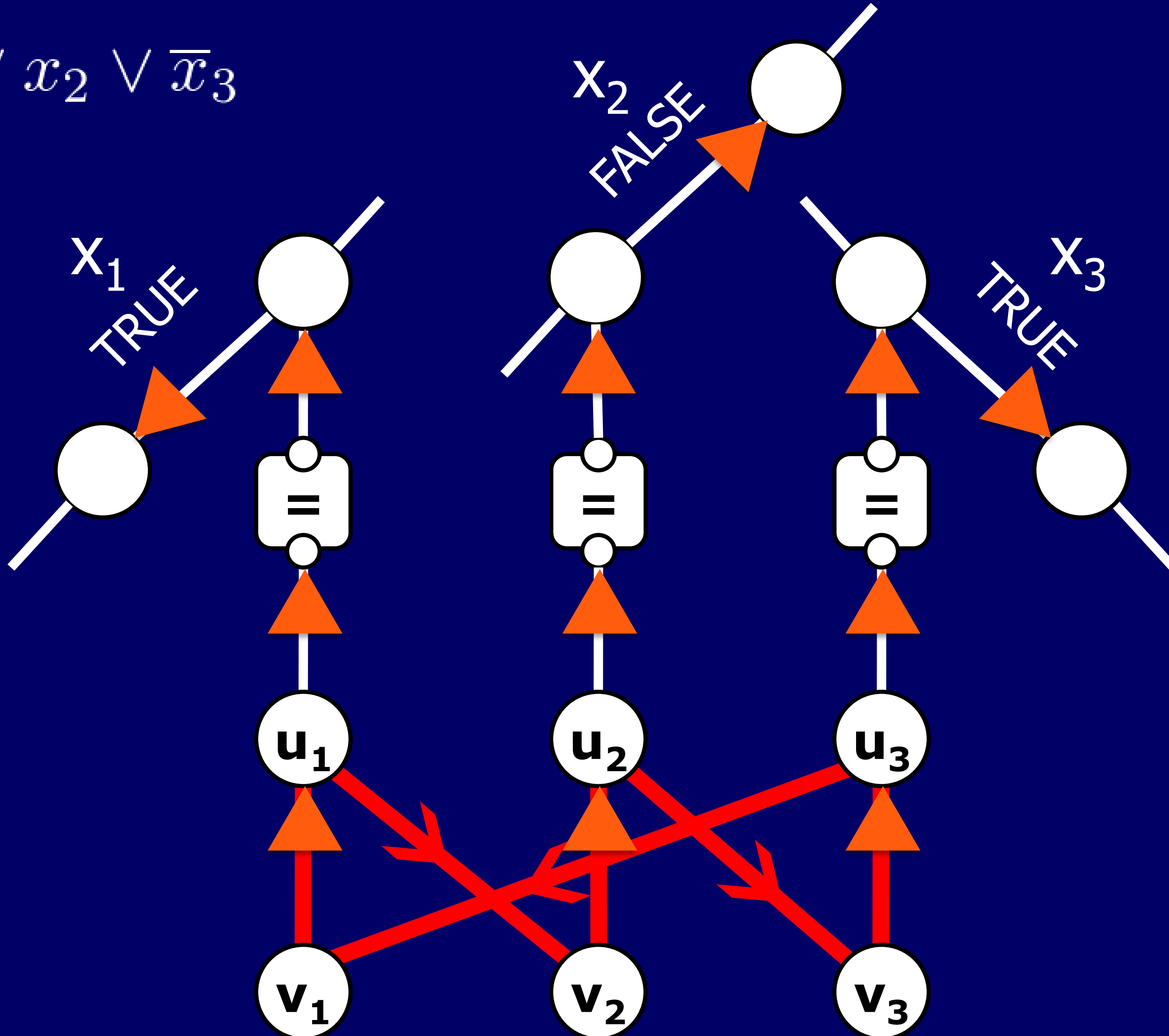
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



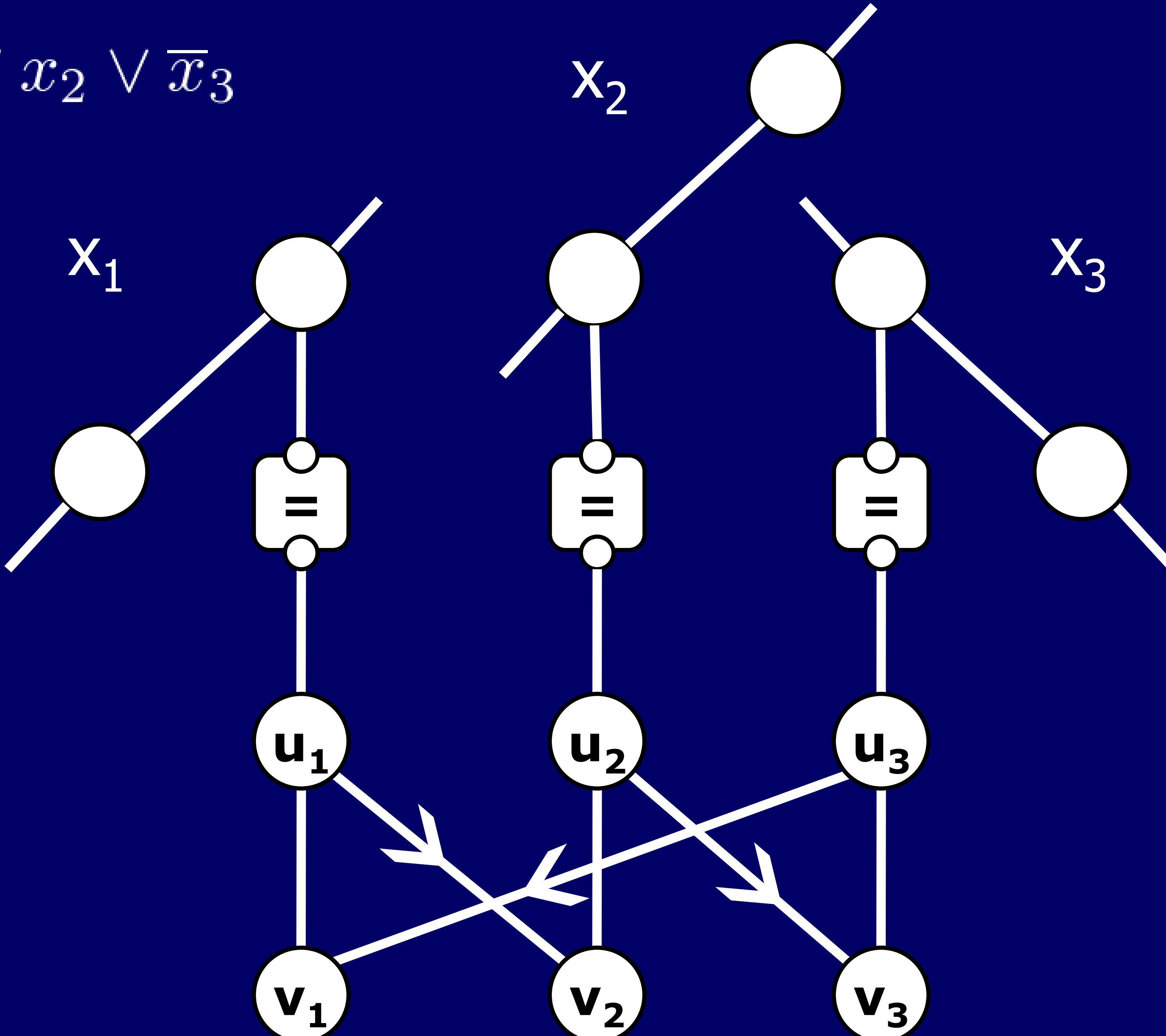
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



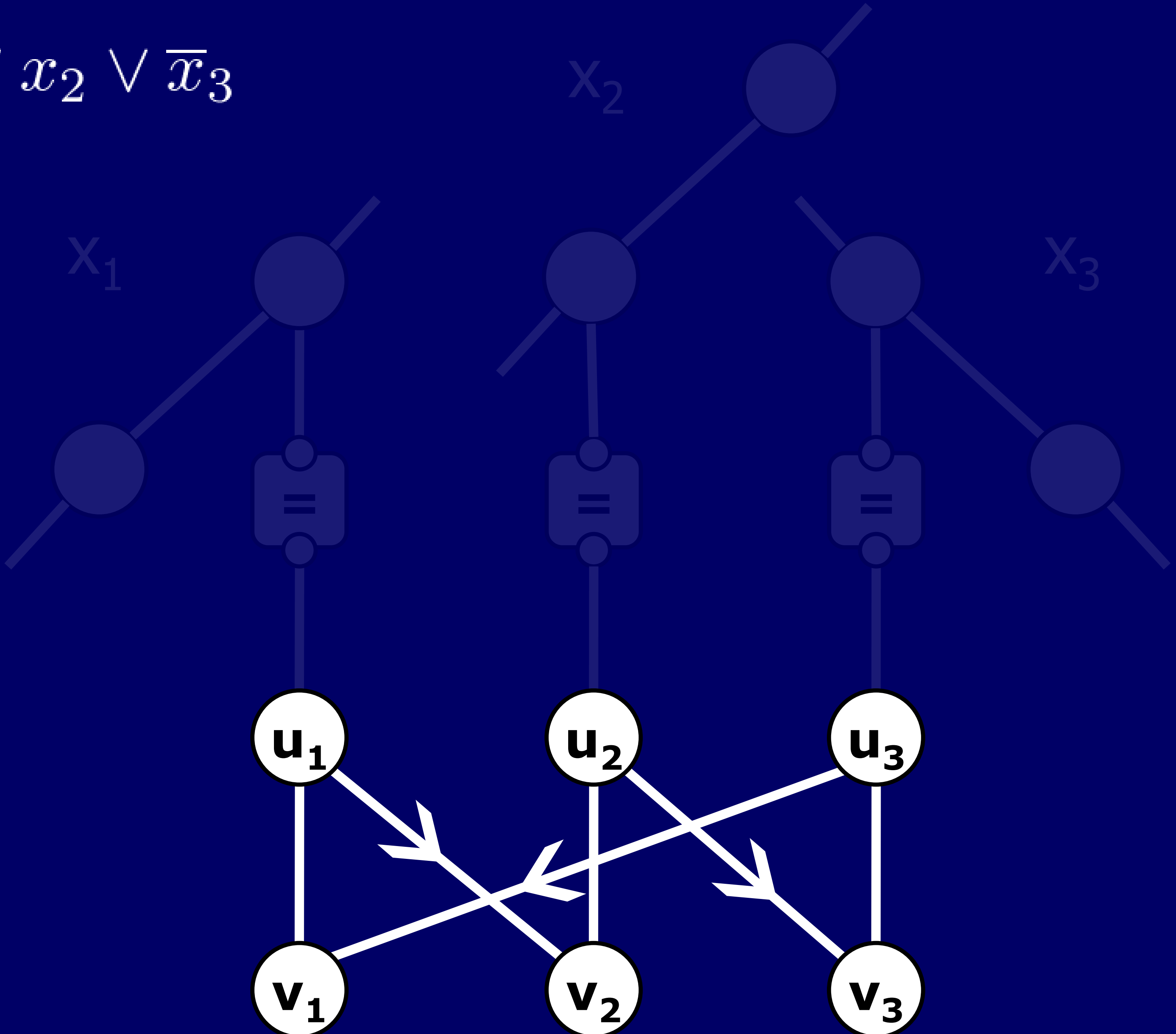
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



$\overline{x}_1 \vee x_2 \vee \overline{x}_3$

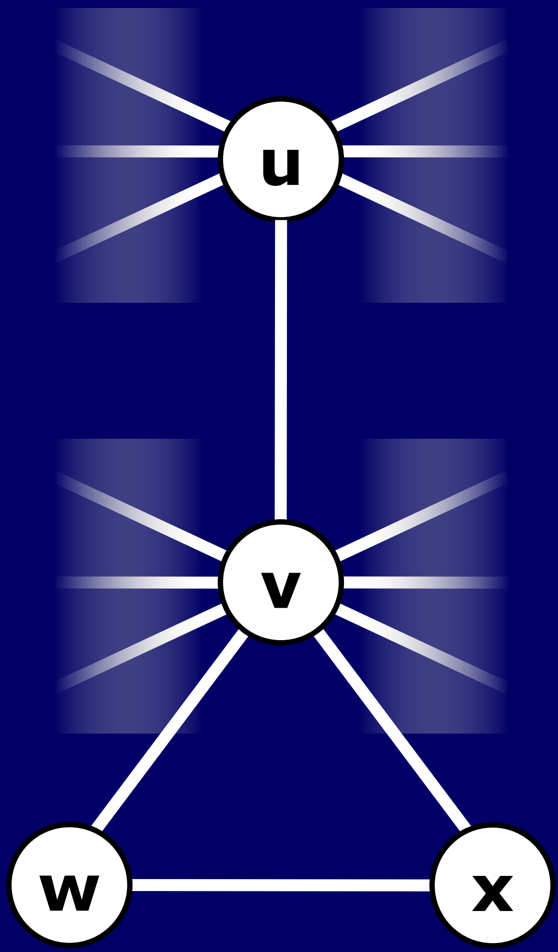


$\overline{x}_1 \lor x_2 \lor \overline{x}_3$



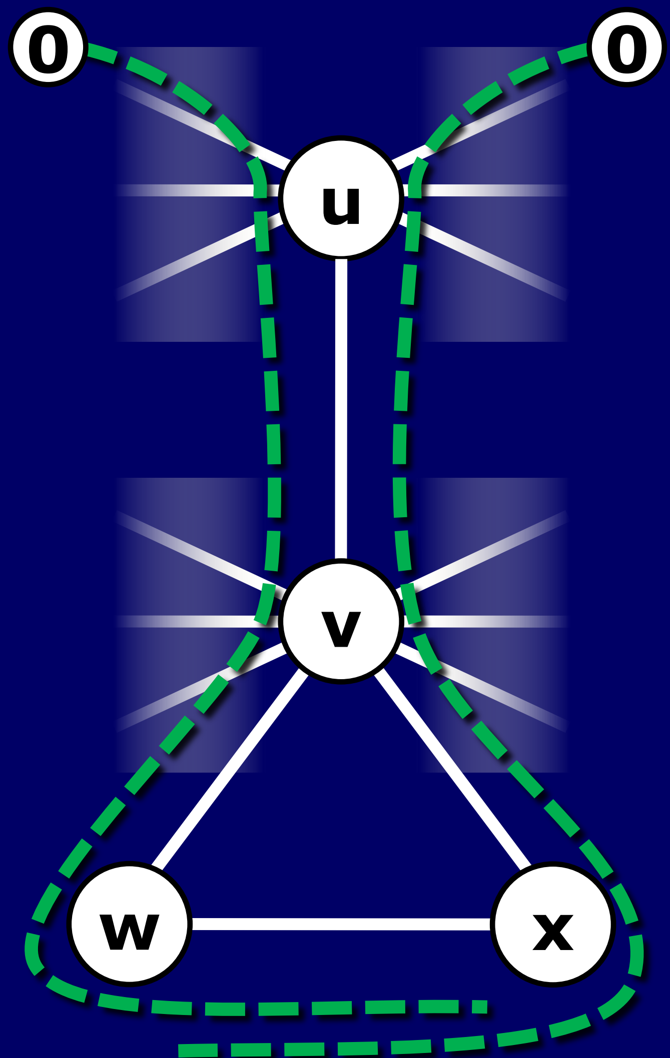
The Forcing Configuration

$\mathcal{FC}(u, v)$



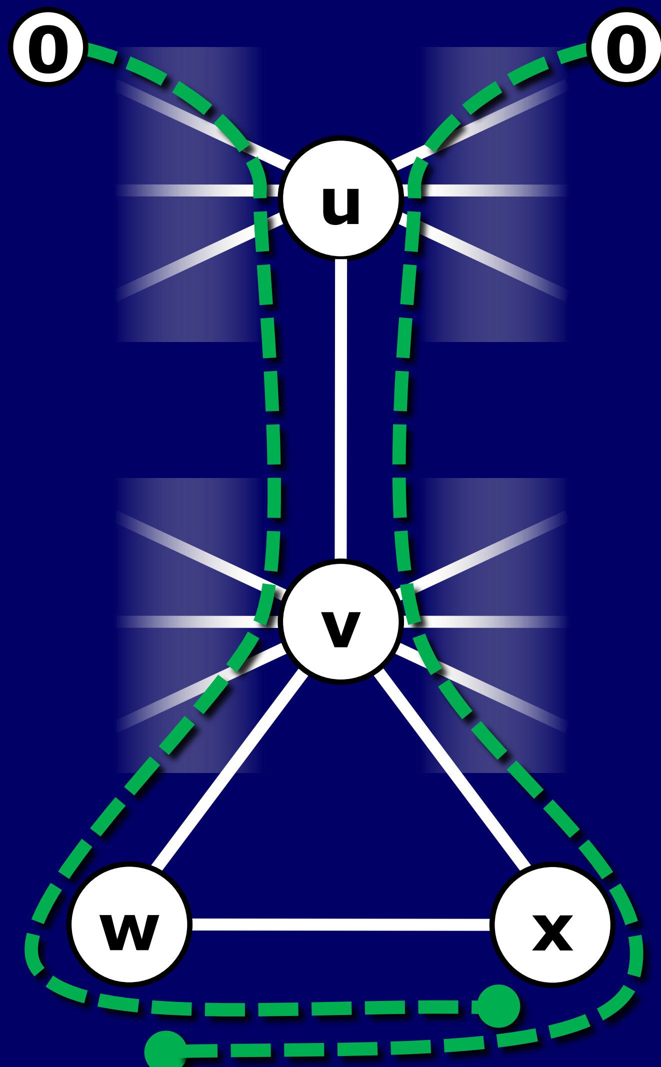
The Forcing Configuration

$\mathcal{FC}(u, v)$



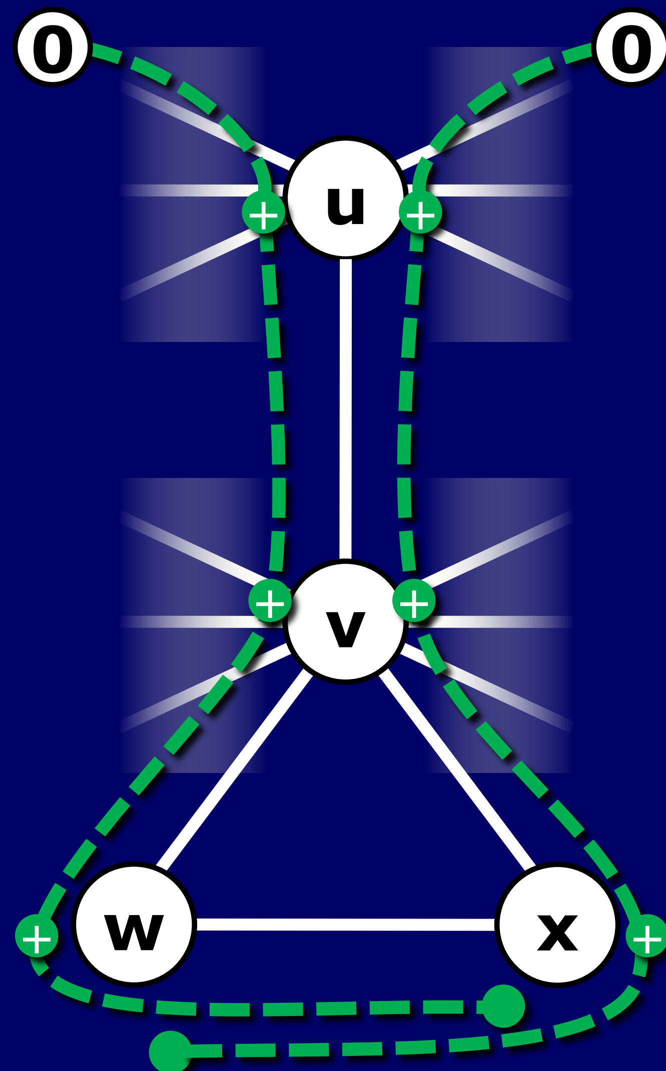
The Forcing Configuration

$\mathcal{FC}(u, v)$



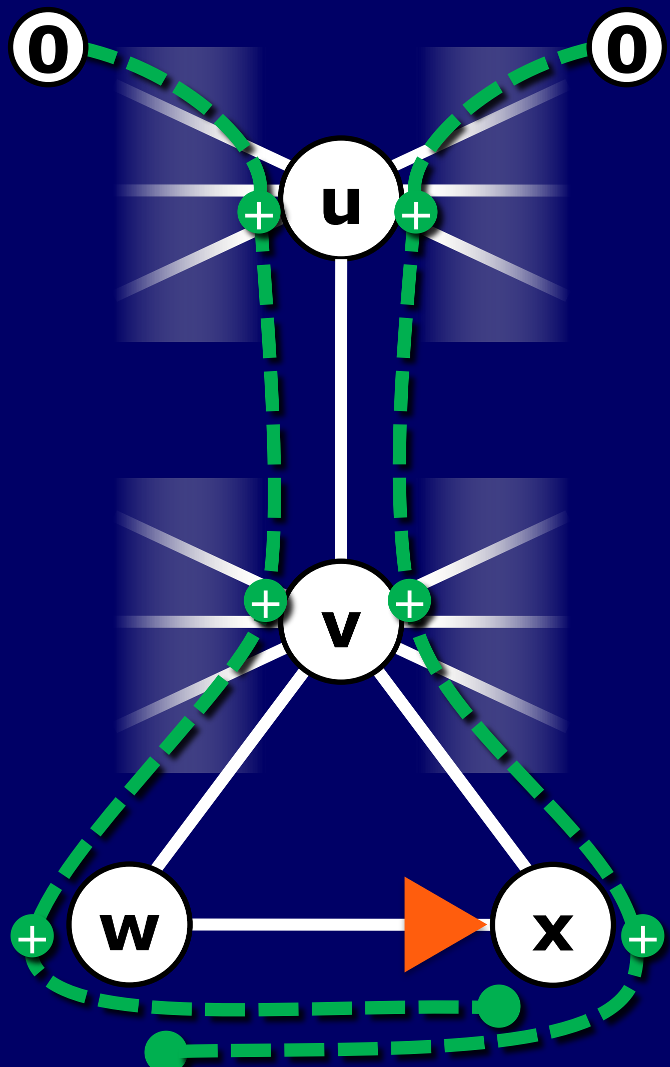
The Forcing Configuration

$\mathcal{FC}(u, v)$



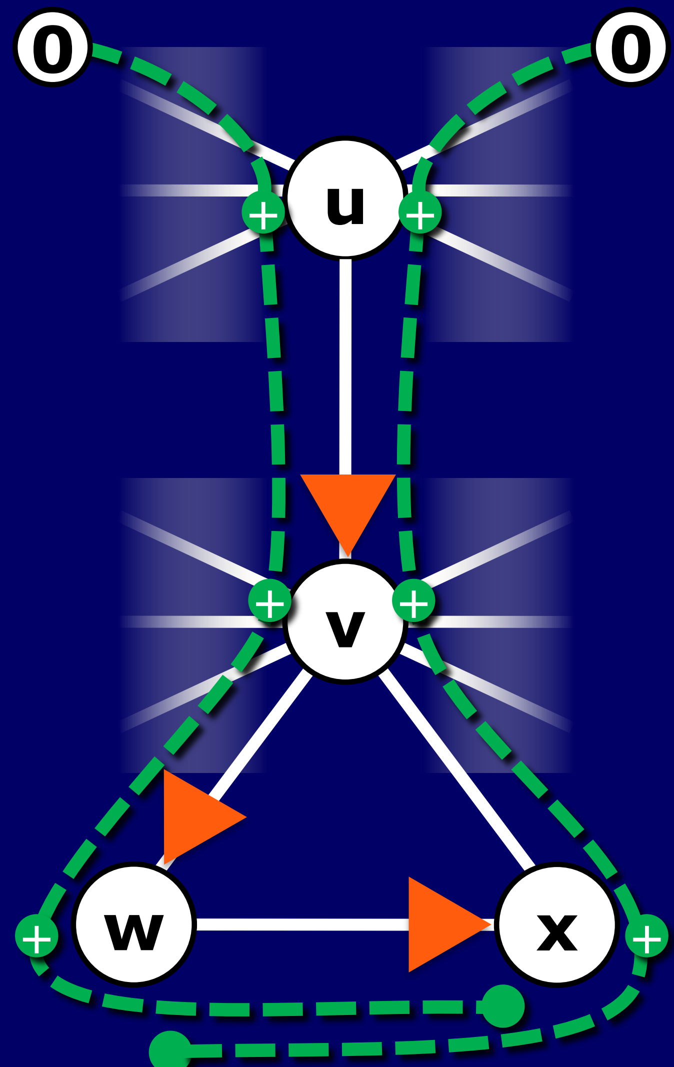
The Forcing Configuration

$\mathcal{FC}(u, v)$



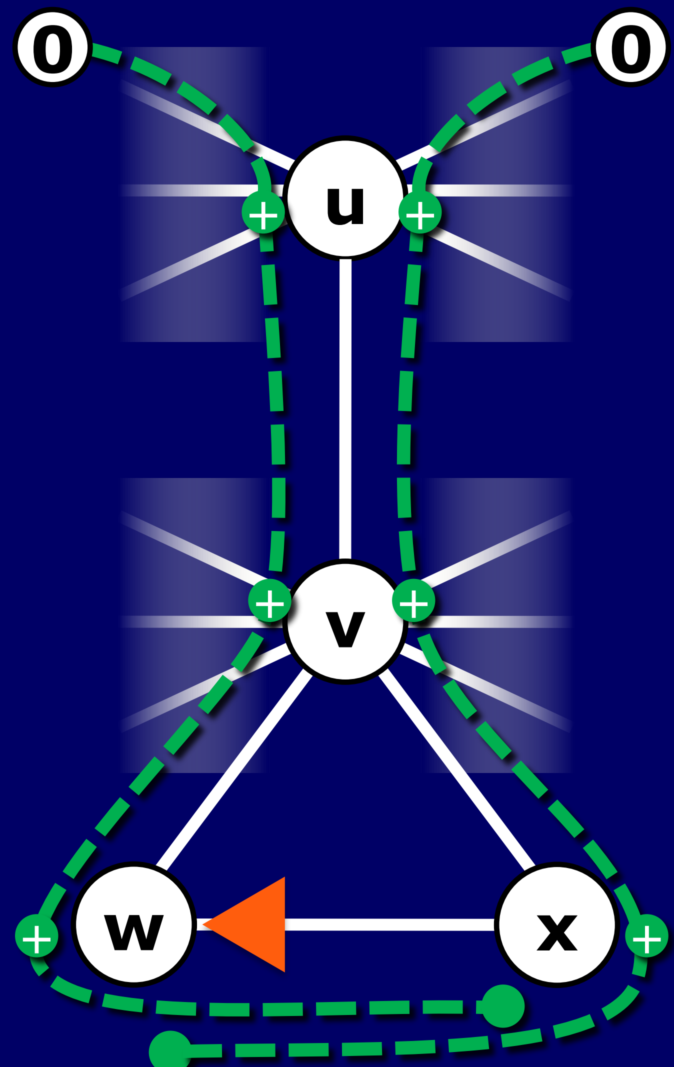
The Forcing Configuration

$\mathcal{FC}(u, v)$



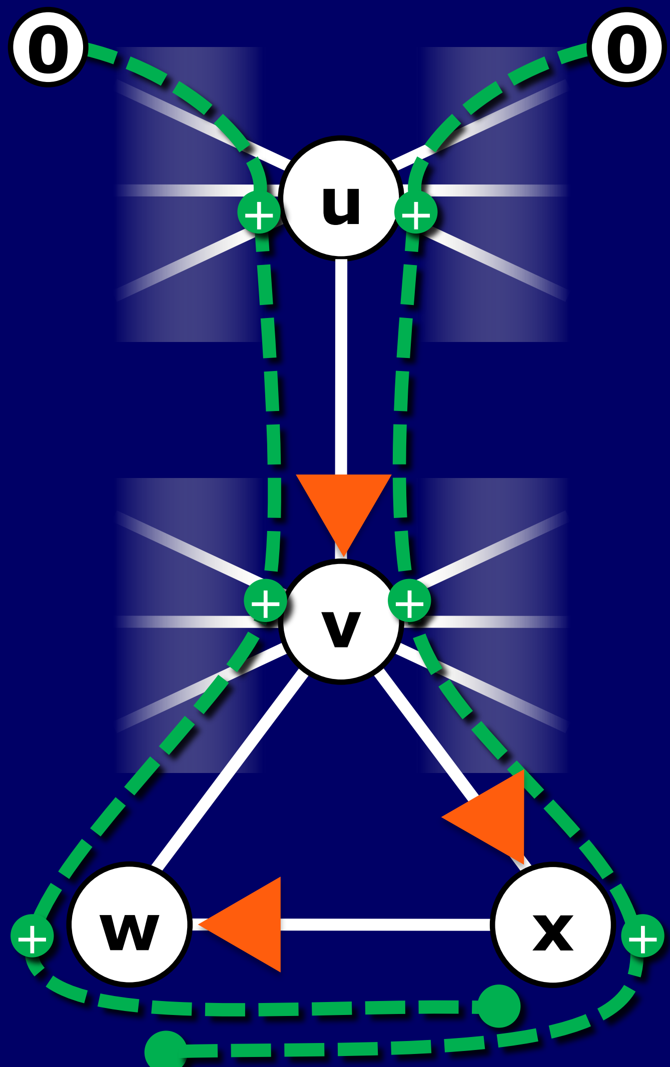
The Forcing Configuration

$\mathcal{FC}(u, v)$



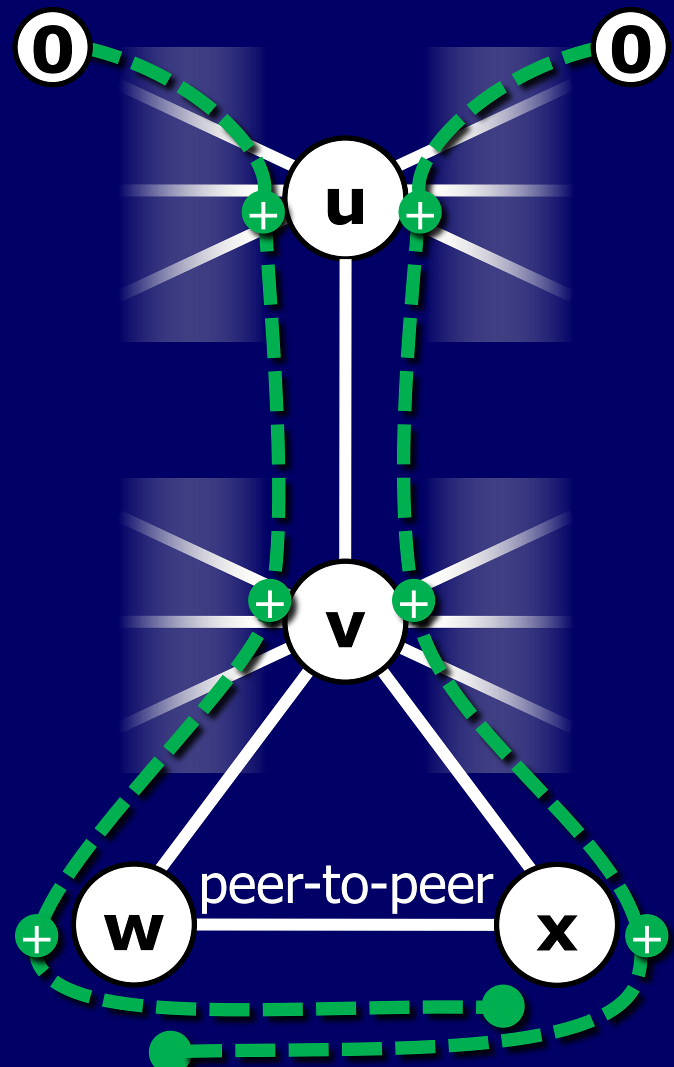
The Forcing Configuration

$\mathcal{FC}(u, v)$



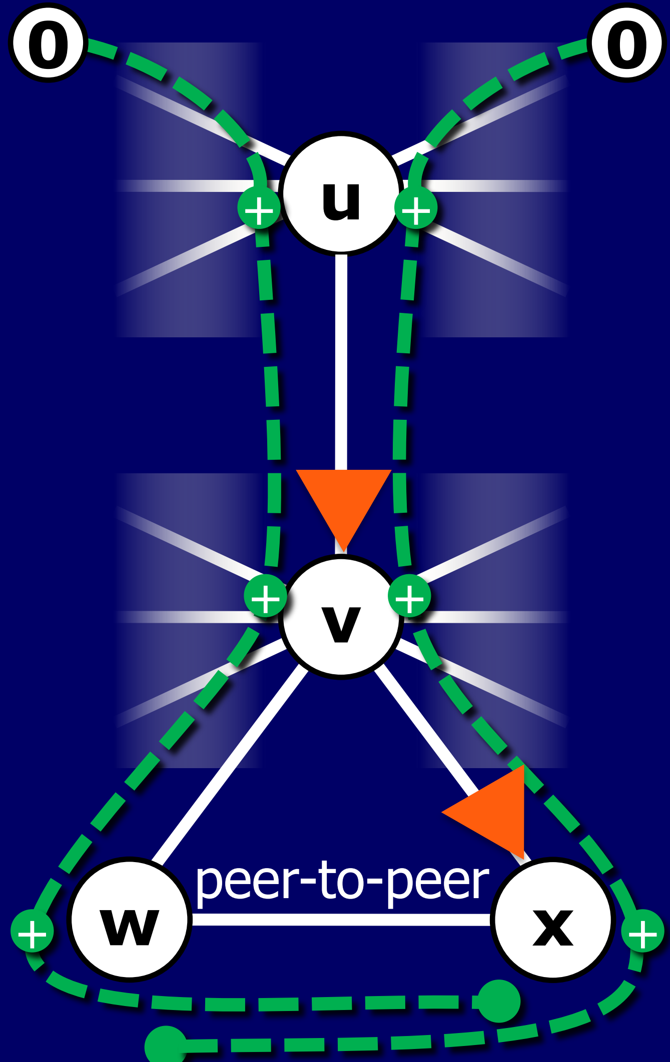
The Forcing Configuration

$\mathcal{FC}(u, v)$



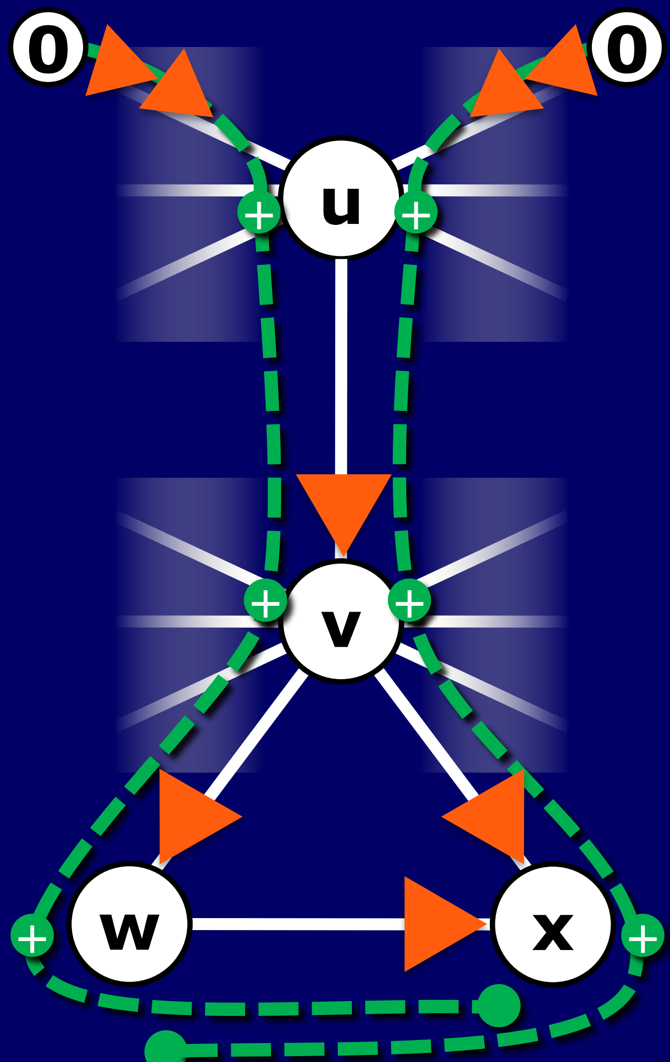
The Forcing Configuration

$\mathcal{FC}(u, v)$



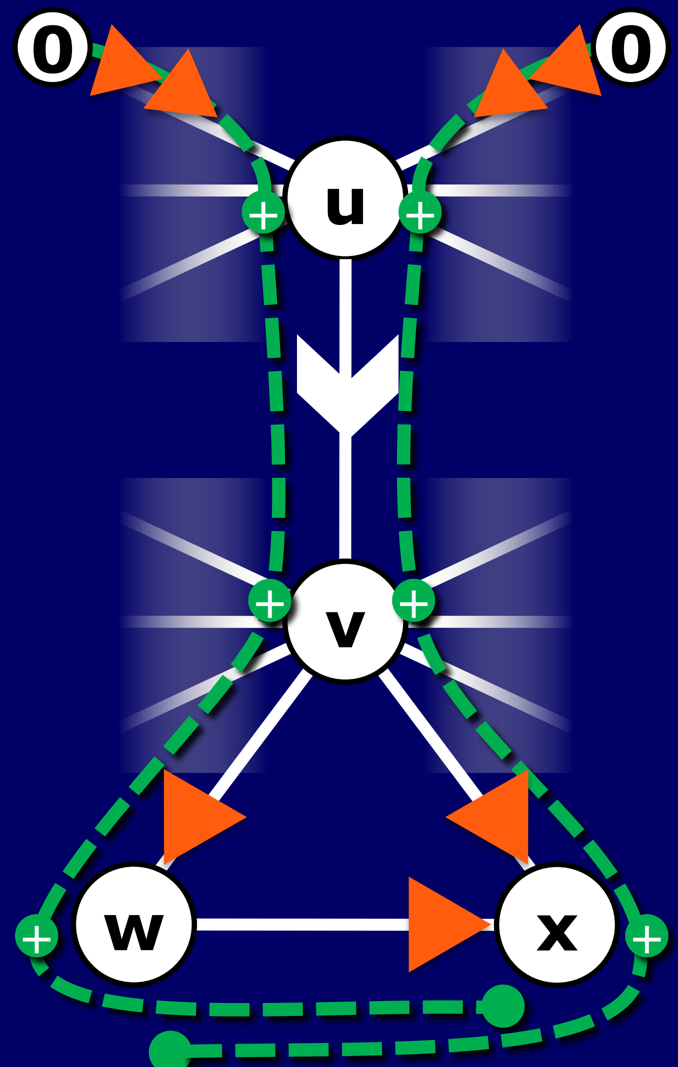
The Forcing Configuration

$\mathcal{FC}(u, v)$

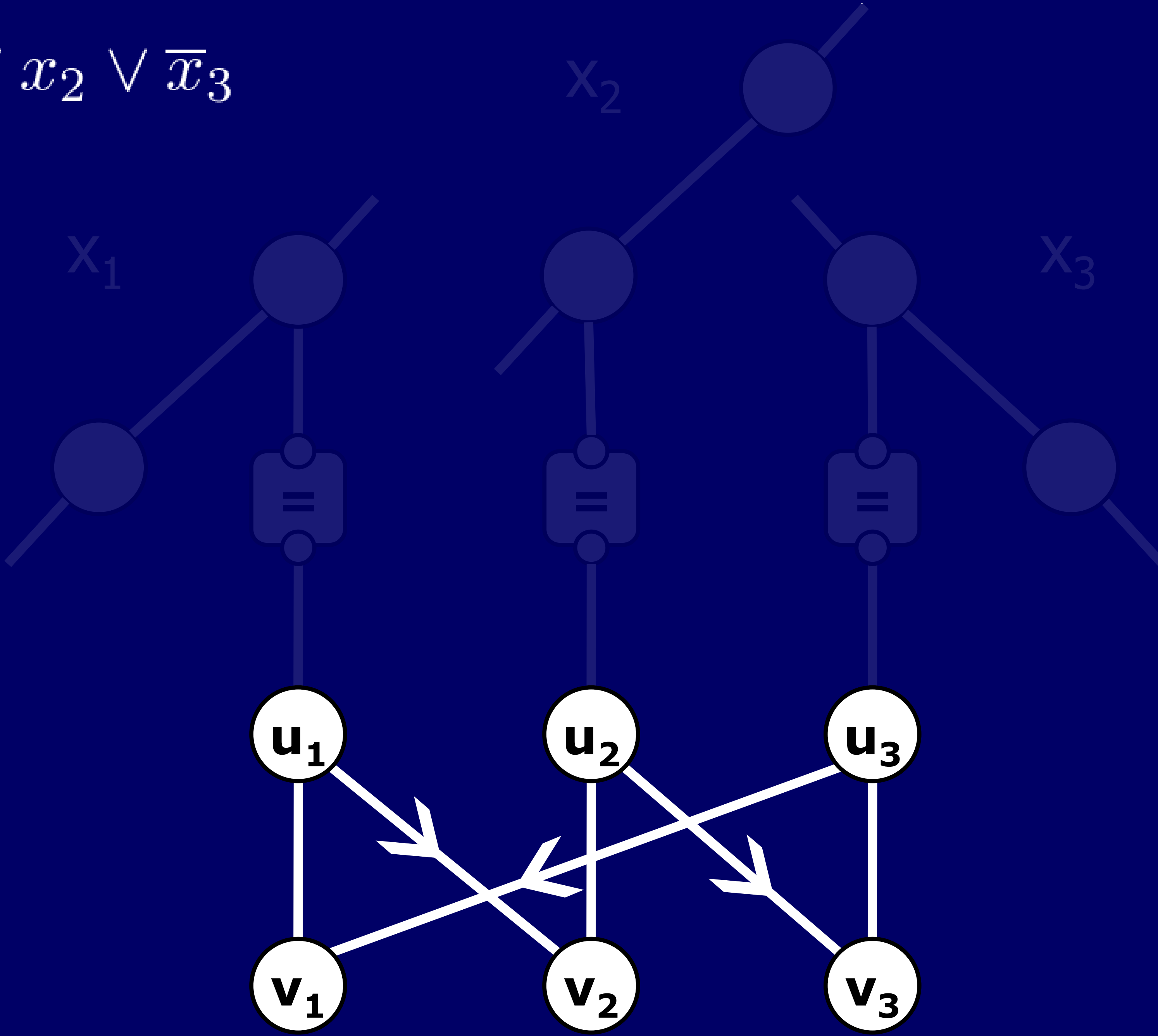


The Forcing Configuration

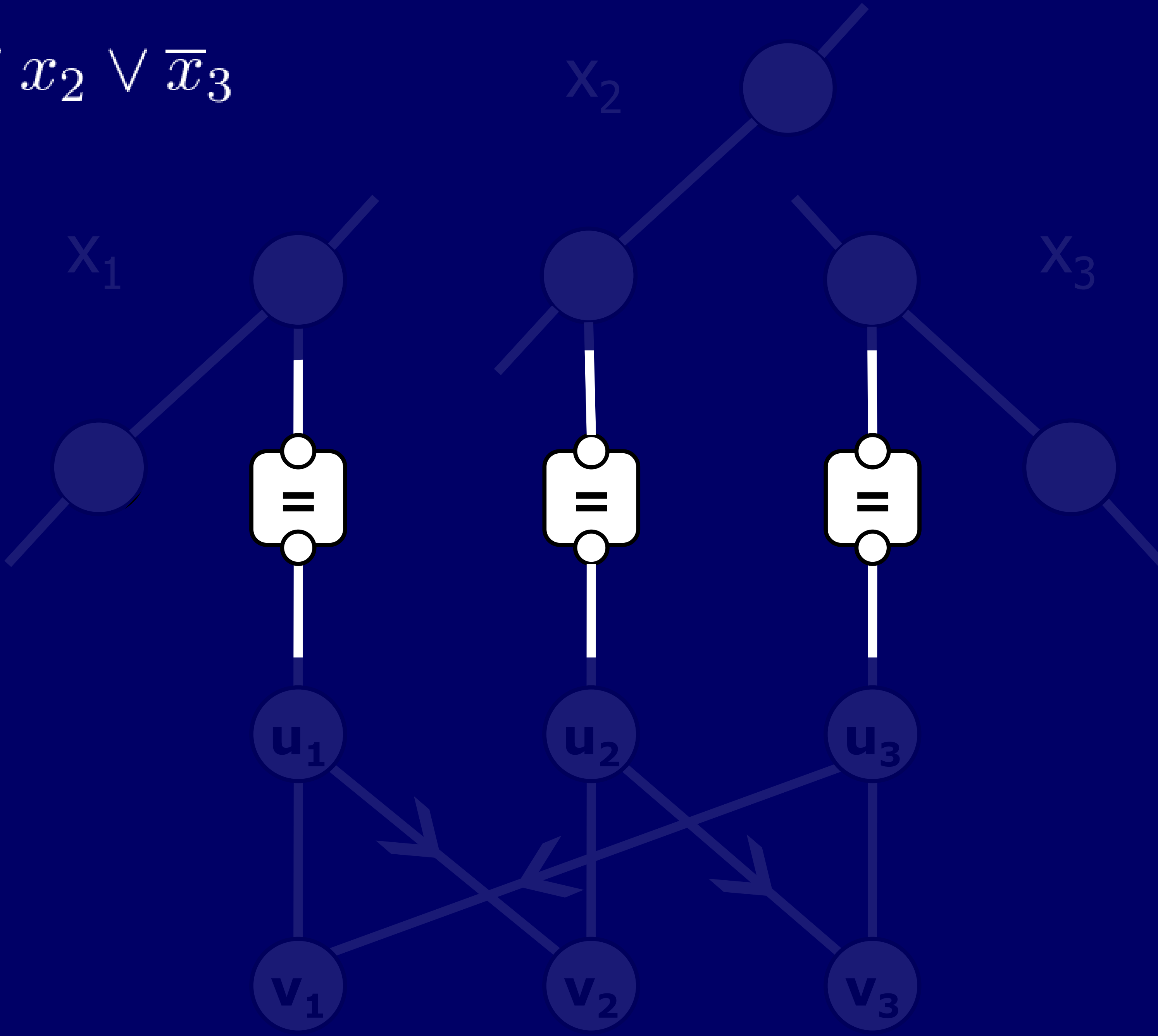
$\mathcal{FC}(u, v)$



$\overline{x}_1 \vee x_2 \vee \overline{x}_3$

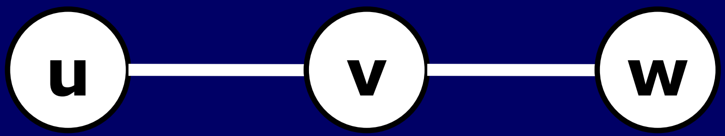


$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



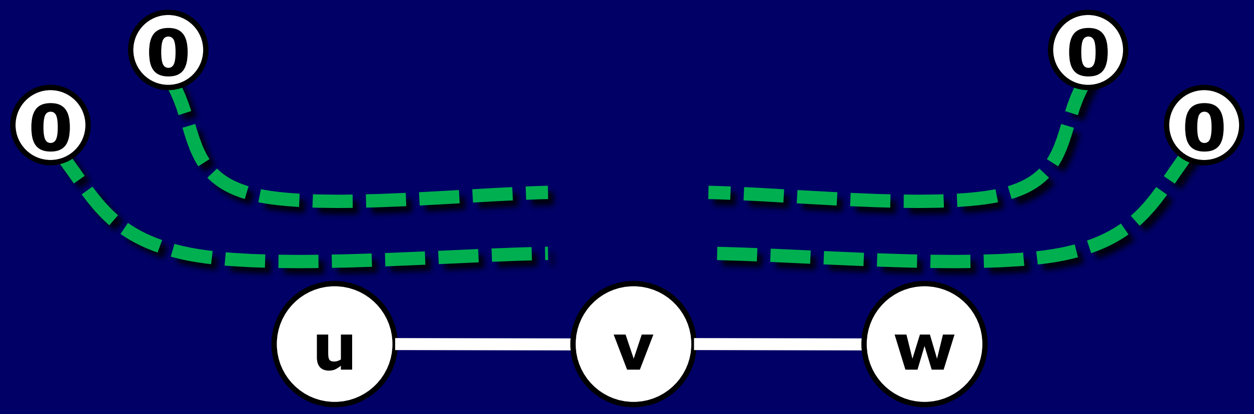
The Symmetric Configuration

$\mathcal{SC}(u, v, w)$



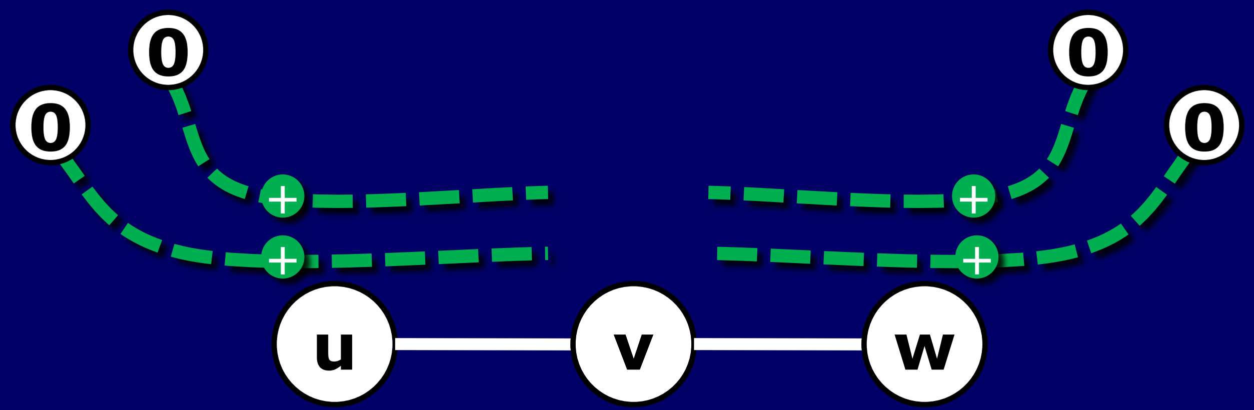
The Symmetric Configuration

$\mathcal{SC}(u, v, w)$



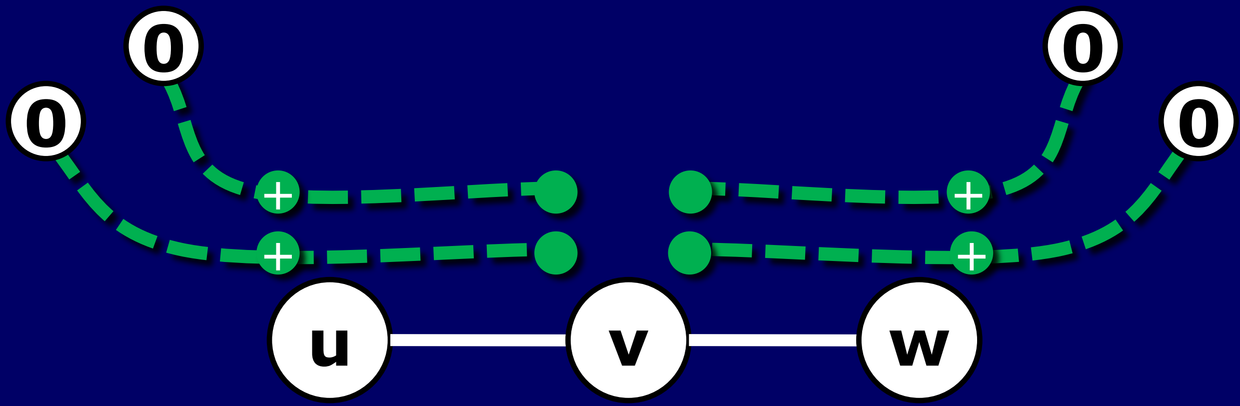
The Symmetric Configuration

$\mathcal{SC}(u, v, w)$



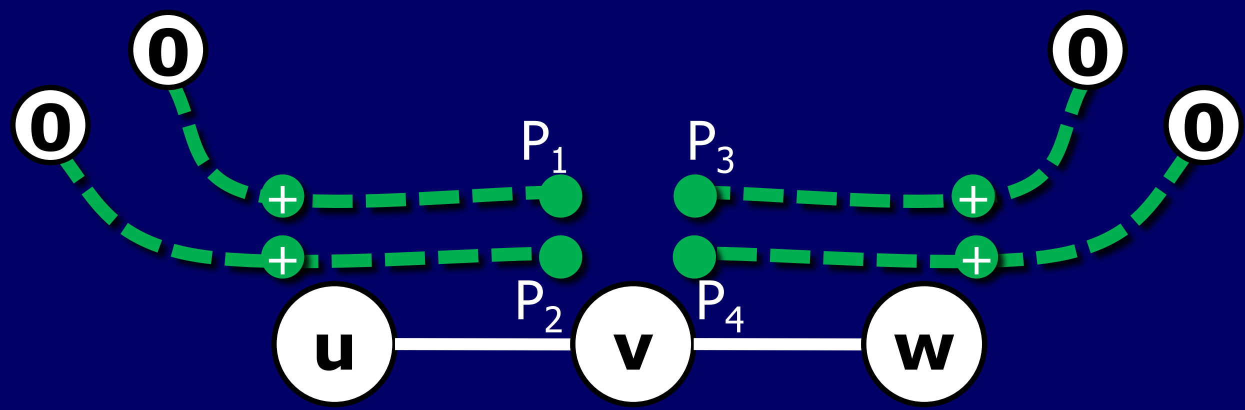
The Symmetric Configuration

$SC(u, v, w)$



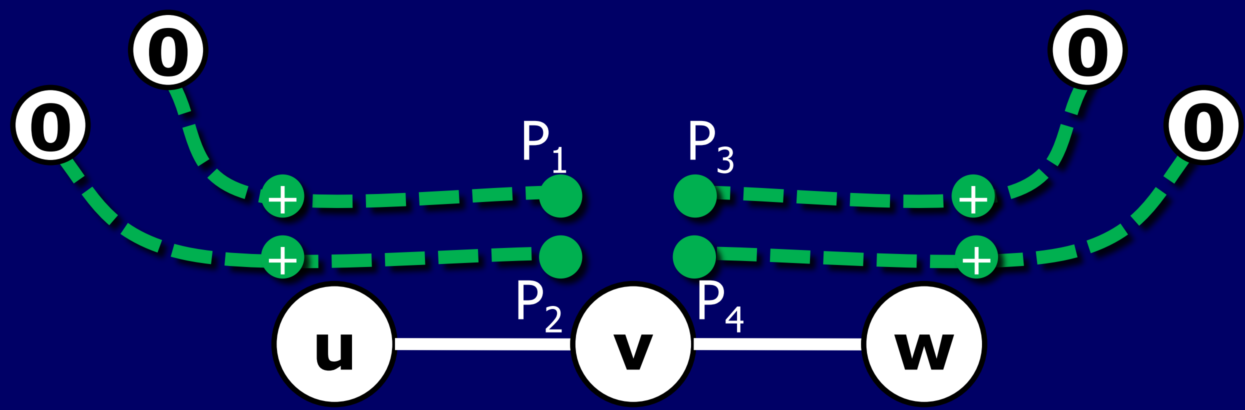
The Symmetric Configuration

$\mathcal{SC}(u, v, w)$



The Symmetric Configuration

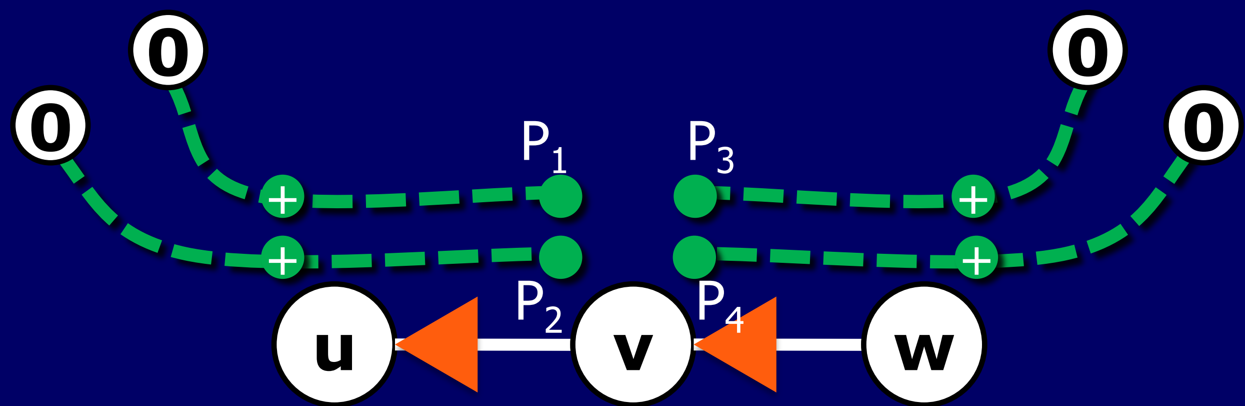
$\mathcal{SC}(u, v, w)$



overall rank:
P_1
P_3
P_4
P_2

The Symmetric Configuration

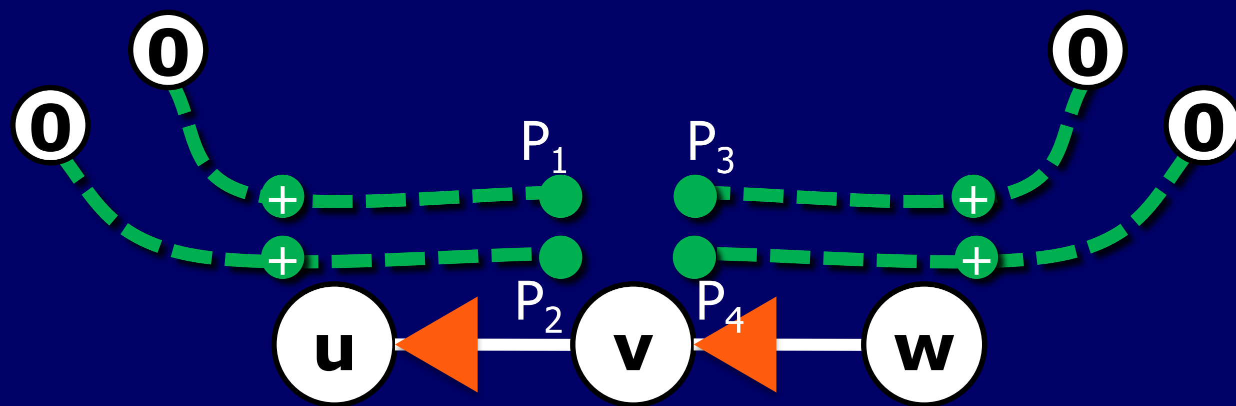
$\mathcal{SC}(u, v, w)$



overall rank:
P_1
P_3
P_4
P_2

The Symmetric Configuration

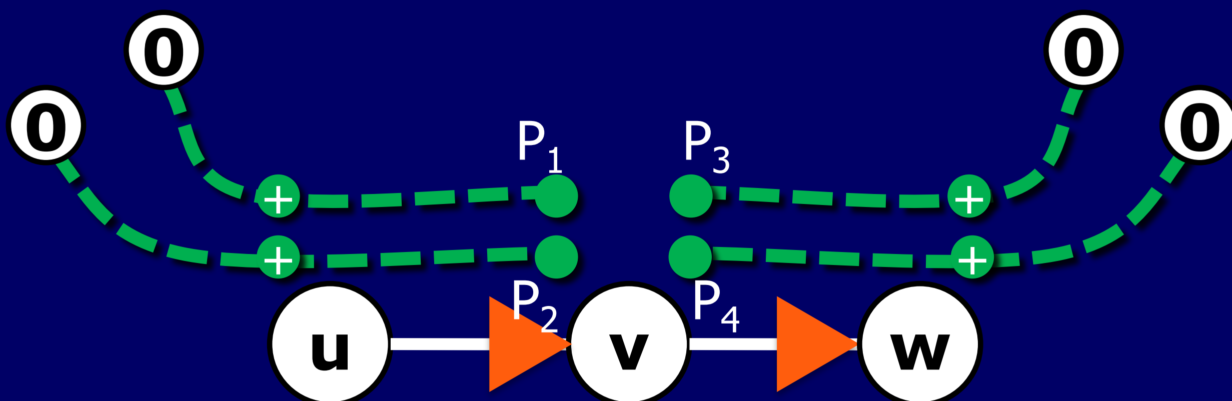
$\mathcal{SC}(u, v, w)$



overall rank:
P_1
P_3
P_4
P_2

The Symmetric Configuration

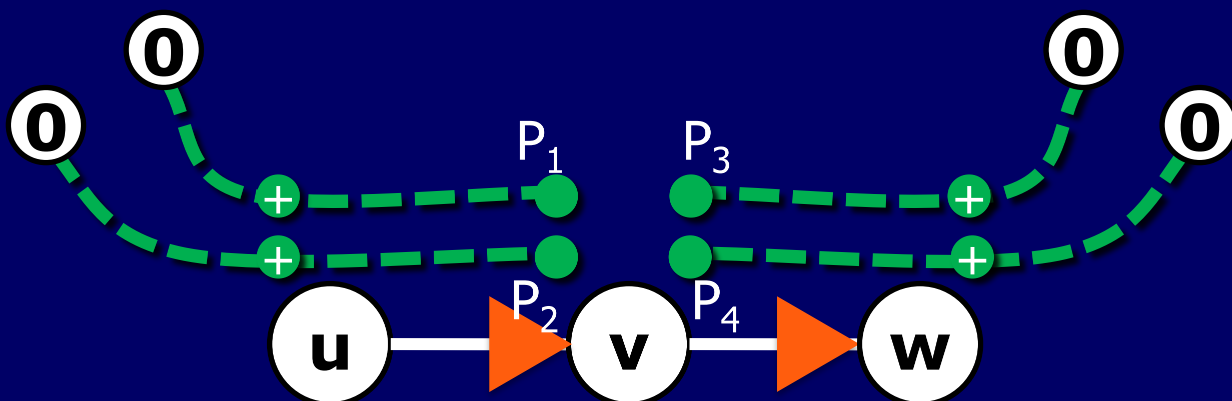
$\mathcal{SC}(u, v, w)$



overall rank:
P_1
P_3
P_4
P_2

The Symmetric Configuration

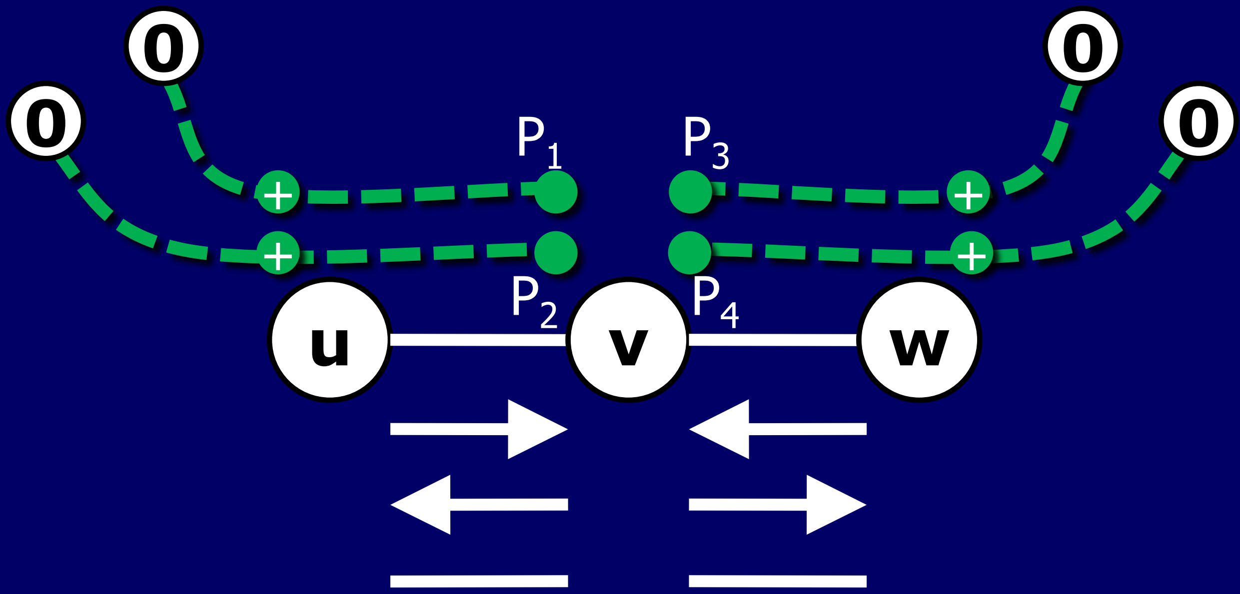
$\mathcal{SC}(u, v, w)$



overall rank:
P_1
P_3
P_4
P_2

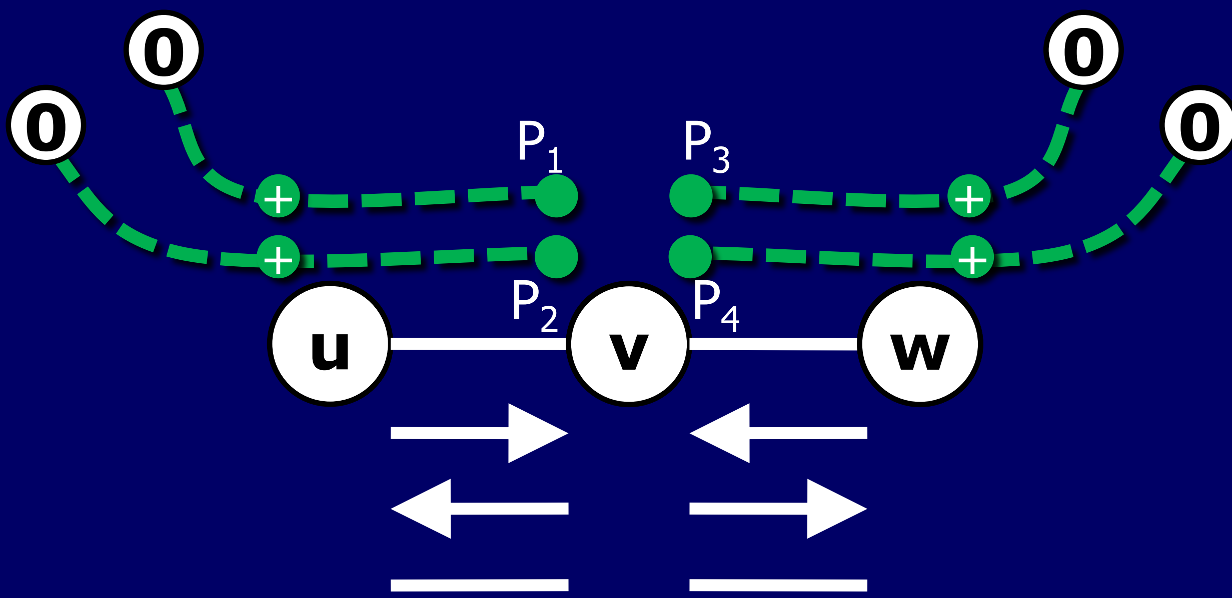
The Symmetric Configuration

$SC(u, v, w)$



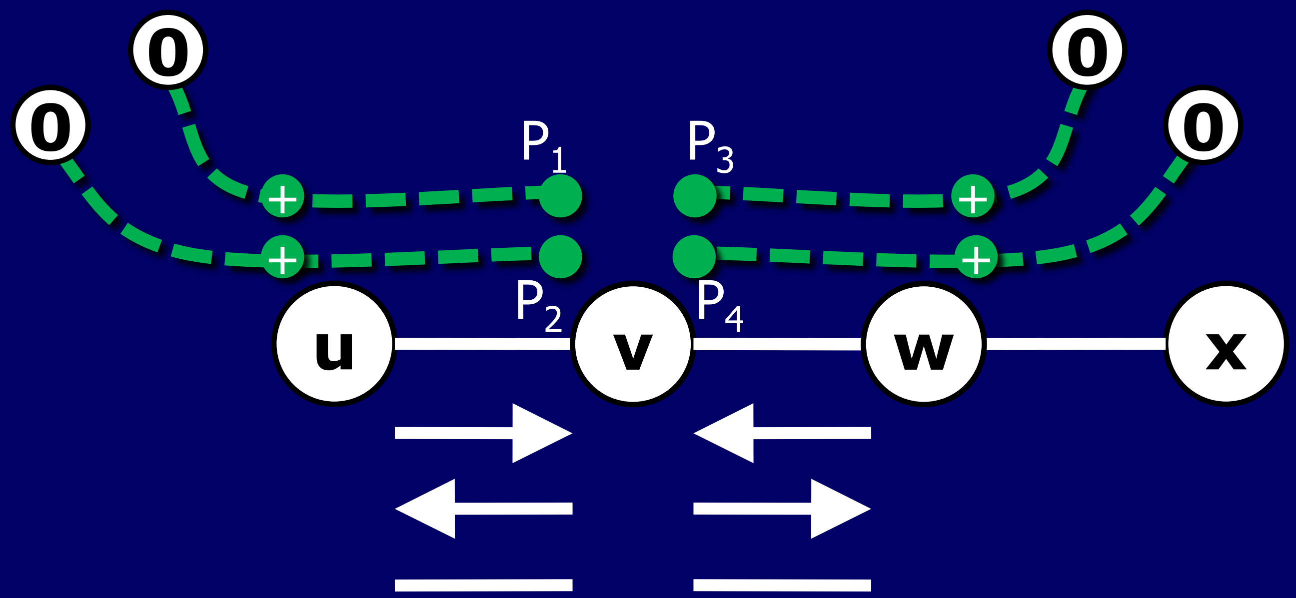
The De-Coupling Configuration

$$\mathcal{SC}(u, v, w) + \mathcal{SC}(v, w, x)$$



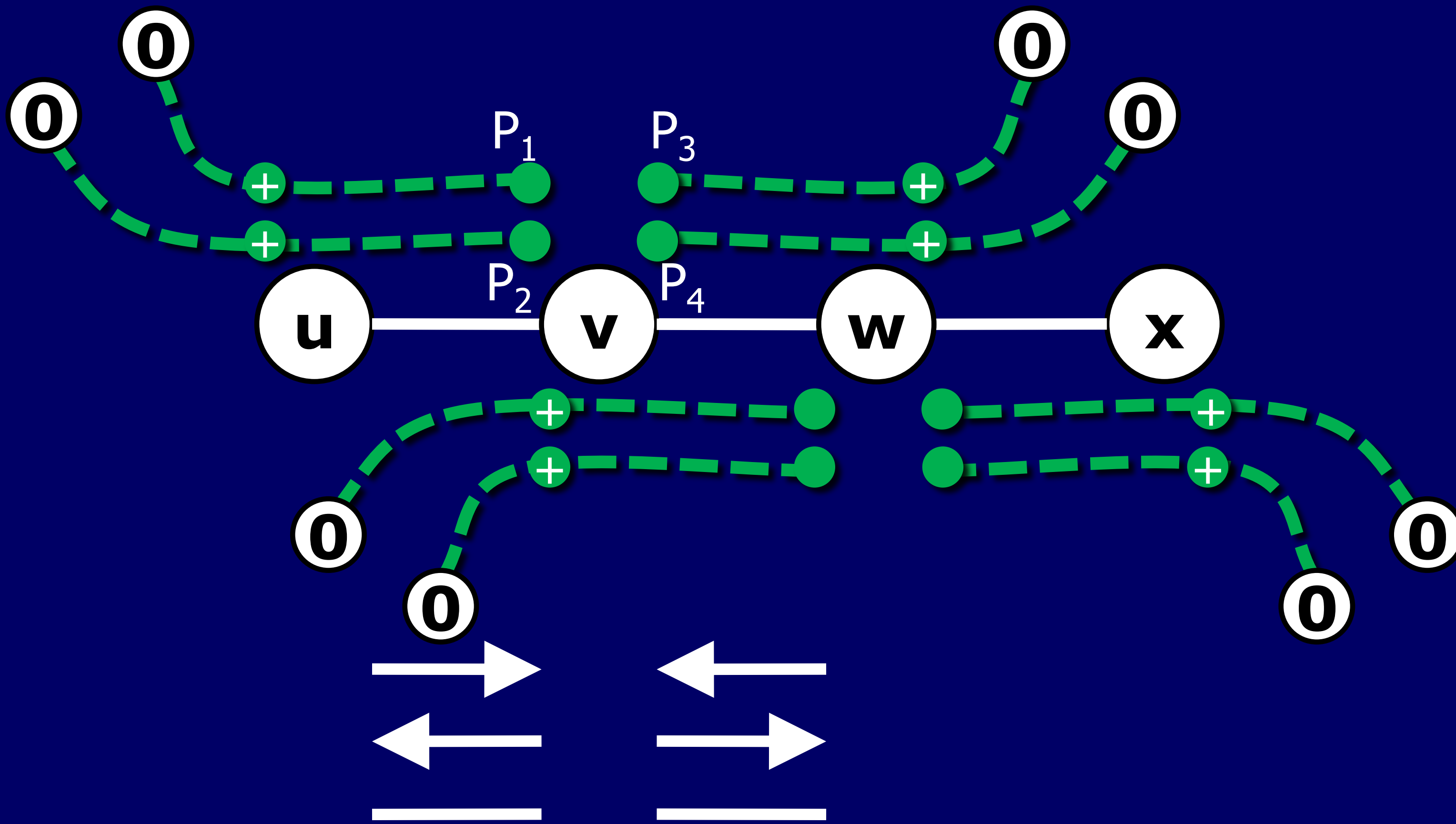
The De-Coupling Configuration

$$\mathcal{SC}(u, v, w) + \mathcal{SC}(v, w, x)$$



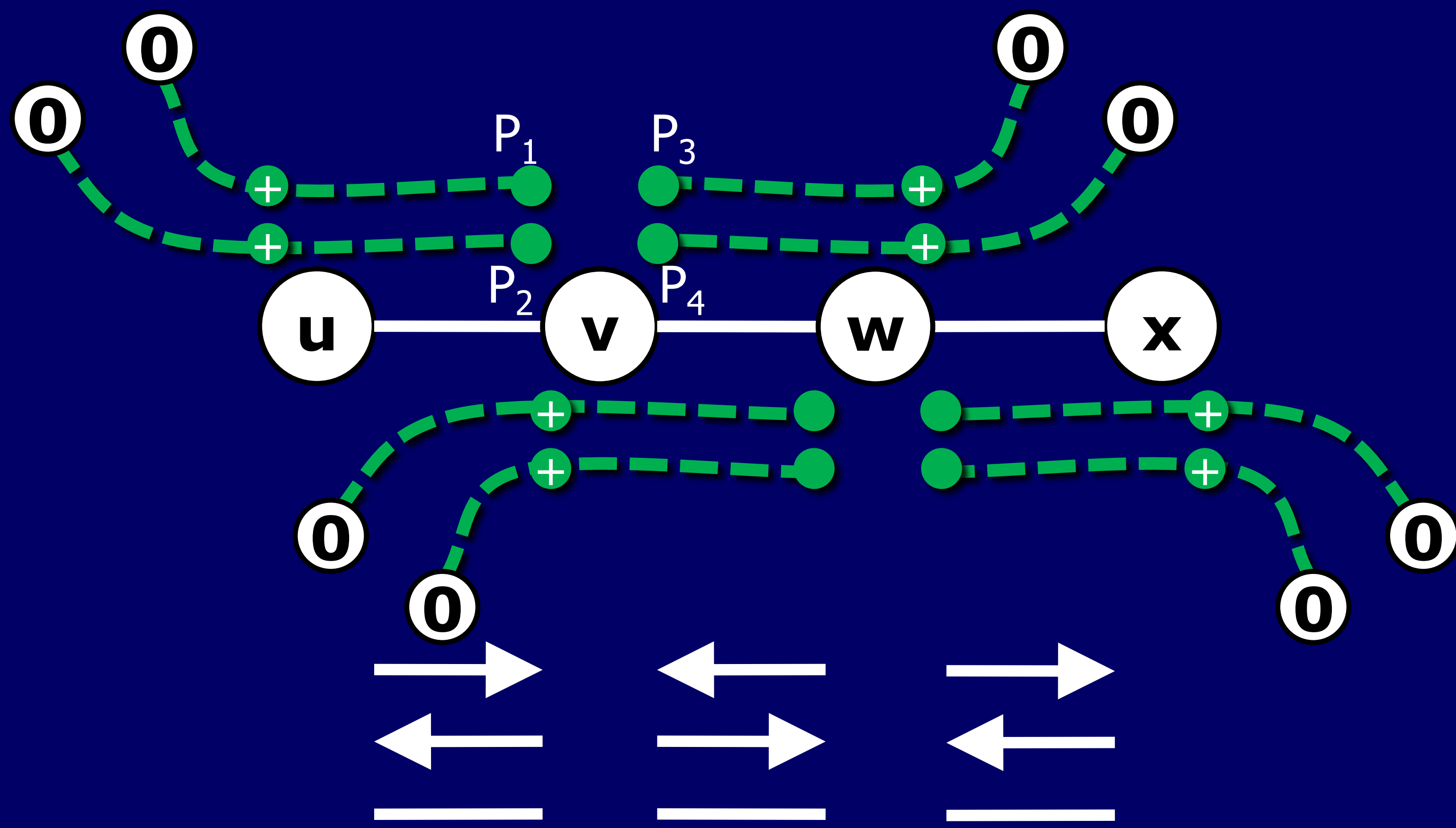
The De-Coupling Configuration

$$\mathcal{SC}(u, v, w) + \mathcal{SC}(v, w, x)$$



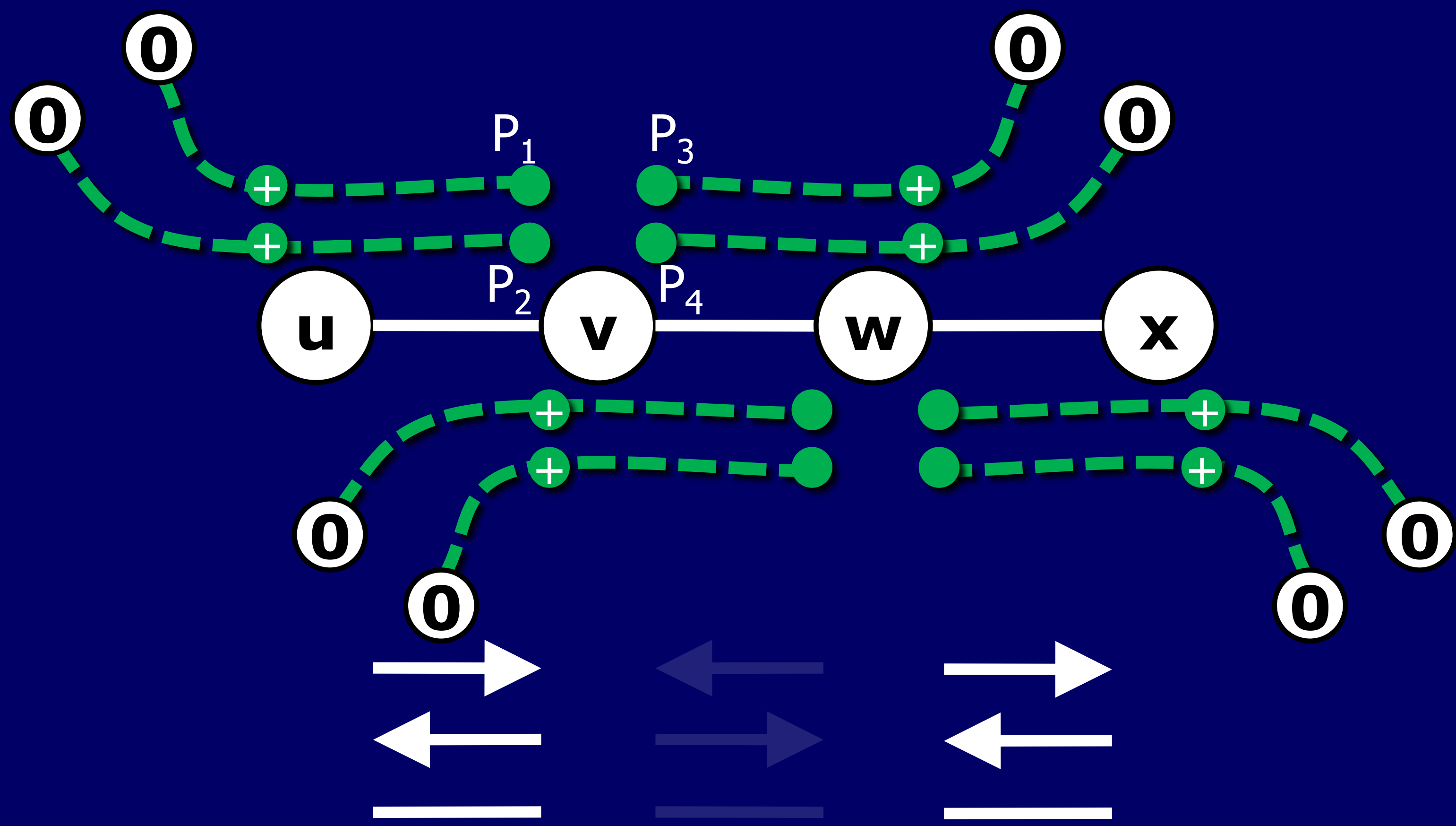
The De-Coupling Configuration

$$\mathcal{SC}(u, v, w) + \mathcal{SC}(v, w, x)$$



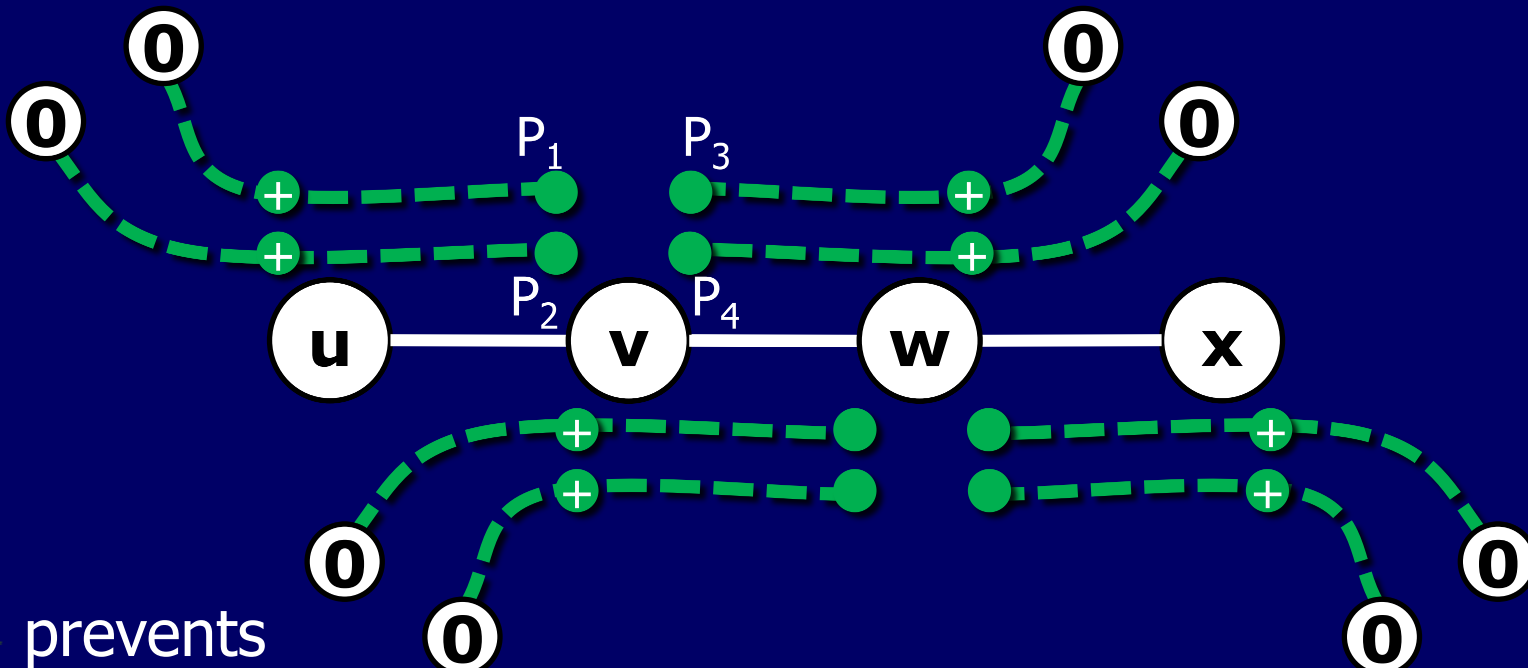
The De-Coupling Configuration

$$\mathcal{SC}(u, v, w) + \mathcal{SC}(v, w, x)$$

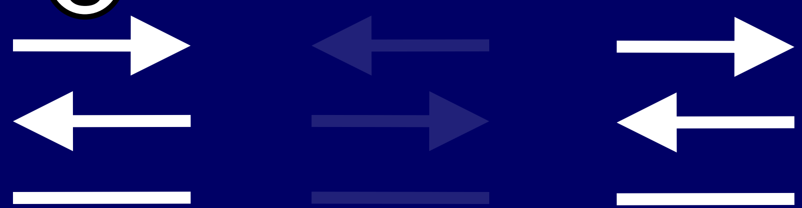


The De-Coupling Configuration

$$\mathcal{SC}(u, v, w) + \mathcal{SC}(v, w, x)$$

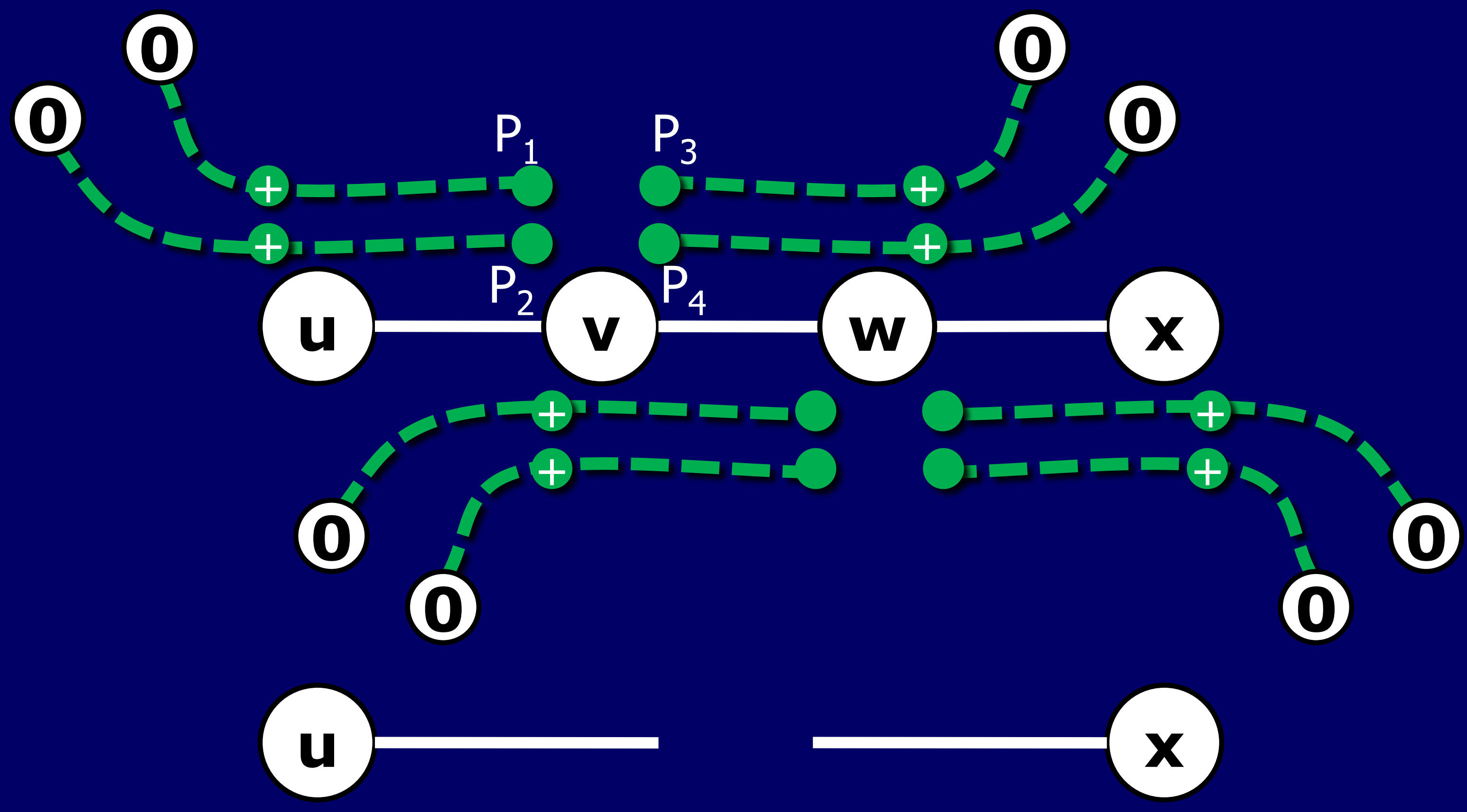


prevents directed cycles



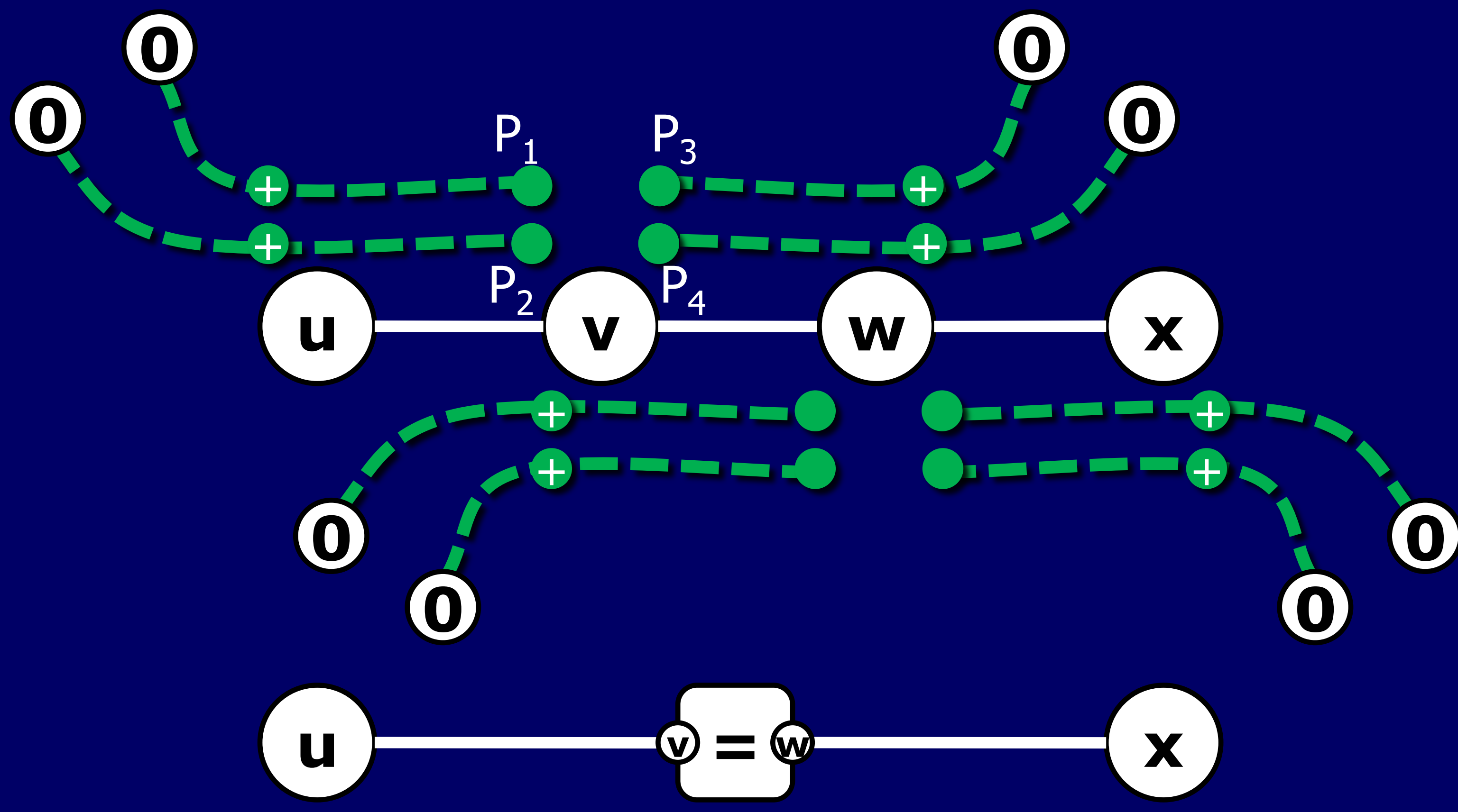
The De-Coupling Configuration

$$\mathcal{SC}(u, v, w) + \mathcal{SC}(v, w, x)$$

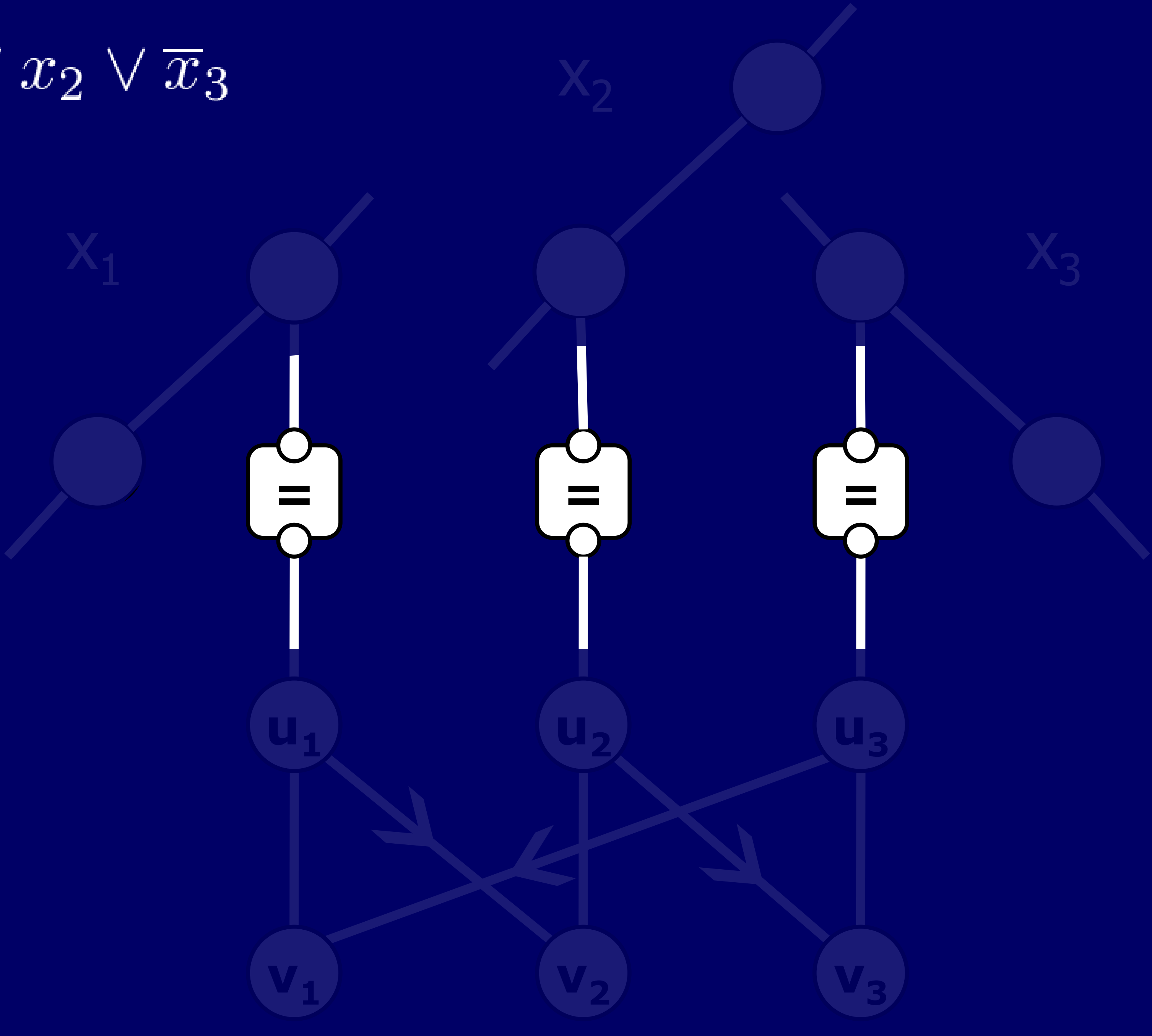


The De-Coupling Configuration

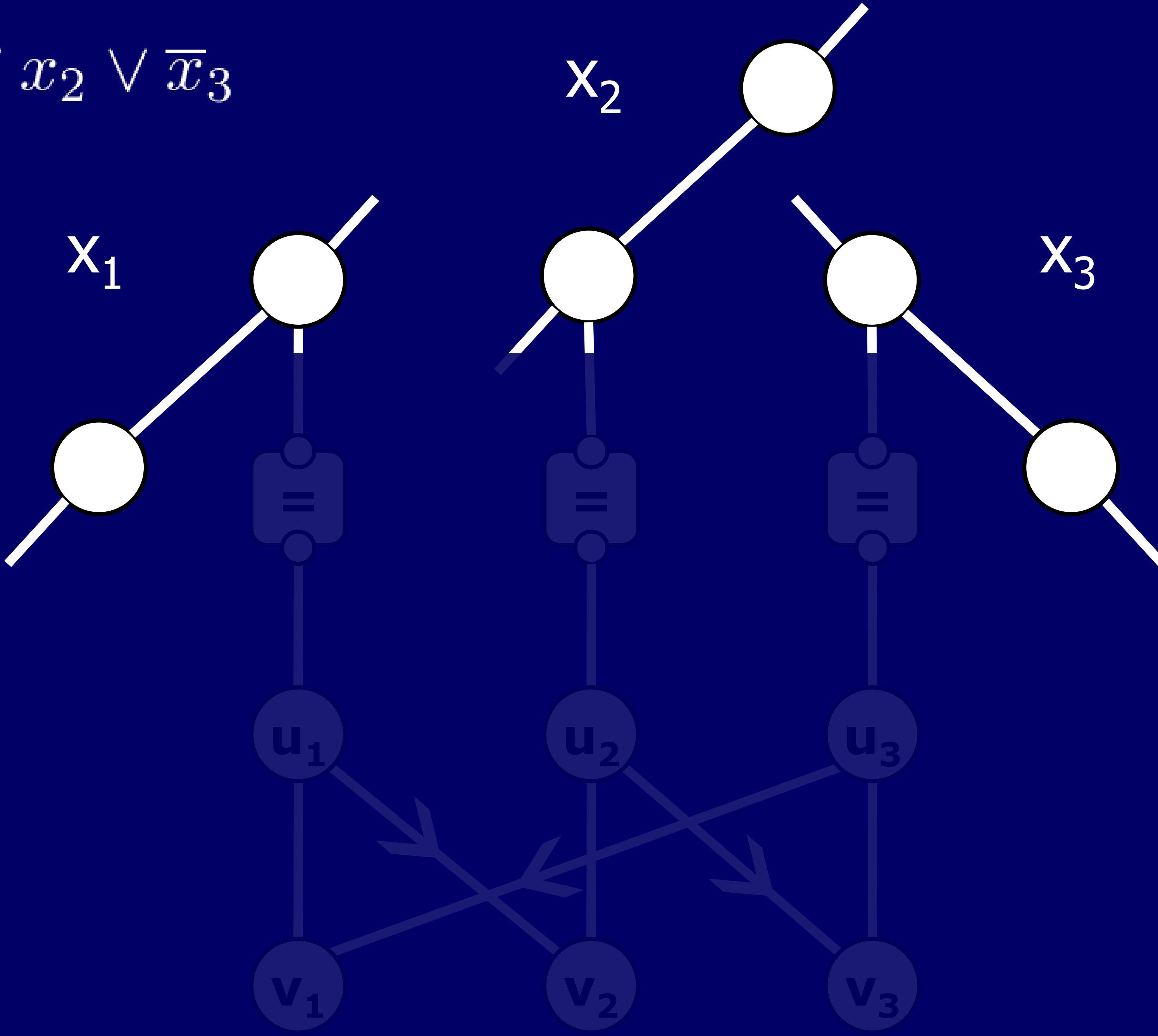
$$\mathcal{SC}(u, v, w) + \mathcal{SC}(v, w, x)$$



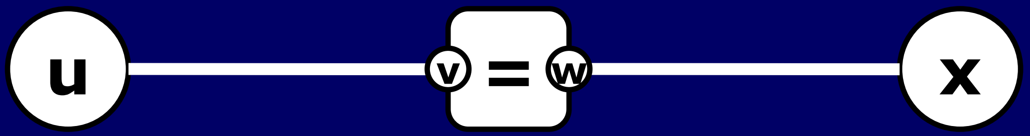
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



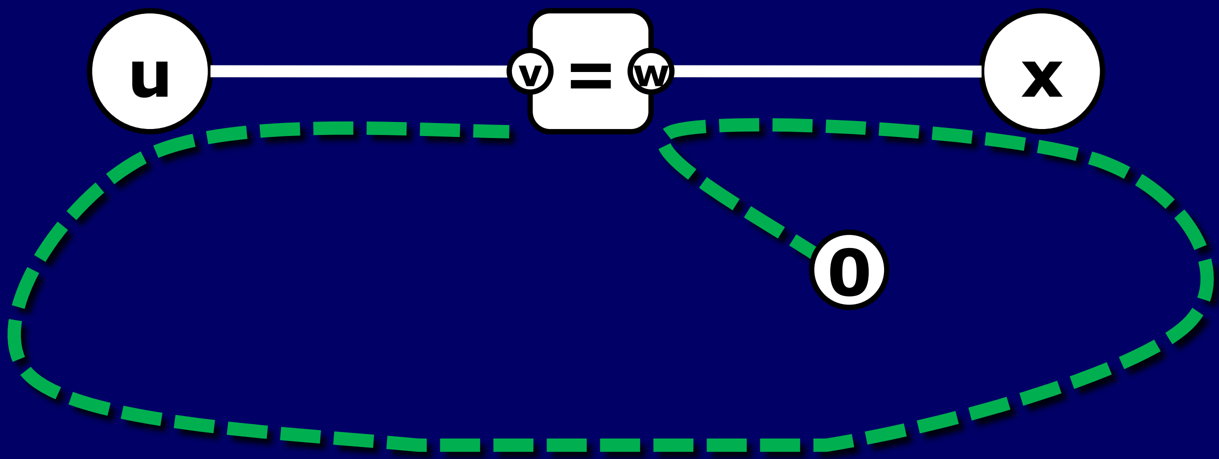
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



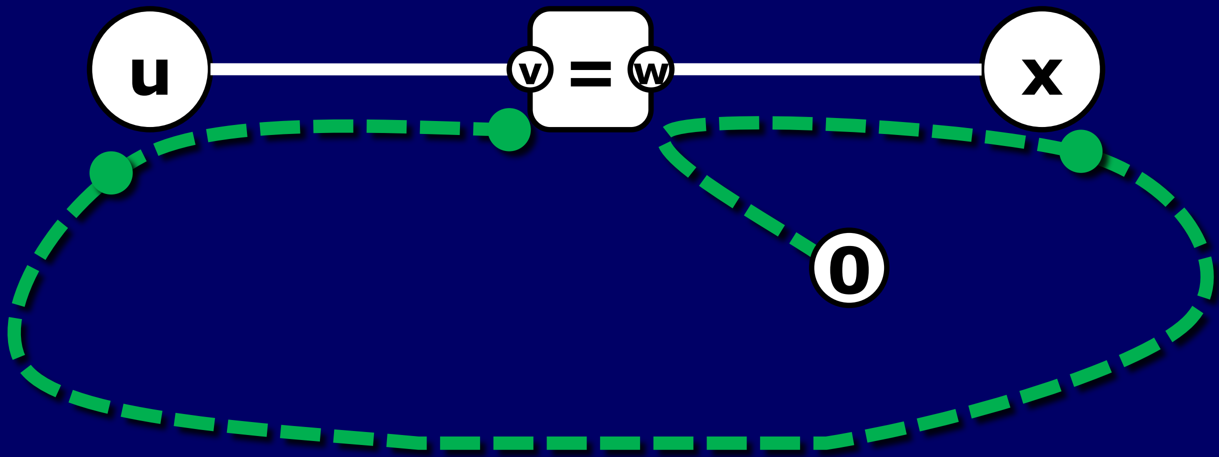
The Variable Gadget



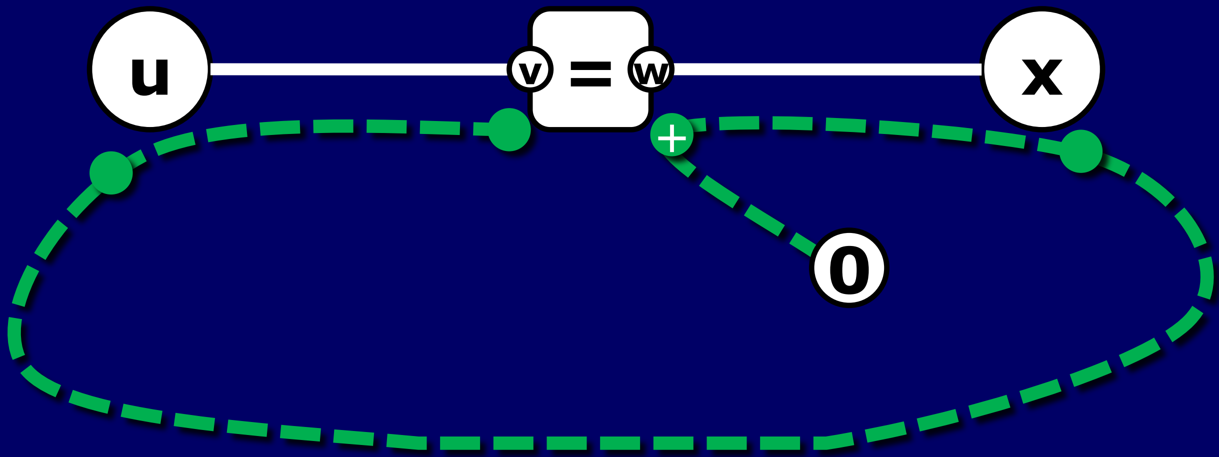
The Variable Gadget



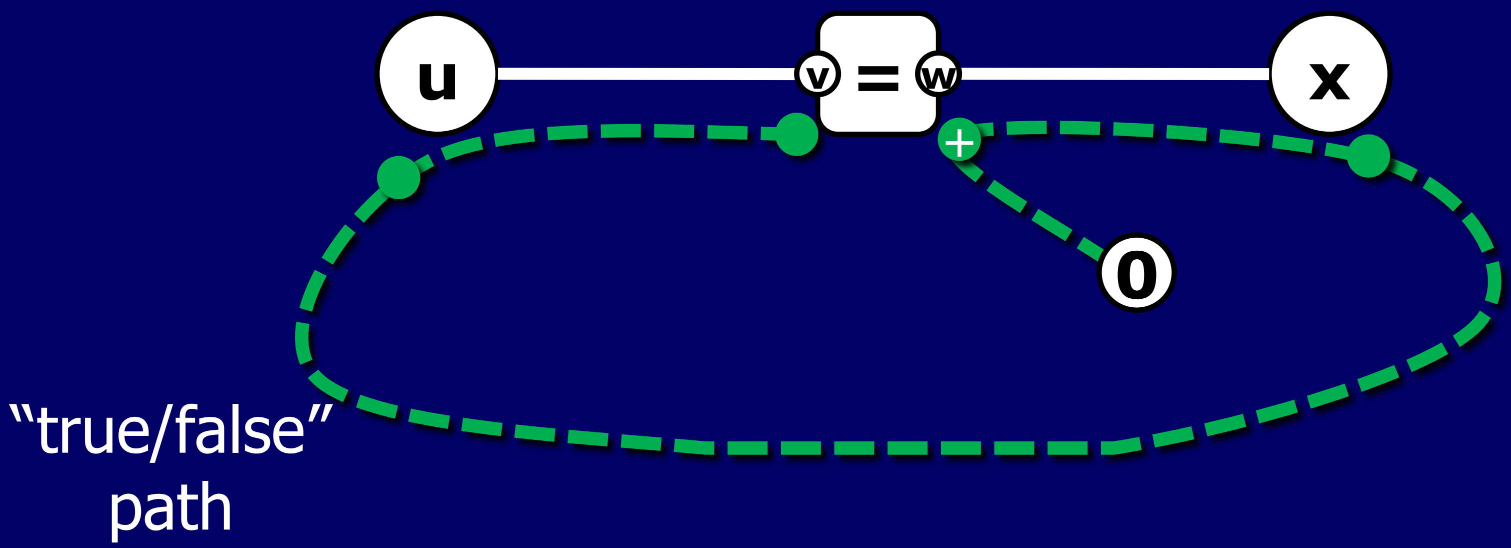
The Variable Gadget



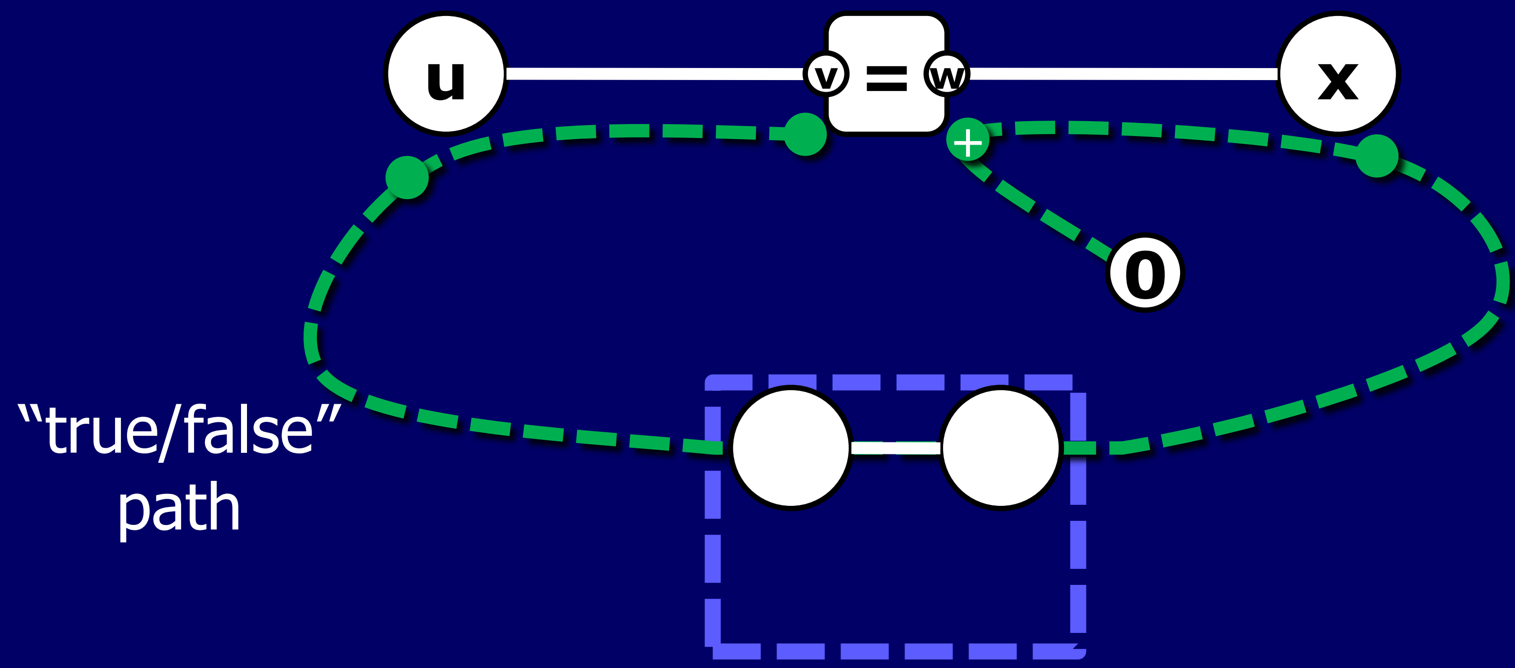
The Variable Gadget



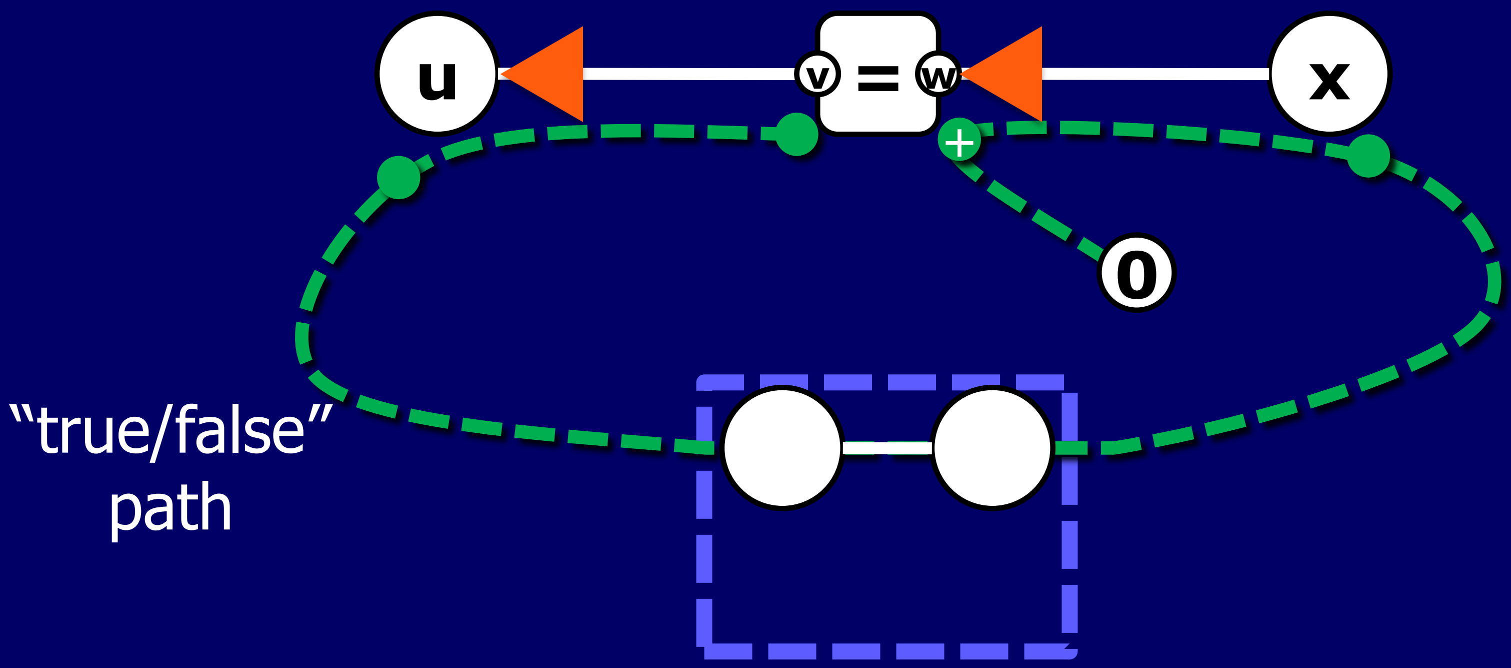
The Variable Gadget



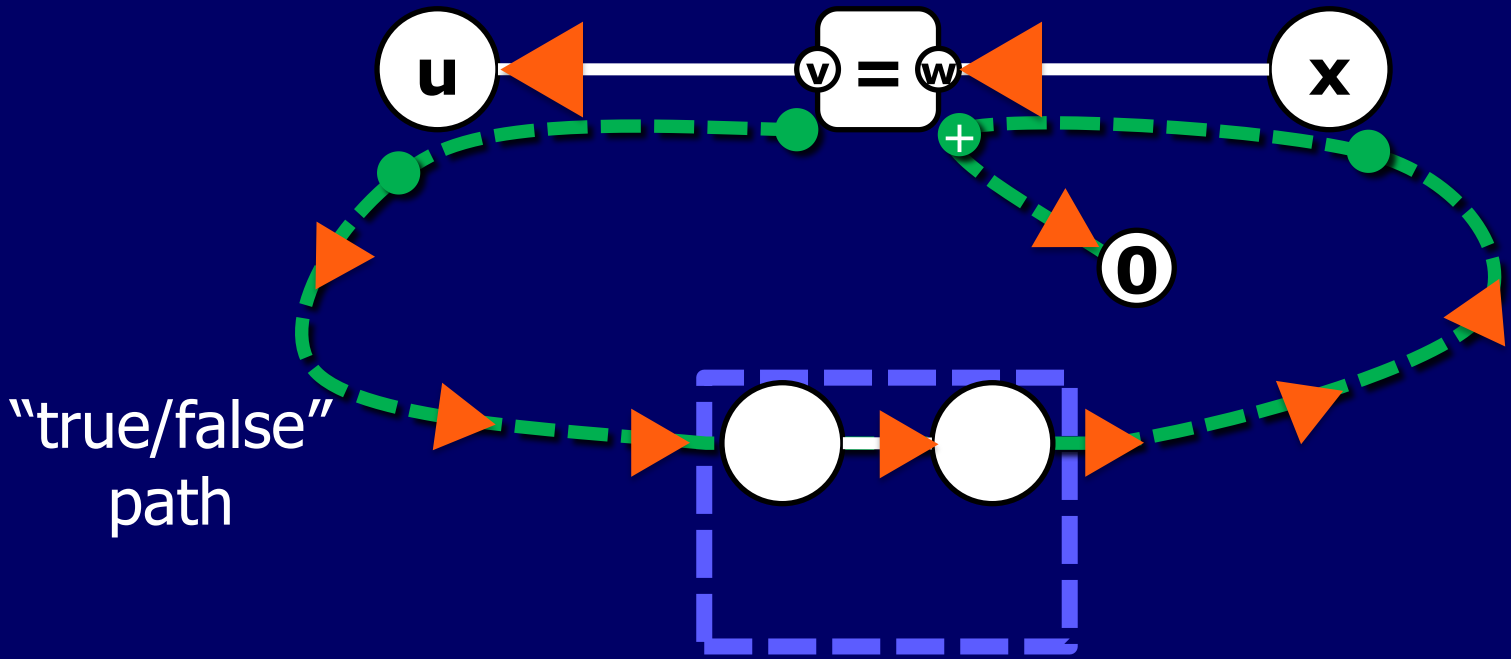
The Variable Gadget



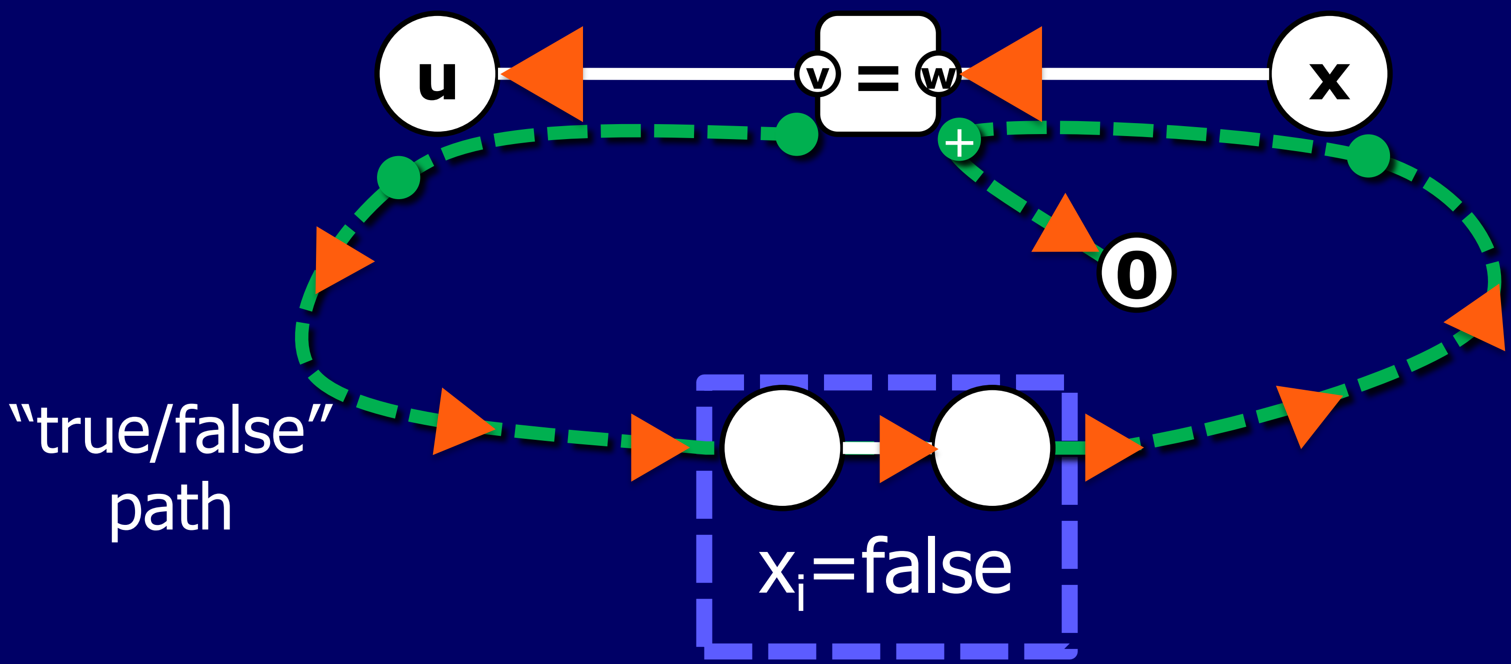
The Variable Gadget



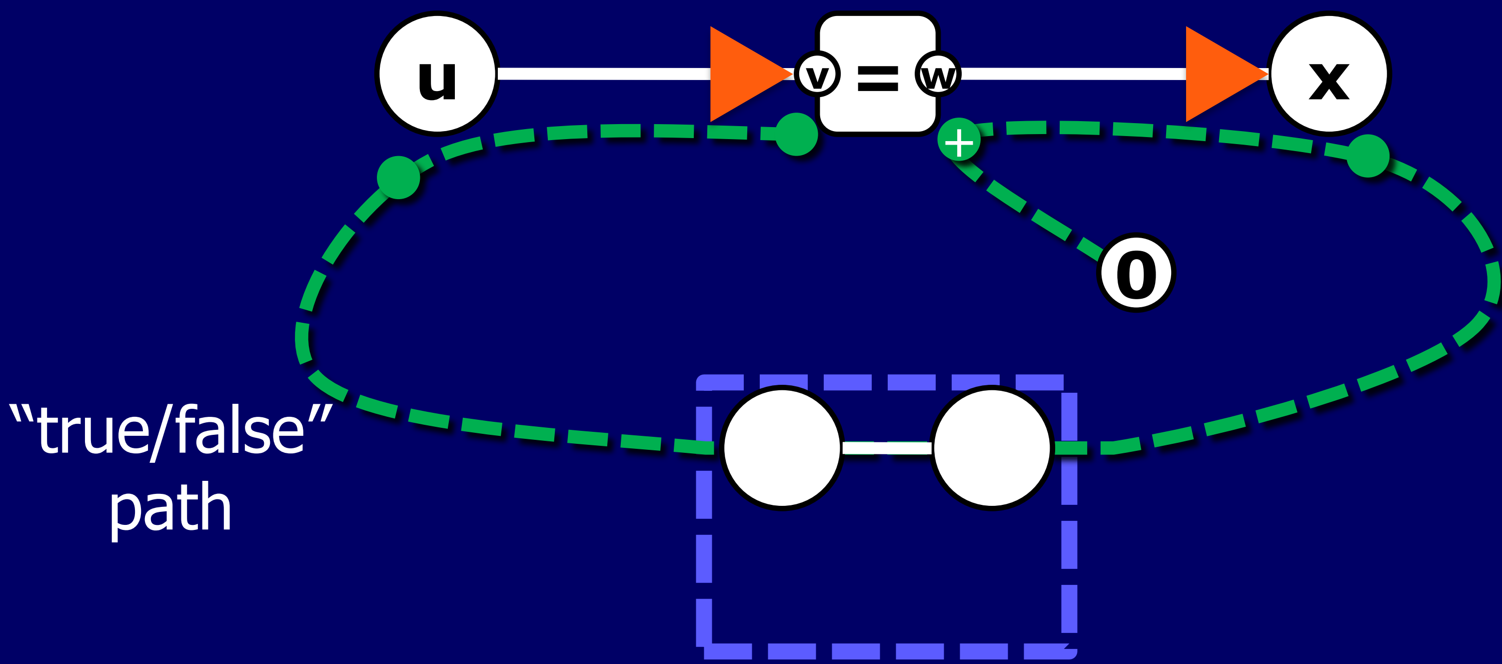
The Variable Gadget



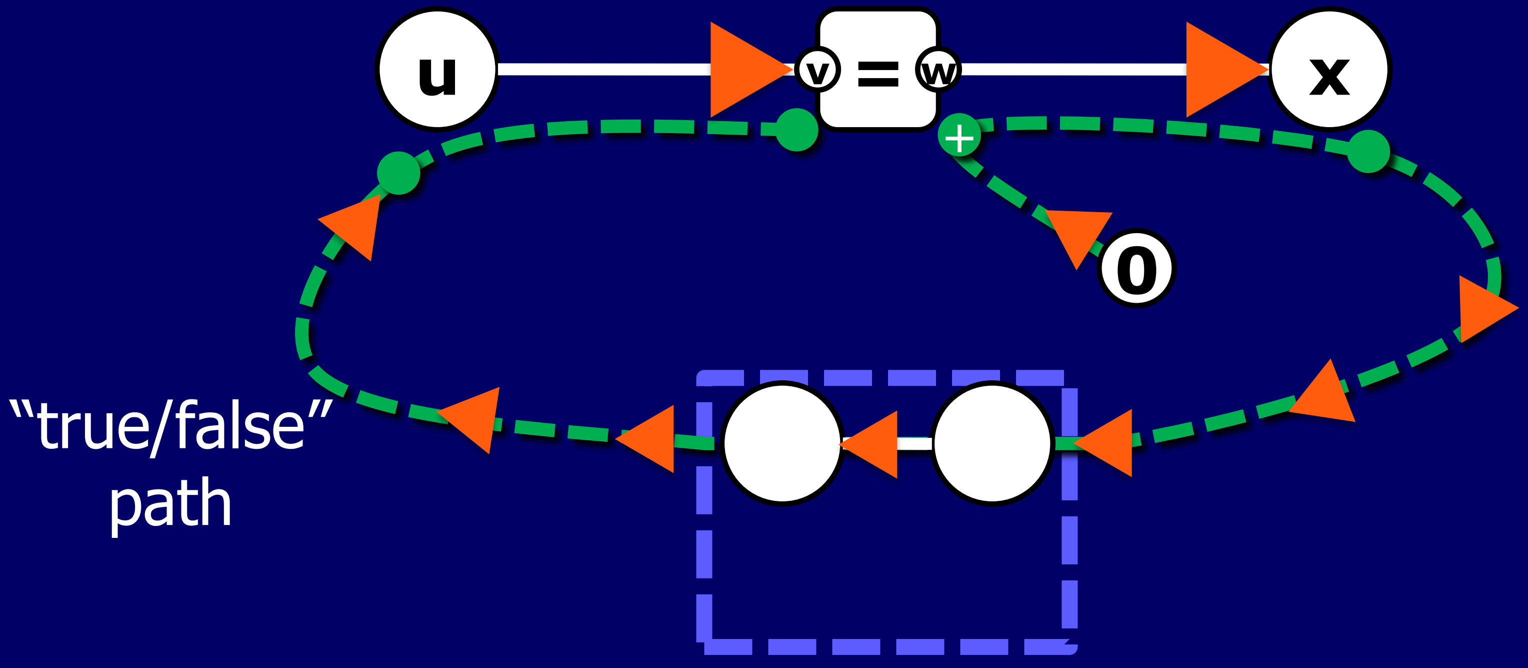
The Variable Gadget



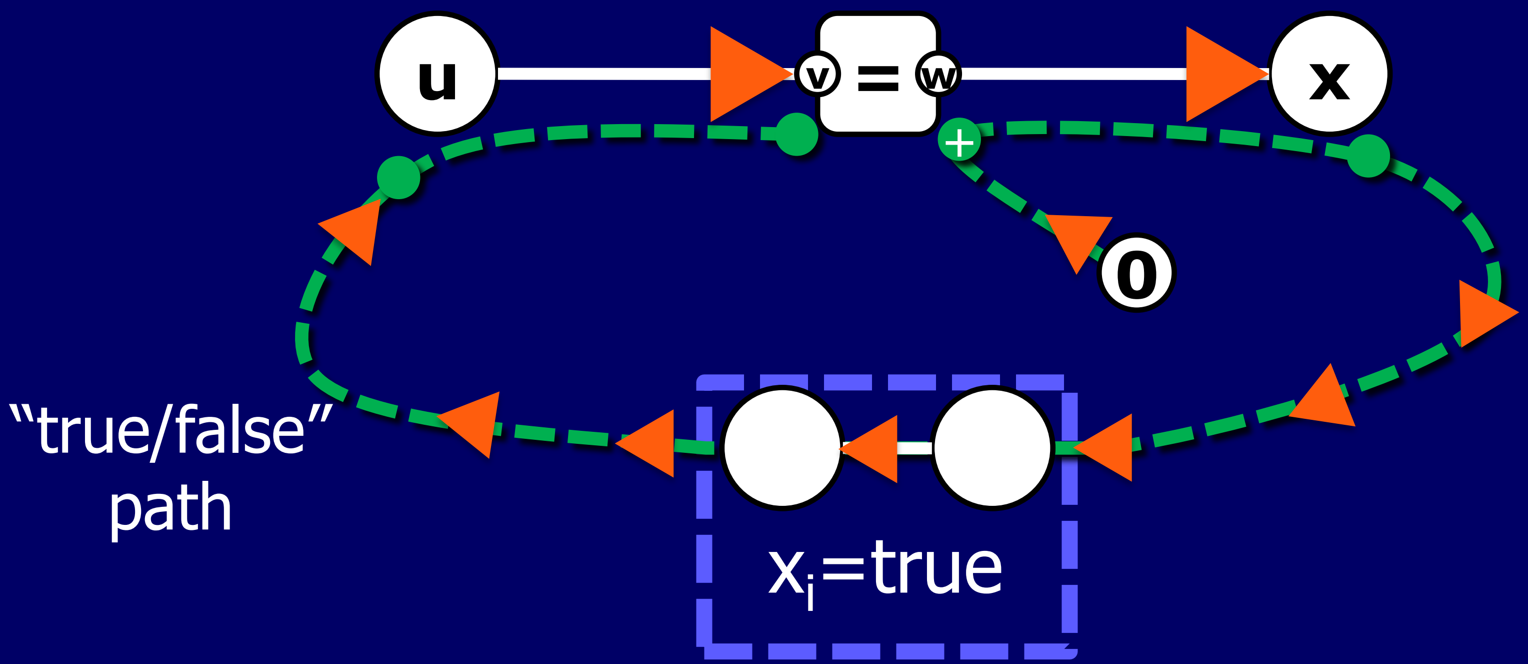
The Variable Gadget



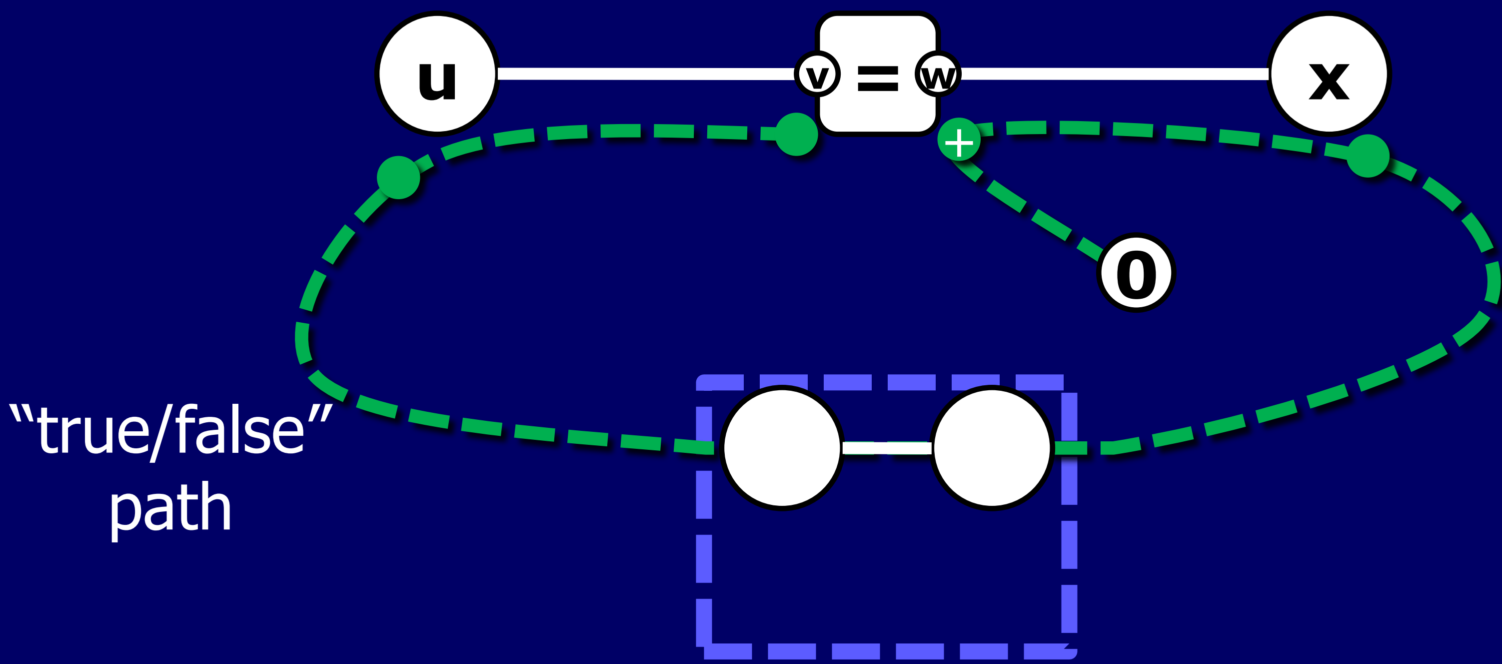
The Variable Gadget



The Variable Gadget

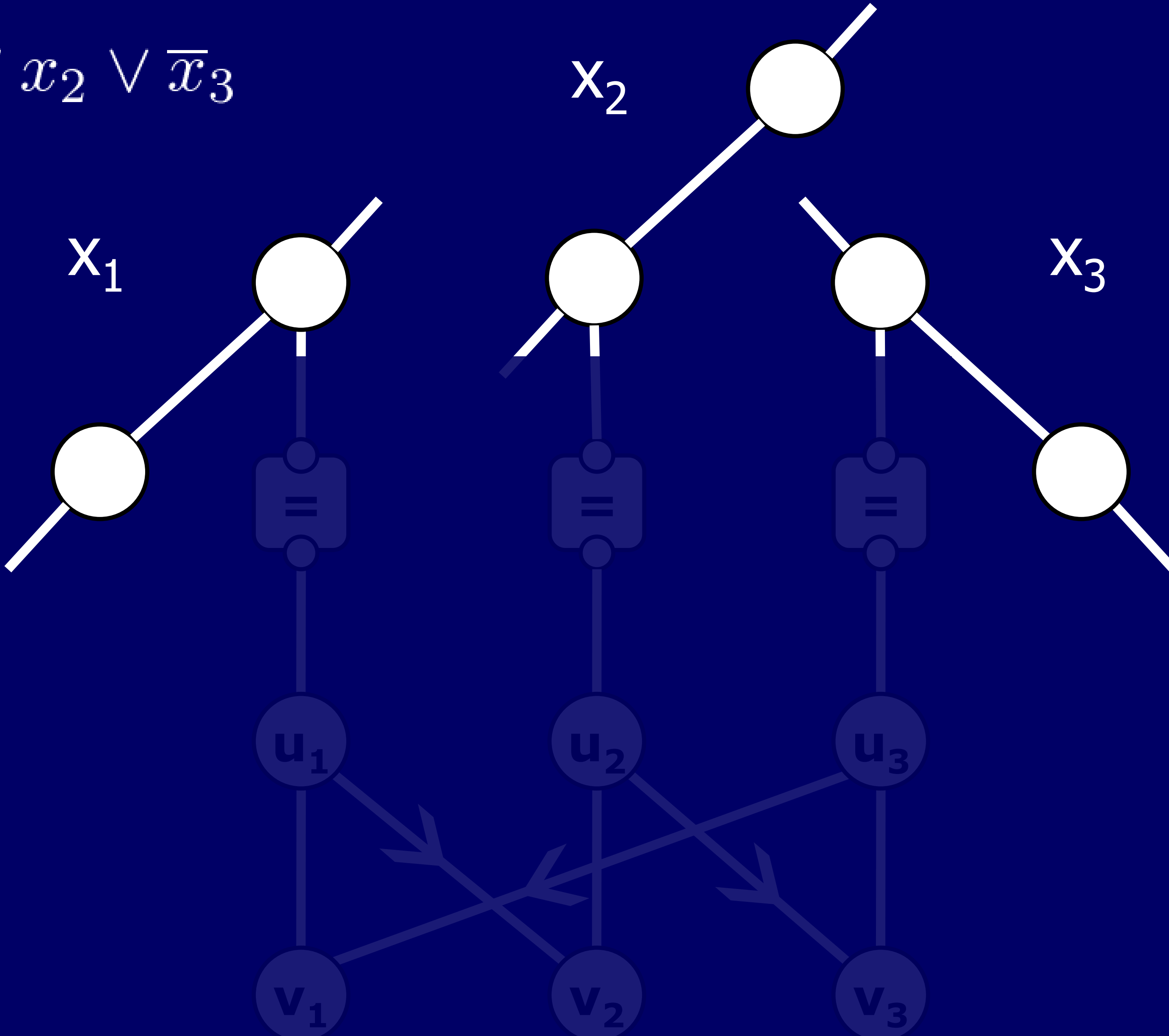


The Variable Gadget

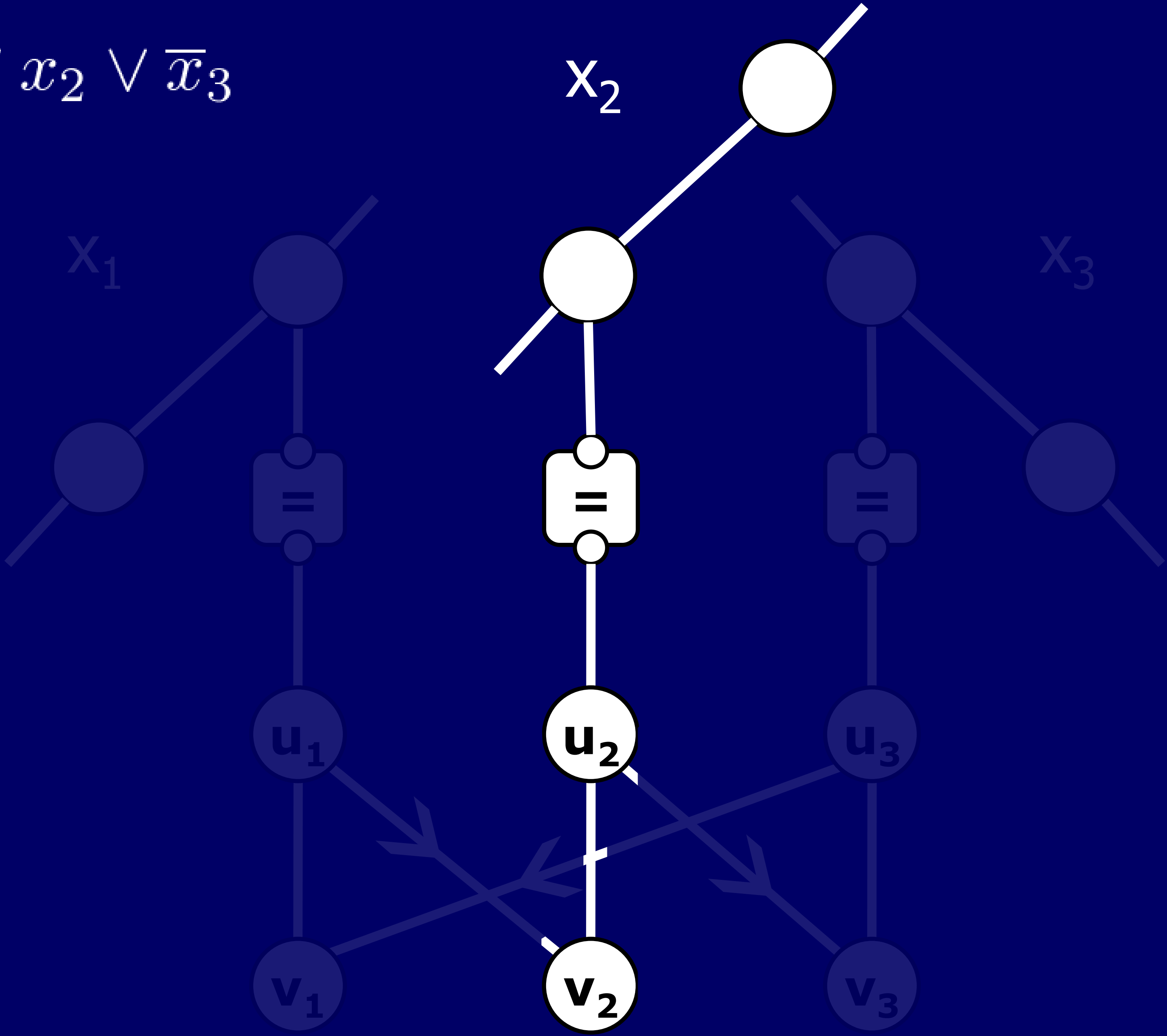


peer-to-peer prevented by valley-freeness

$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



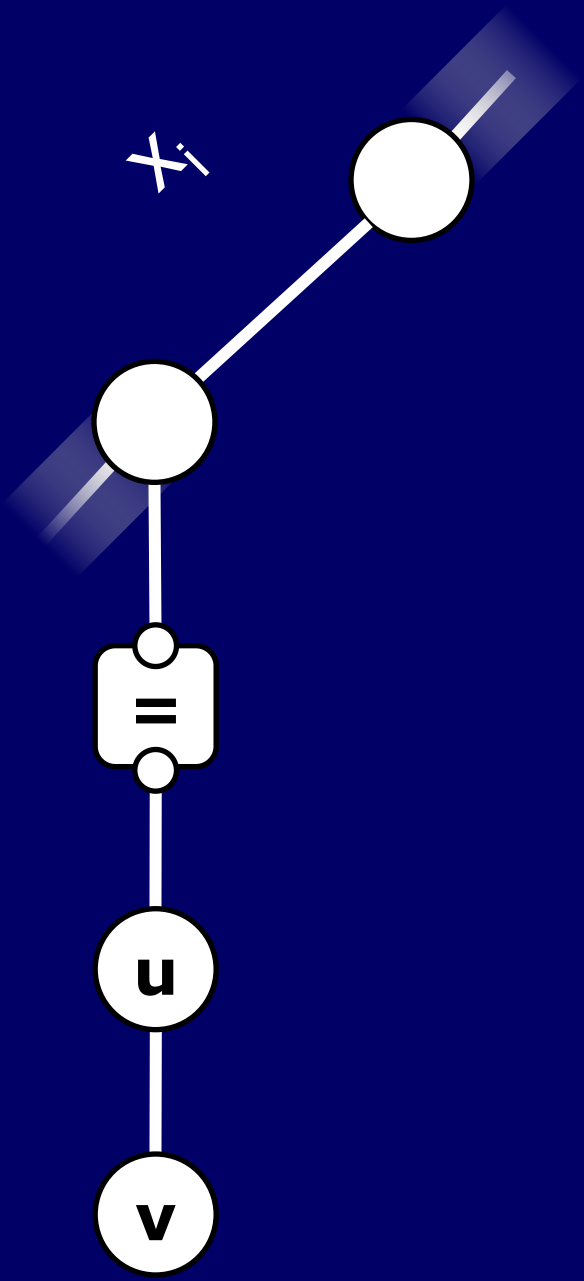
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



The Tap Configuration

$\mathcal{TC}(u, v)$

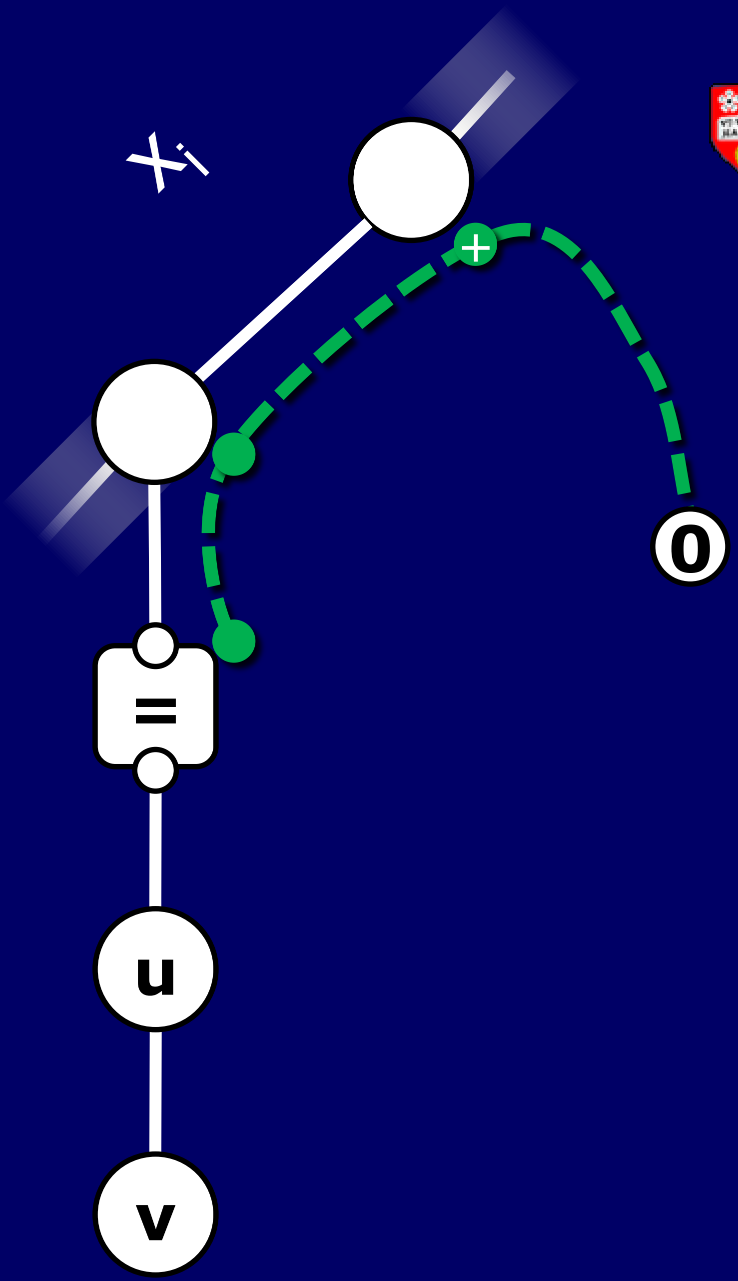
x_i



The Tap Configuration

$\mathcal{TC}(u, v)$

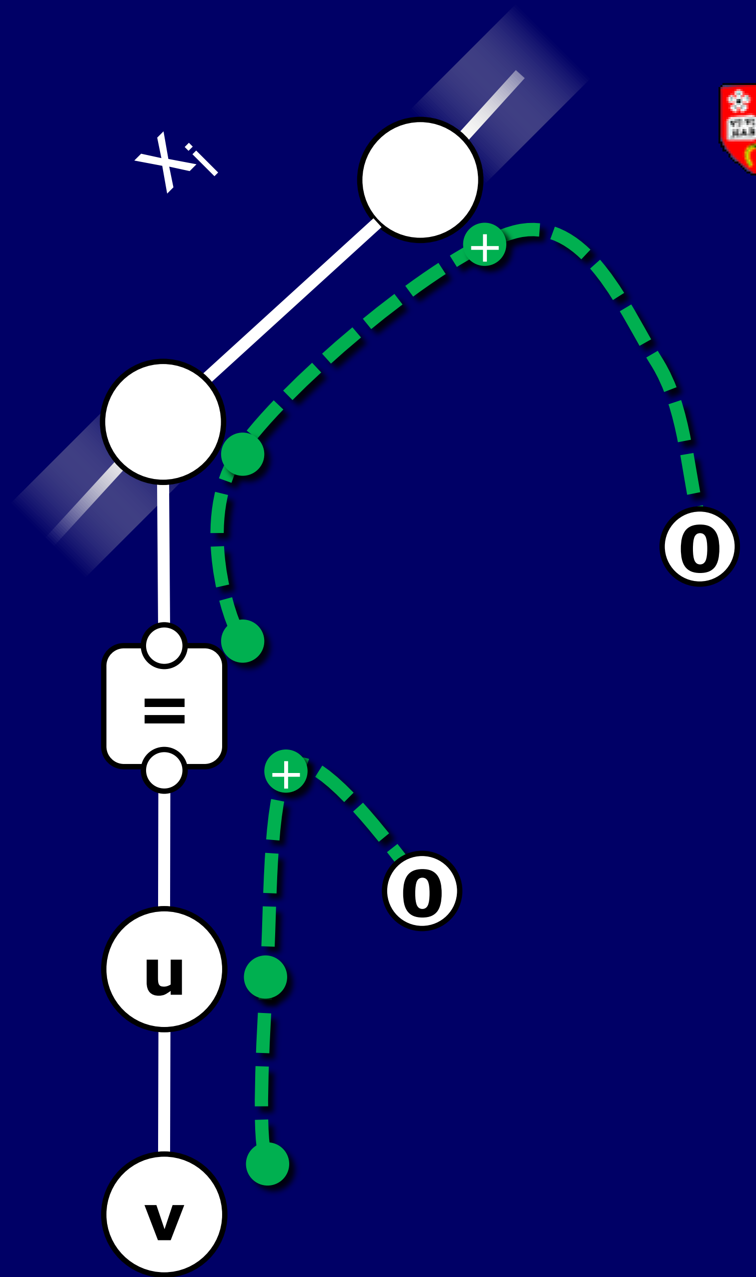
x_i



The Tap Configuration

$\mathcal{TC}(u, v)$

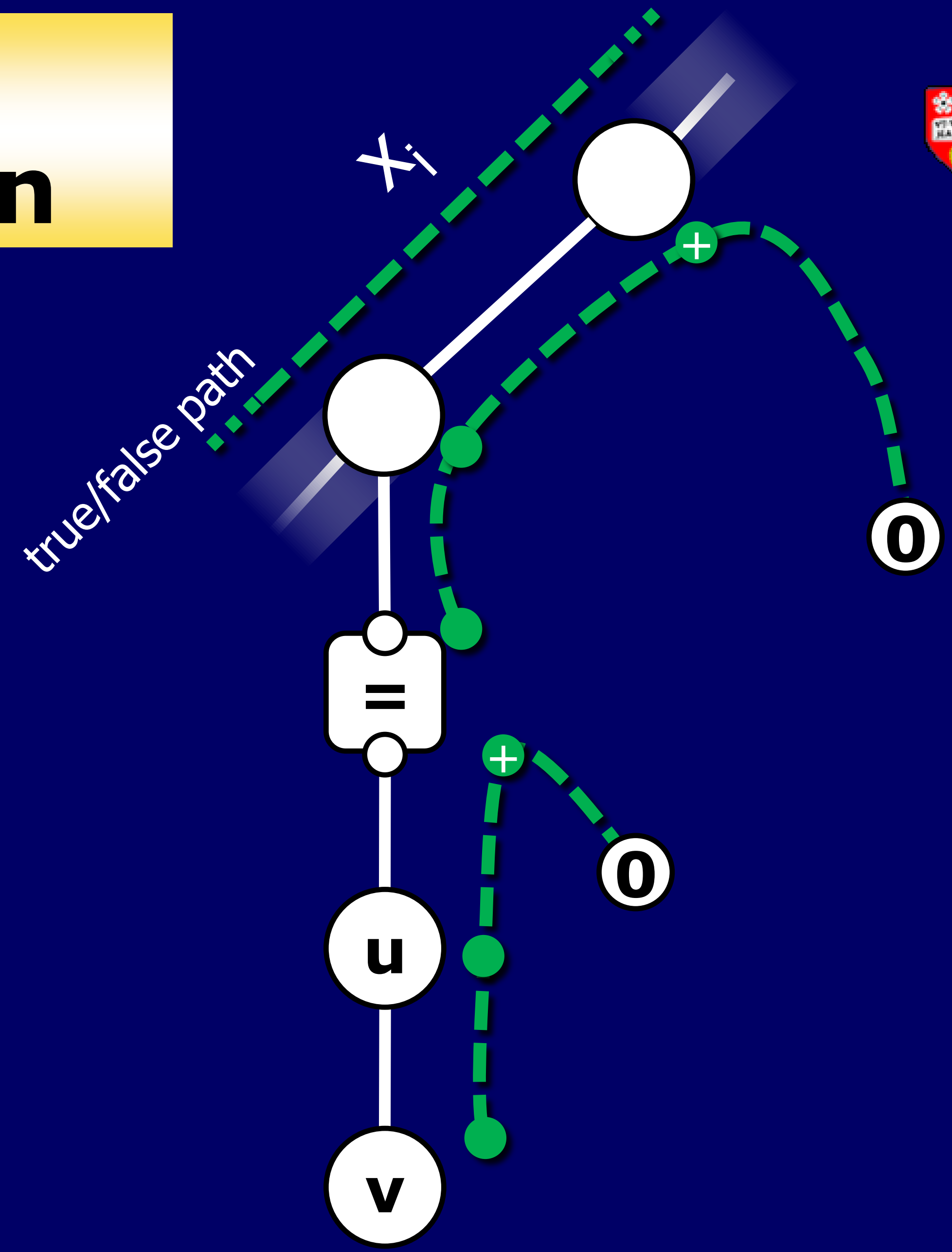
x_i



The Tap Configuration

$\mathcal{TC}(u, v)$

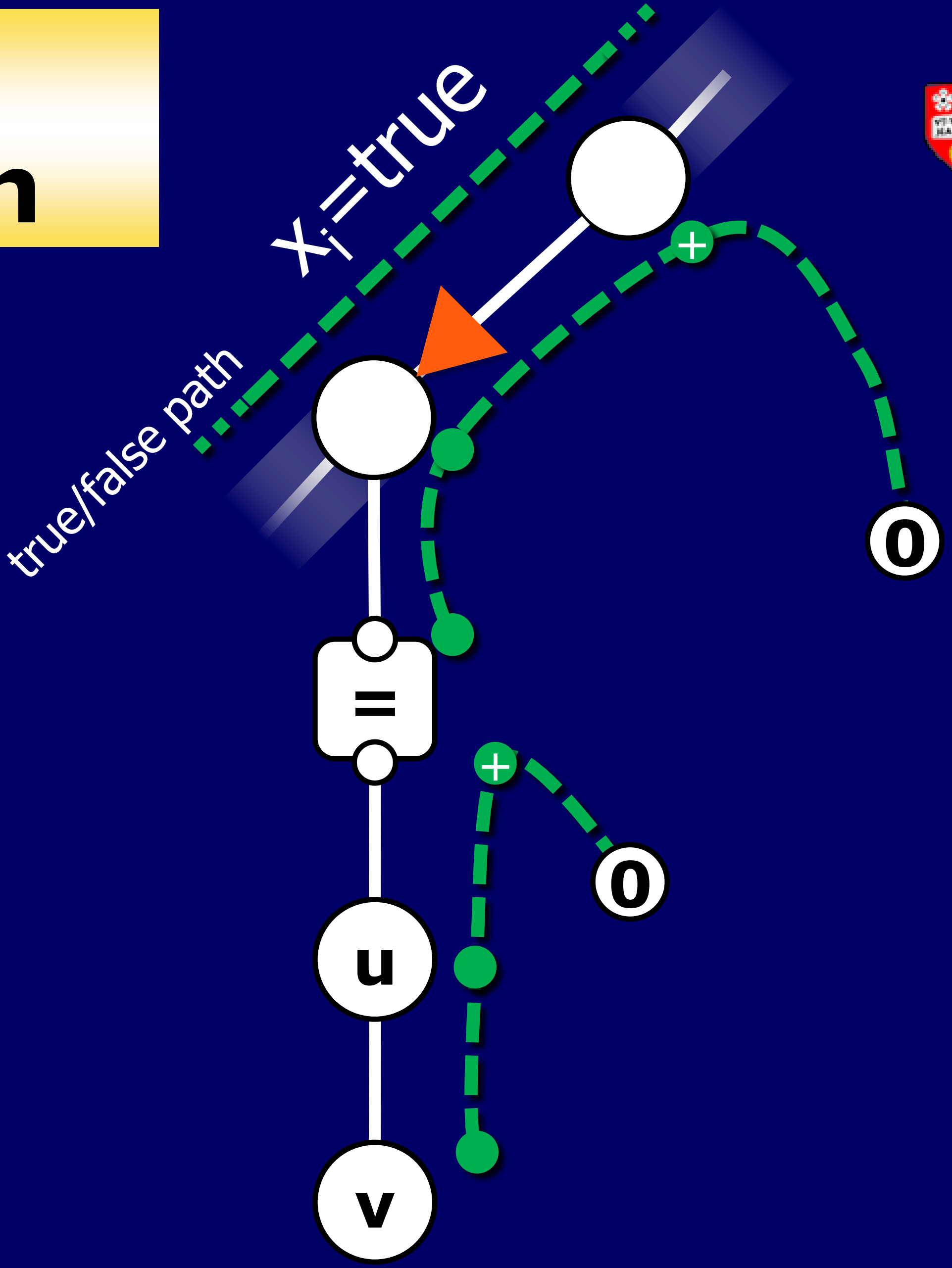
x_i



The Tap Configuration

$\mathcal{TC}(u, v)$

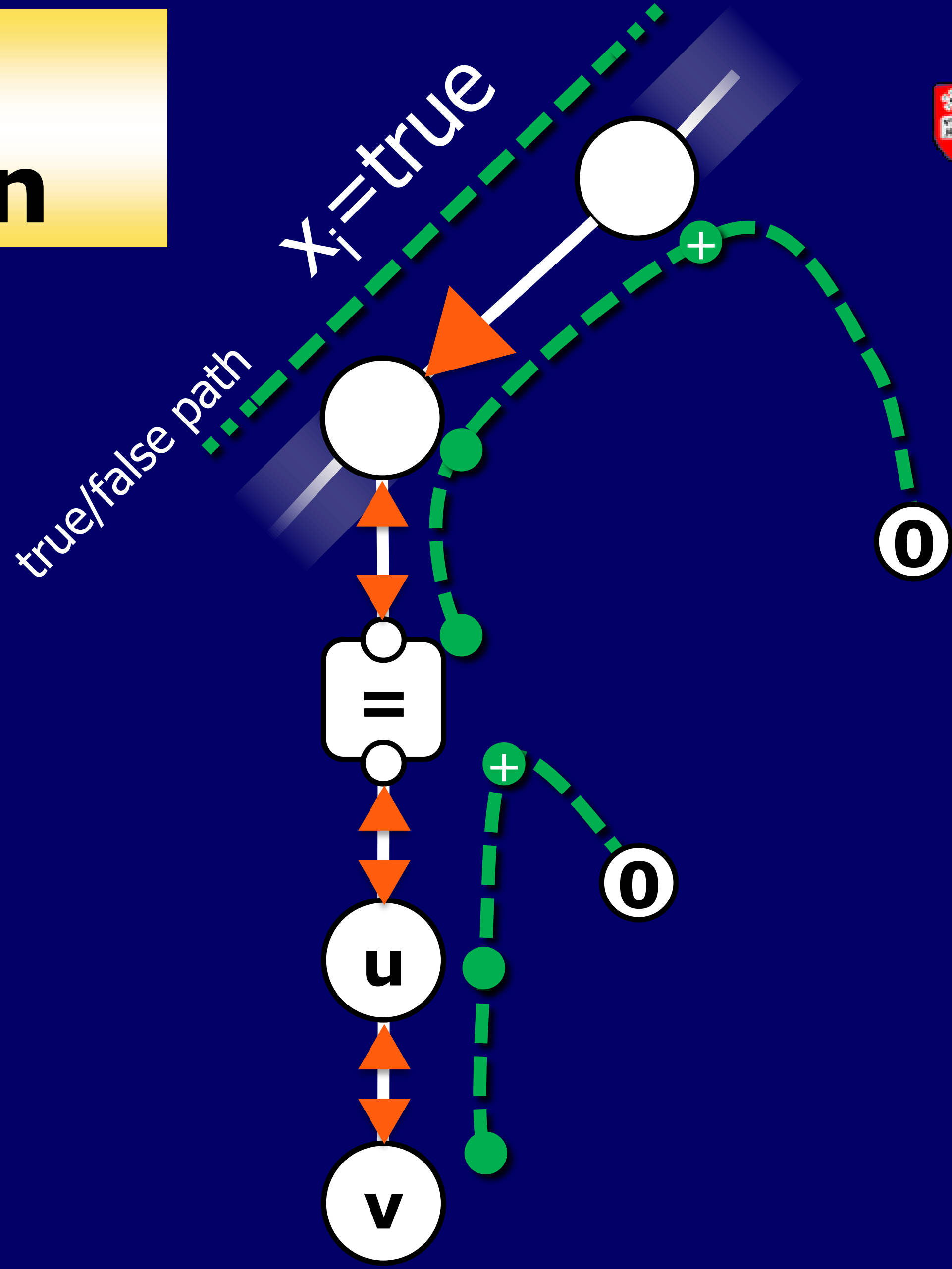
x_i



The Tap Configuration

$\mathcal{TC}(u, v)$

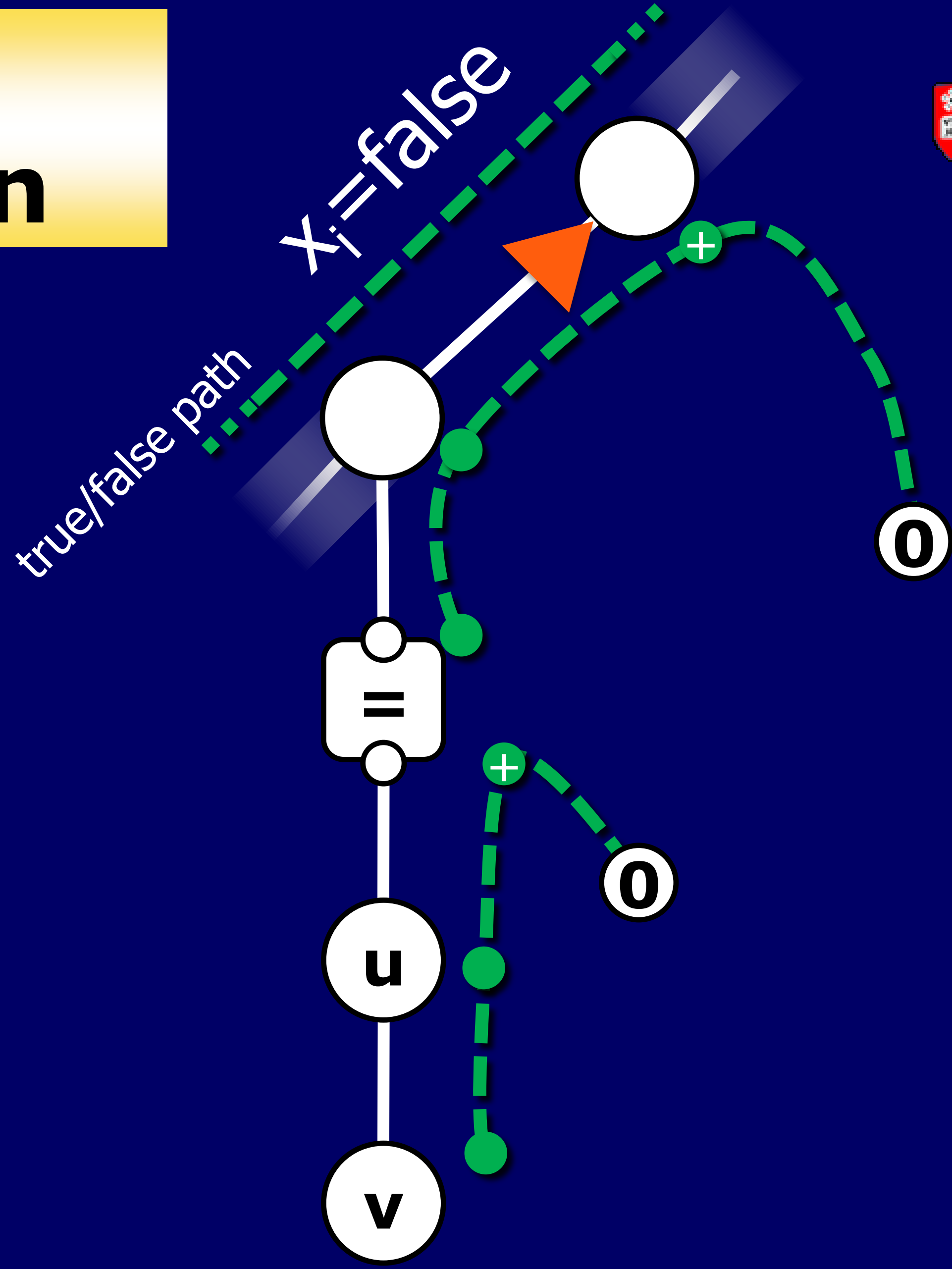
x_i



The Tap Configuration

$\mathcal{TC}(u, v)$

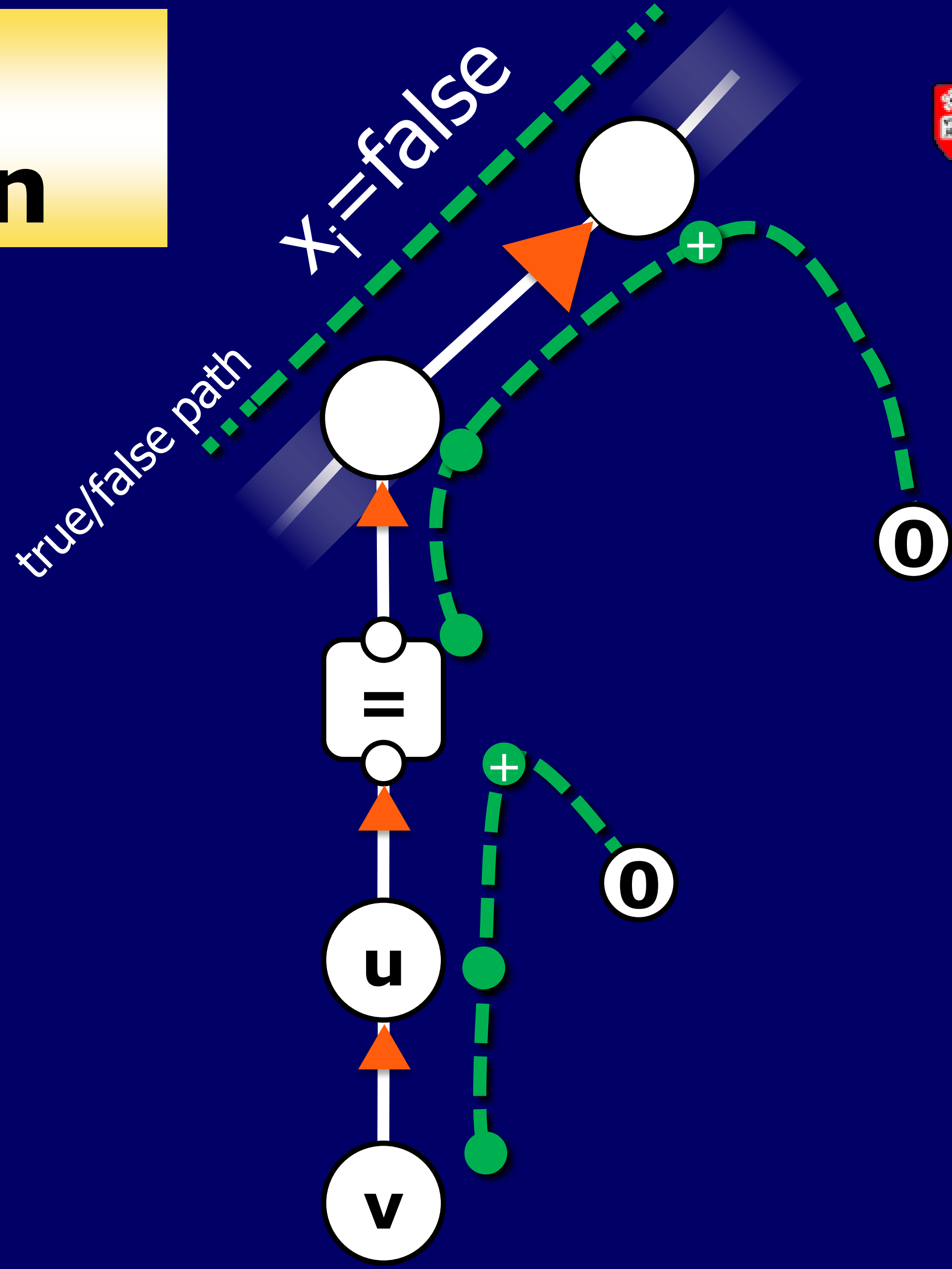
x_i



The Tap Configuration

$\mathcal{TC}(u, v)$

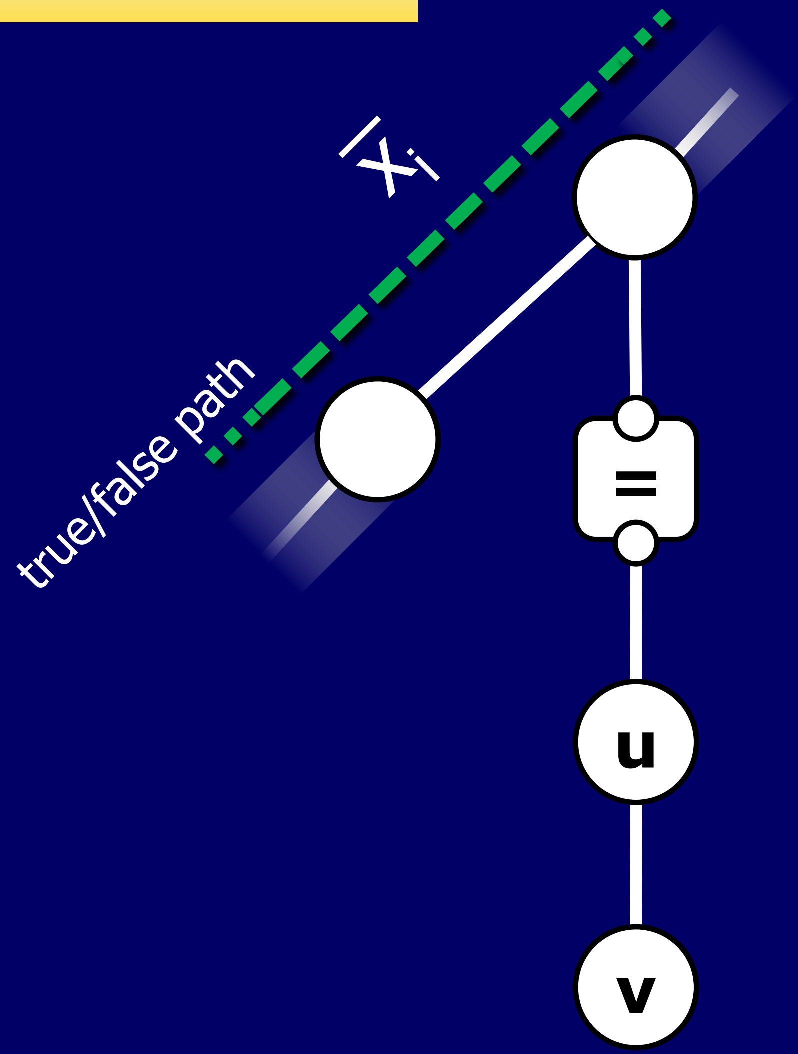
x_i



The Tap Configuration

$\mathcal{TC}(u, v)$

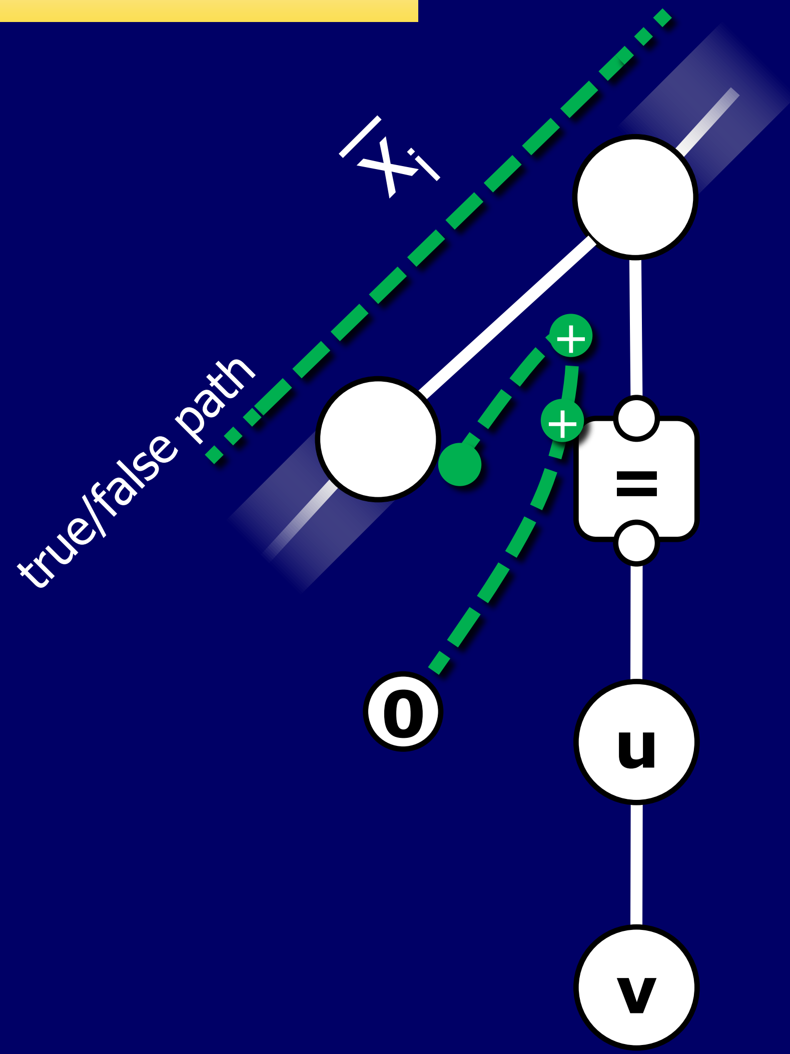
$\overline{x}_i$



The Tap Configuration

$\mathcal{TC}(u, v)$

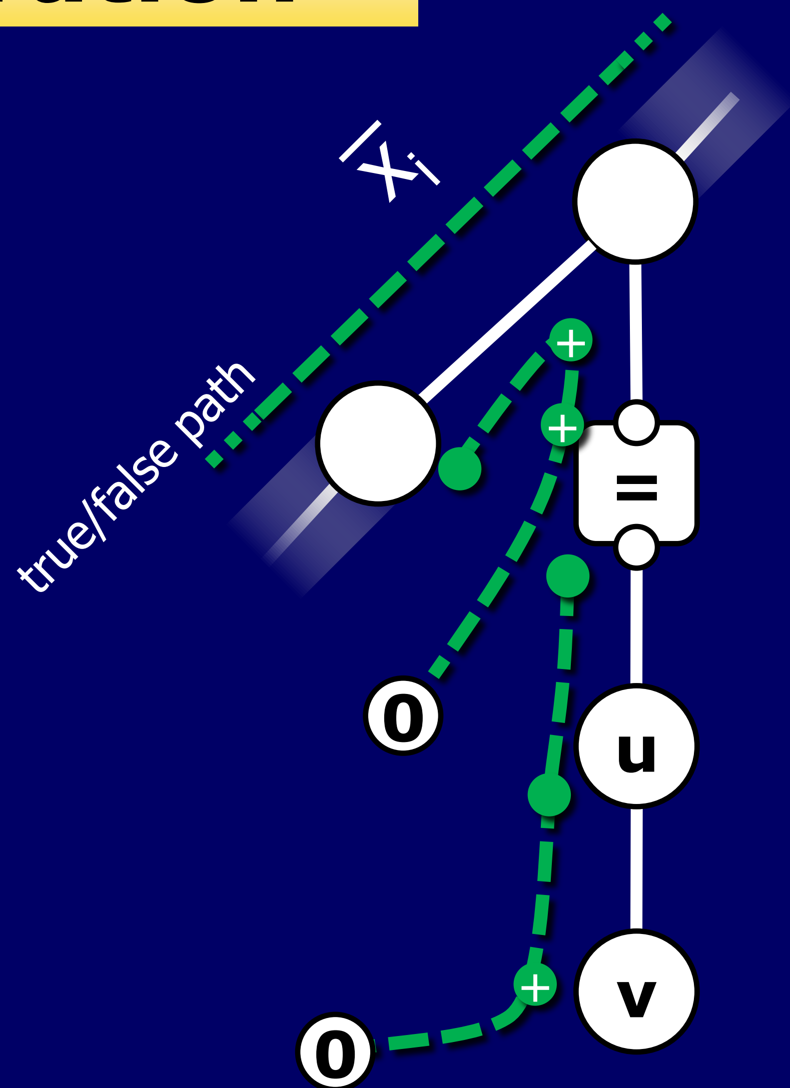
$\overline{x}_i$



The Tap Configuration

$\mathcal{TC}(u, v)$

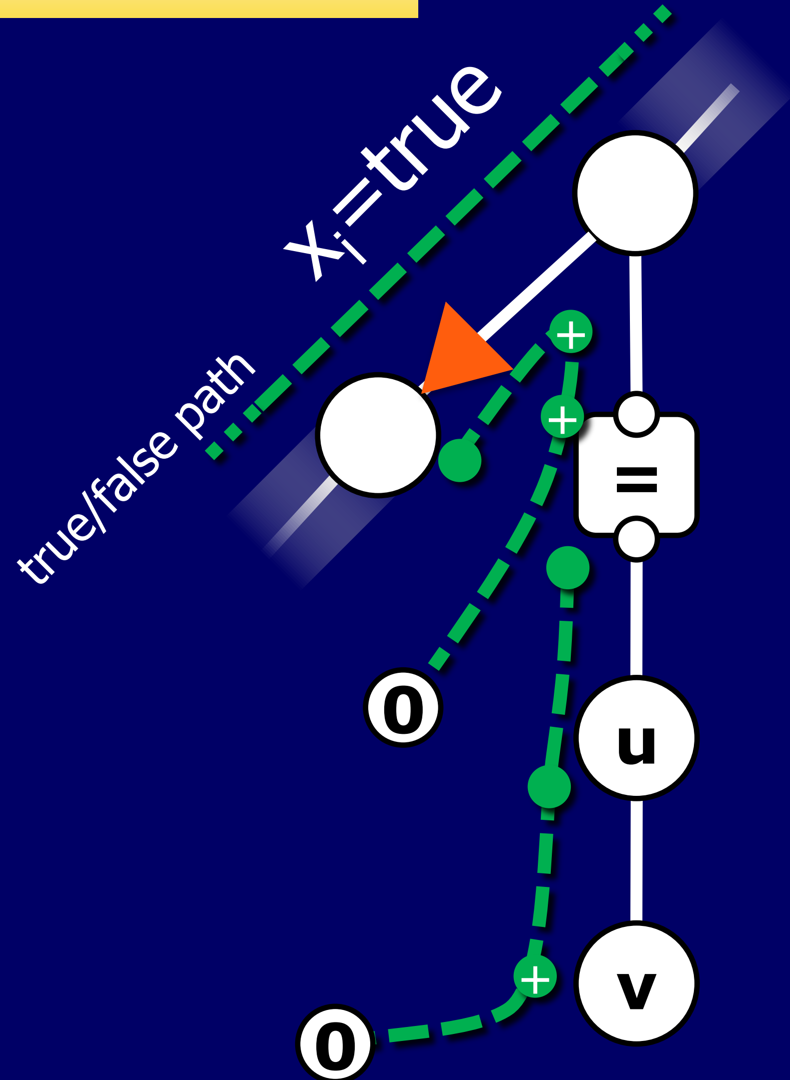
$\overline{x}_i$



The Tap Configuration

$\mathcal{TC}(u, v)$

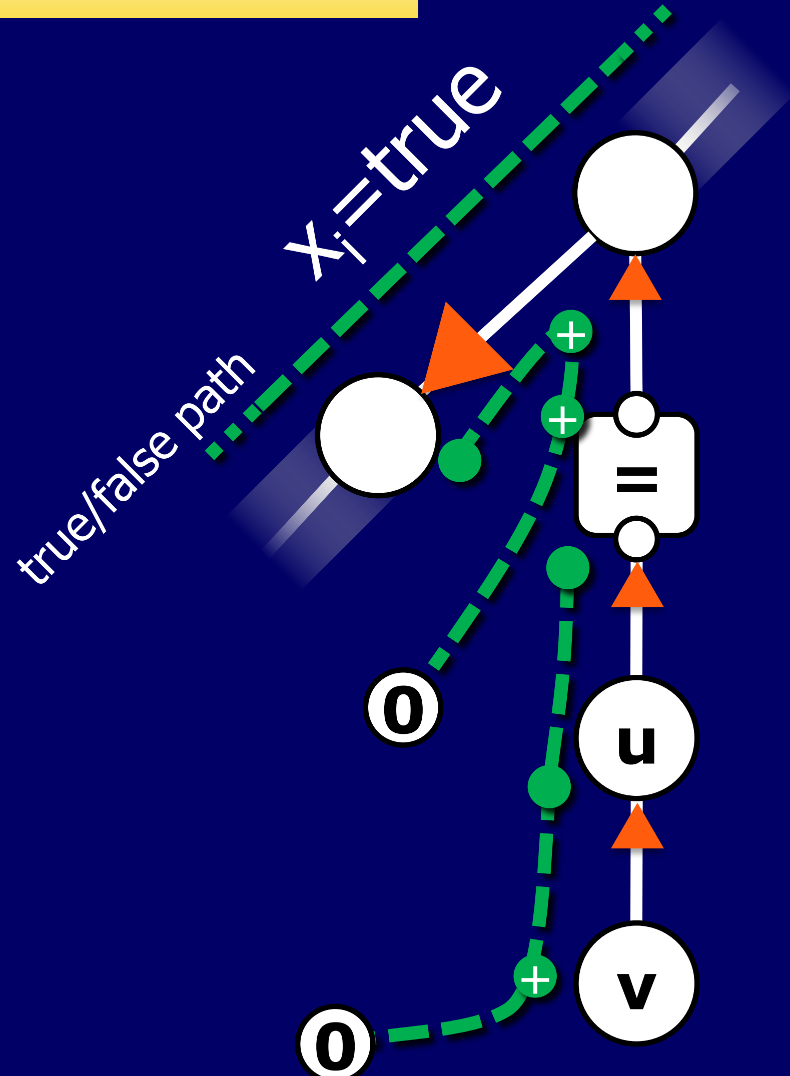
$\overline{x}_i$



The Tap Configuration

$\mathcal{TC}(u, v)$

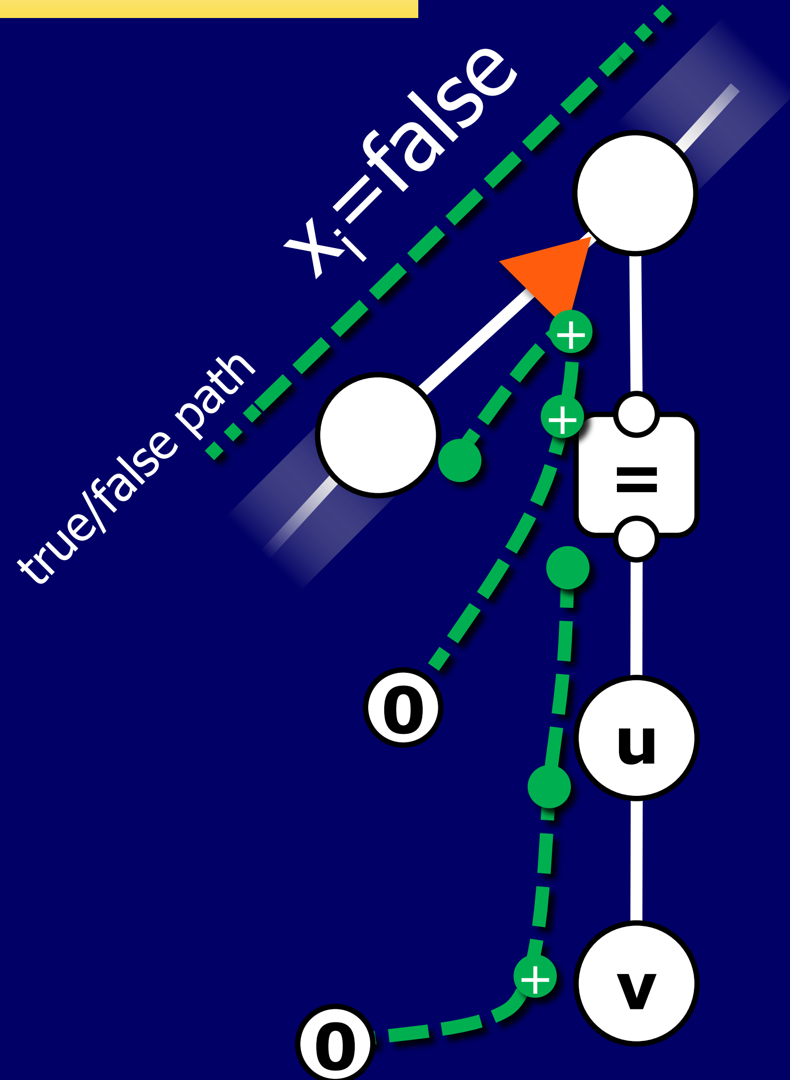
$\overline{x}_i$



The Tap Configuration

$\mathcal{TC}(u, v)$

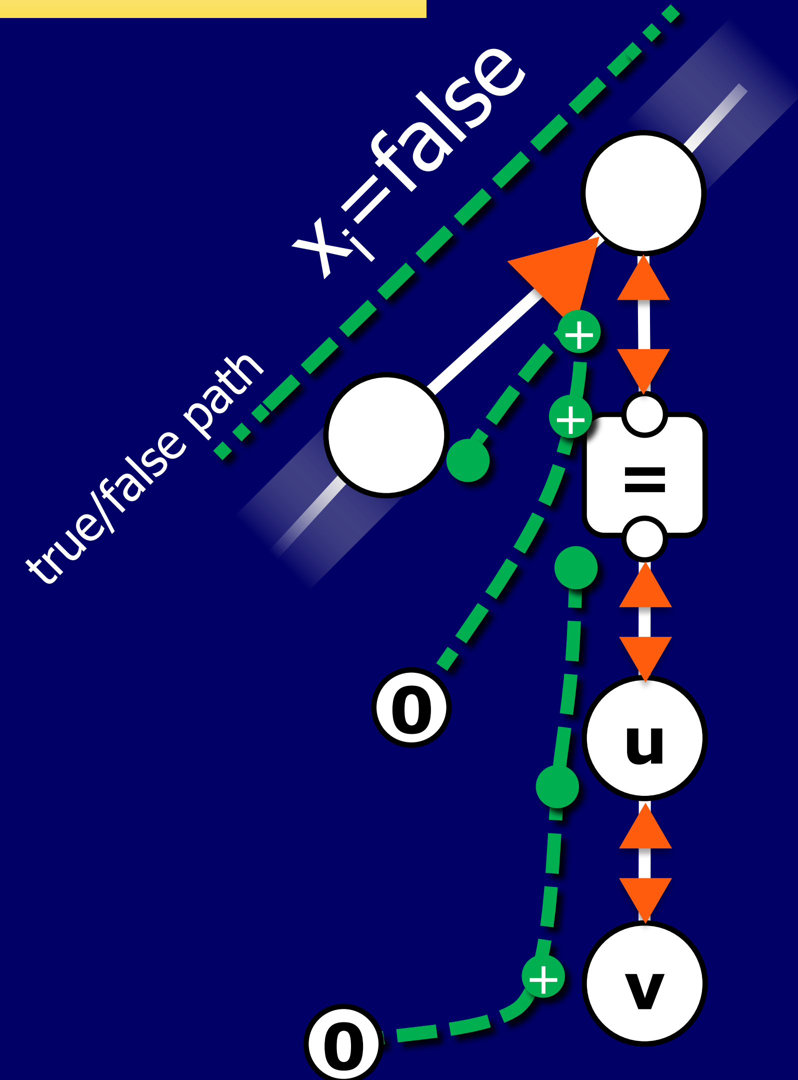
$\overline{x}_i$

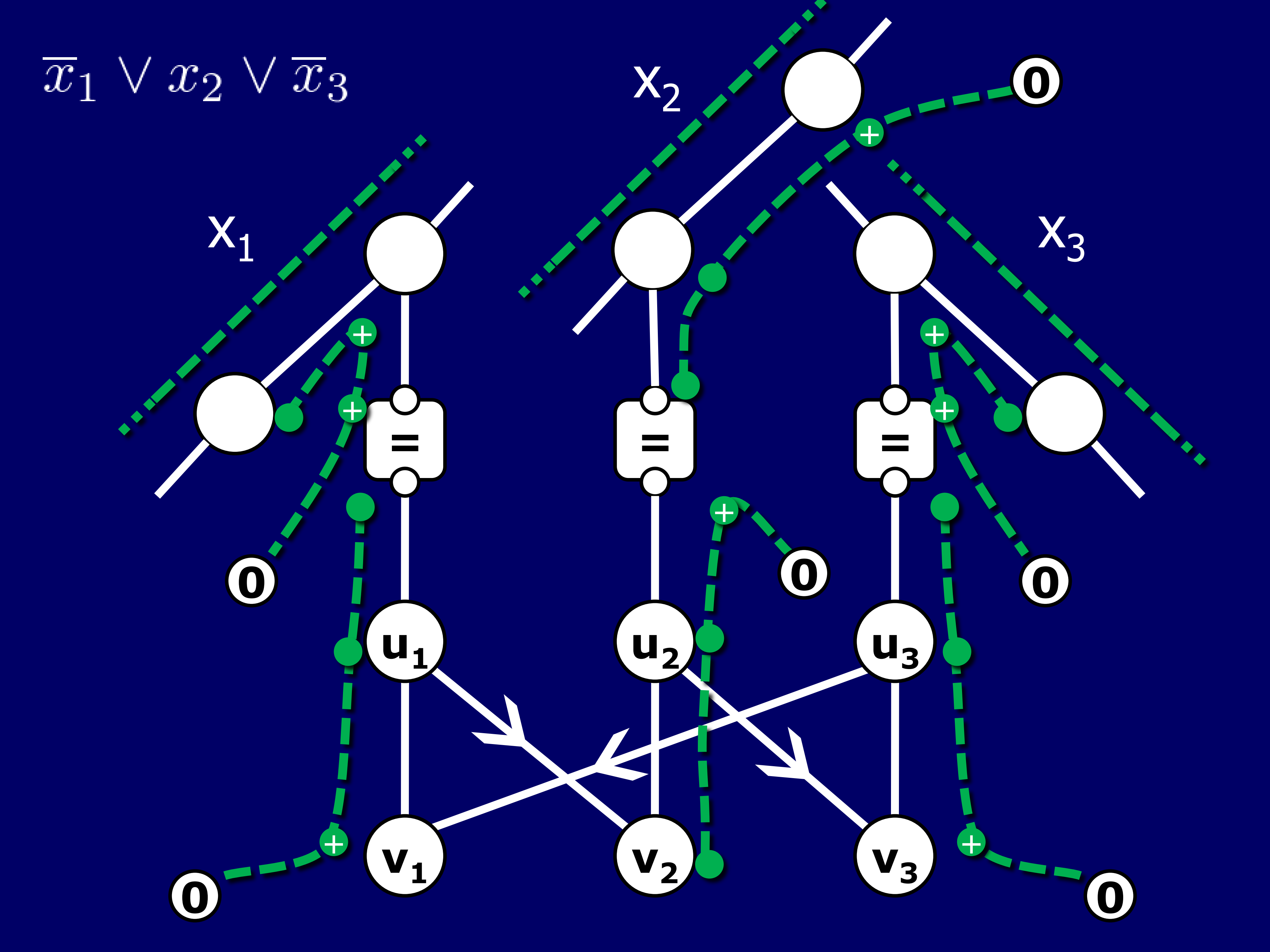


The Tap Configuration

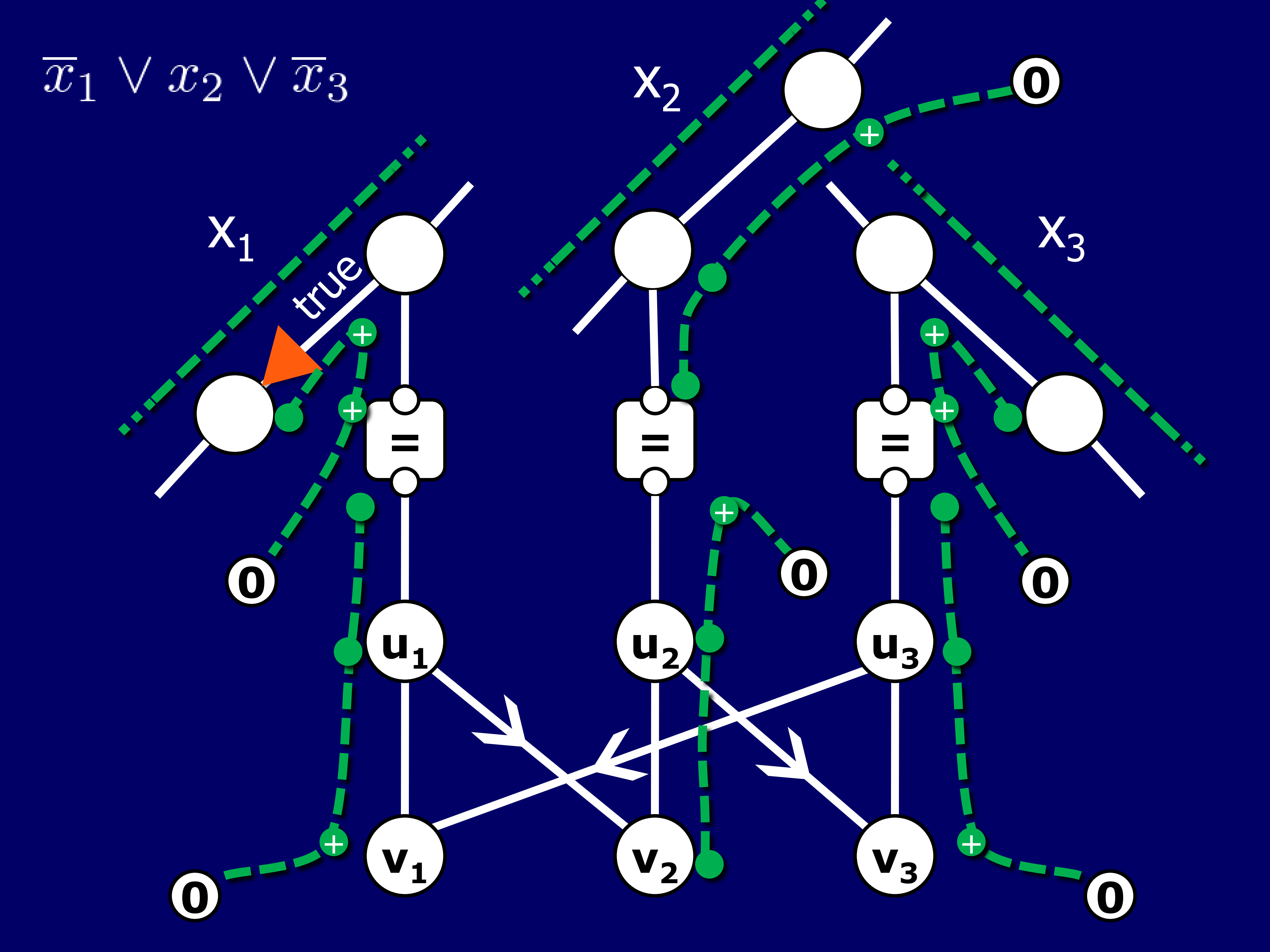
$\mathcal{TC}(u, v)$

$\overline{x}_i$



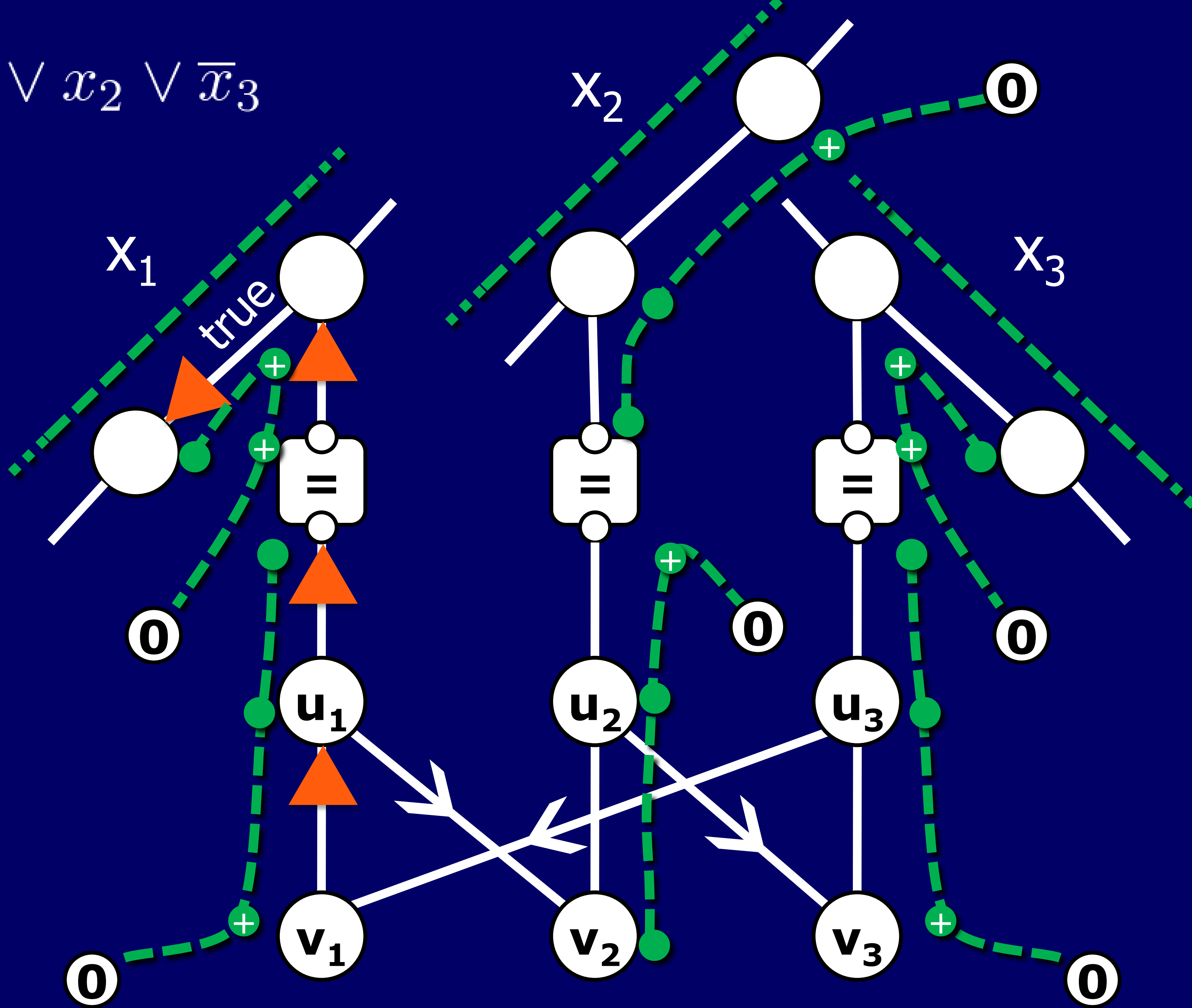


$\overline{x}_1 \vee x_2 \vee \overline{x}_3$
x_1
x_2
x_3
0
0
0
0
0
0
=
=
=
u_1
u_2
u_3
v_1
v_2
v_3

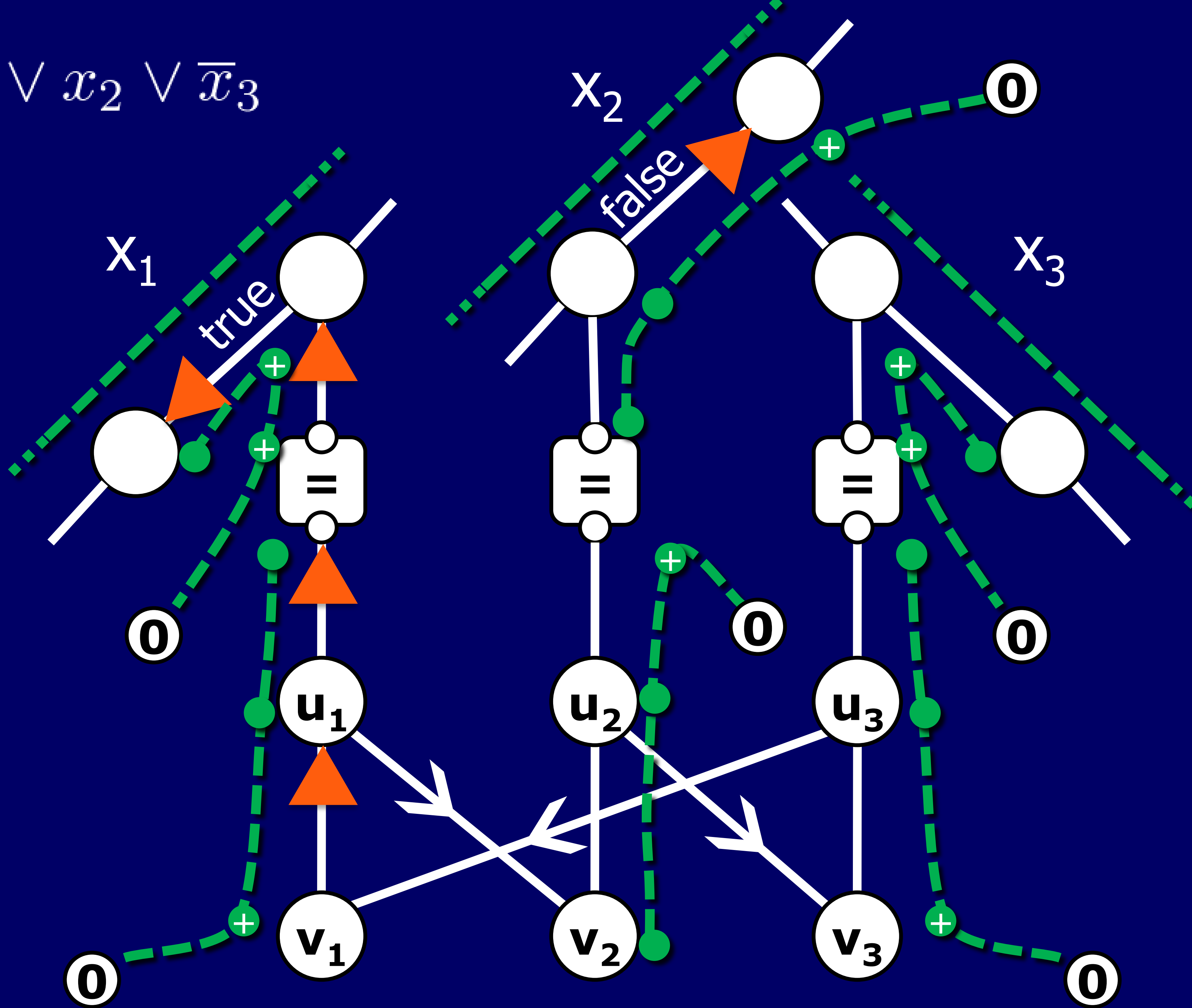


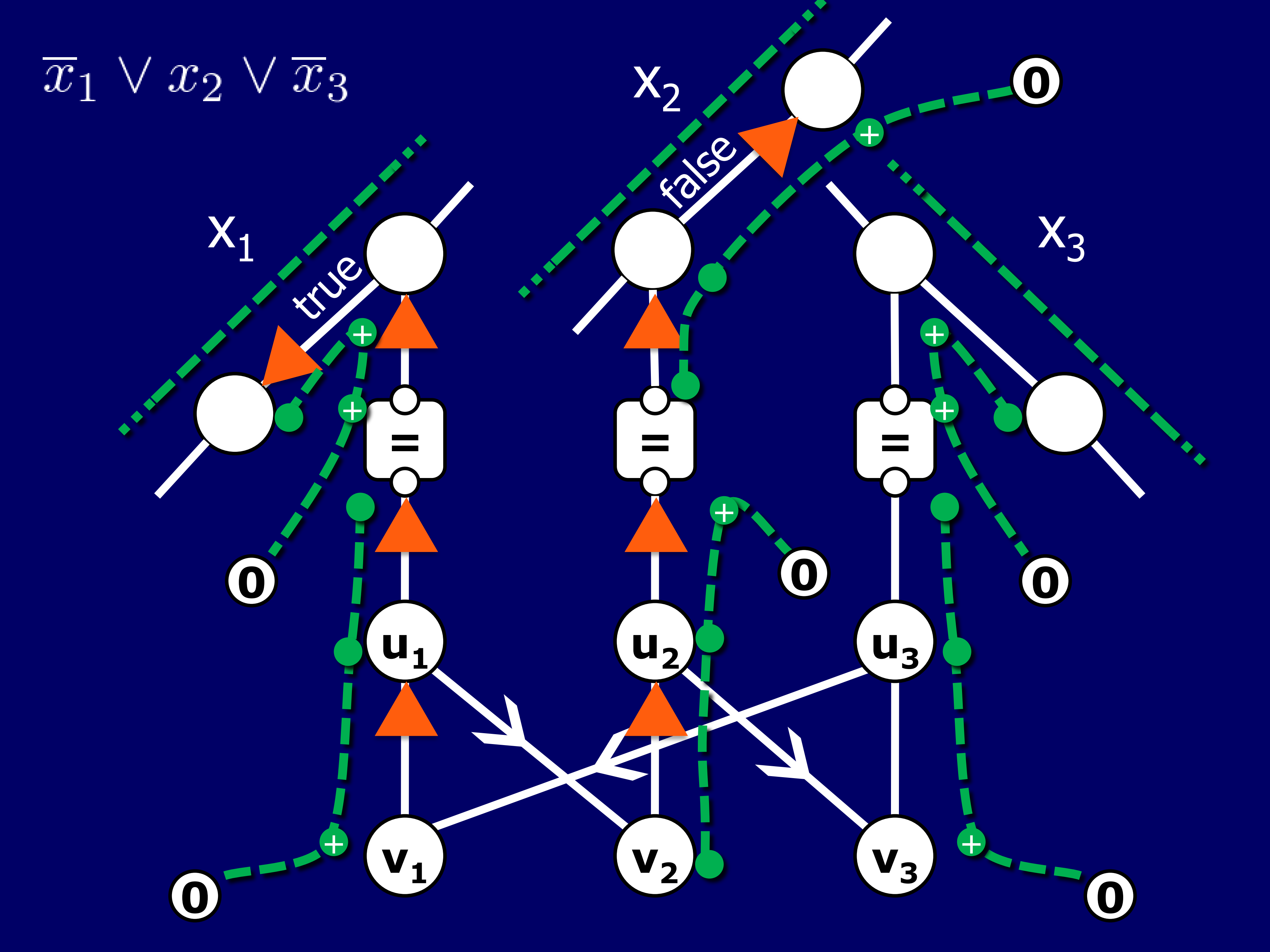
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$
x_2
x_1
x_3
true
0
=
u_1
u_2
u_3
v_1
v_2
v_3

$\overline{x}_1 \vee x_2 \vee \overline{x}_3$

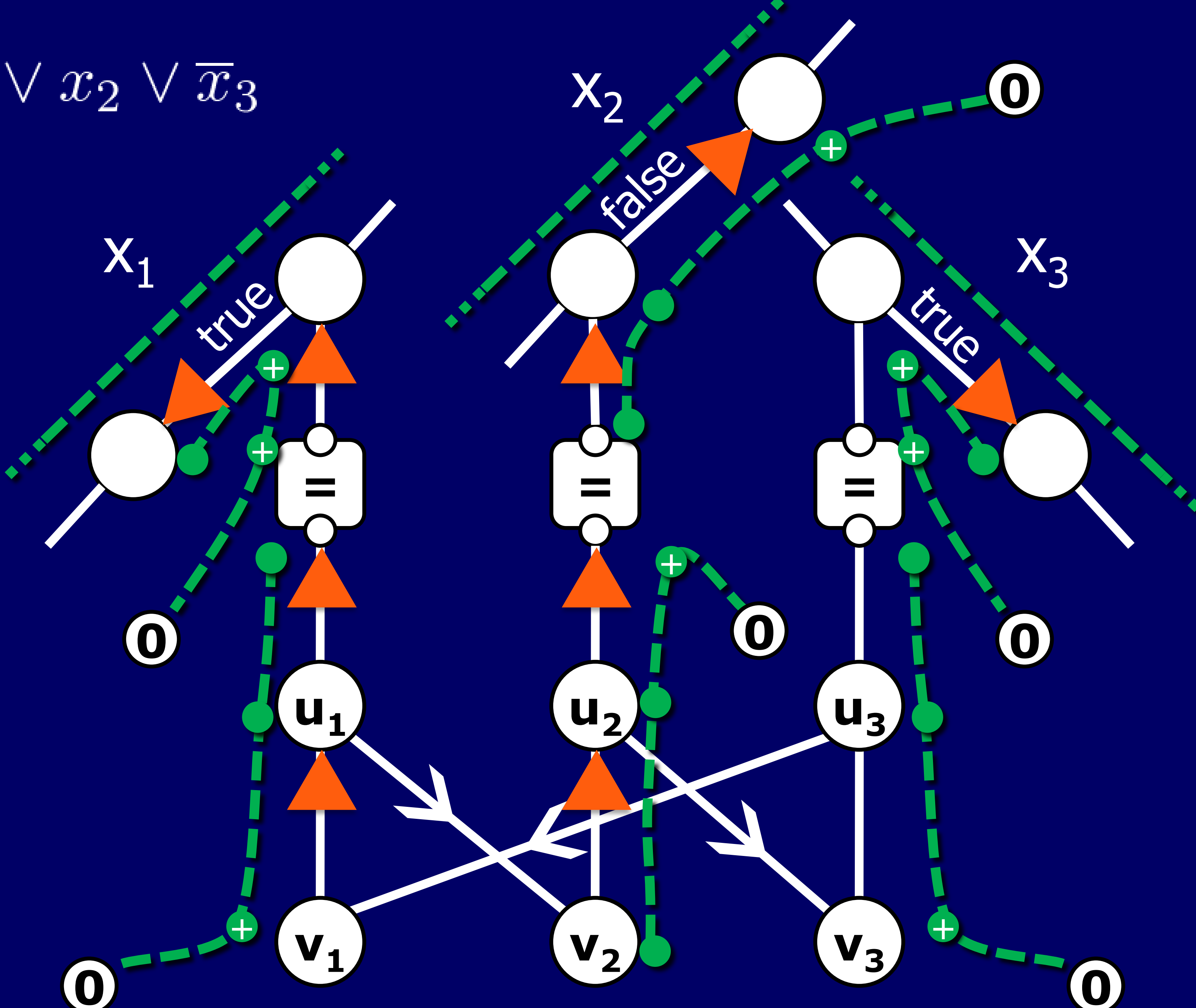


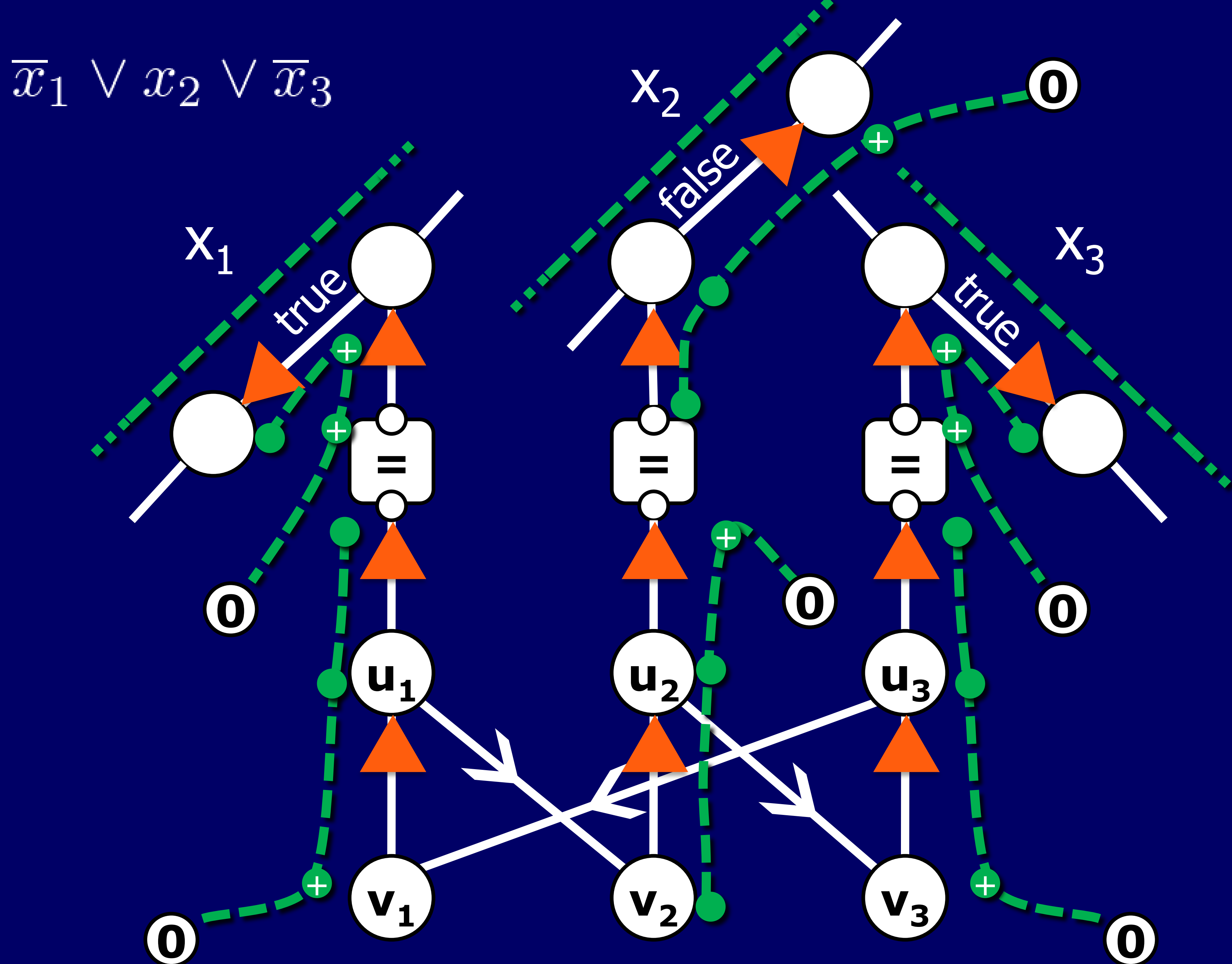
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$




$\overline{x}_1 \vee x_2 \vee \overline{x}_3$
x_1
x_2
x_3
true
false
=
=
=
u_1
u_2
u_3
v_1
v_2
v_3
0
0
0
0
0
0
+

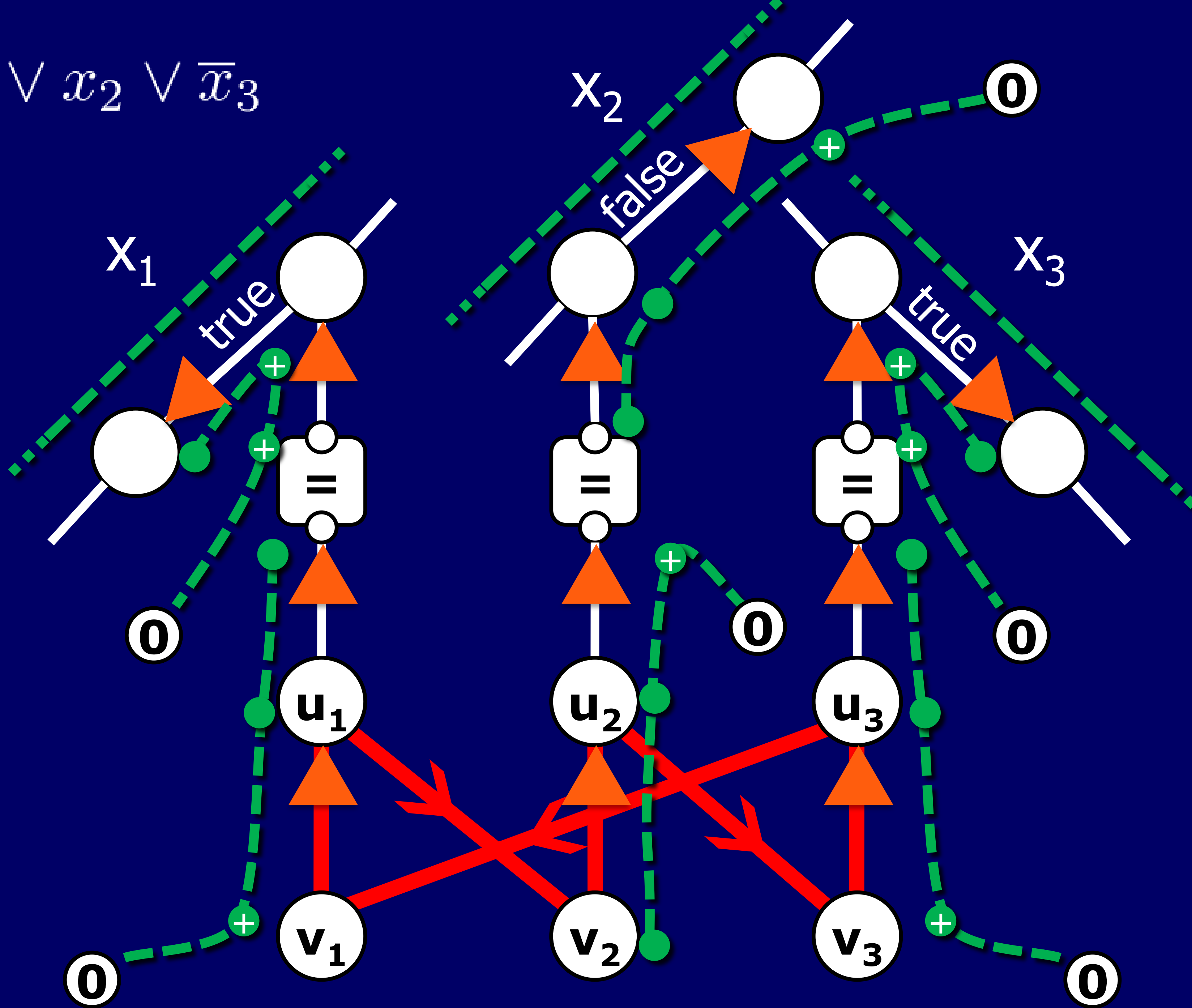
$\overline{x}_1 \vee x_2 \vee \overline{x}_3$





$\overline{x}_1 \vee x_2 \vee \overline{x}_3$
x_2
0
false
x_1
true
x_3
true
=
=
=
0
0
0
u_1
u_2
u_3
0
v_1
v_2
v_3
0

$\overline{x}_1 \vee x_2 \vee \overline{x}_3$



NP-hardness Proof

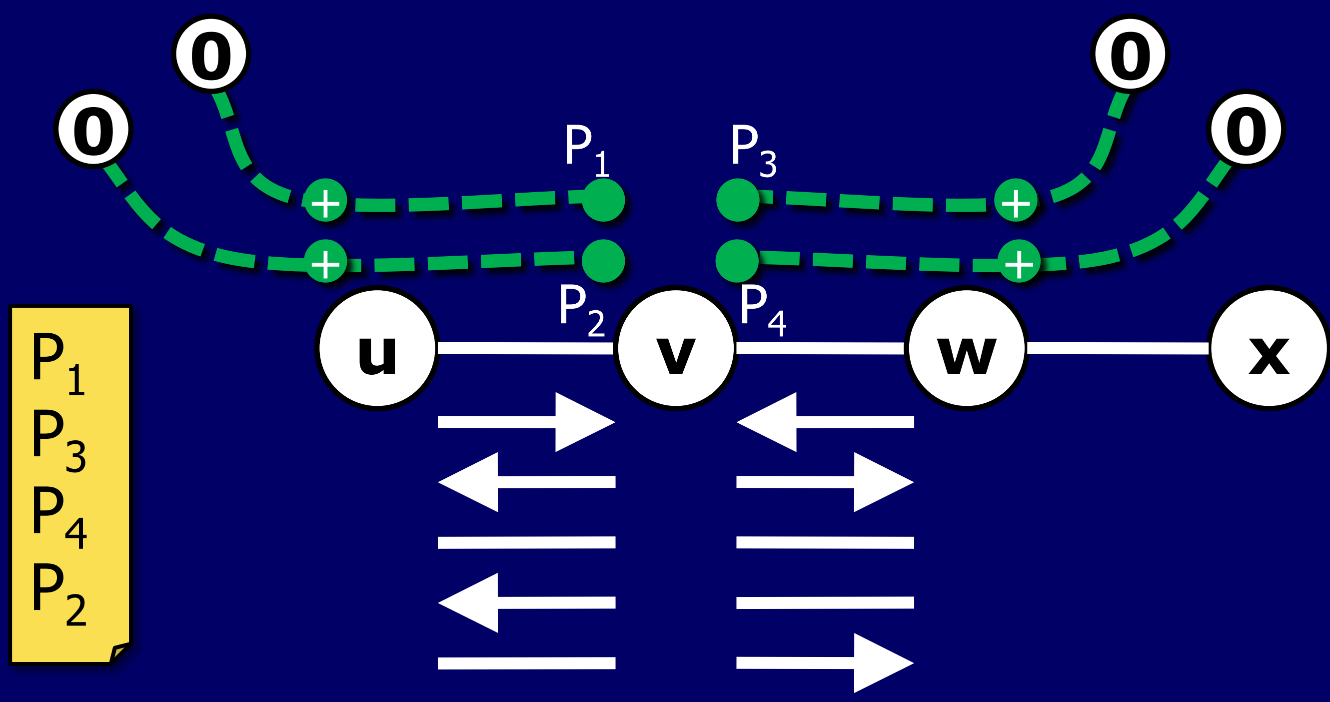
- Polynomial construction
- Also valid in the succinct model (with minor tweaks)

NP-hardness Proof

- Polynomial construction
- Also valid in the succinct model (with minor tweaks)
- Would not work with the original Gao-Rexford conditions

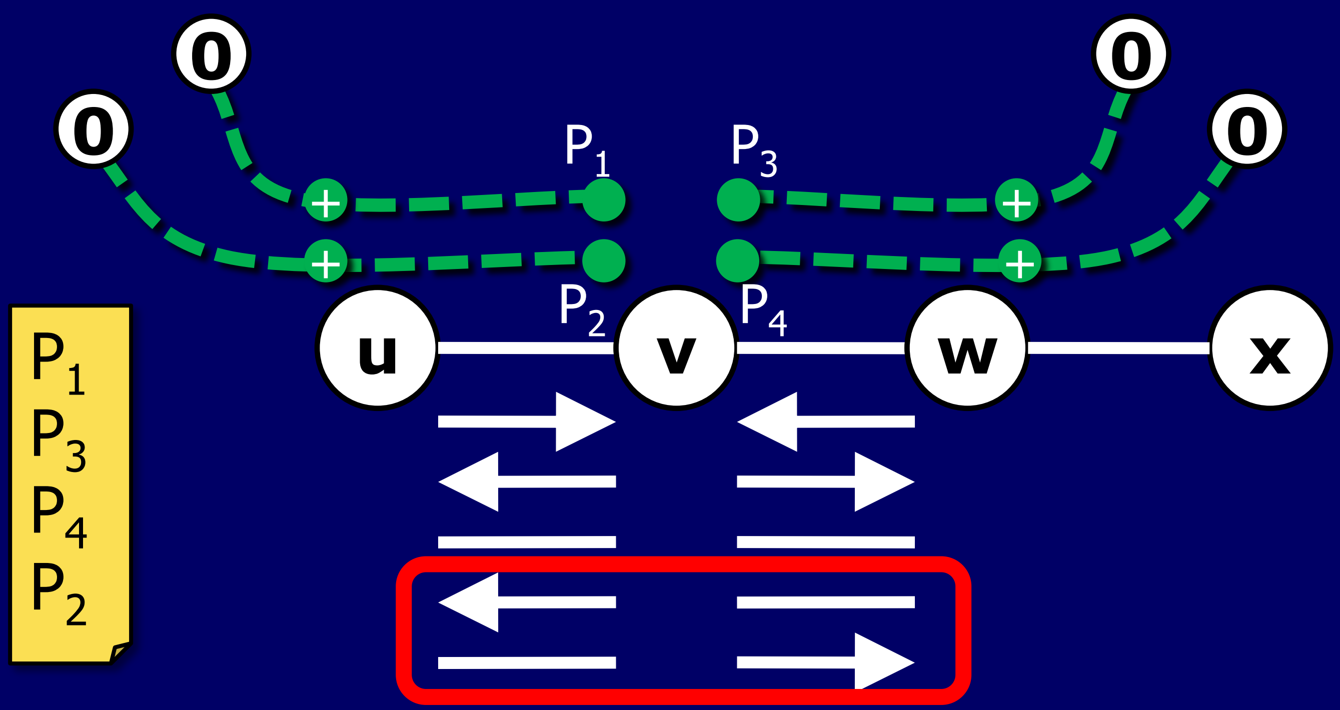
NP-hardness Proof

- Would not work with the original Gao-Rexford conditions



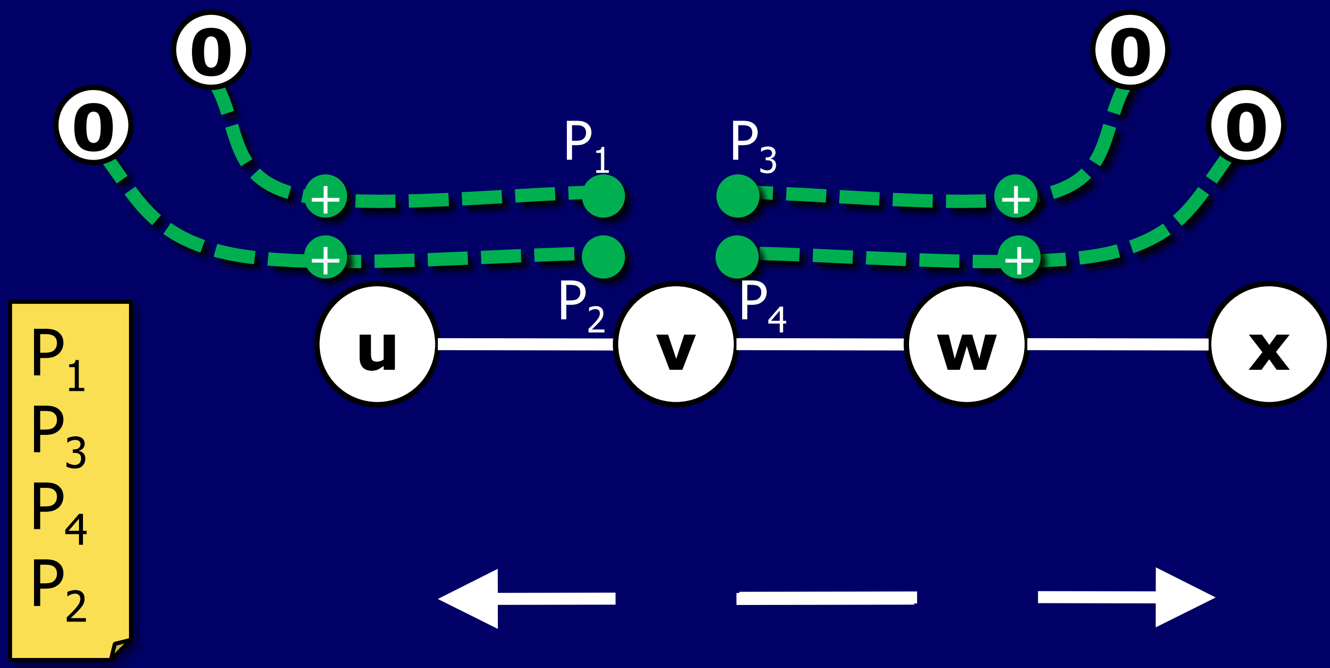
NP-hardness Proof

- Would not work with the original Gao-Rexford conditions



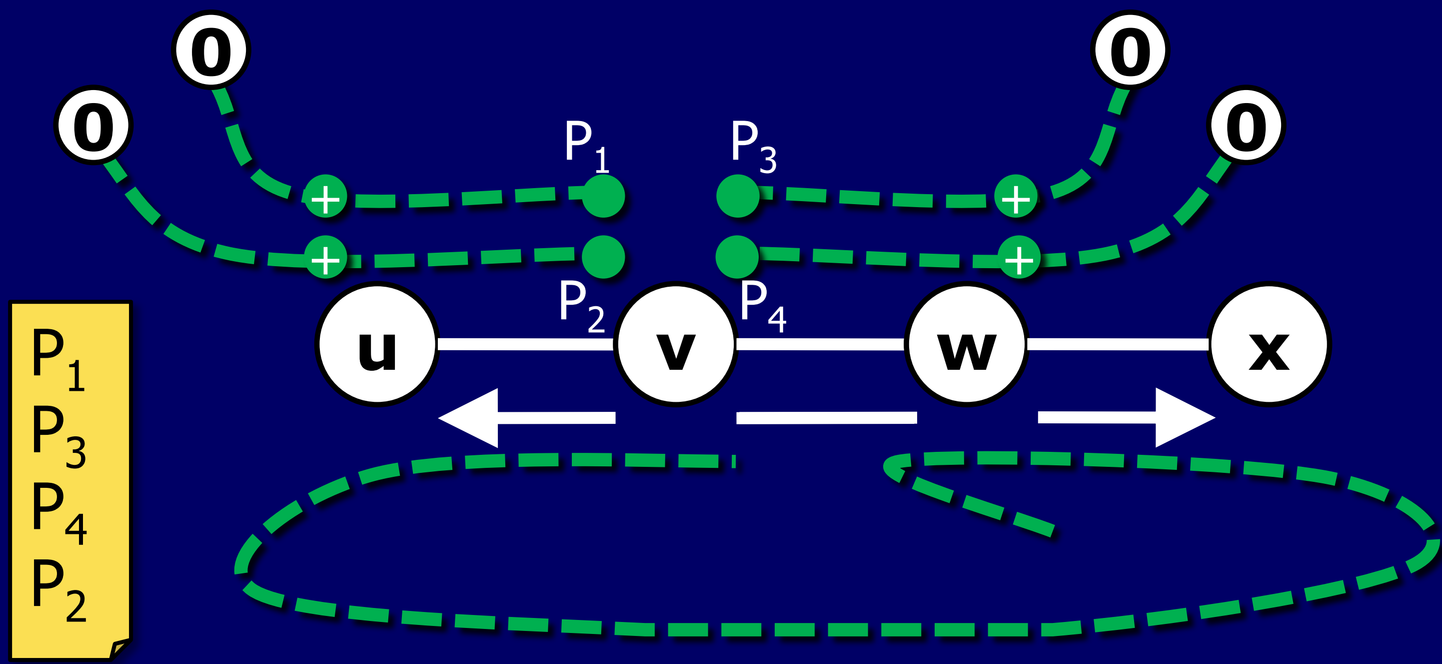
NP-hardness Proof

- Would not work with the original Gao-Rexford conditions



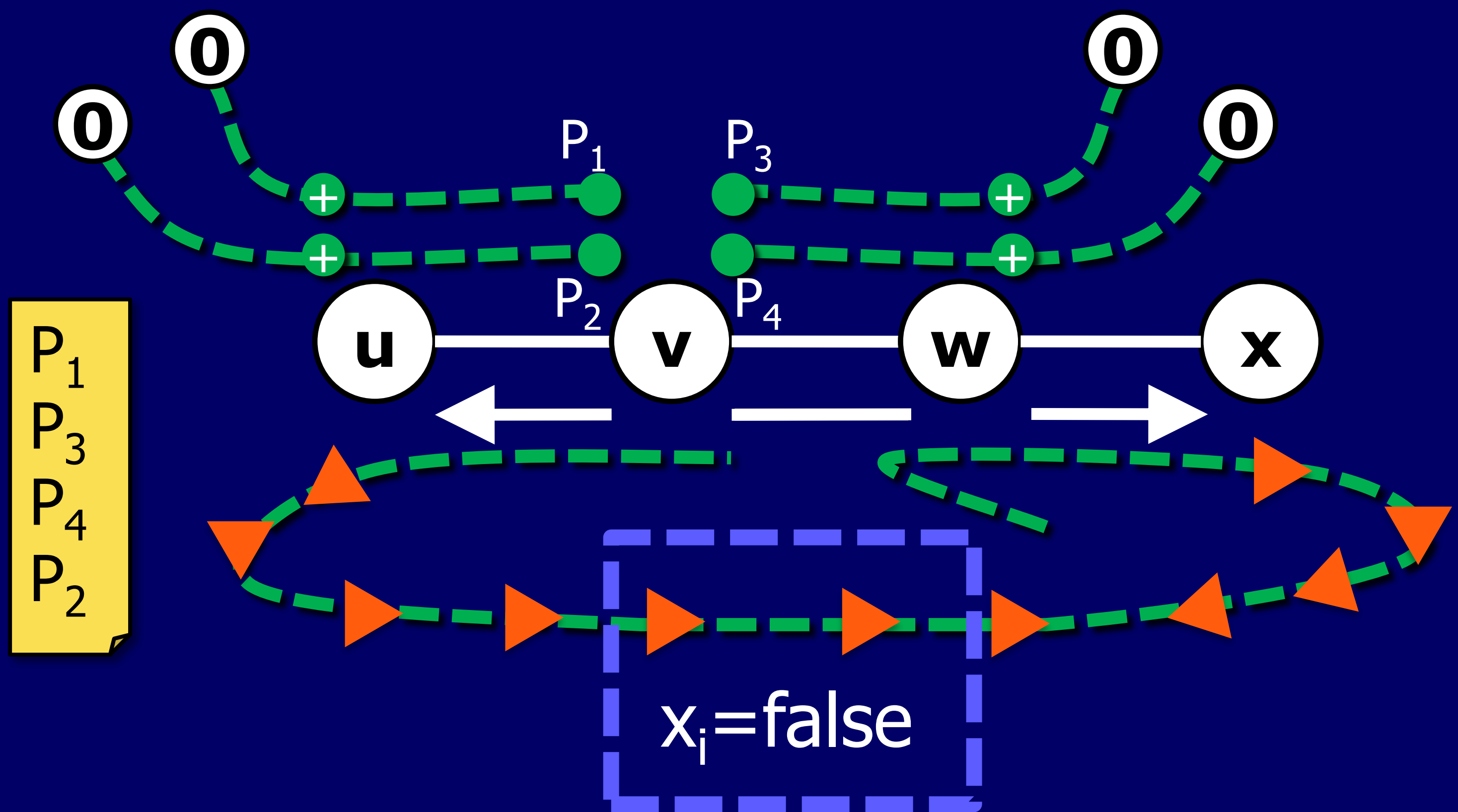
NP-hardness Proof

- Would not work with the original Gao-Rexford conditions



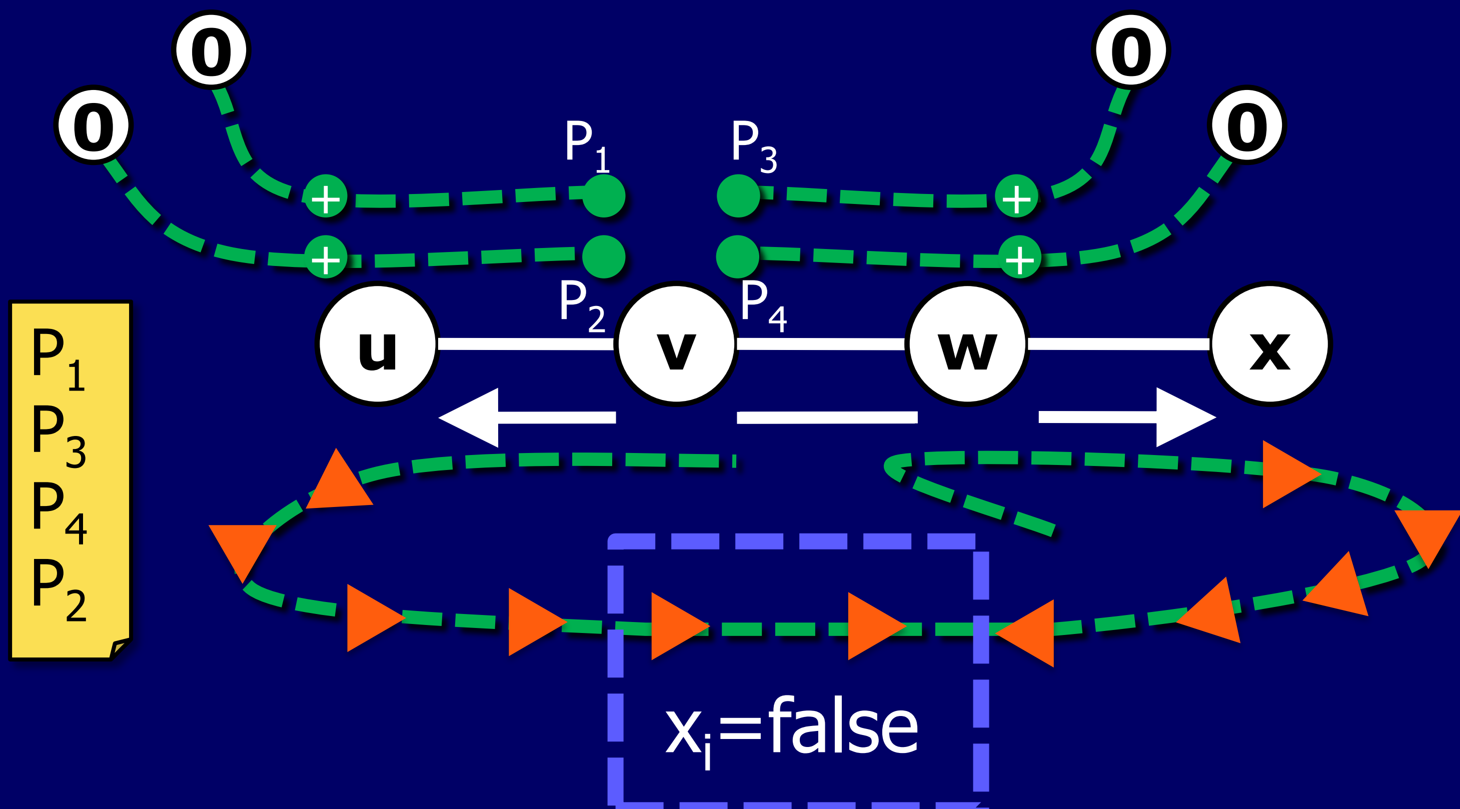
NP-hardness Proof

- Would not work with the original Gao-Rexford conditions



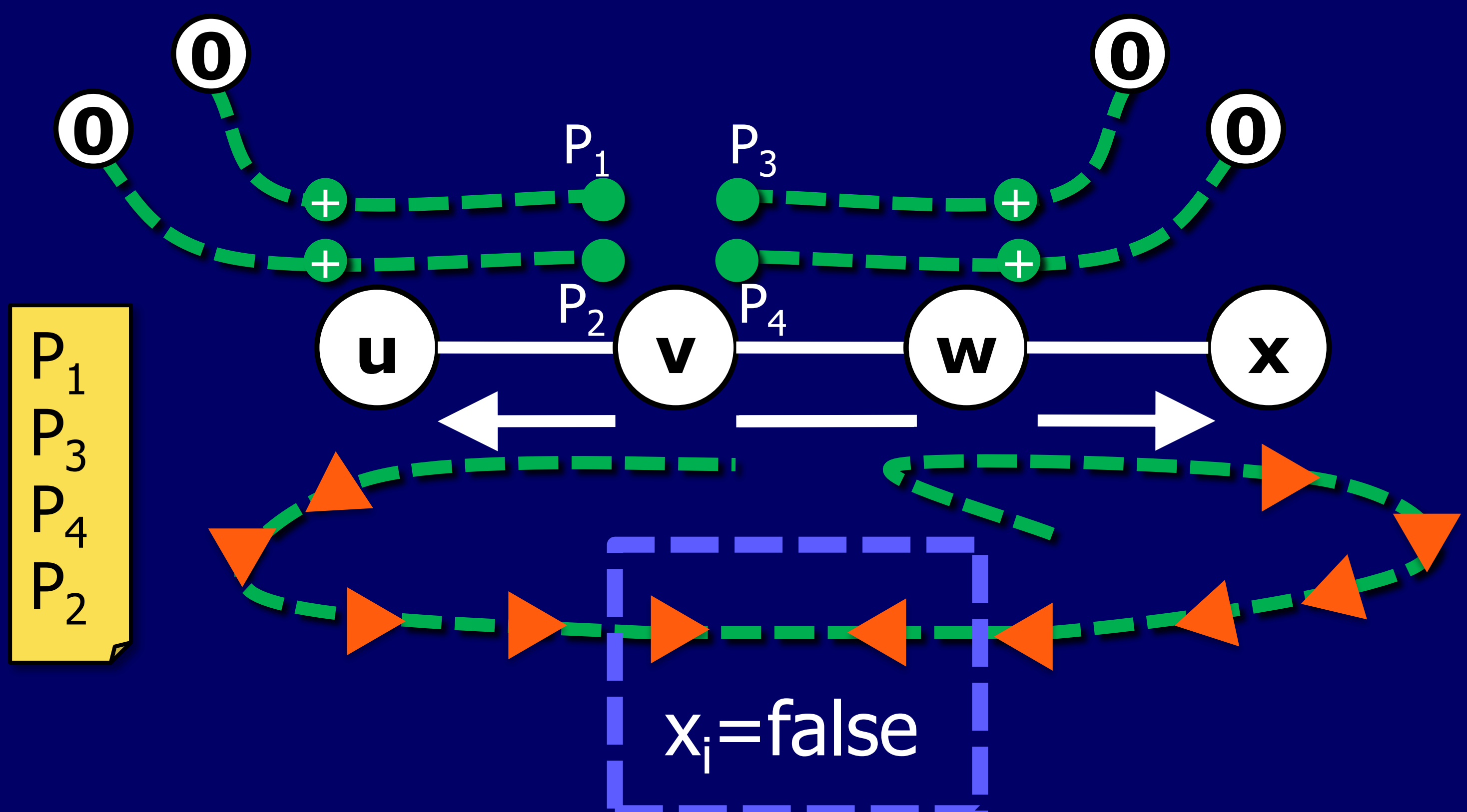
NP-hardness Proof

- Would not work with the original Gao-Rexford conditions



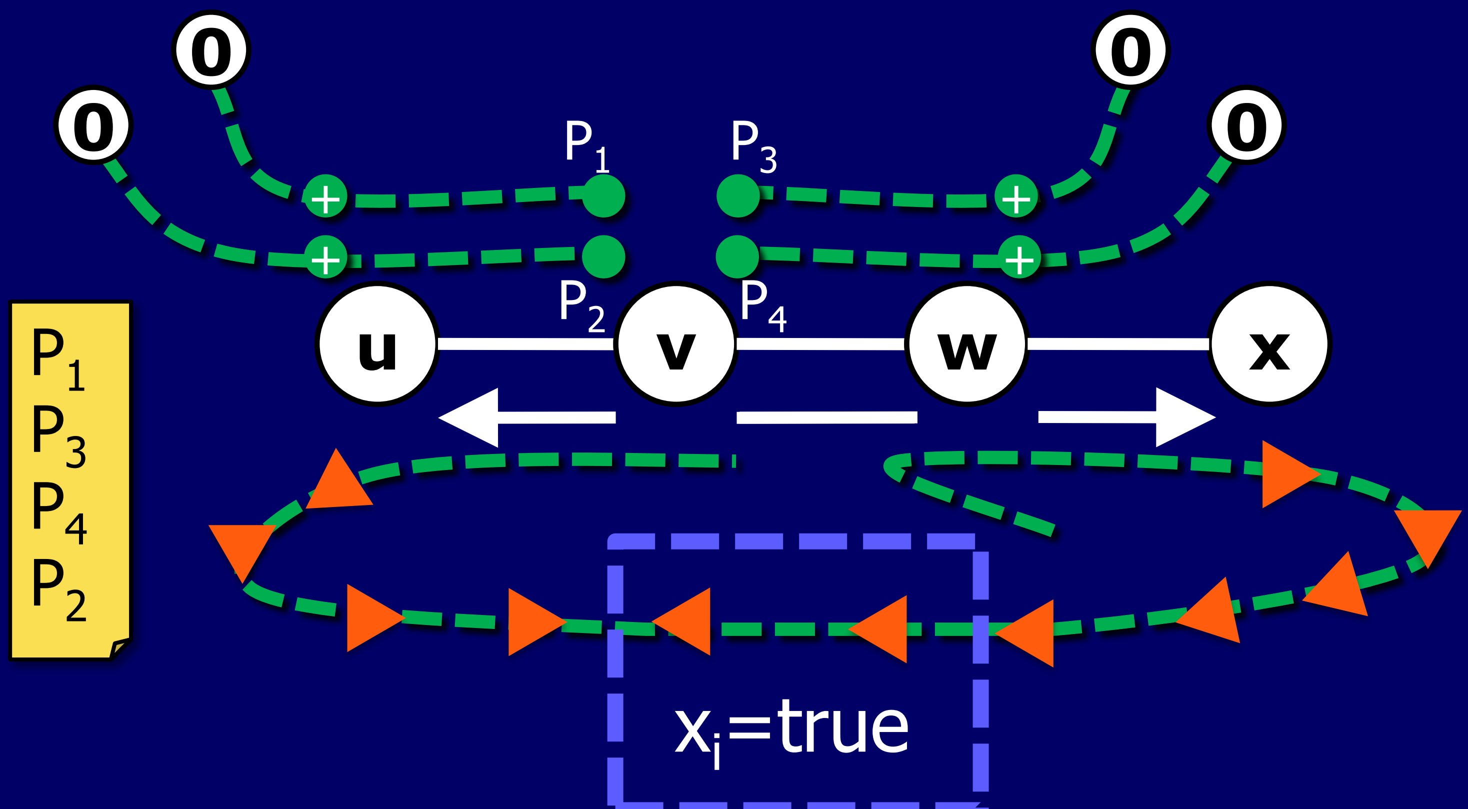
NP-hardness Proof

- Would not work with the original Gao-Rexford conditions



NP-hardness Proof

- Would not work with the original Gao-Rexford conditions



Concluding Remarks

Concluding Remarks

- Our contribution:
- Applicability:
- Open Problems:

Concluding Remarks

- Our contribution (in 4 words):
- Applicability:
- Open Problems:

Concluding Remarks

- Our contribution (in 4 words):
  - feasibility of checking GR
    - relevant for routing stability
- Applicability:
- Open Problems:

Concluding Remarks

- Our contribution (in 4 words):
  - feasibility of checking GR
    - relevant for routing stability
- Applicability (hints):
- Open Problems:

Concluding Remarks

- Our contribution (in 4 words):
  - feasibility of checking GR
    - relevant for routing stability
- Applicability (hints):
  - network simulators
  - iBGP, confederations
- Open Problems:

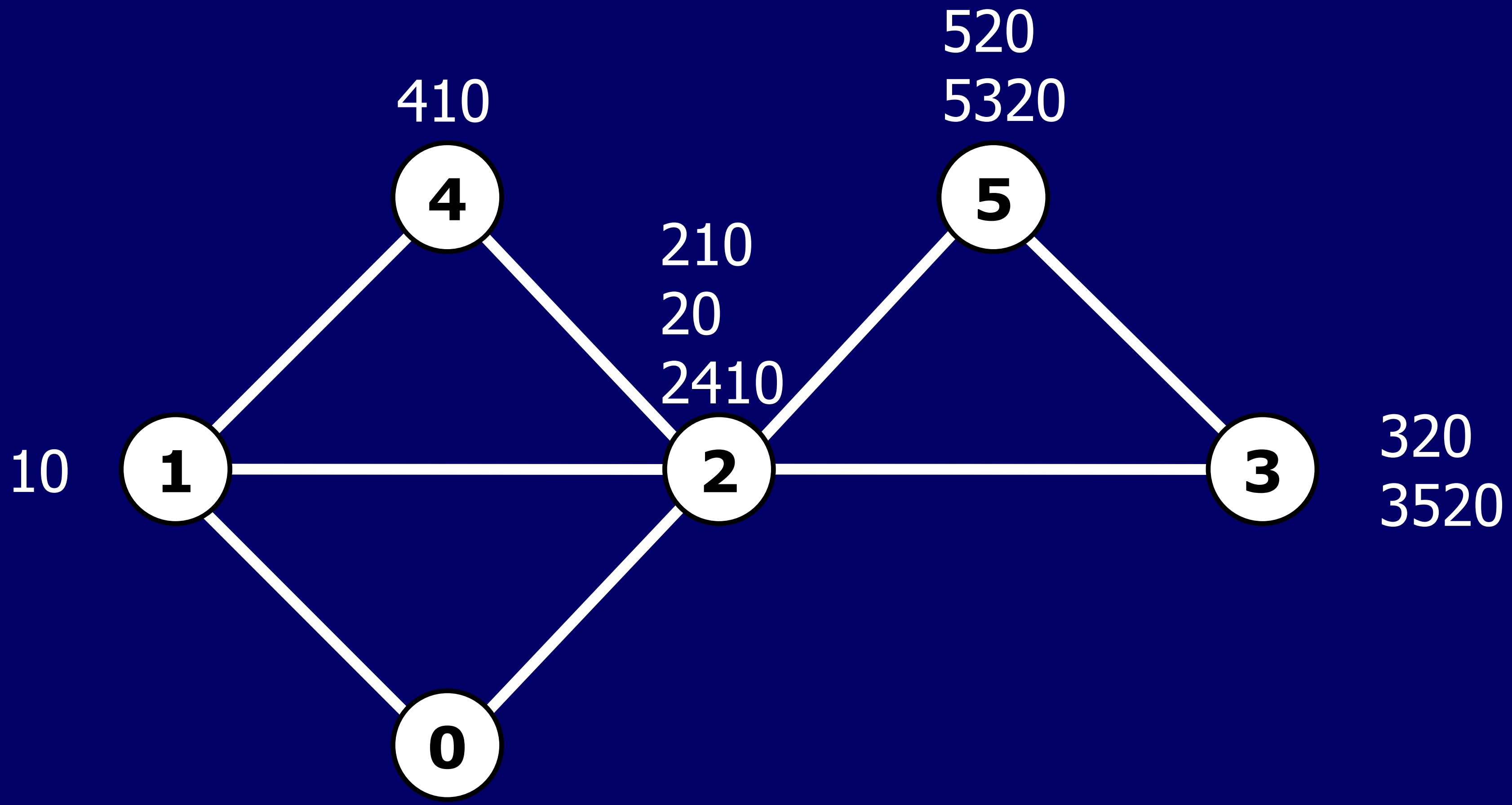
Concluding Remarks

- Our contribution (in 4 words):
  - feasibility of checking GR
    - relevant for routing stability
- Applicability (hints):
  - network simulators
  - iBGP, confederations
- Open Problems:
  - backup routing policies?
  - complexity of other conditions (no DW, etc.)?
  - other models (e.g., [13])

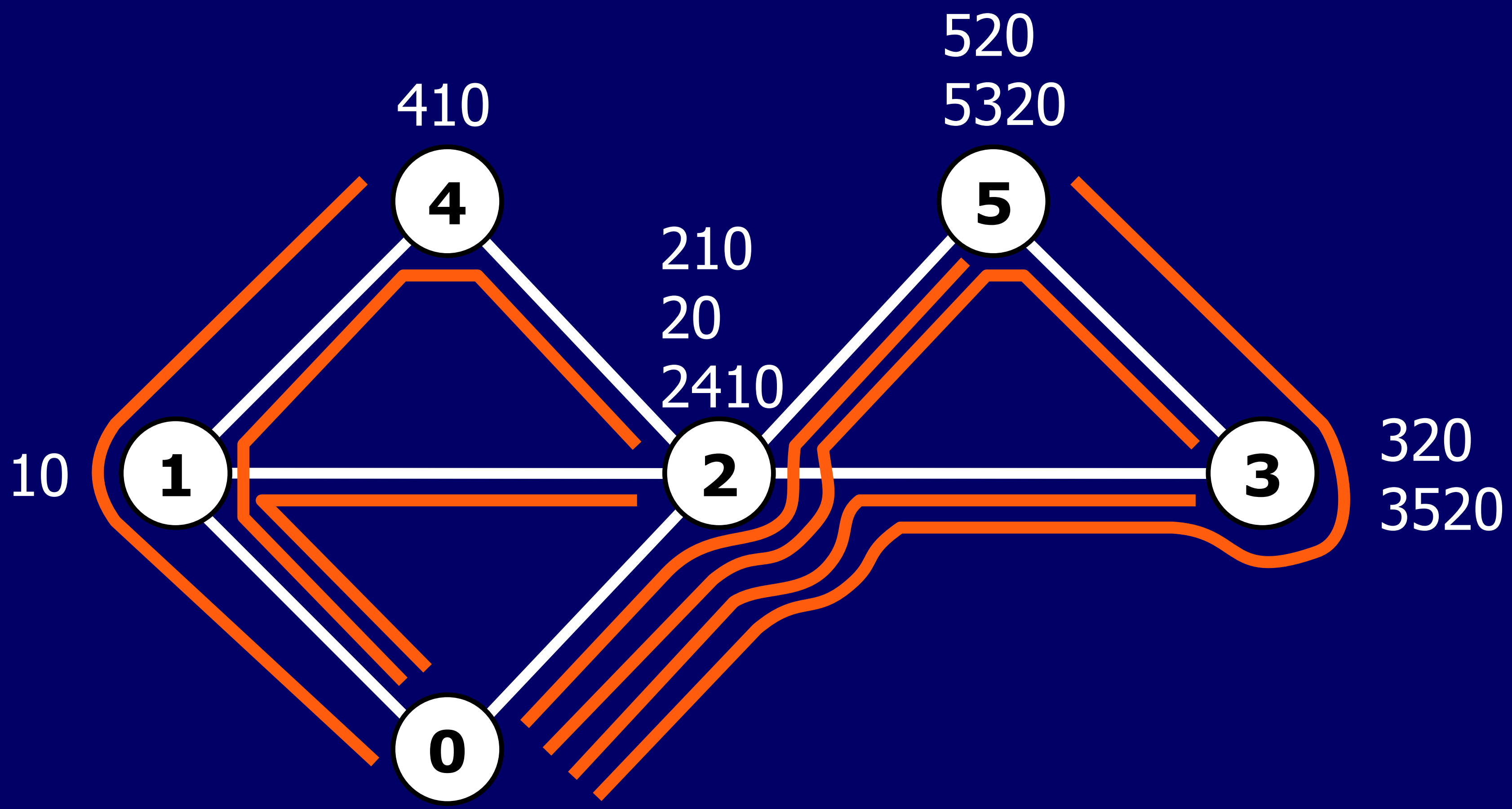
[13] T. Griffin, J. Sobrinho. Metarouting. SIGCOMM 2005.

ども　ありがとう
ございます。
(should read:
“thank you very much”)

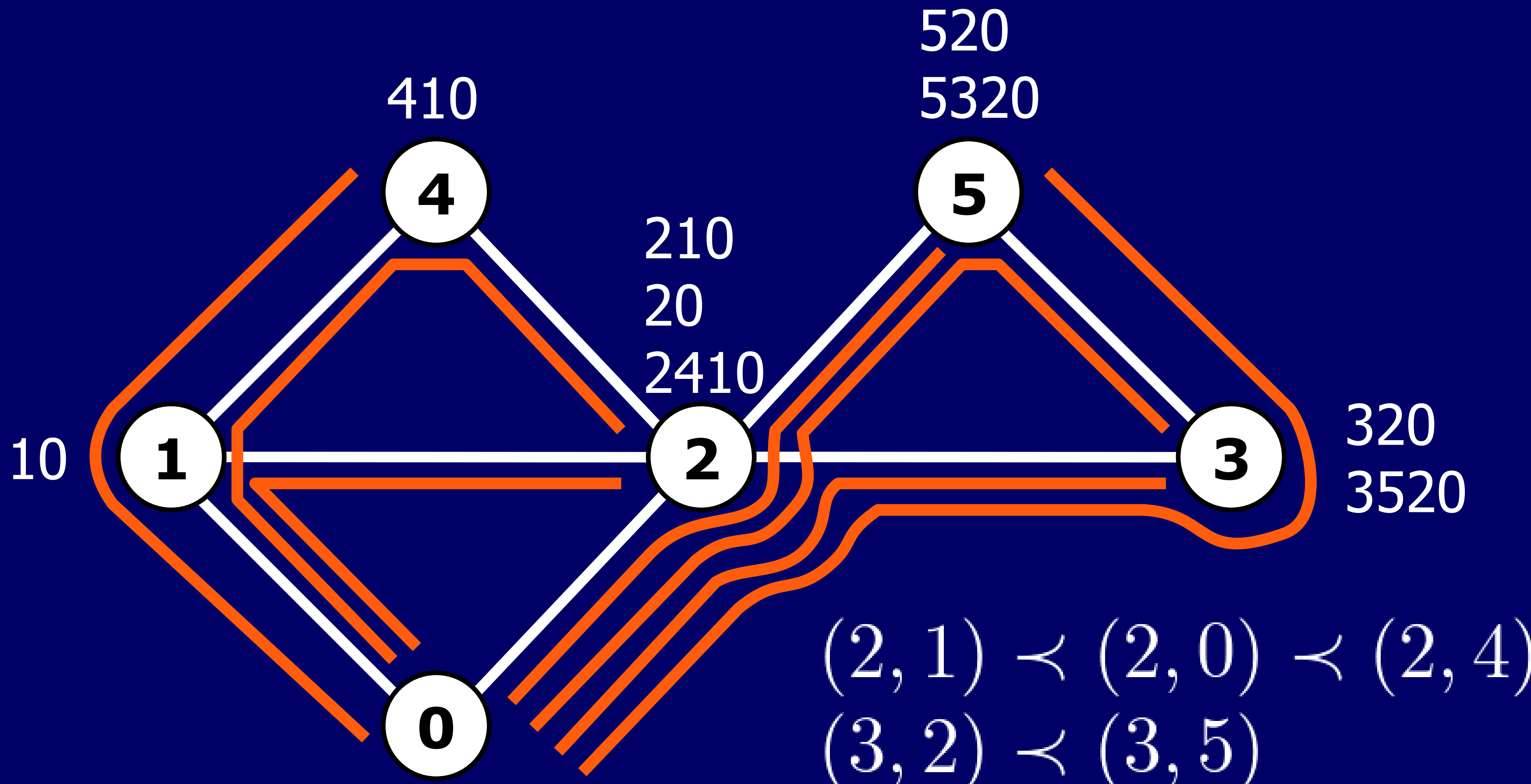
Running Example



Running Example

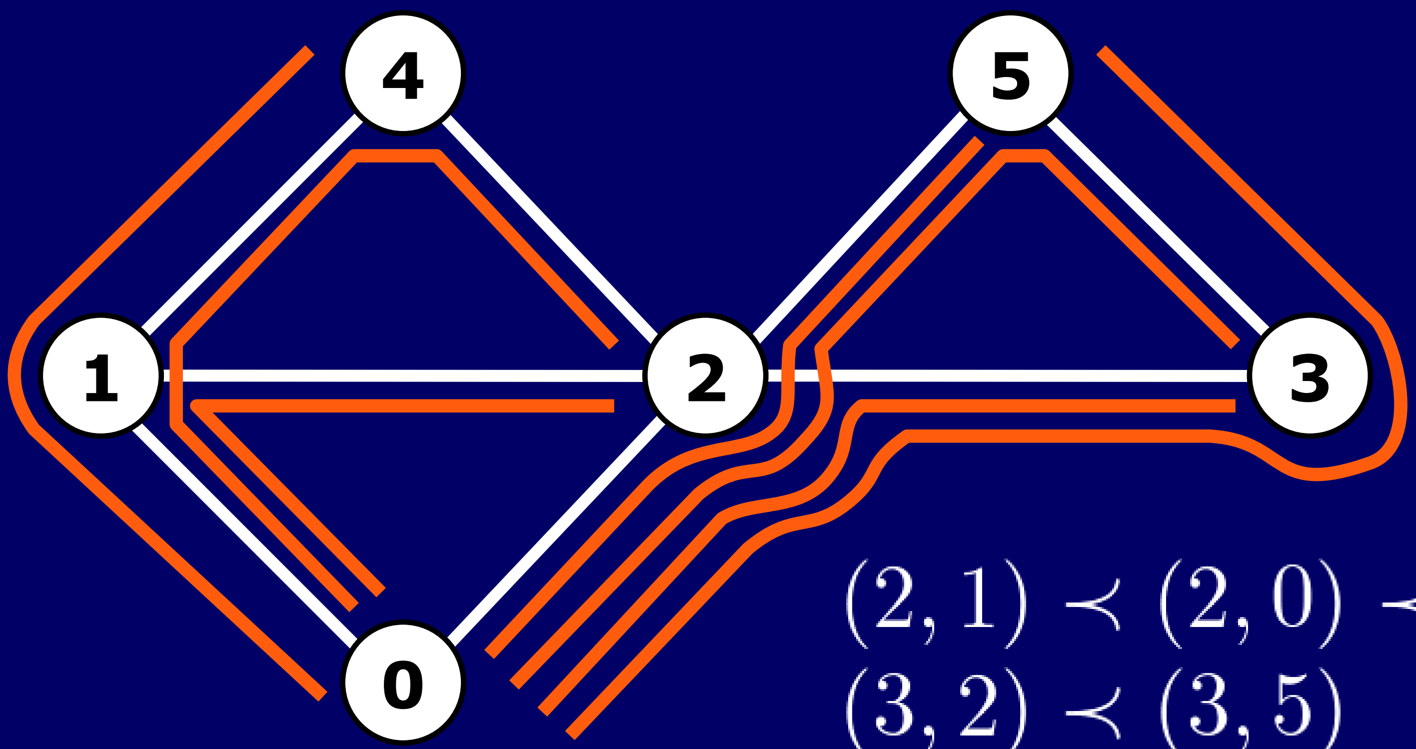


Running Example



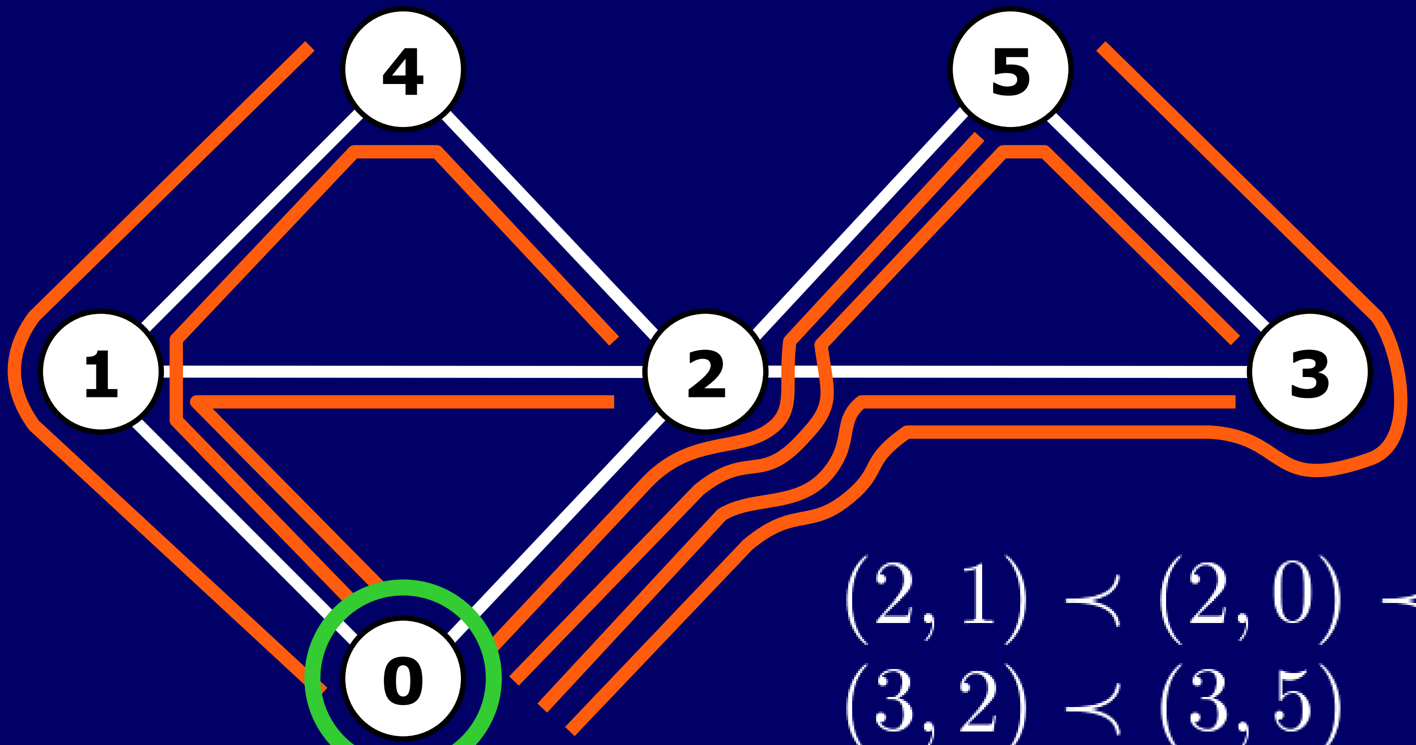
$$(2,1) \prec (2,0) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example



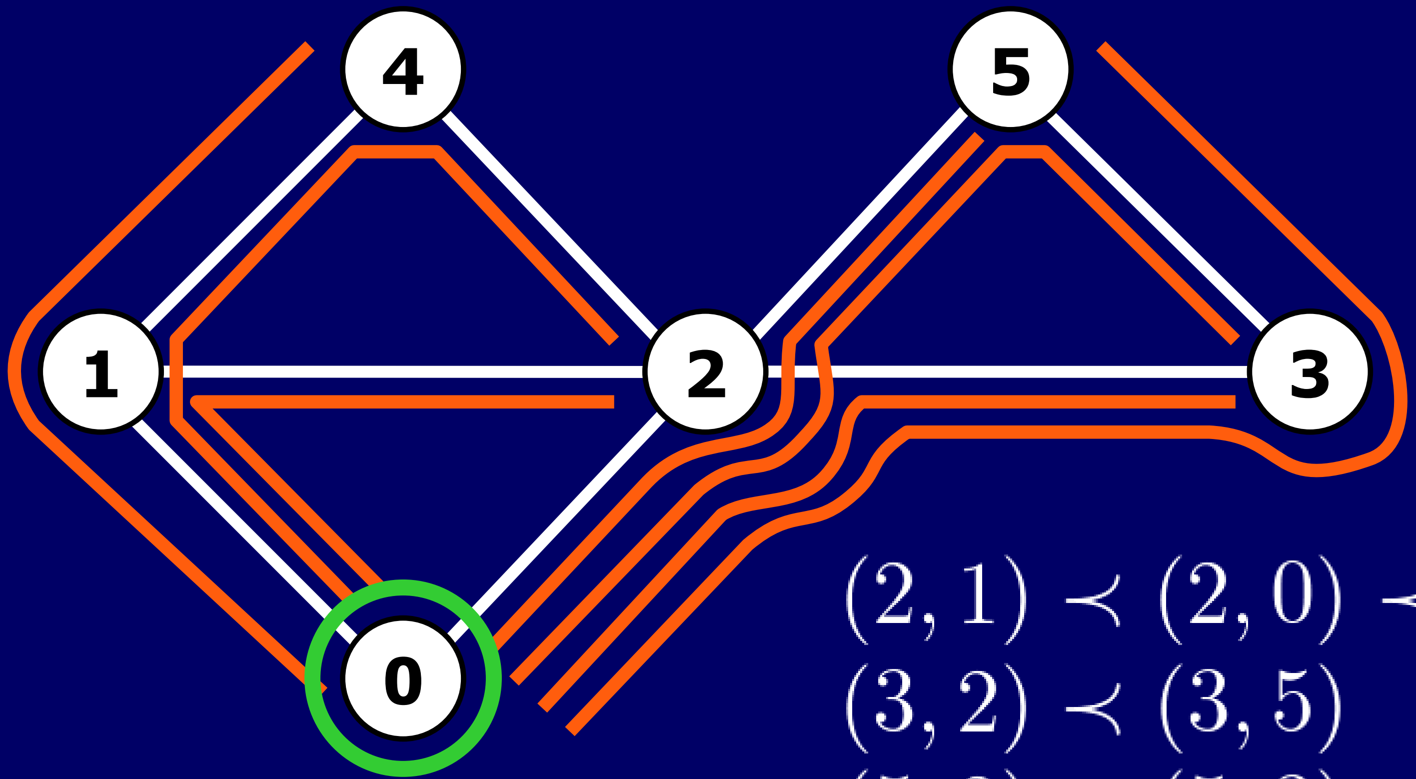
$$(2,1) \prec (2,0) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example



$$(2,1) \prec (2,0) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

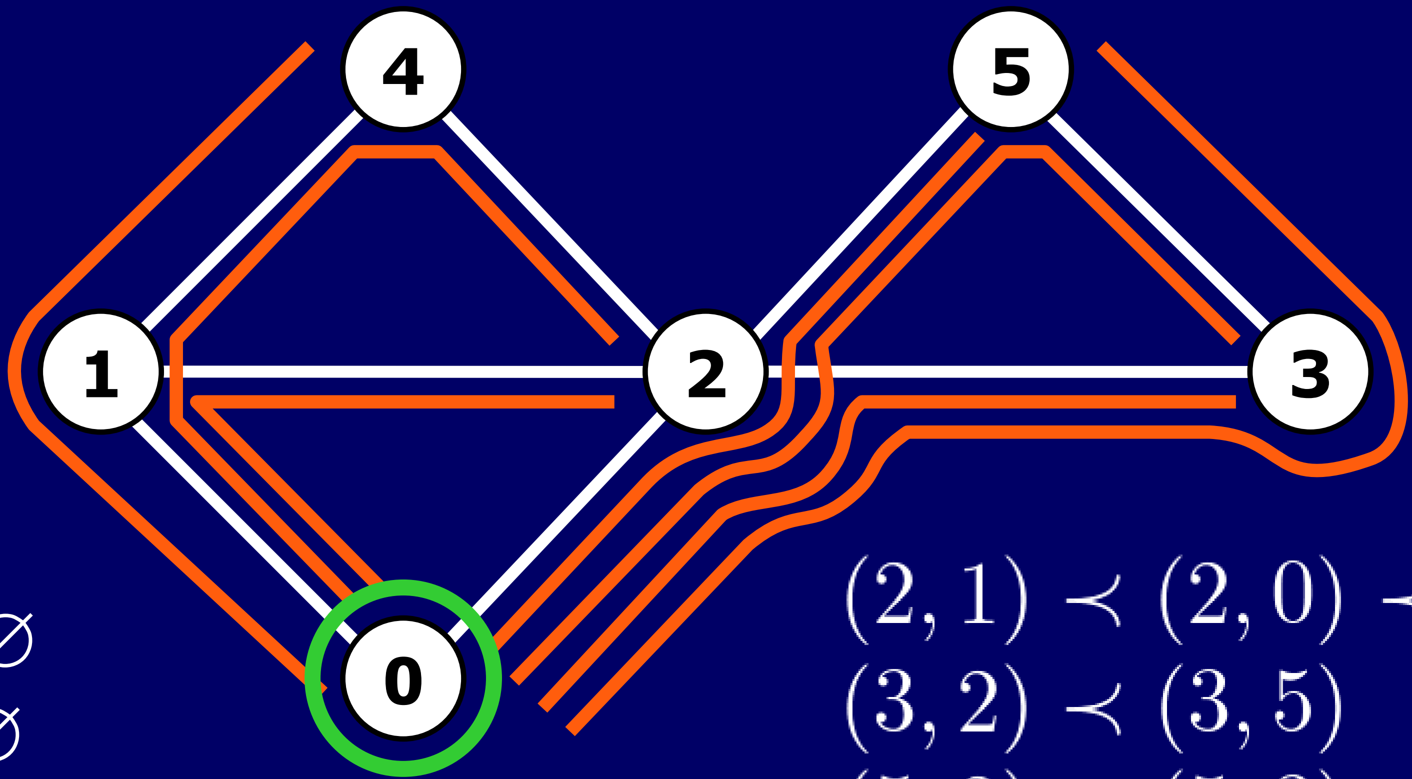
Running Example



$$(2,1) \prec (2,0) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

not traversed by any paths
no incoming edge

Running Example



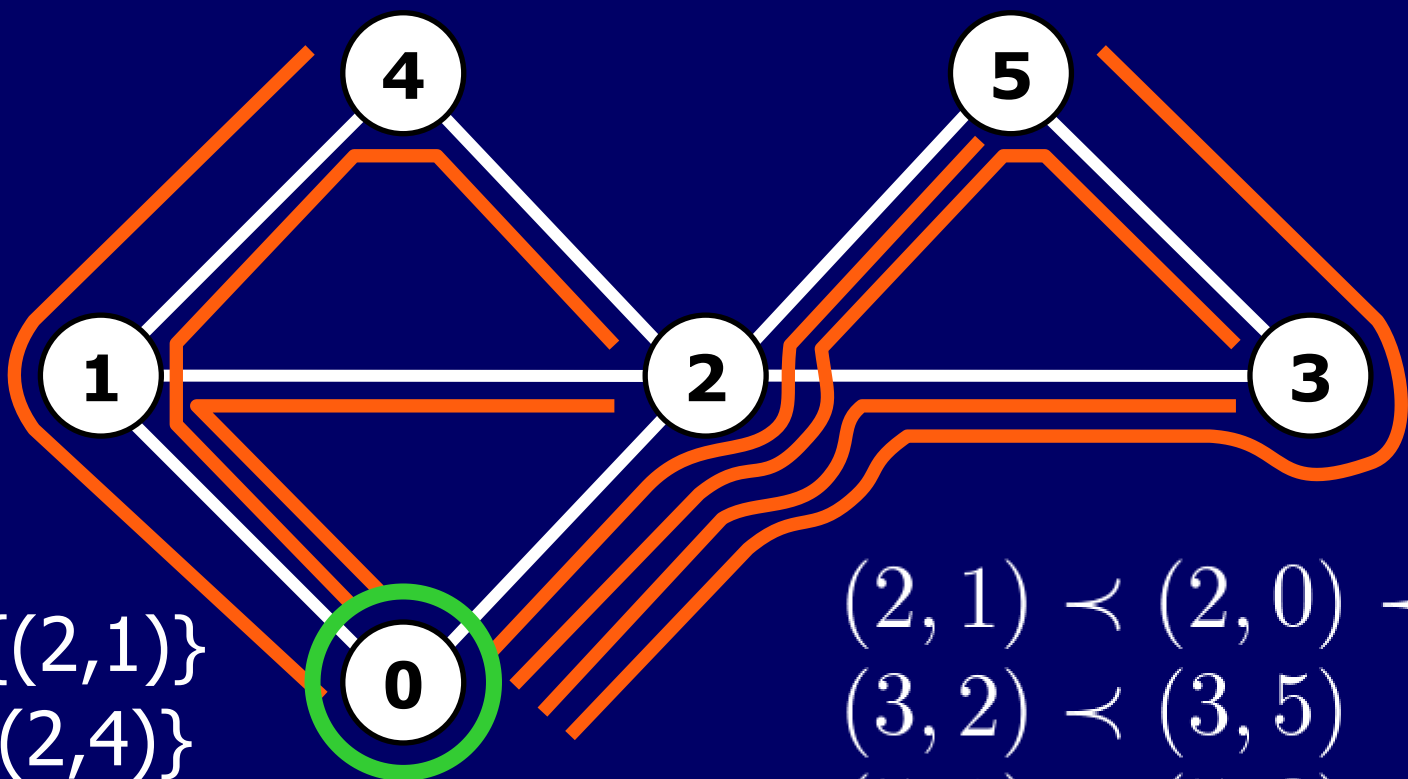
$$(2,1) \prec (2,0) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

$H_{10} = \varnothing$

$L_{10} = \varnothing$

$F_{10} = \{(1,2),(1,4)\}$

Running Example



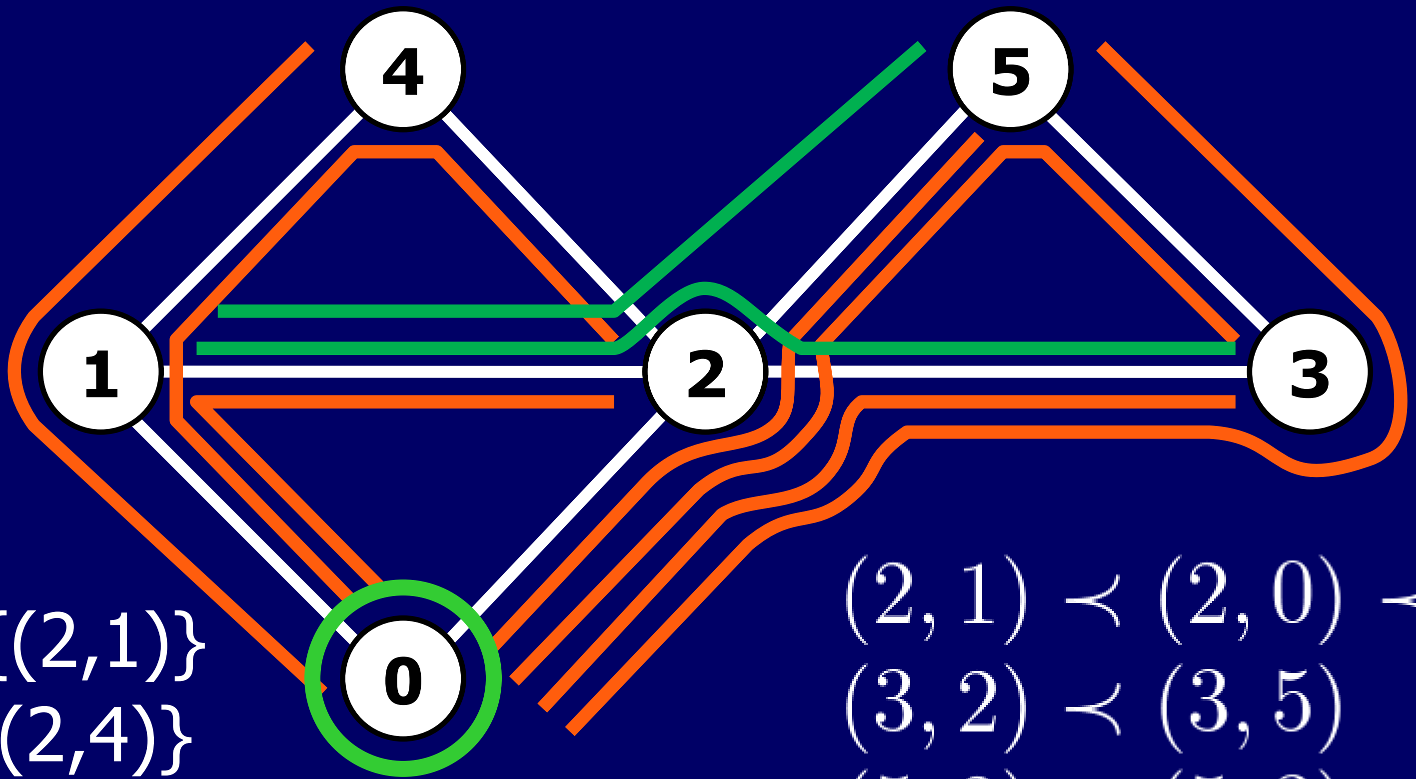
$H_{20}=\{(2,1)\}$

$L_{20}=\{(2,4)\}$

$F_{20}=\{(2,5),(2,3)\}$

$$(2,1) \prec (2,0) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example



$H_{20}=\{(2,1)\}$

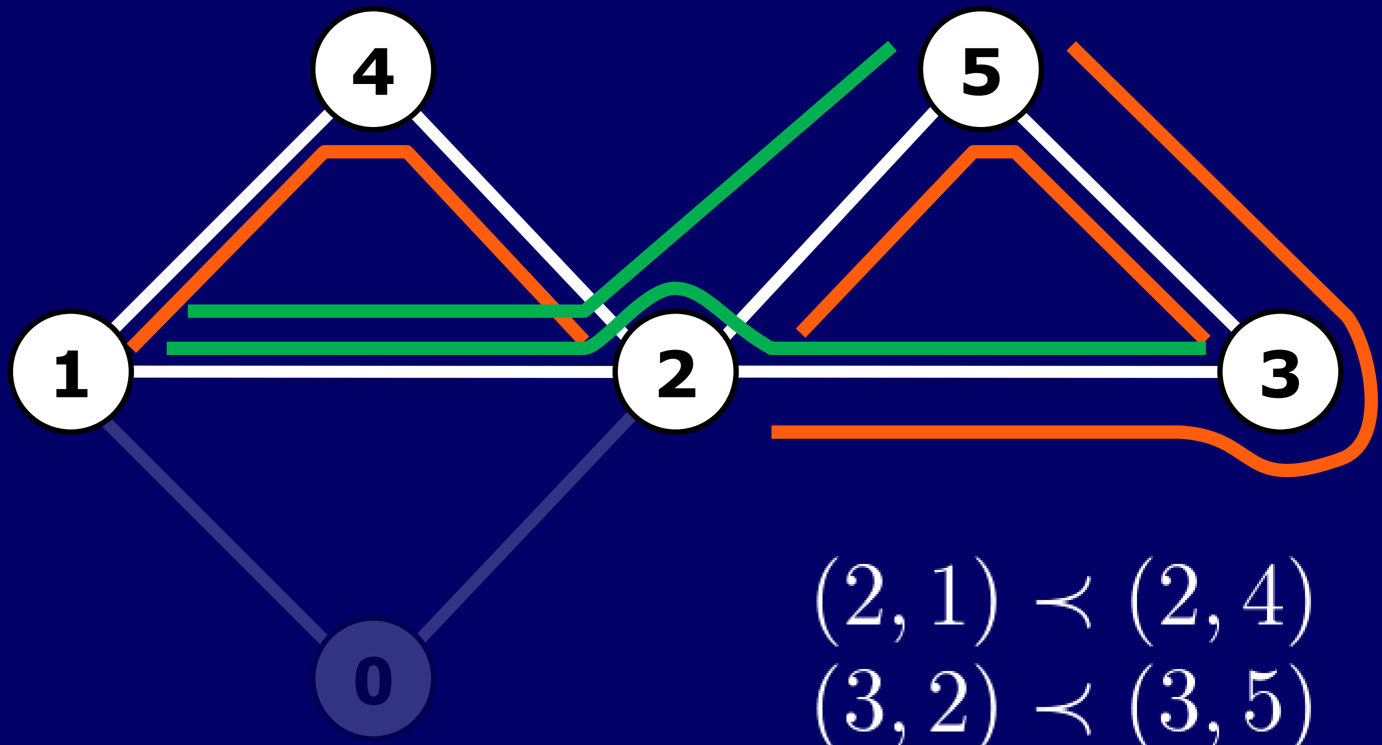
$L_{20}=\{(2,4)\}$

$F_{20}=\{(2,5),(2,3)\}$

$$(2,1) \prec (2,0) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example

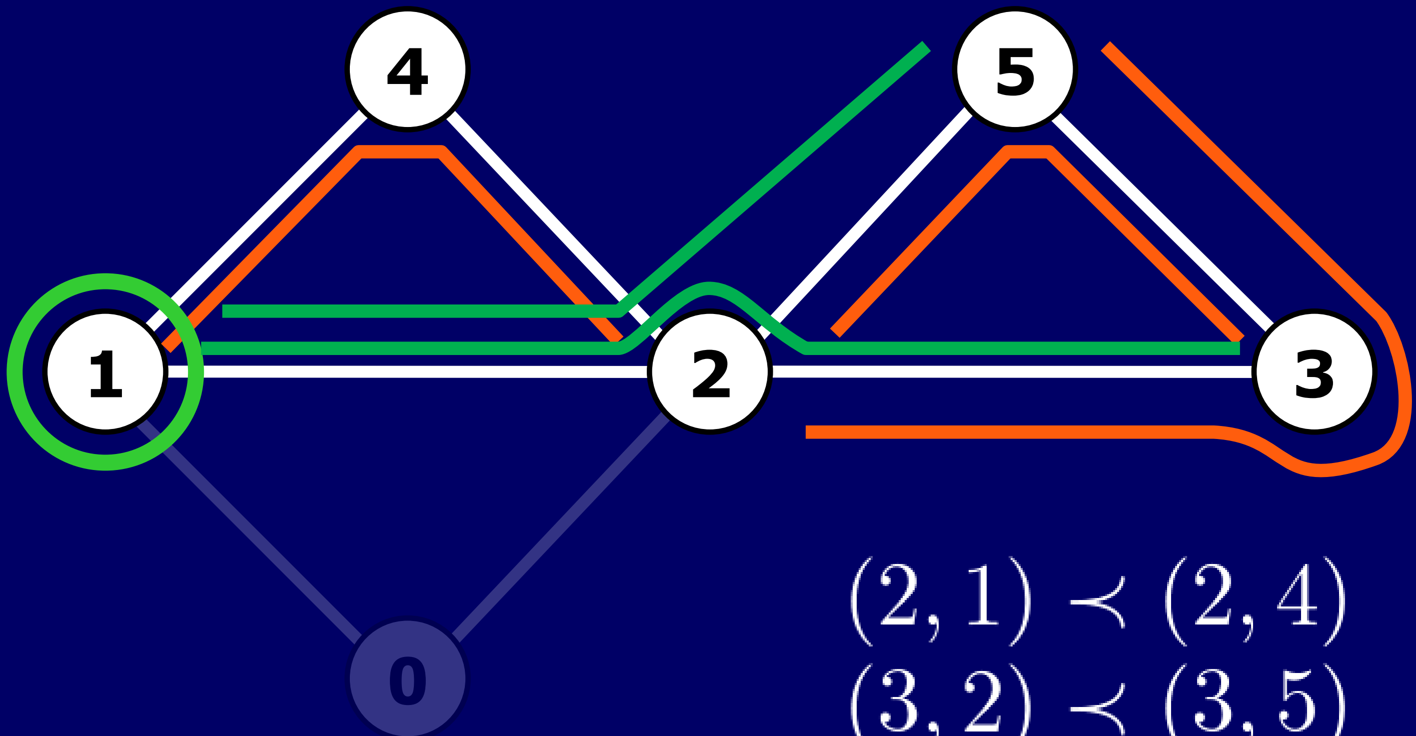
Recursion



$$(2,1) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example

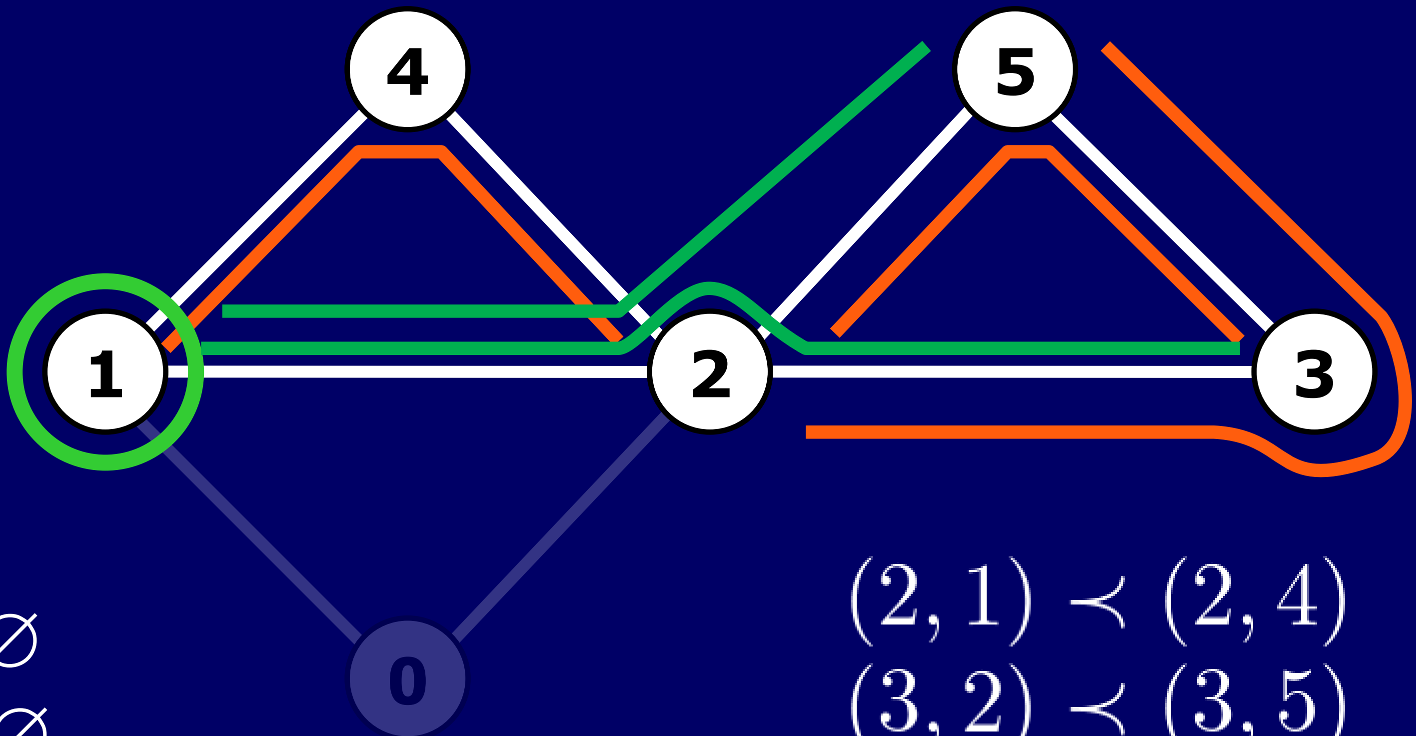
Recursion



$$(2,1) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example

Recursion



$H_{41} = \varnothing$

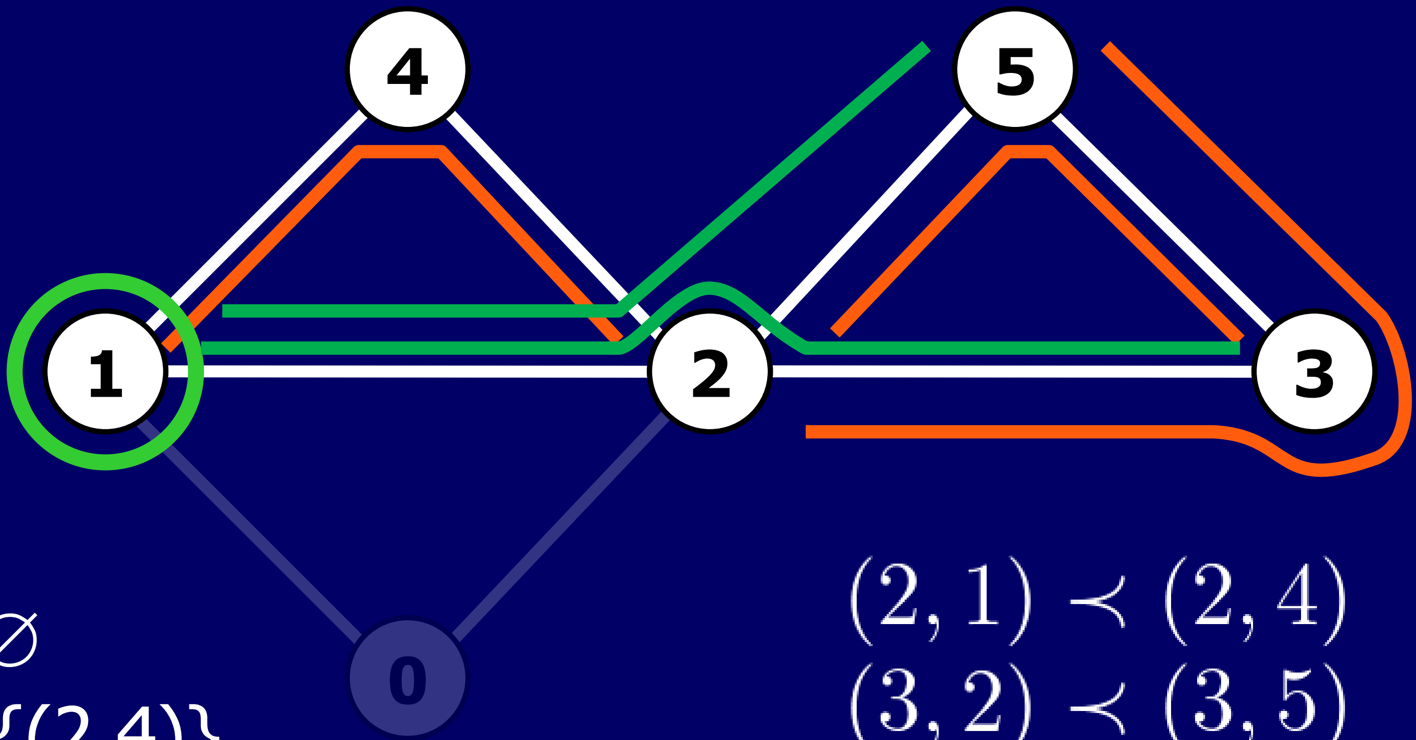
$L_{41} = \varnothing$

$F_{41} = \{(4,2)\}$

$$(2,1) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example

Recursion



$H_{21}=\varnothing$

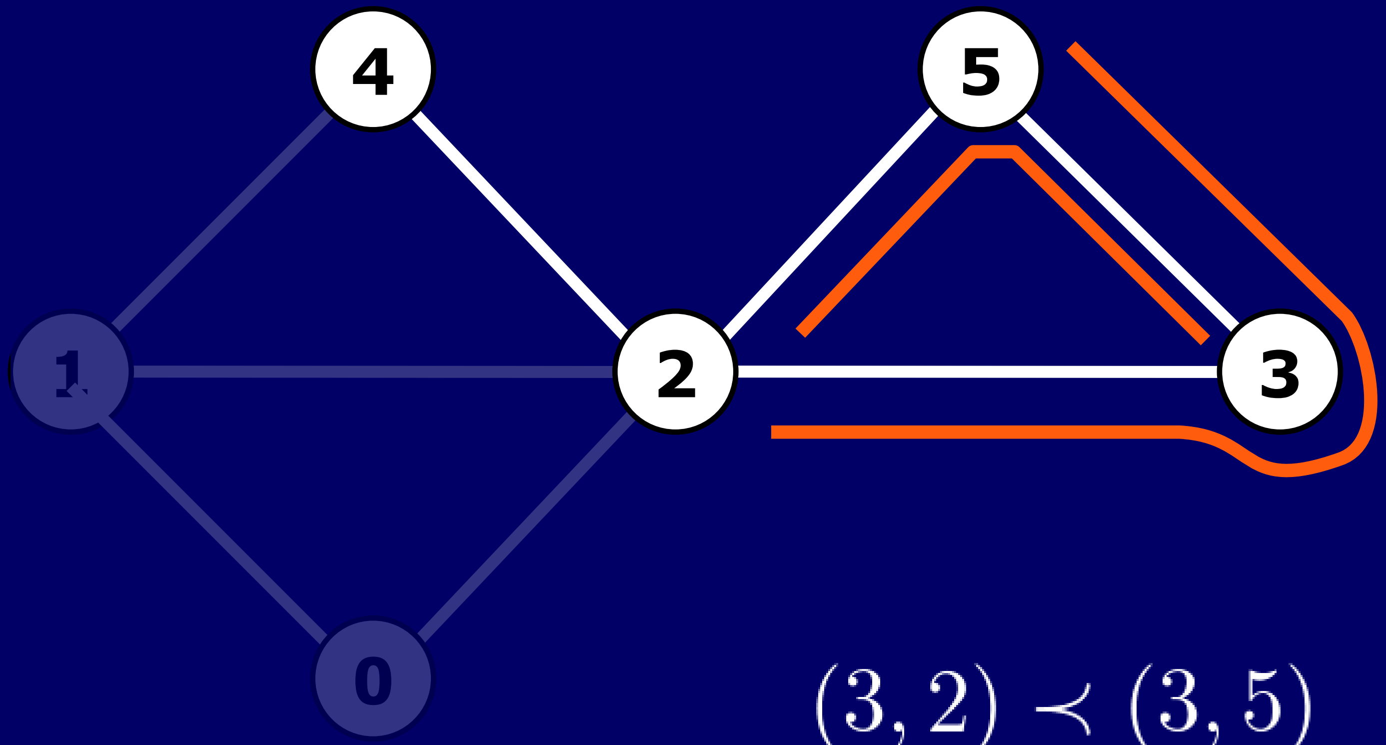
$L_{21}=\{(2,4)\}$

$F_{21}=\{(2,5),(2,3)\}$

$$(2,1) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example

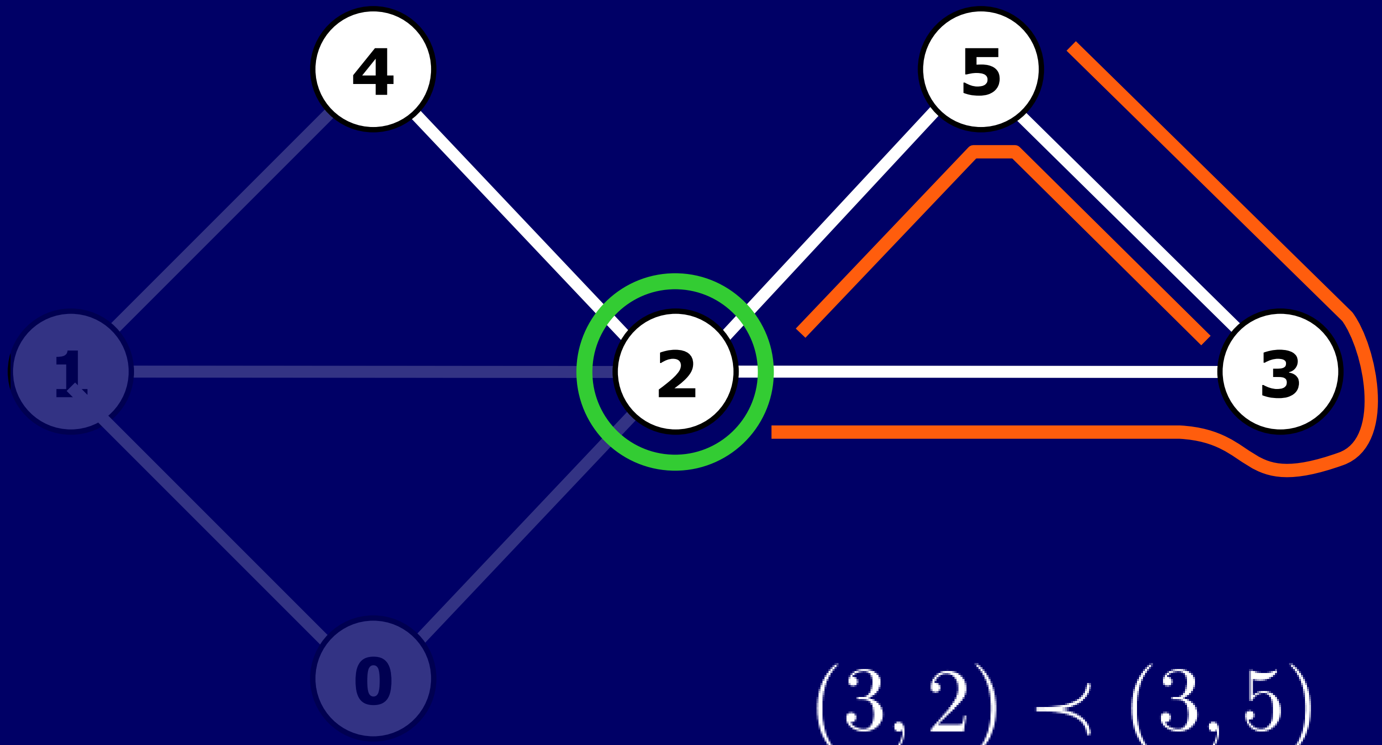
Recursion



$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example

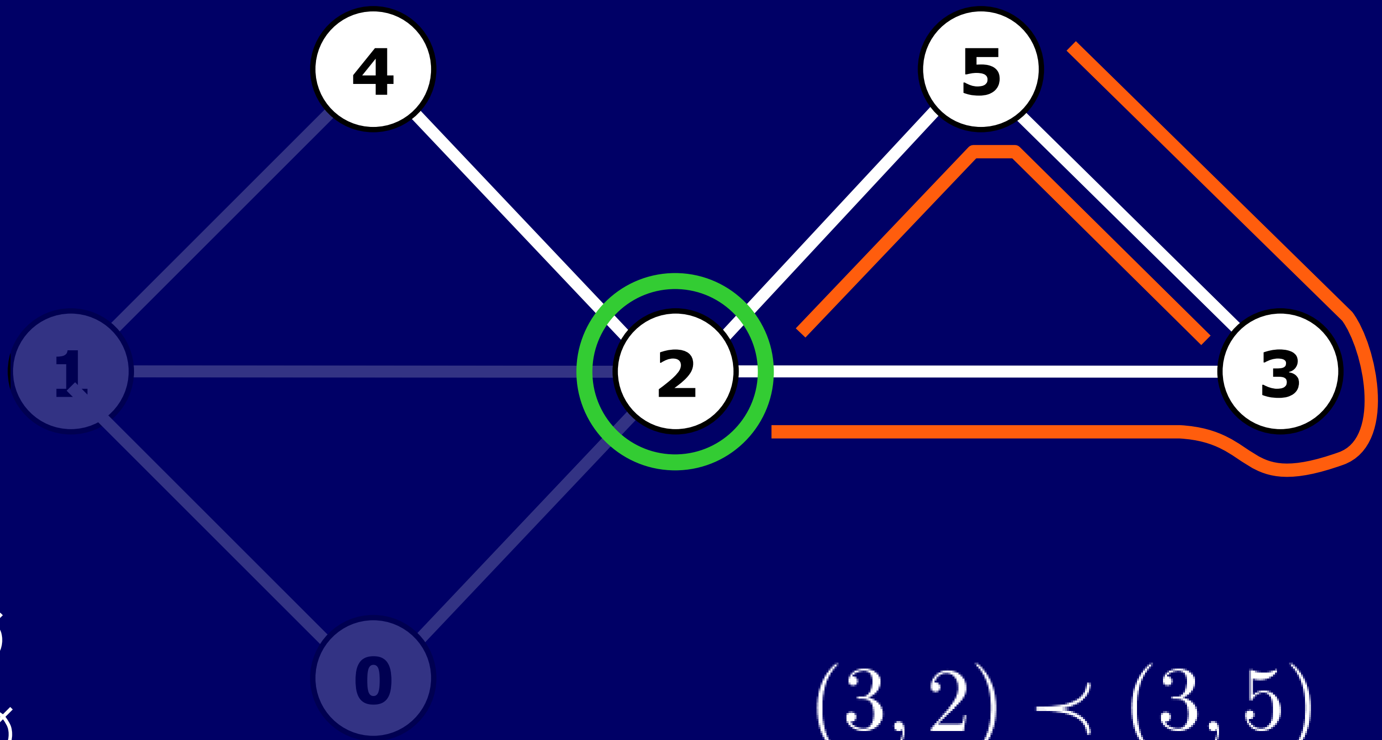
Recursion



$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example

Recursion



$H_{42}=\varnothing$

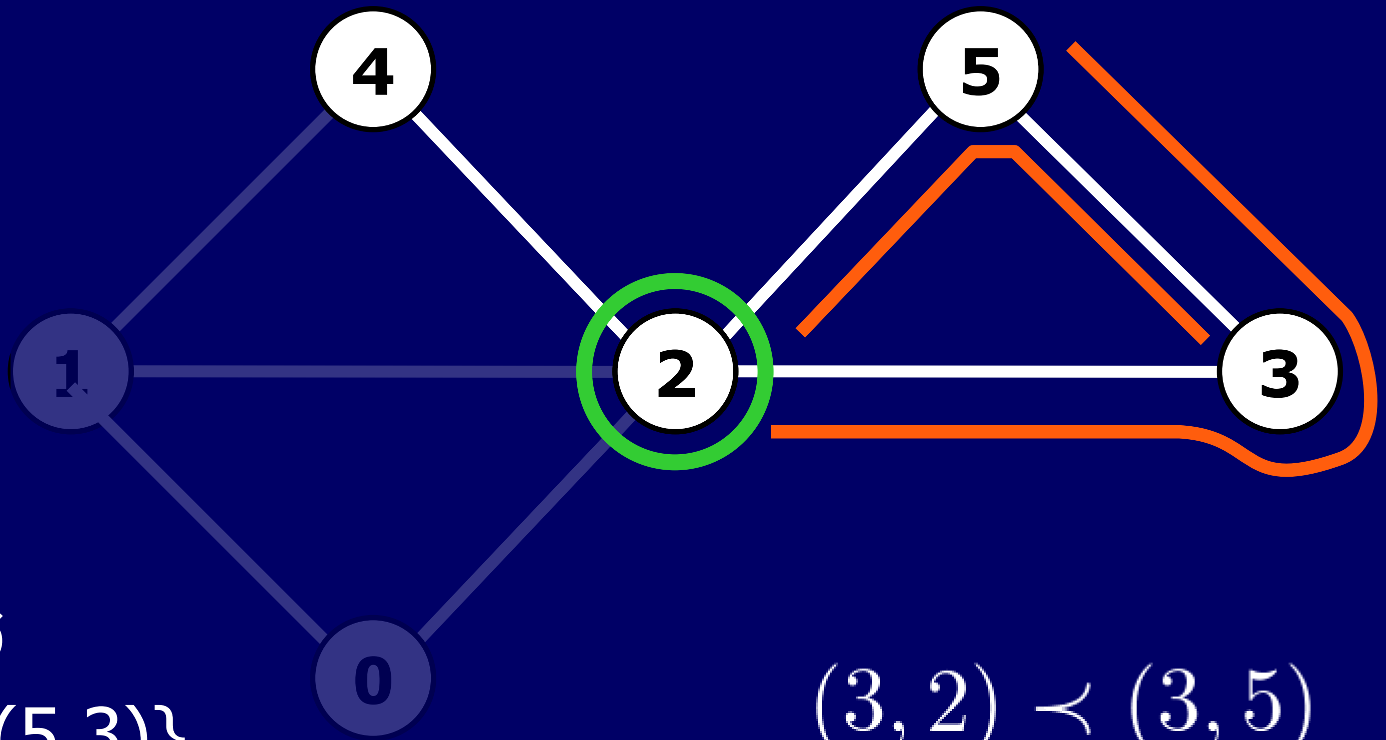
$L_{42}=\varnothing$

$F_{42}=\varnothing$

$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example

Recursion



$H_{52}=\emptyset$

$L_{52}=\{(5,3)\}$

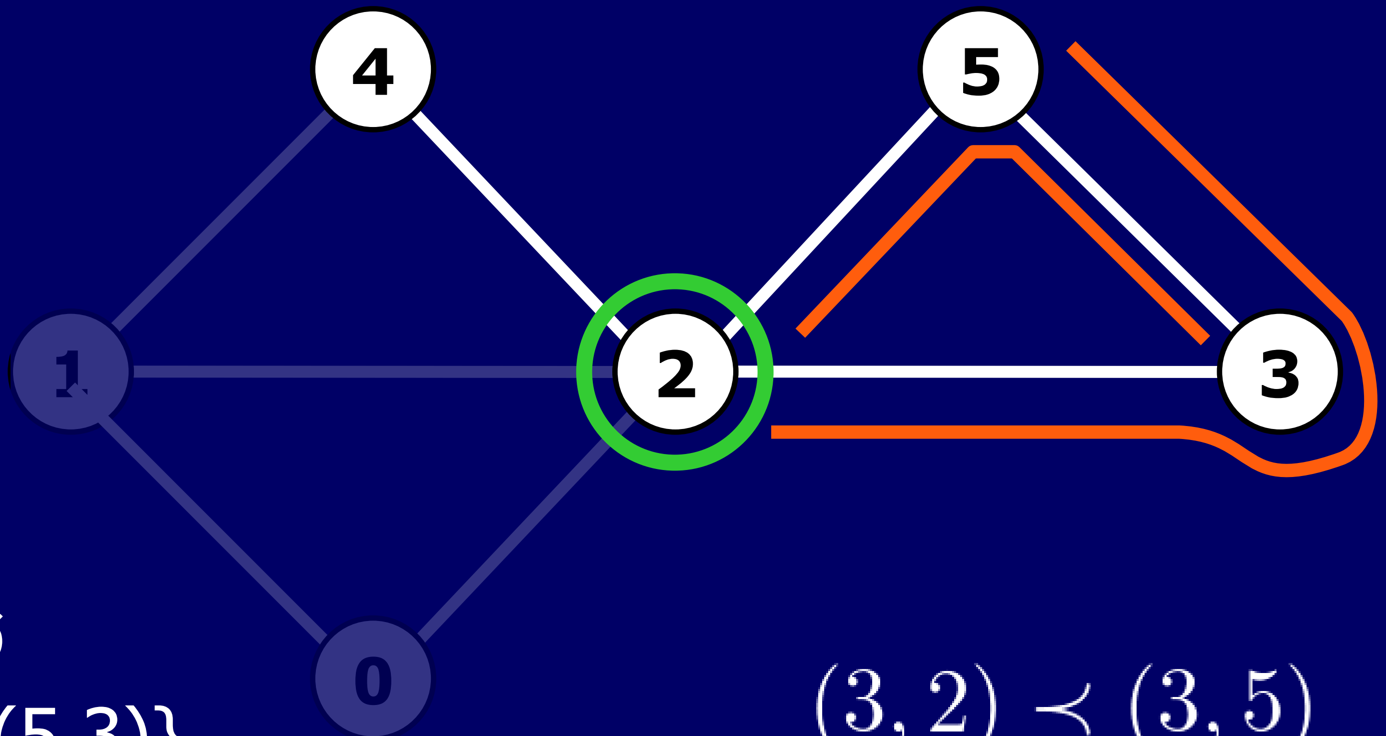
$F_{52}=\{(5,3)\}$

$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example

Recursion

$F_{52} \cap L_{52} \neq \emptyset$!



$H_{52}=\emptyset$

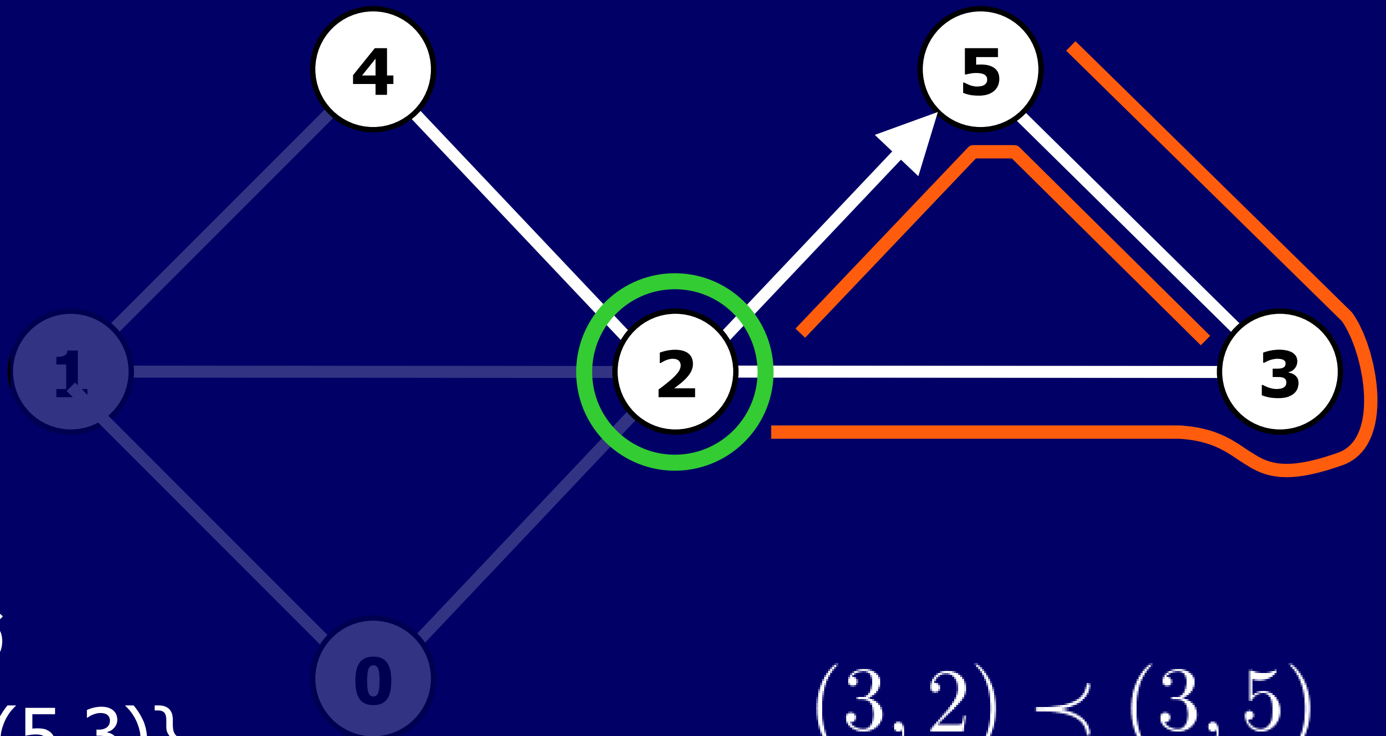
$L_{52}= \{(5,3)\}$

$F_{52}=\{(5,3)\}$

$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example

Recursion



$H_{52}=\varnothing$

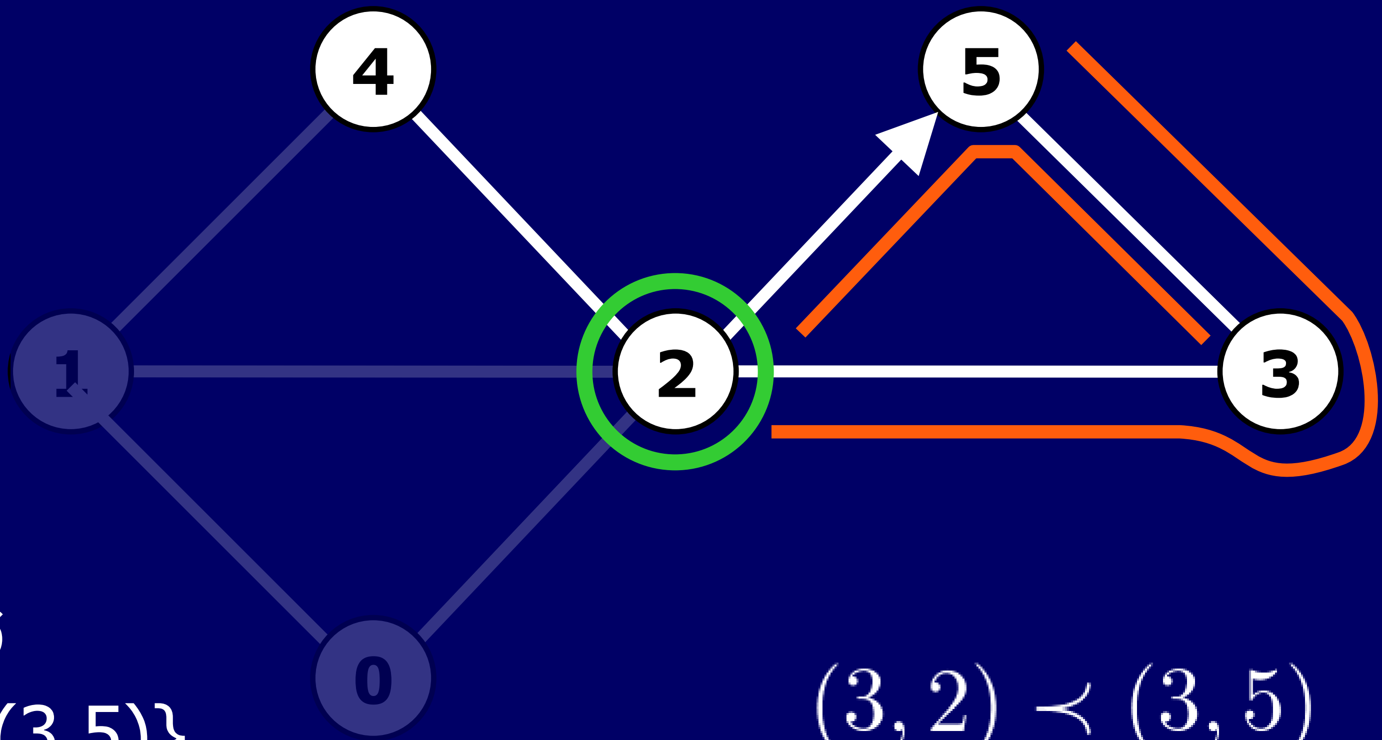
$L_{52}= \{(5,3)\}$

$F_{52}=\{(5,3)\}$

$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example

Recursion



$H_{32}=\varnothing$

$L_{32}= \{(3,5)\}$

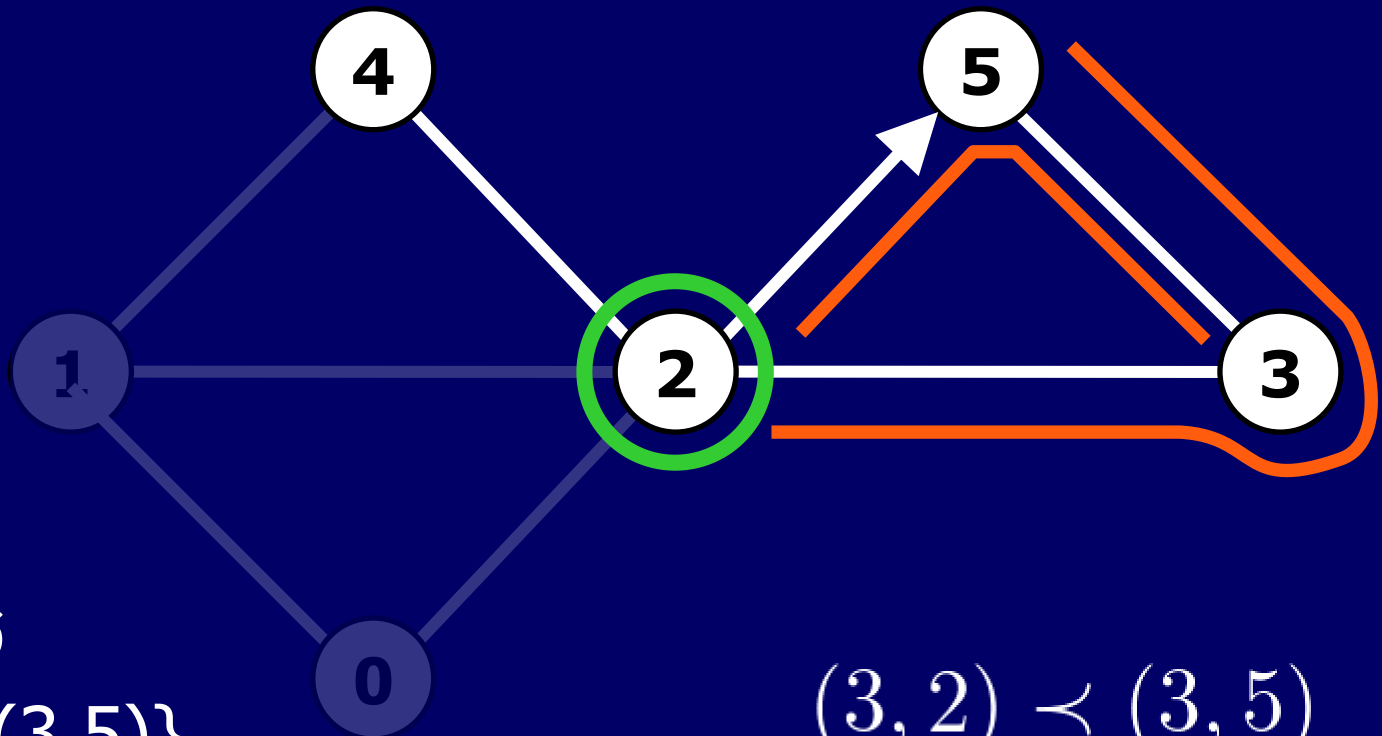
$F_{32}=\{(3,5)\}$

$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example

Recursion

$F_{32} \cap L_{32} \neq \emptyset$!



$H_{32}=\emptyset$

$L_{32}= \{(3,5)\}$

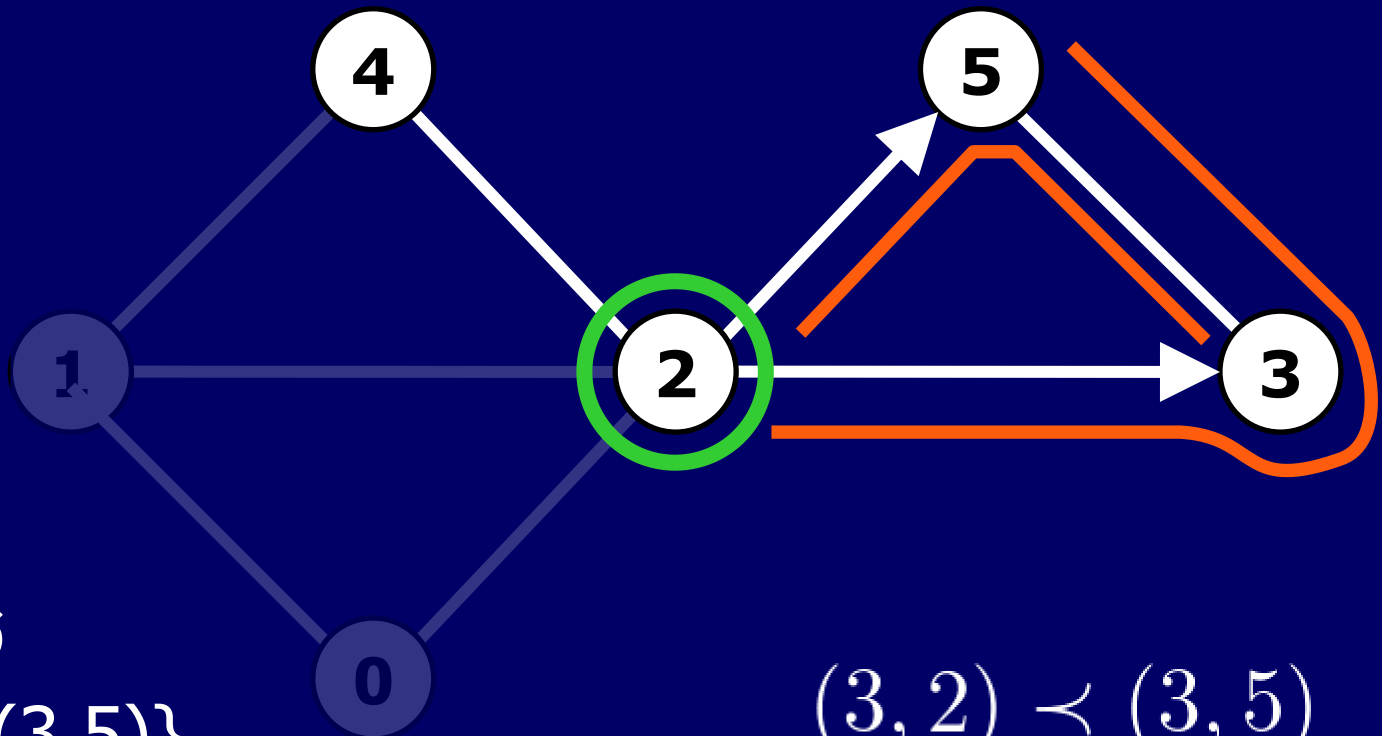
$F_{32}=\{(3,5)\}$

$(3,2) \prec (3,5)$

$(5,2) \prec (5,3)$

Running Example

Recursion



$H_{32}=\varnothing$

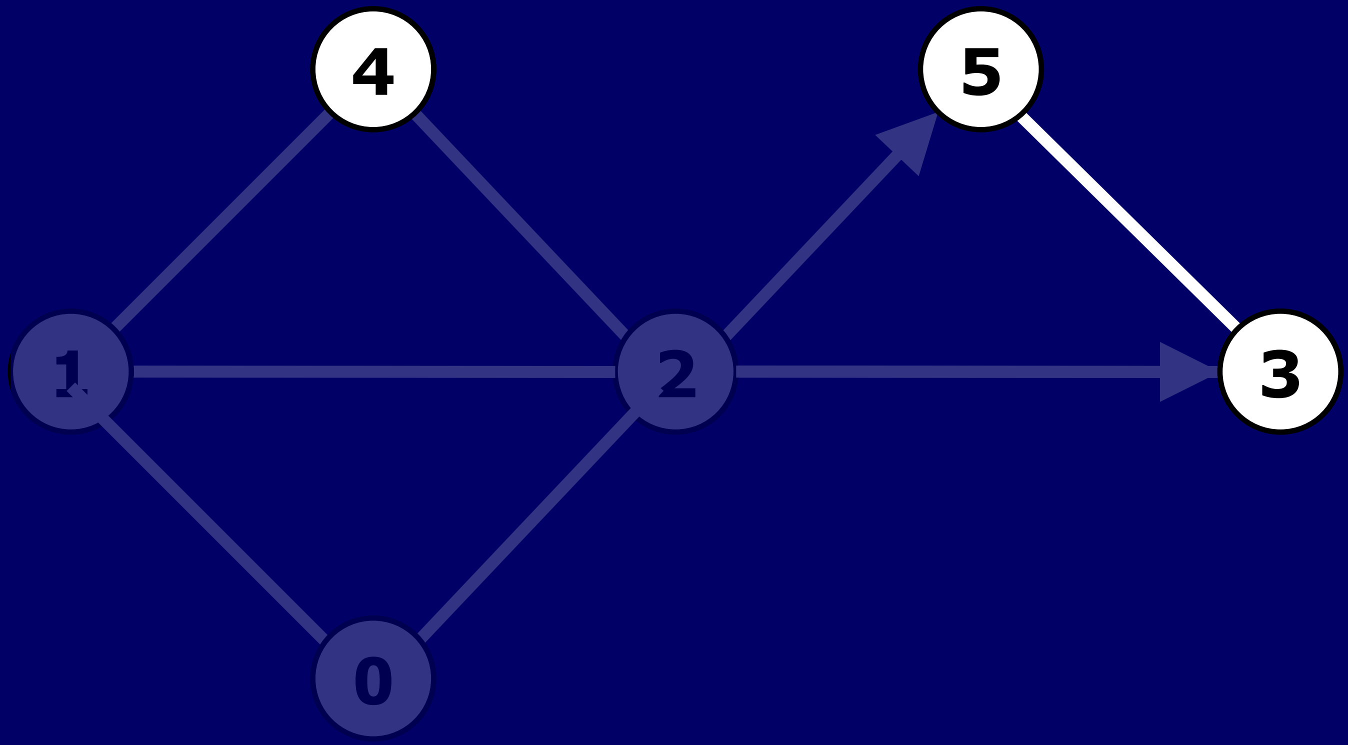
$L_{32}= \{(3,5)\}$

$F_{32}=\{(3,5)\}$

$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

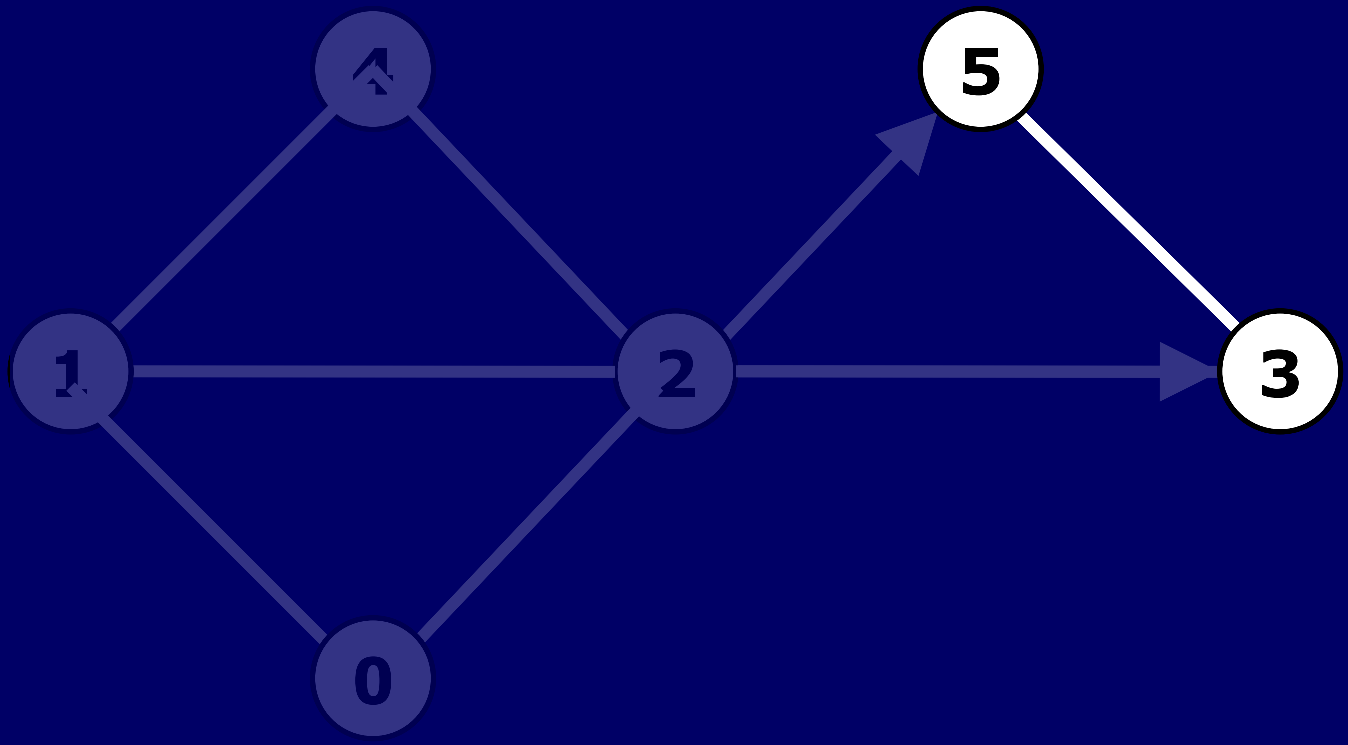
Running Example

Recursion



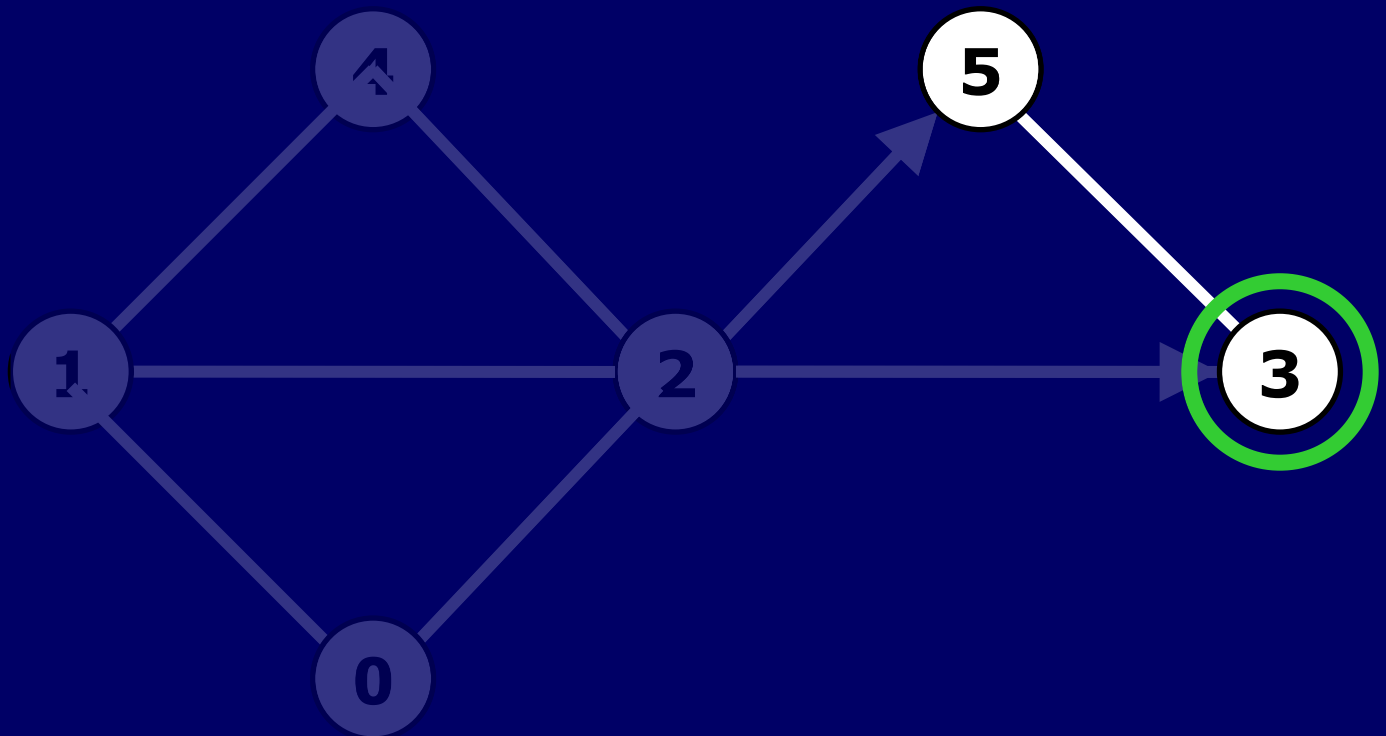
Running Example

Recursion



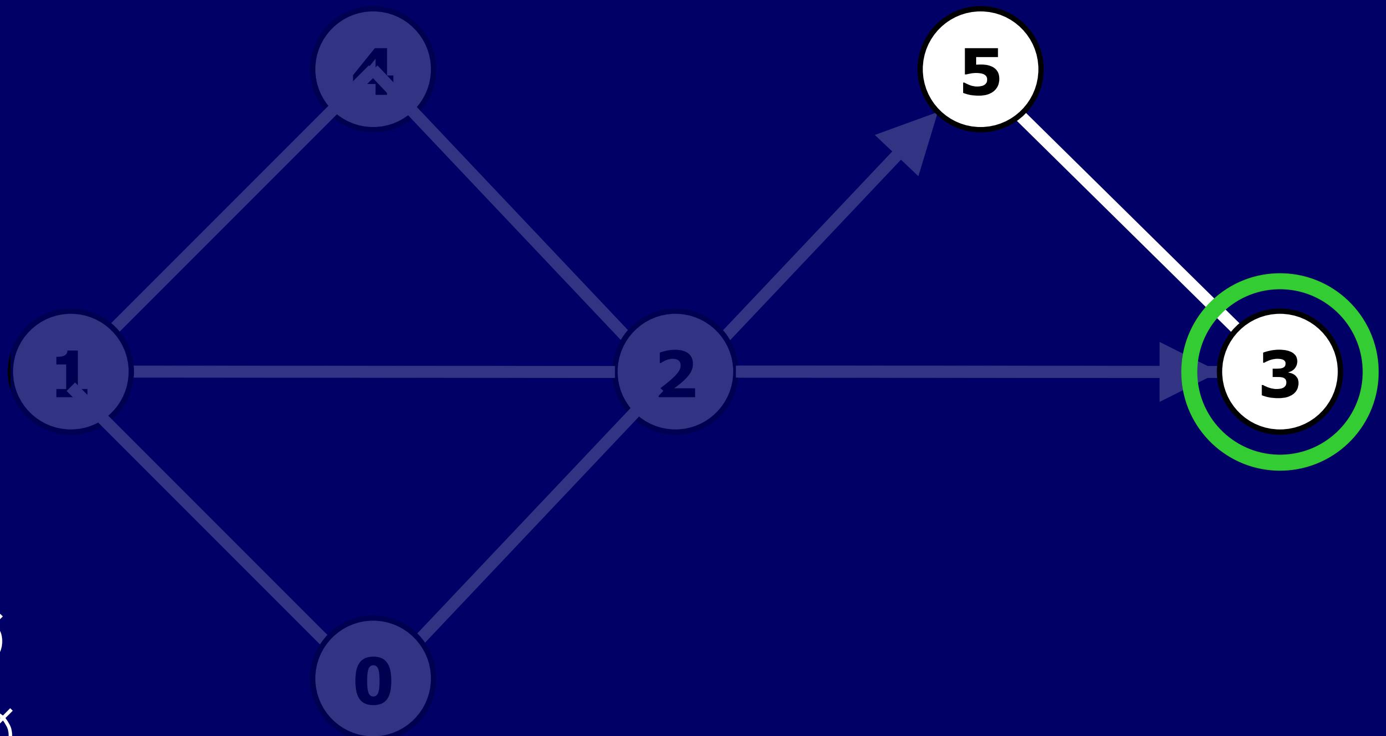
Running Example

Recursion



Running Example

Recursion



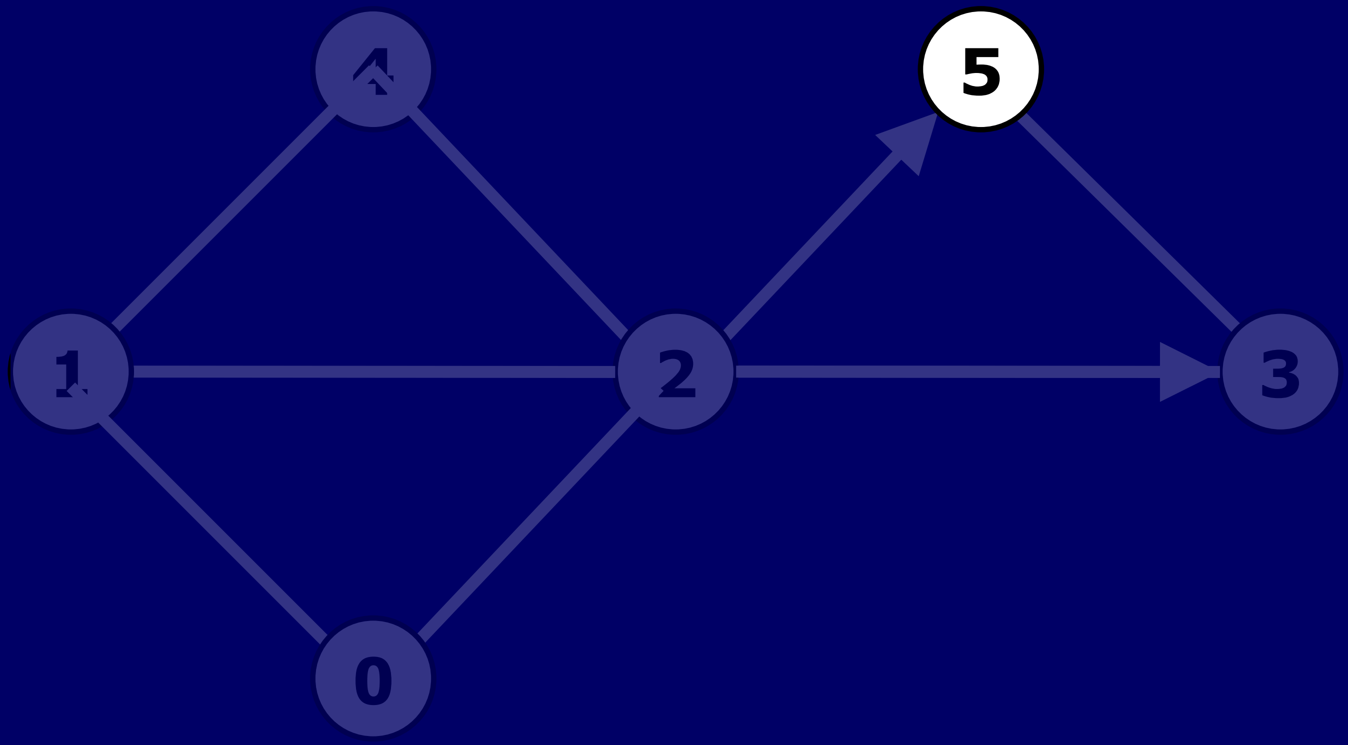
$H_{53} = \varnothing$

$L_{53} = \varnothing$

$F_{53} = \varnothing$

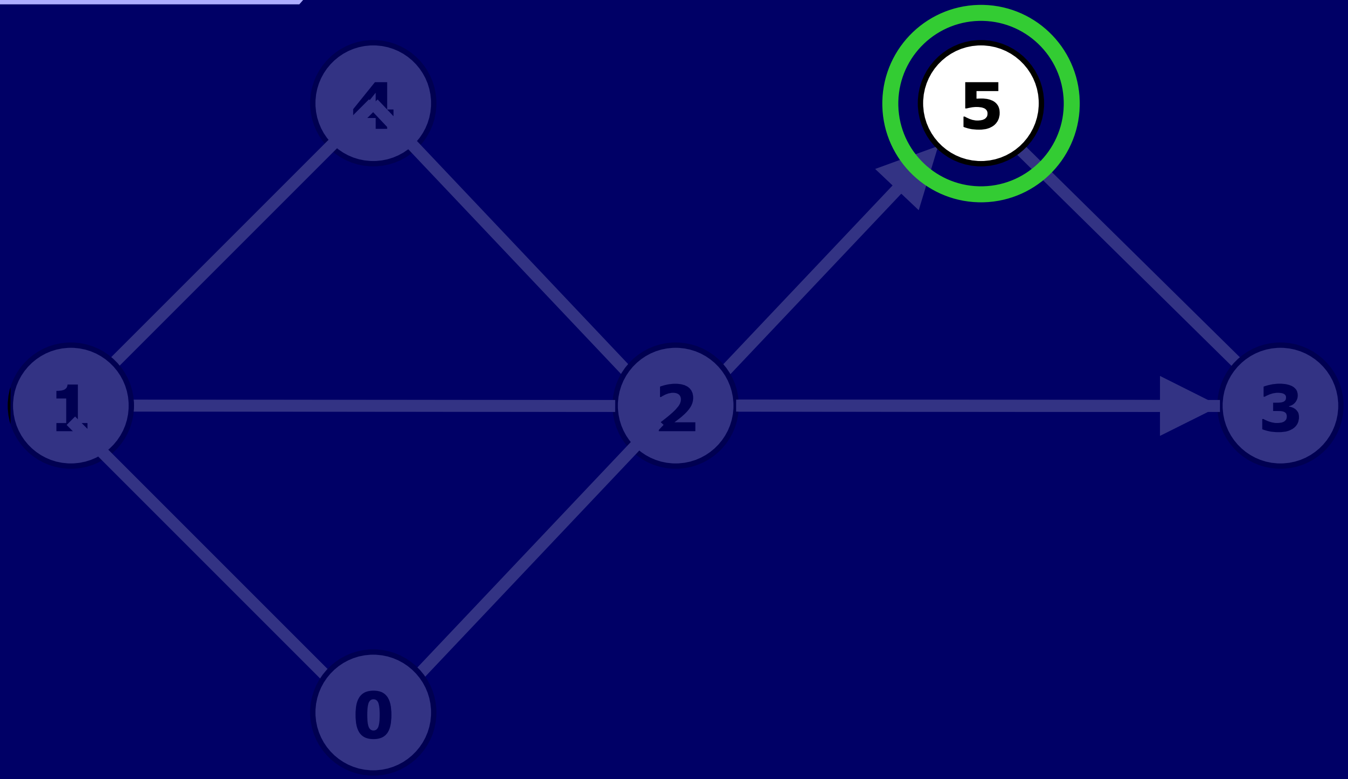
Running Example

Recursion



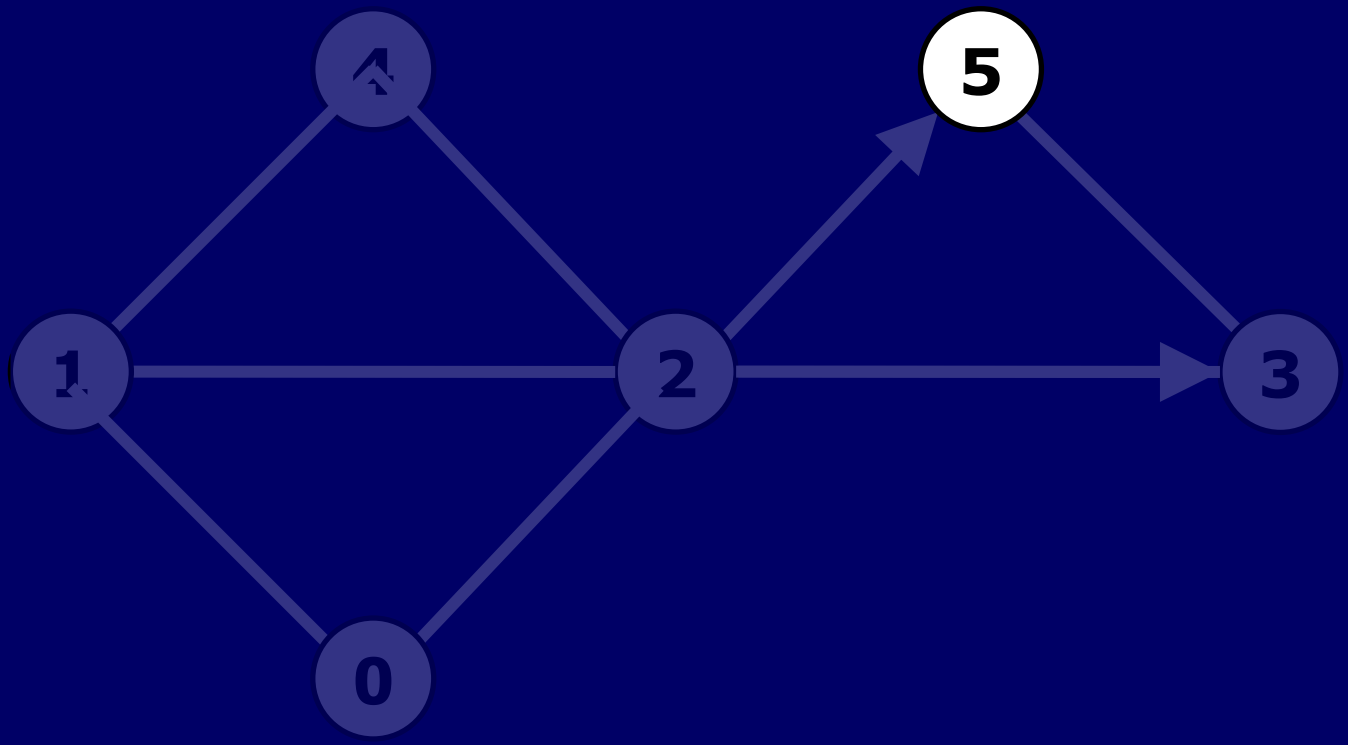
Running Example

Recursion



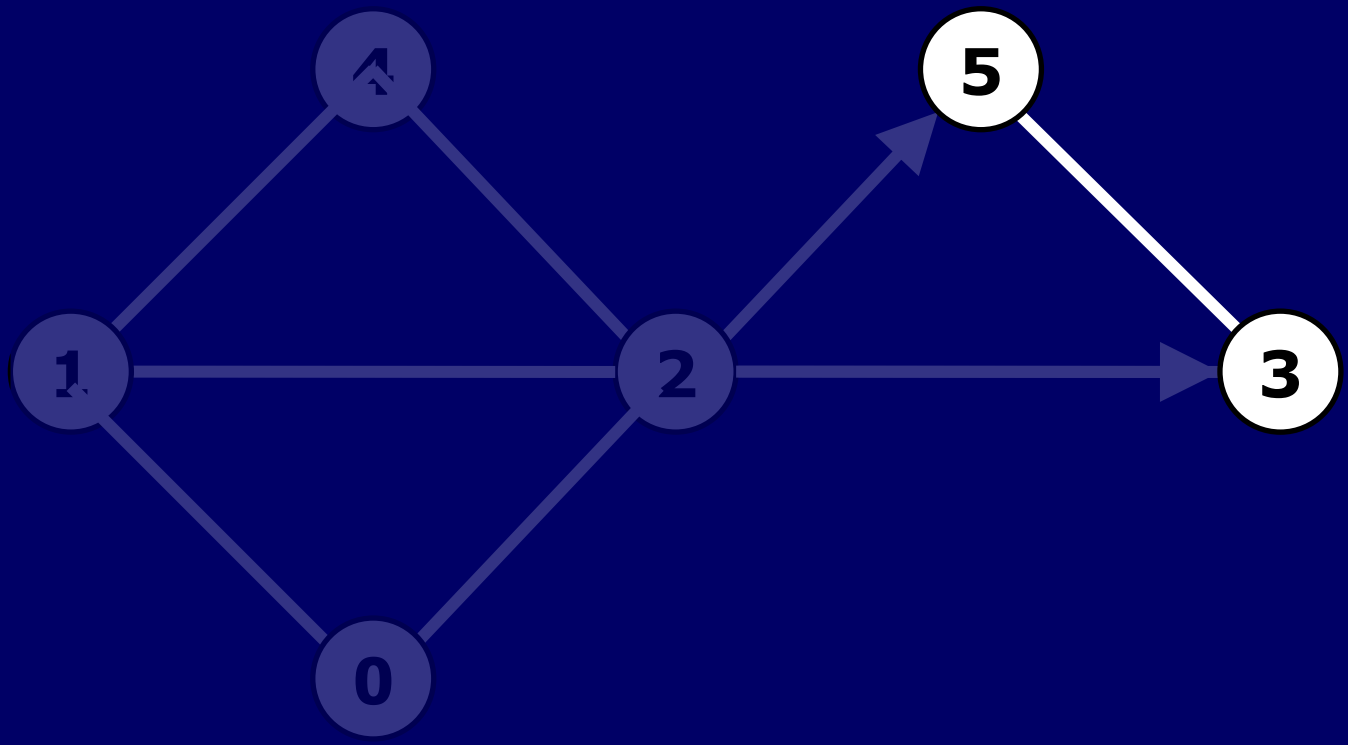
Running Example

Back from recursion



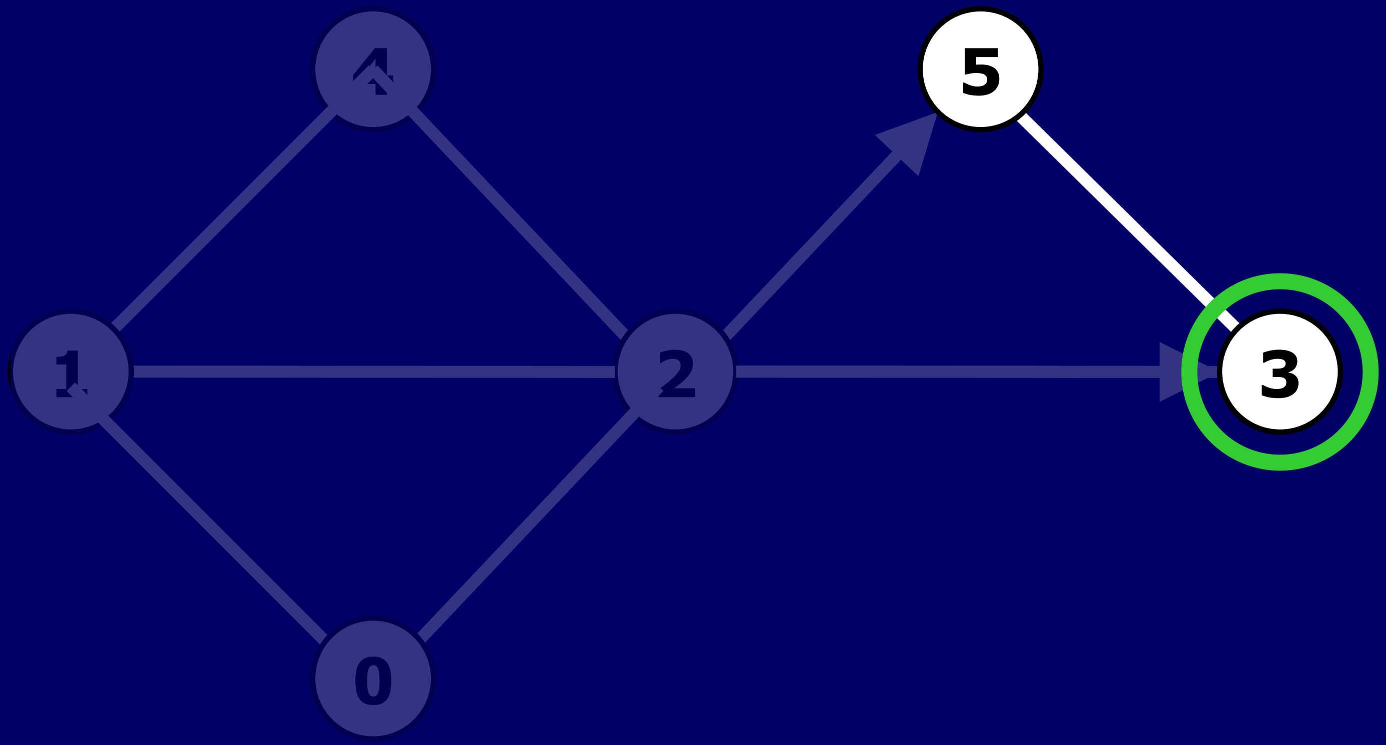
Running Example

Back from recursion



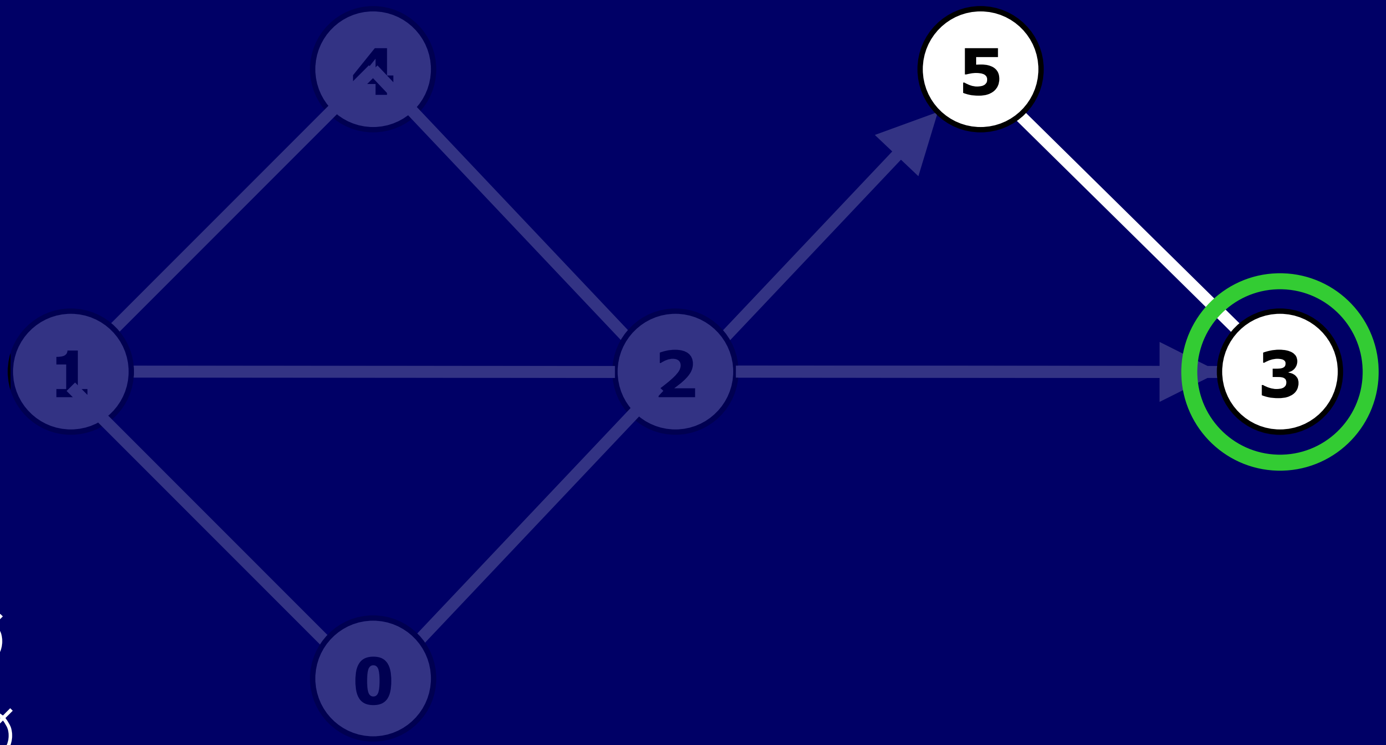
Running Example

Back from recursion



Running Example

Back from recursion



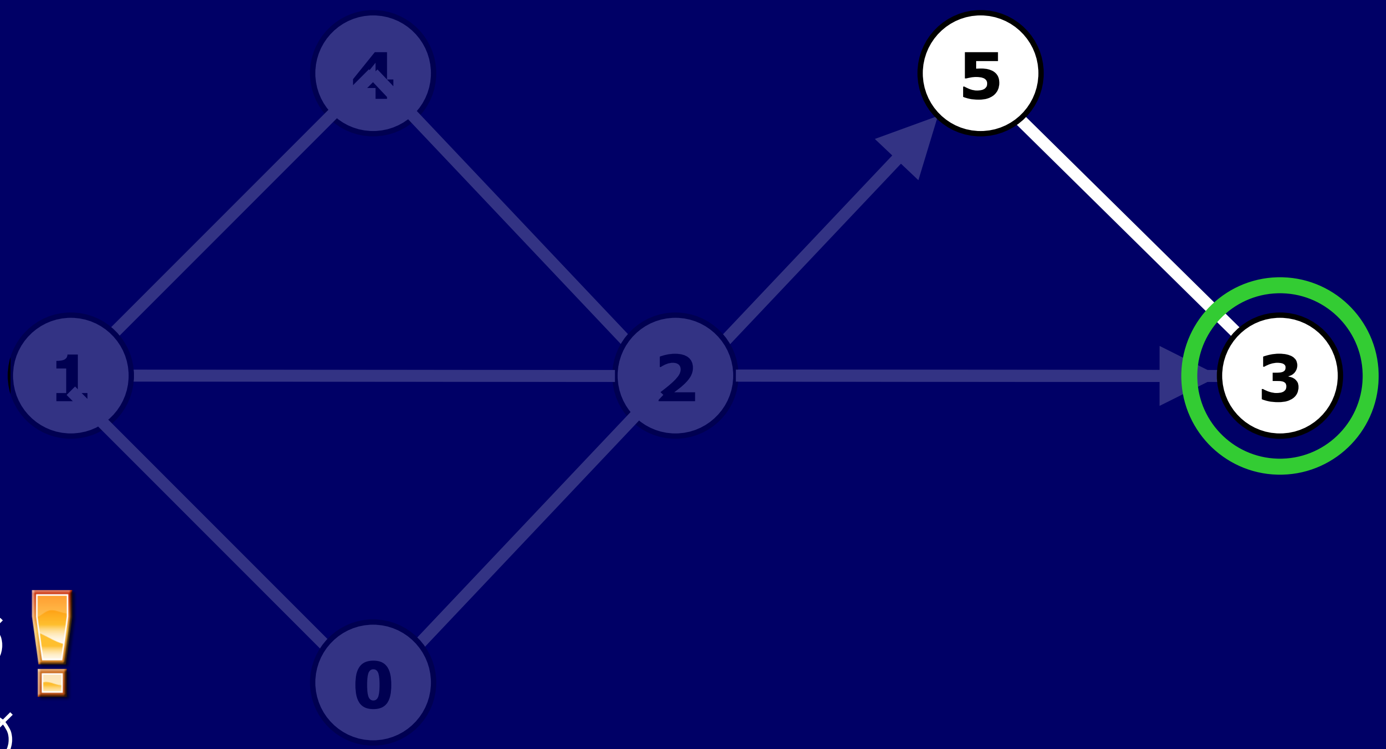
$H_{53} = \emptyset$

$L_{53} = \emptyset$

$F_{53} = \emptyset$

Running Example

Back from recursion



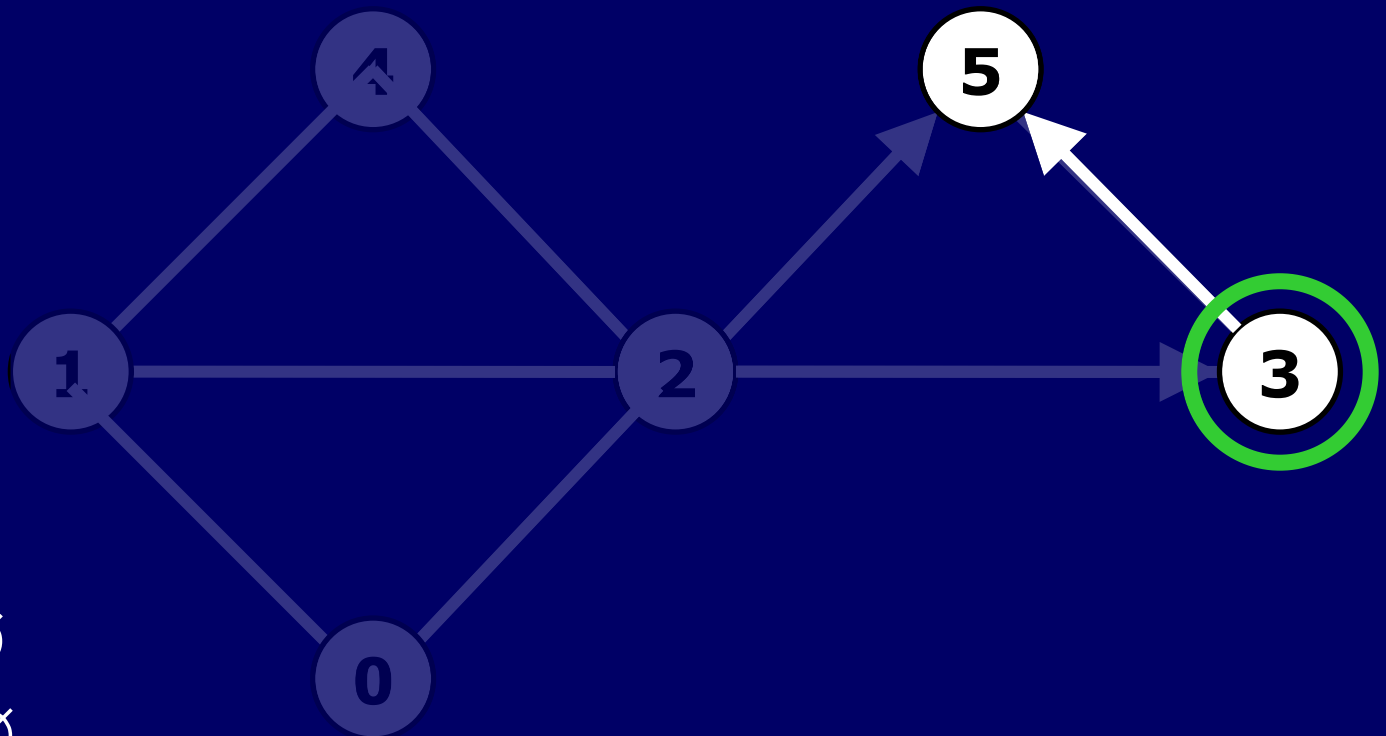
$H_{53} = \varnothing$

$L_{53} = \varnothing$

$F_{53} = \varnothing$

Running Example

Back from recursion



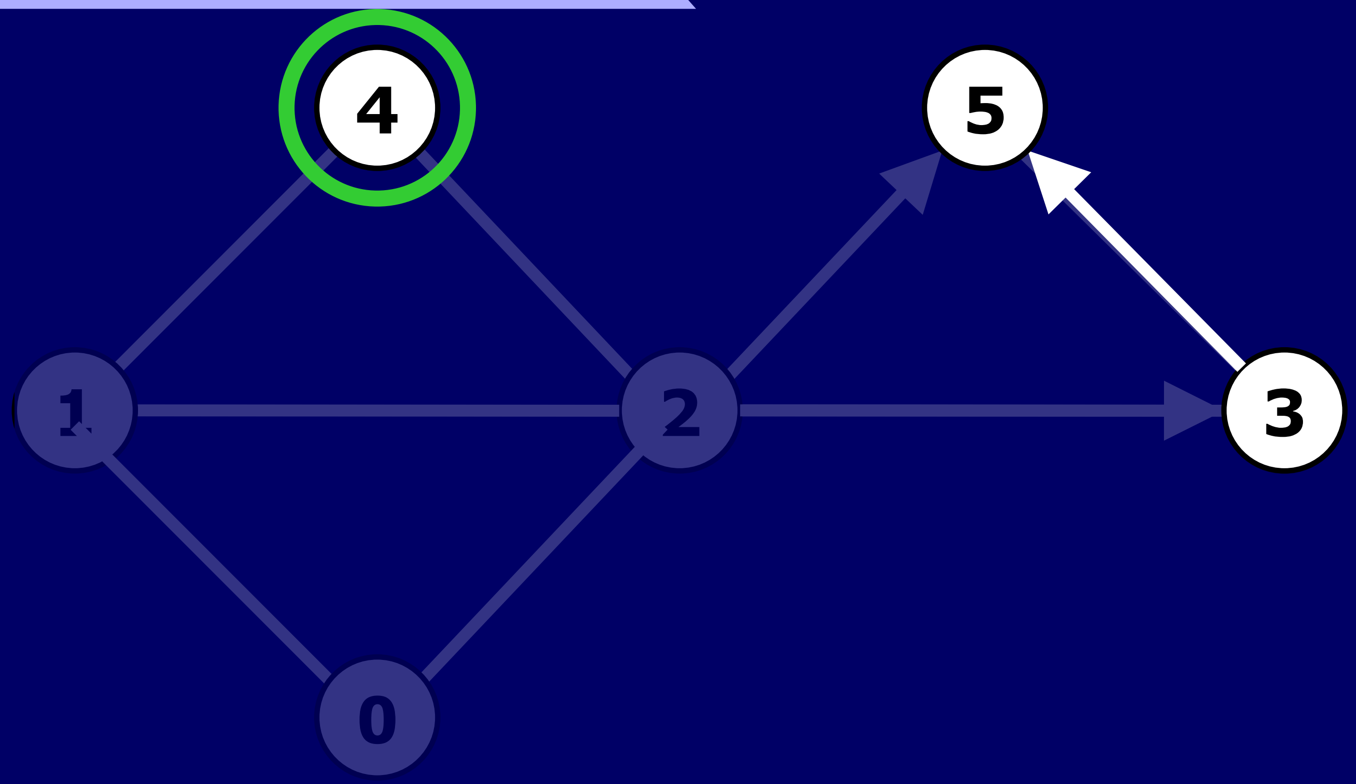
$H_{53} = \emptyset$

$L_{53} = \emptyset$

$F_{53} = \emptyset$

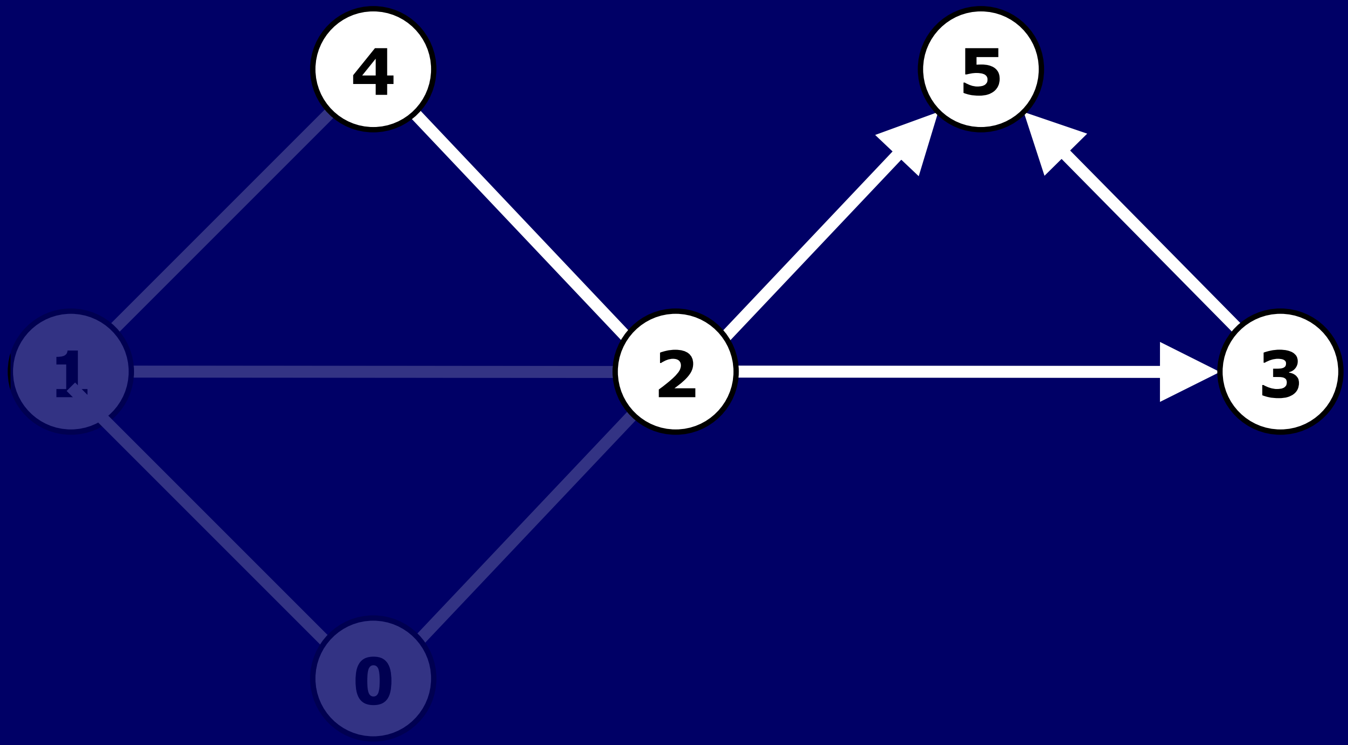
Running Example

Back from recursion



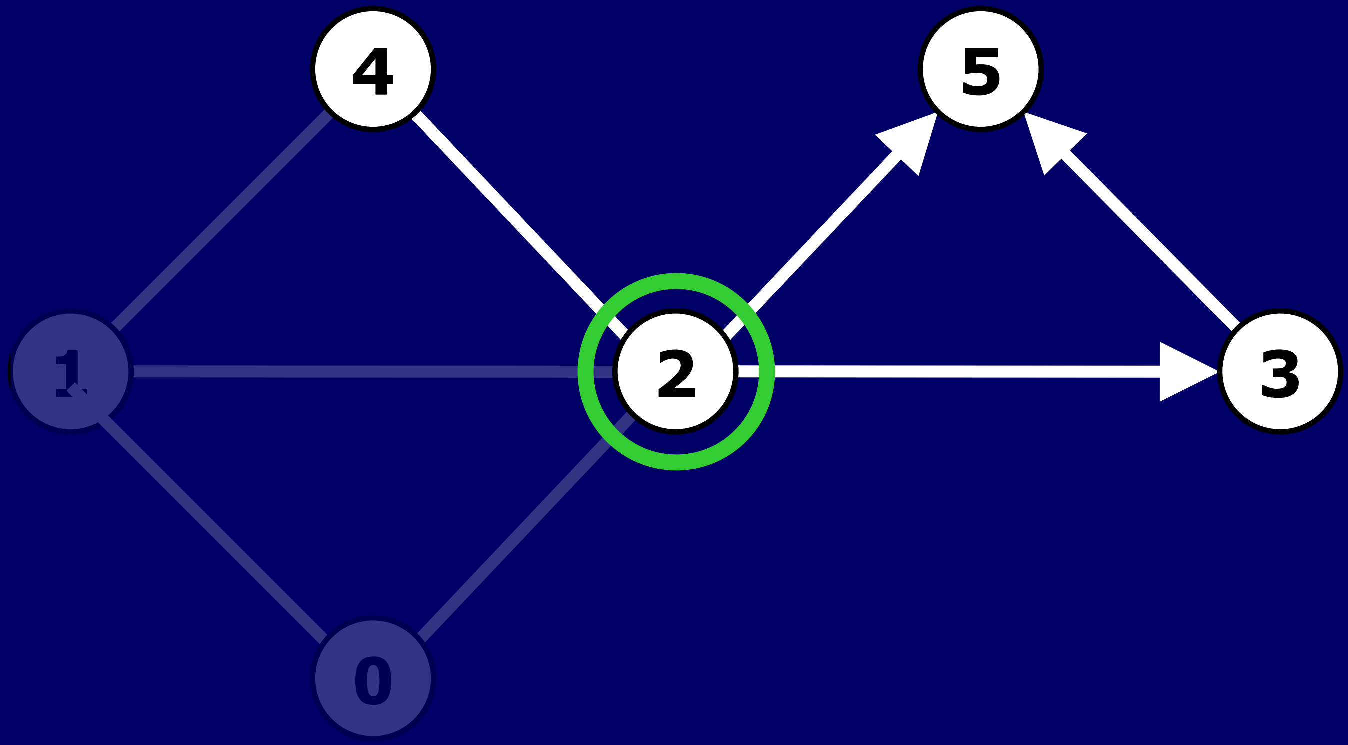
Running Example

Back from recursion



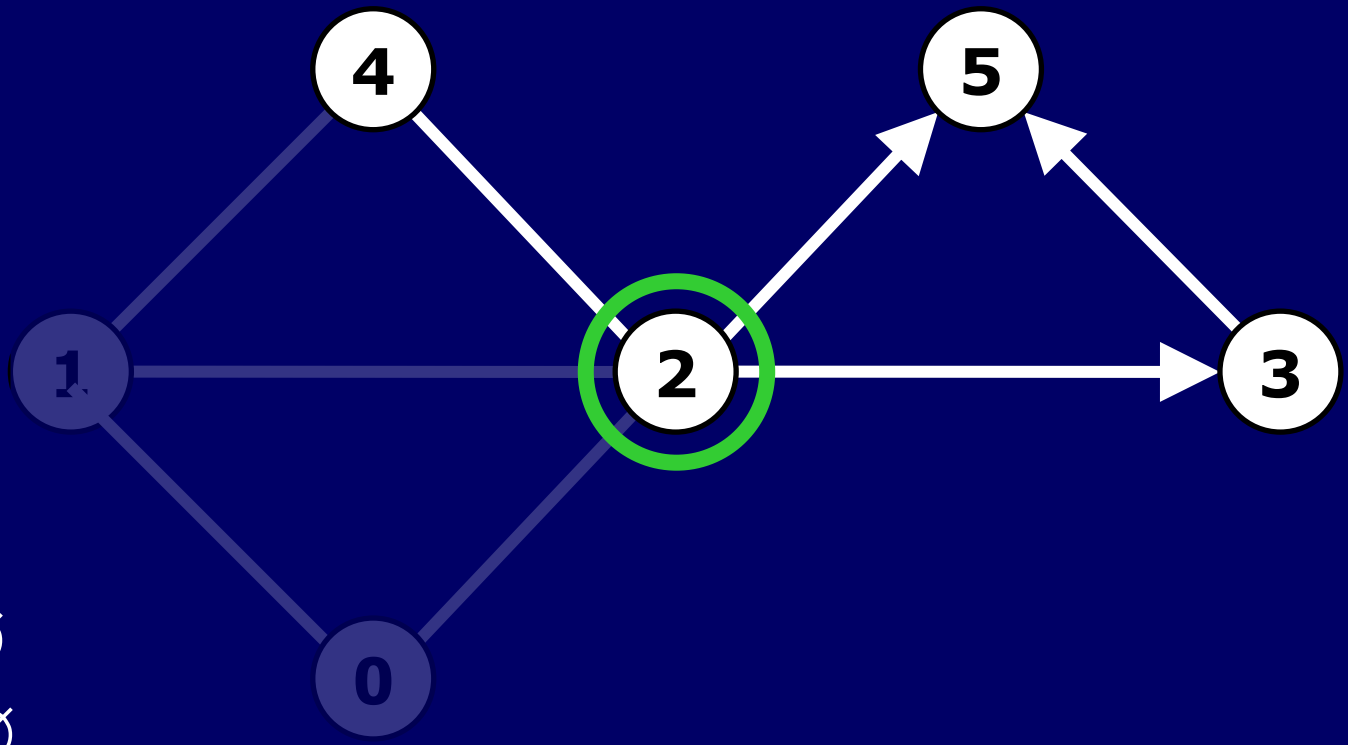
Running Example

Back from recursion



Running Example

Back from recursion



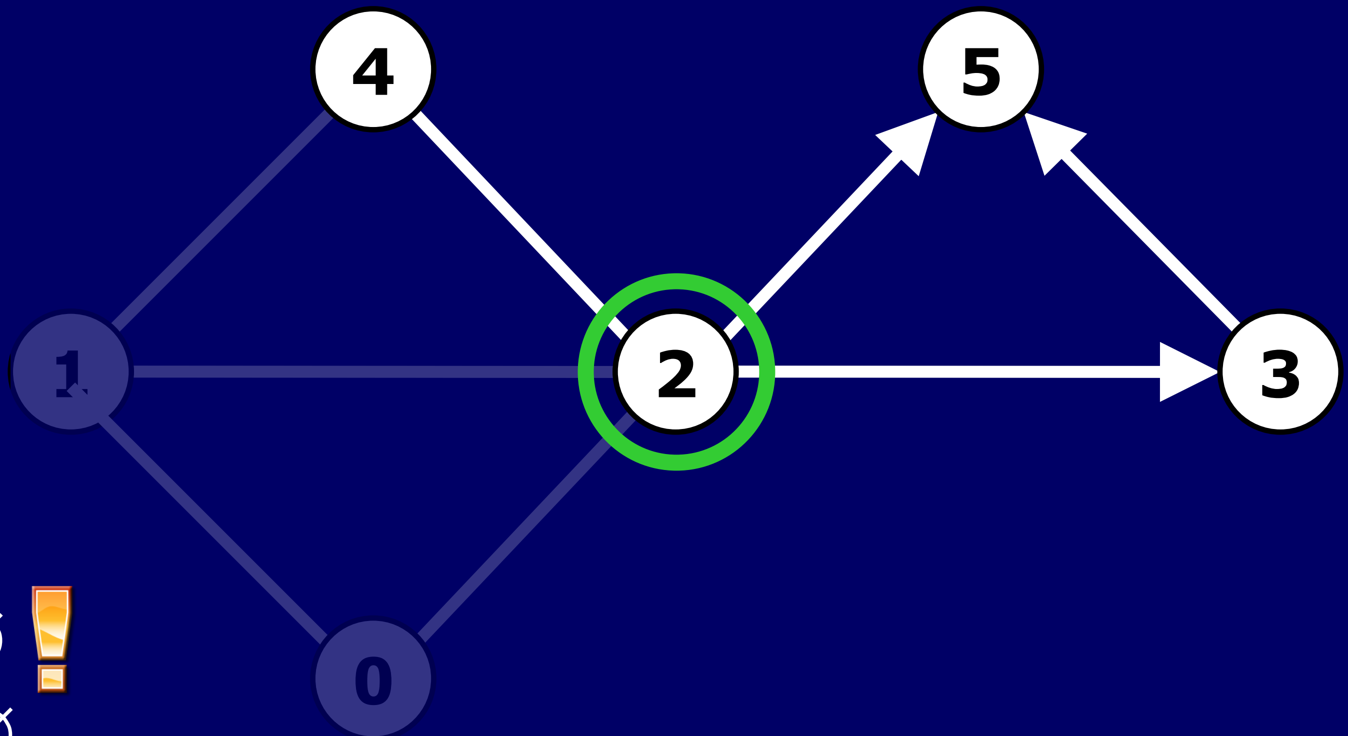
$H_{42}=\varnothing$

$L_{42}= \varnothing$

$F_{42}=\varnothing$

Running Example

Back from recursion



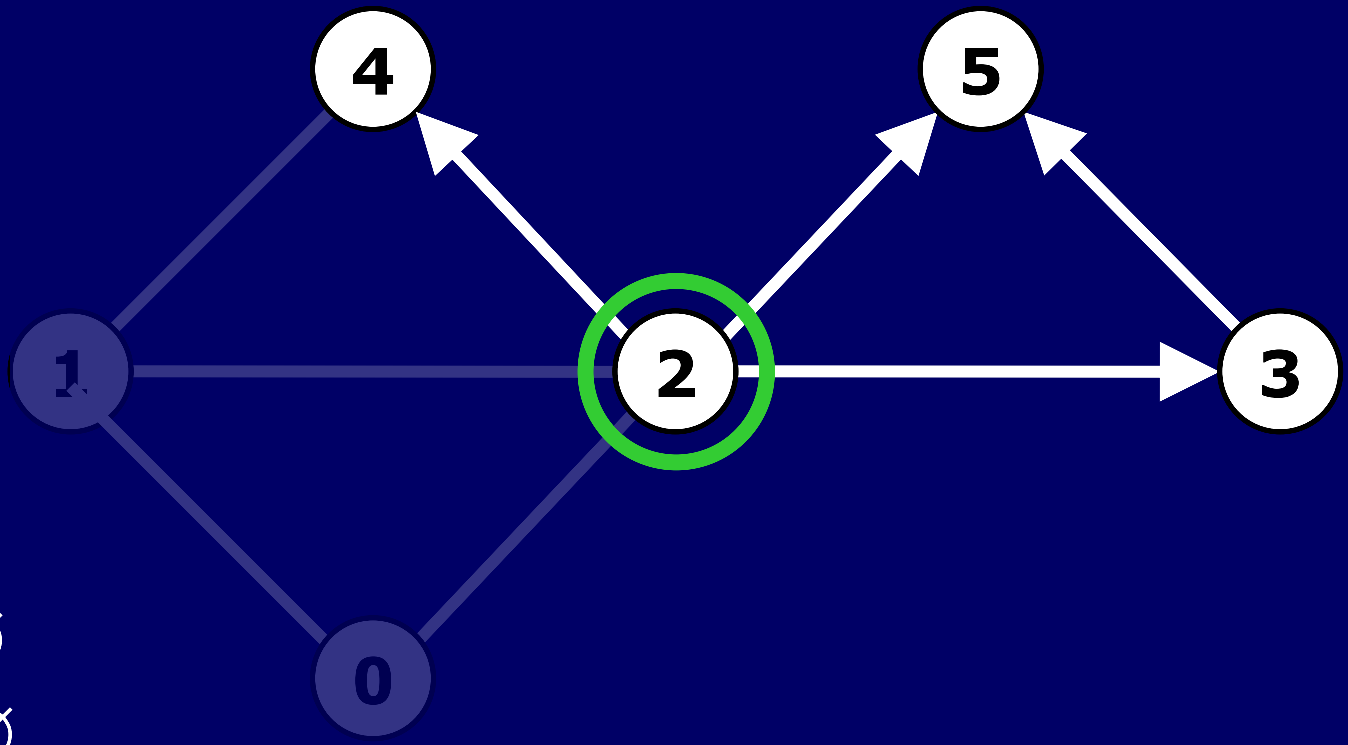
$H_{42} = \varnothing$

$L_{42} = \varnothing$

$F_{42} = \varnothing$

Running Example

Back from recursion



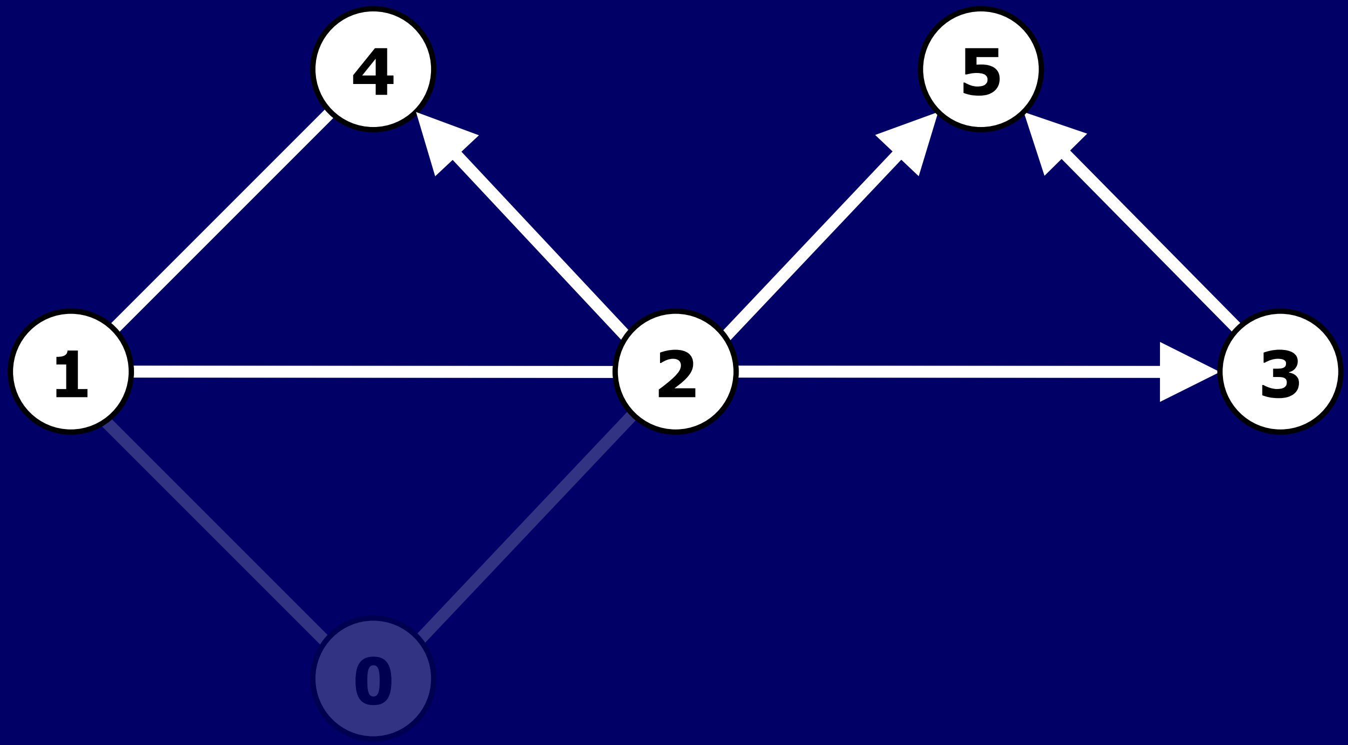
$H_{42}=\varnothing$

$L_{42}=\varnothing$

$F_{42}=\varnothing$

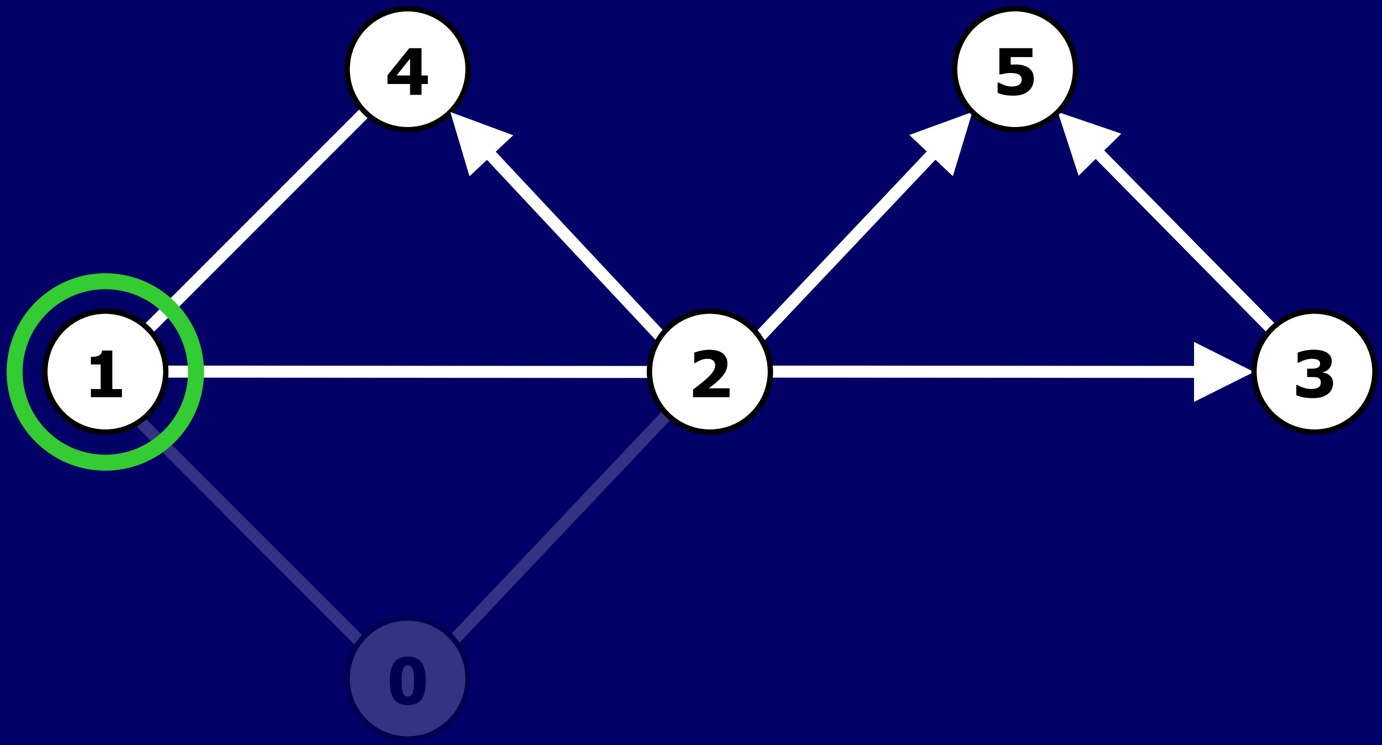
Running Example

Back from recursion



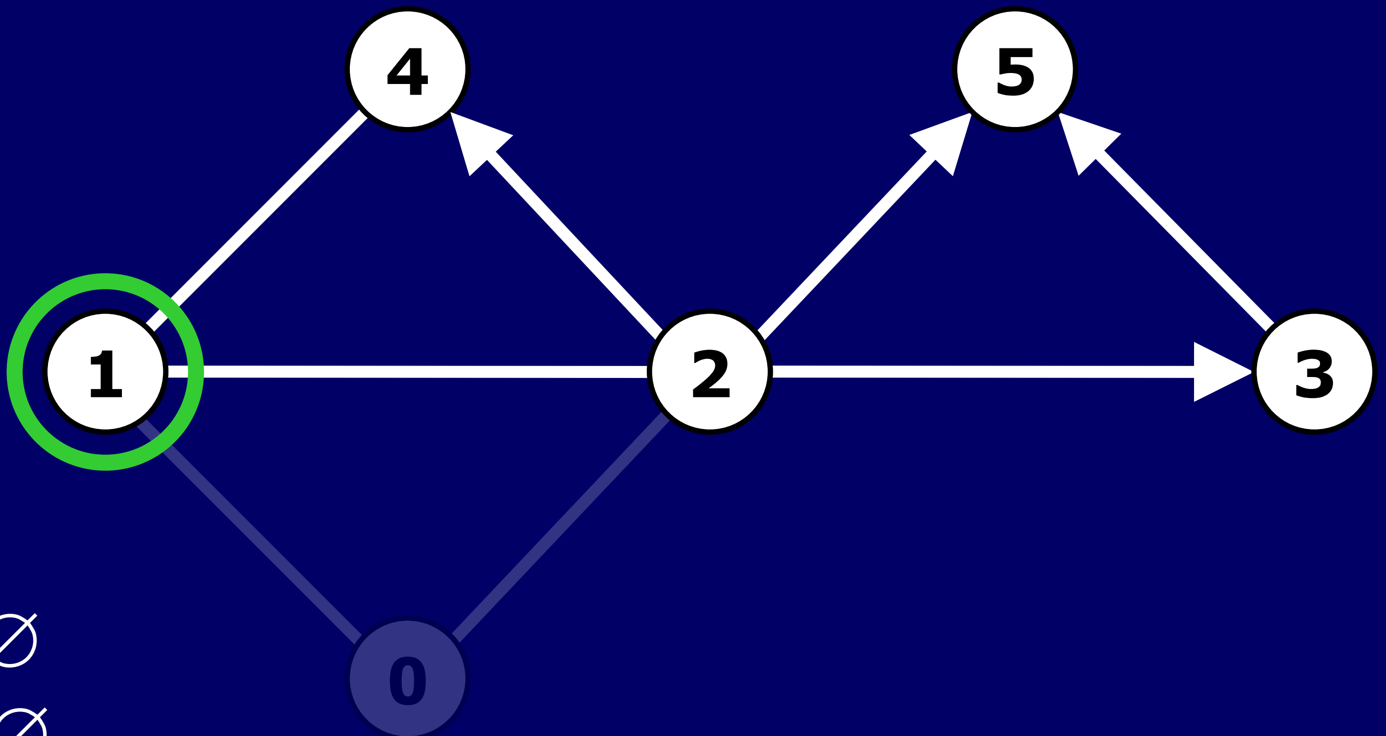
Running Example

Back from recursion



Running Example

Back from recursion



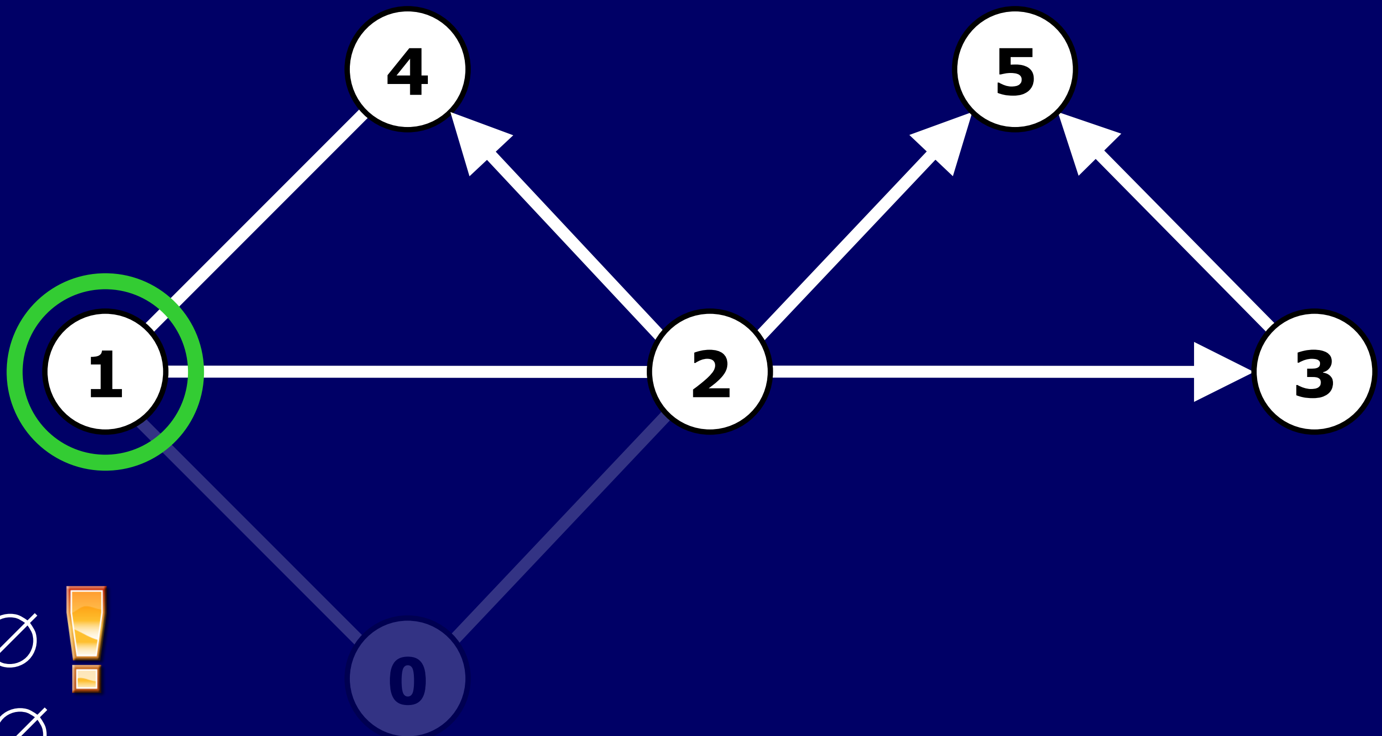
$H_{41}=\varnothing$

$L_{41}= \varnothing$

$F_{41}=\{(4,2)\}$

Running Example

Back from recursion



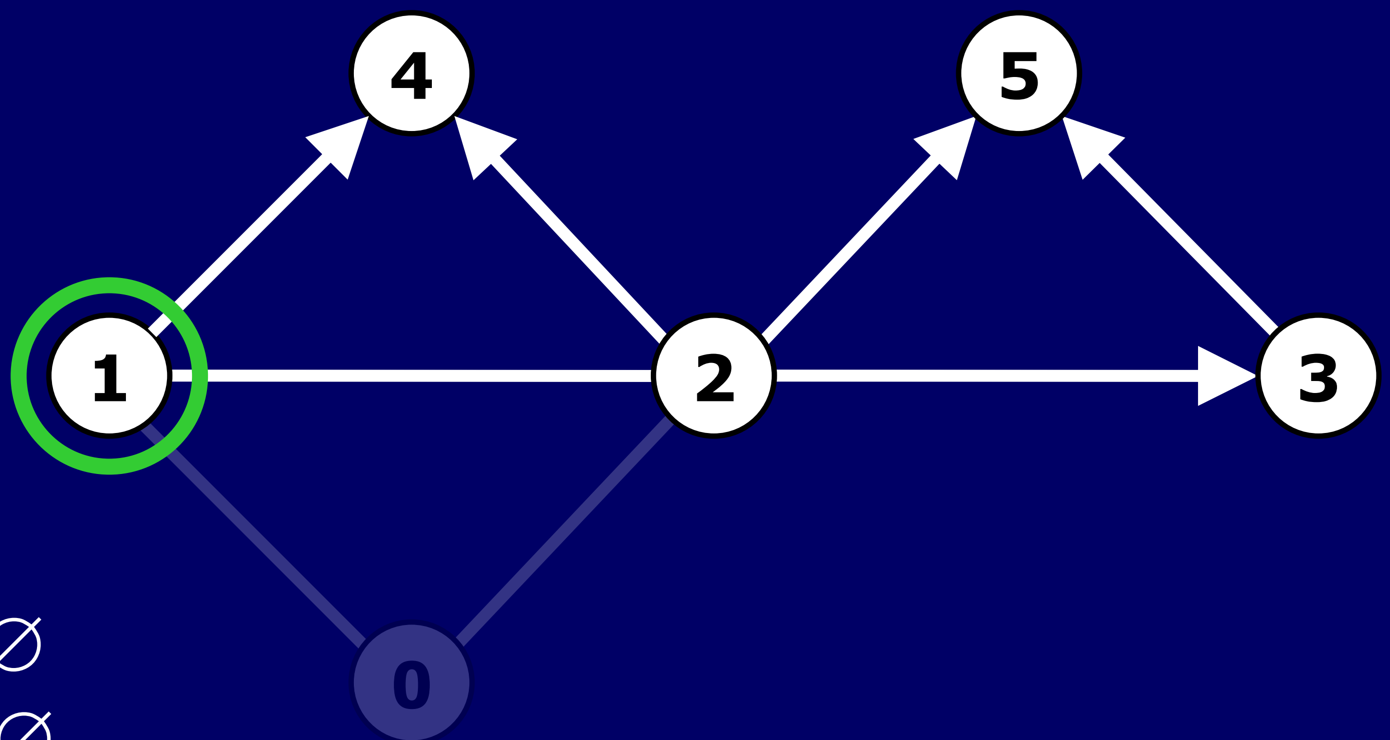
$H_{41} = \varnothing$!

$L_{41} = \varnothing$

$F_{41} = \{(4,2)\}$

Running Example

Back from recursion



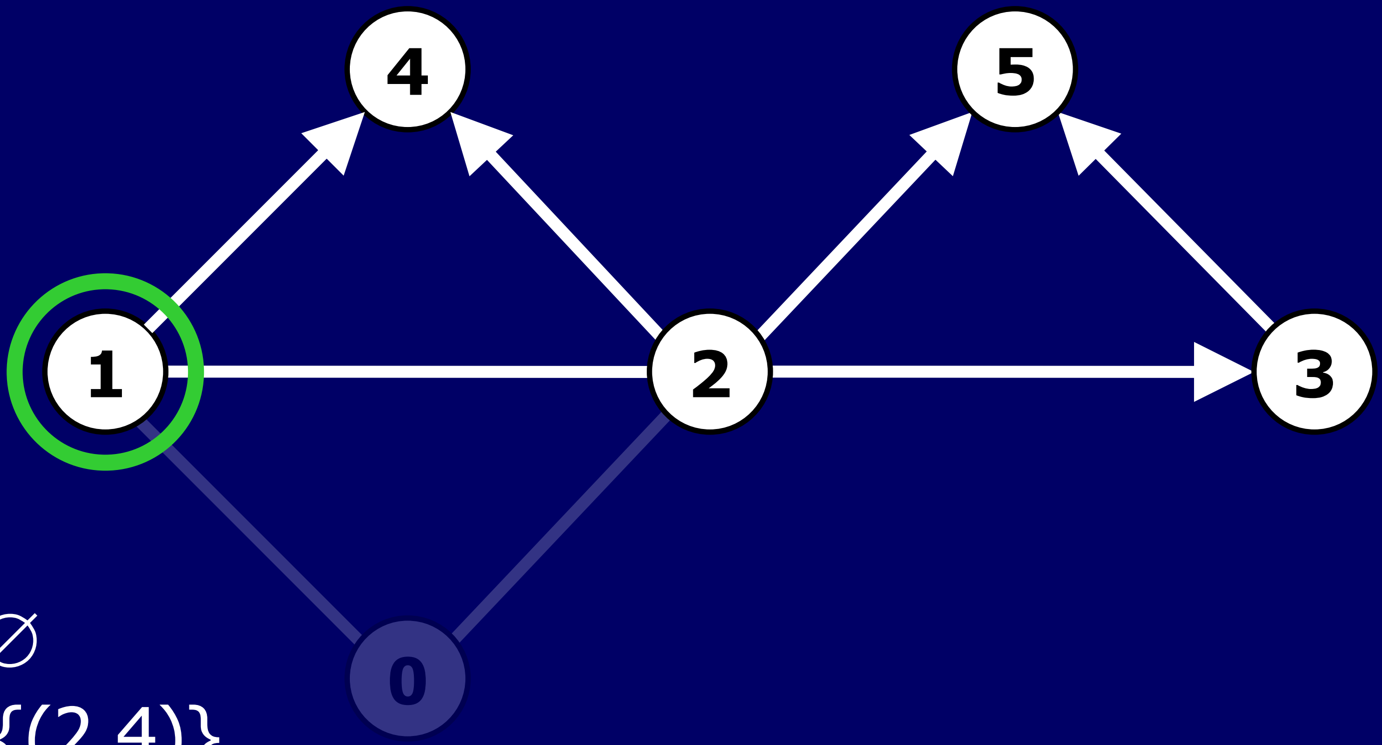
$H_{41} = \emptyset$

$L_{41} = \emptyset$

$F_{41} = \{(4,2)\}$

Running Example

Back from recursion



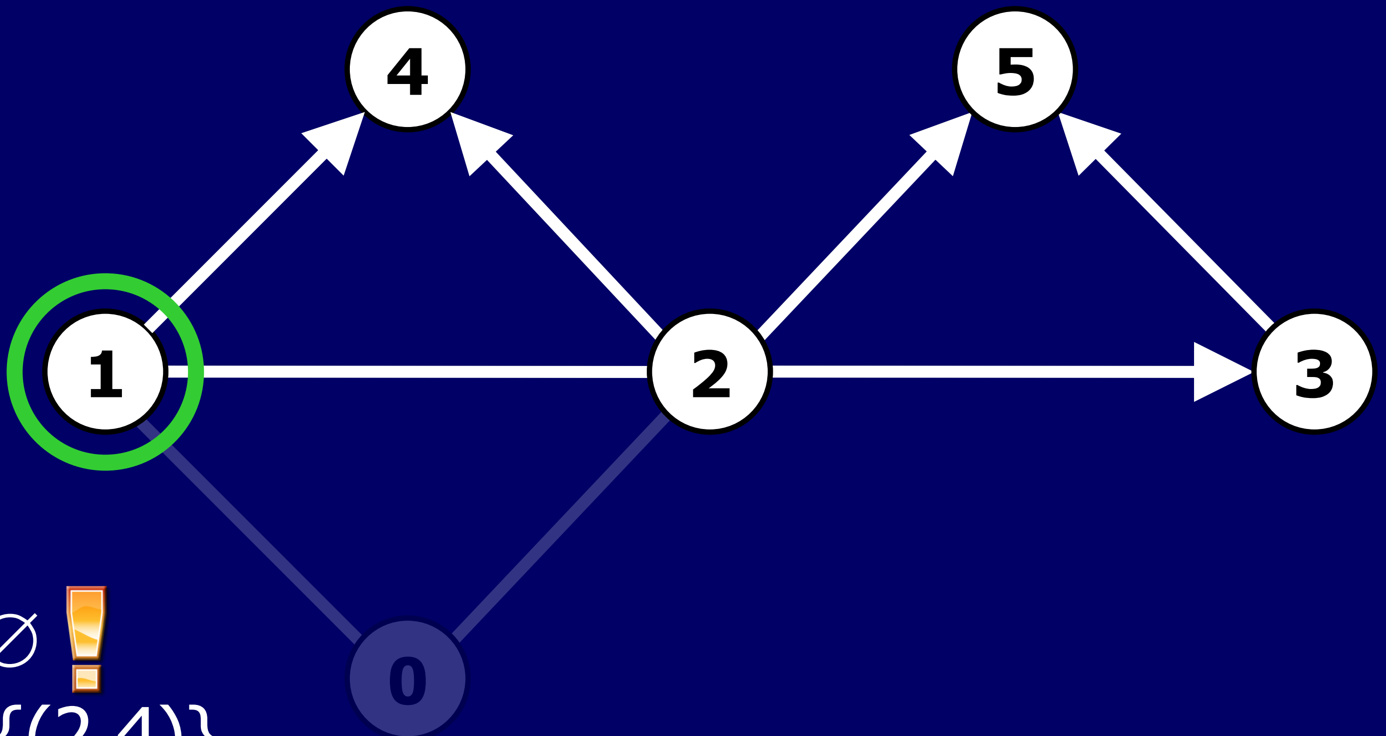
$H_{21}=\varnothing$

$L_{21}= \{(2,4)\}$

$F_{21}=\{(2,5),(2,3)\}$

Running Example

Back from recursion



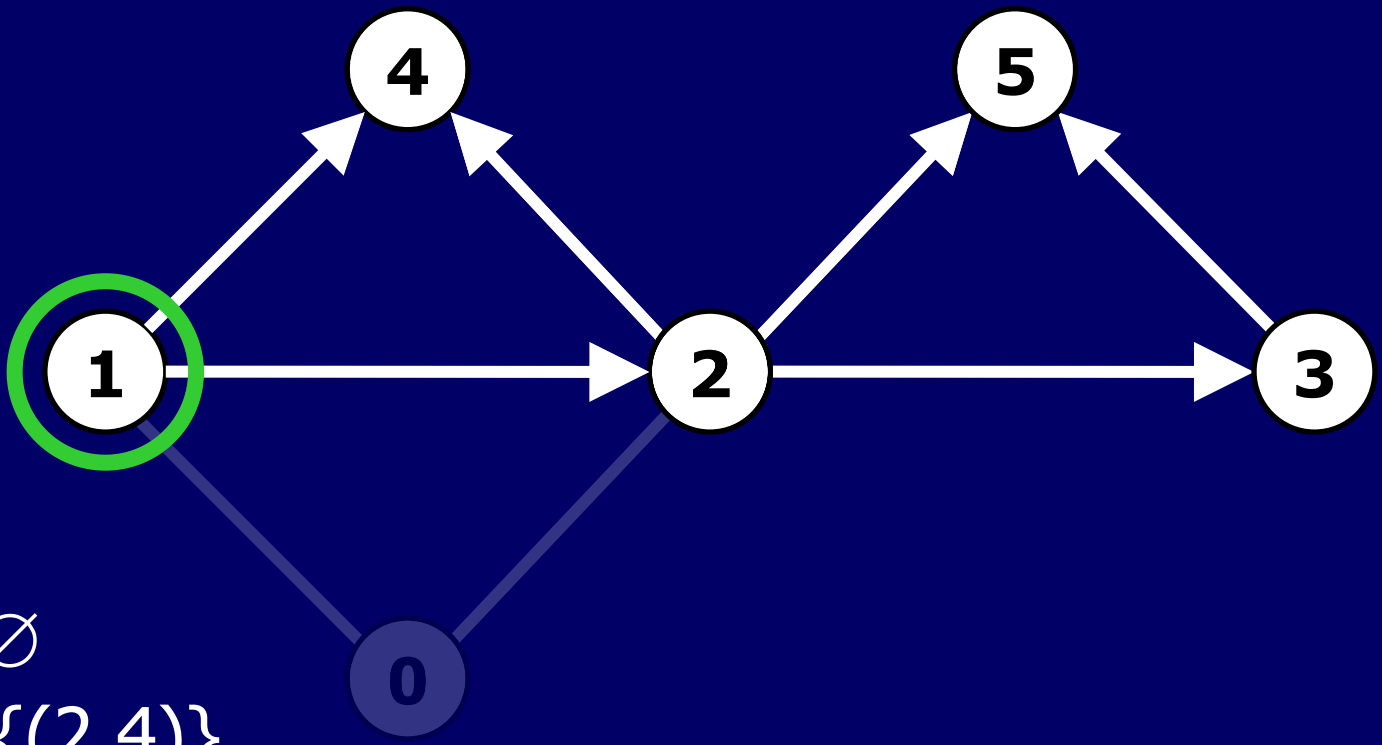
$H_{21}=\varnothing$!

$L_{21}=\{(2,4)\}$

$F_{21}=\{(2,5),(2,3)\}$

Running Example

Back from recursion



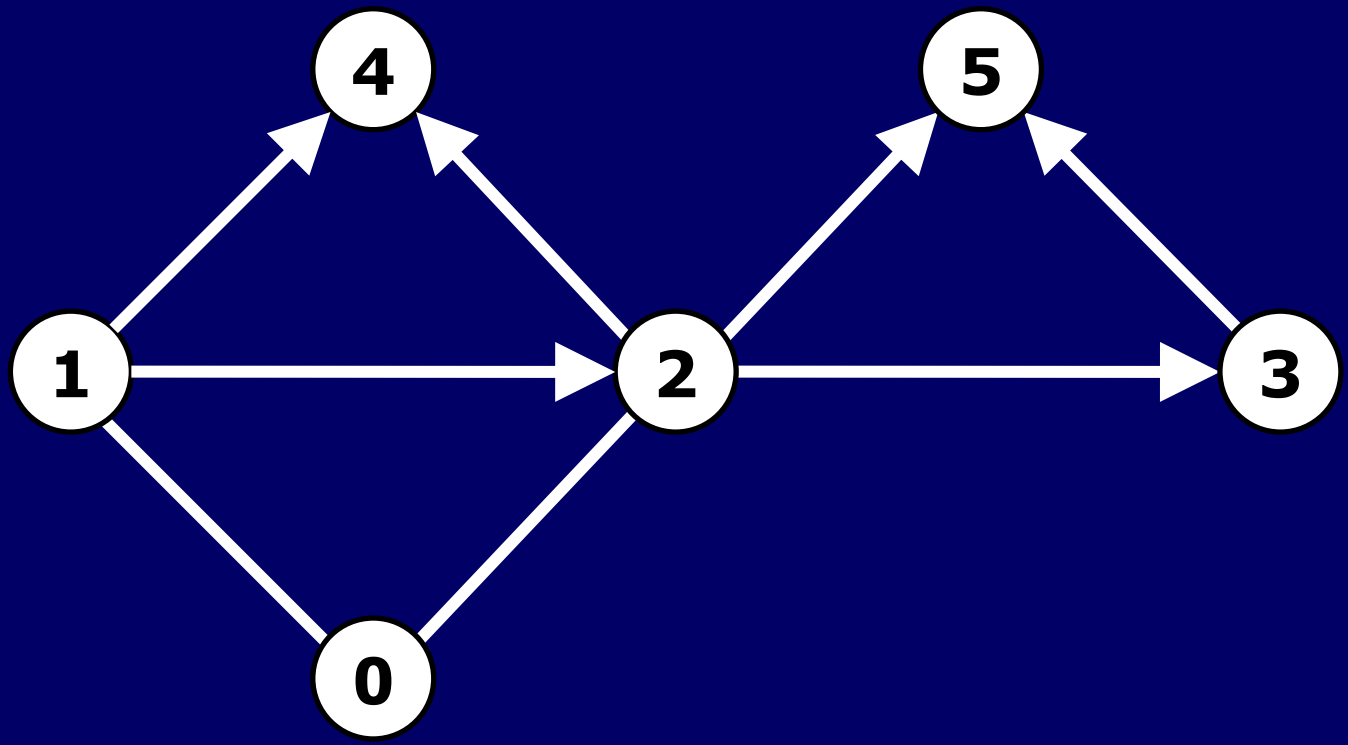
$H_{21}=\emptyset$

$L_{21}= \{(2,4)\}$

$F_{21}=\{(2,5),(2,3)\}$

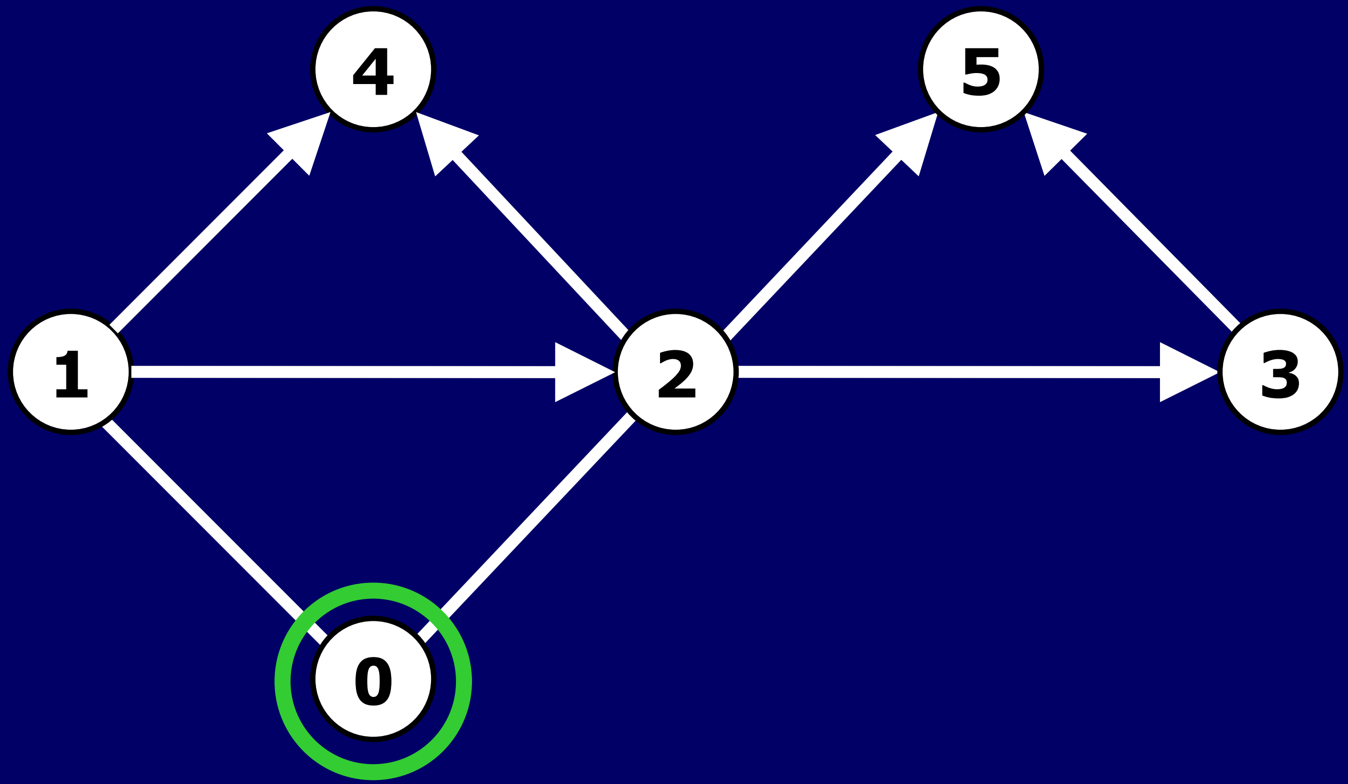
Running Example

Back from recursion



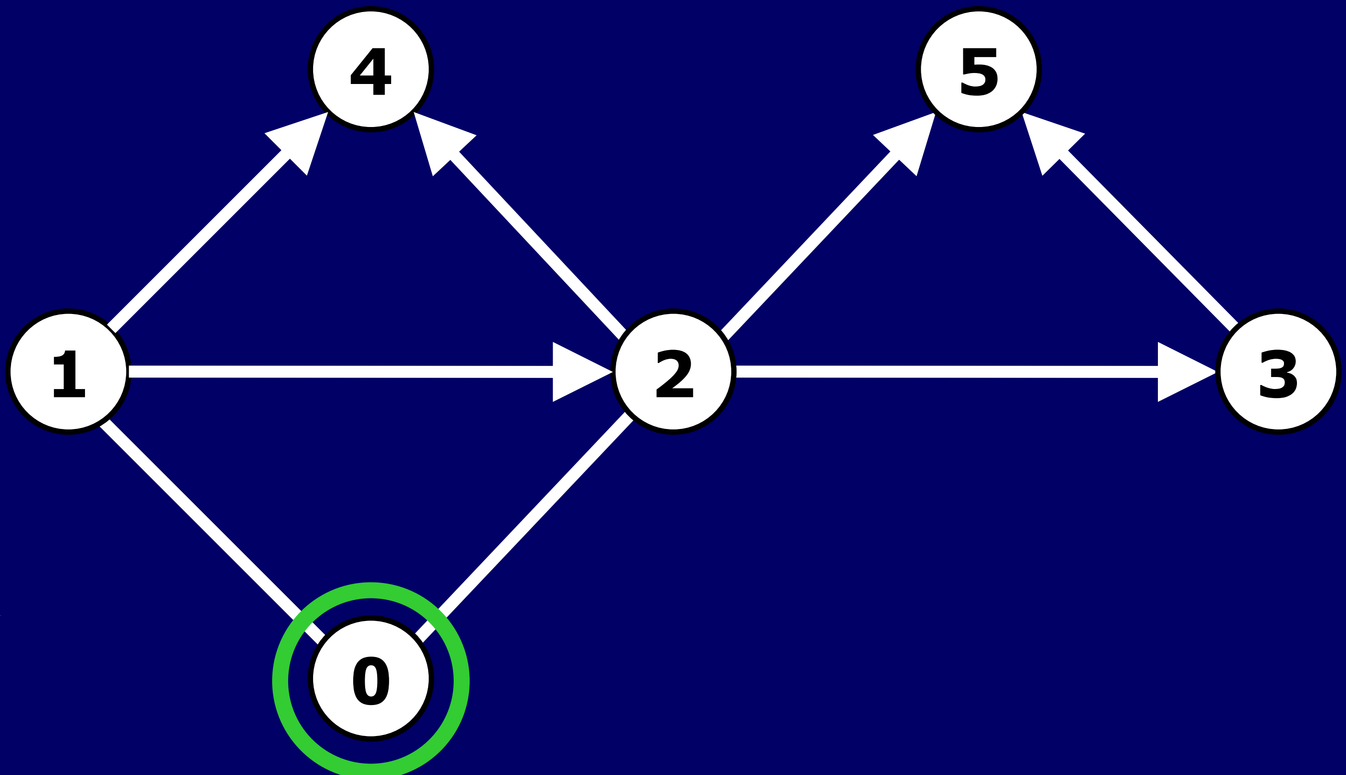
Running Example

Back from recursion



Running Example

Back from recursion



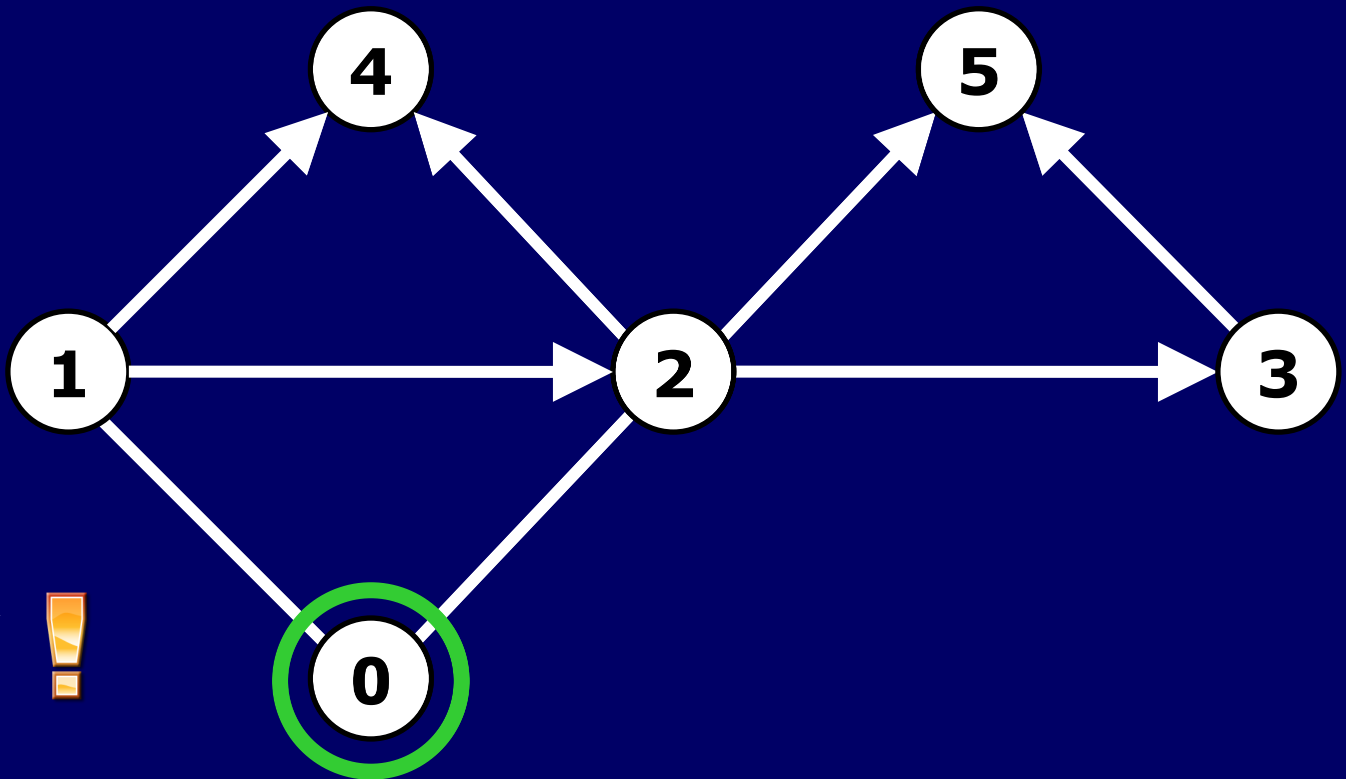
$H_{10}=\varnothing$

$L_{10}=\varnothing$

$F_{10}=\{(1,2),(1,4)\}$

Running Example

Back from recursion



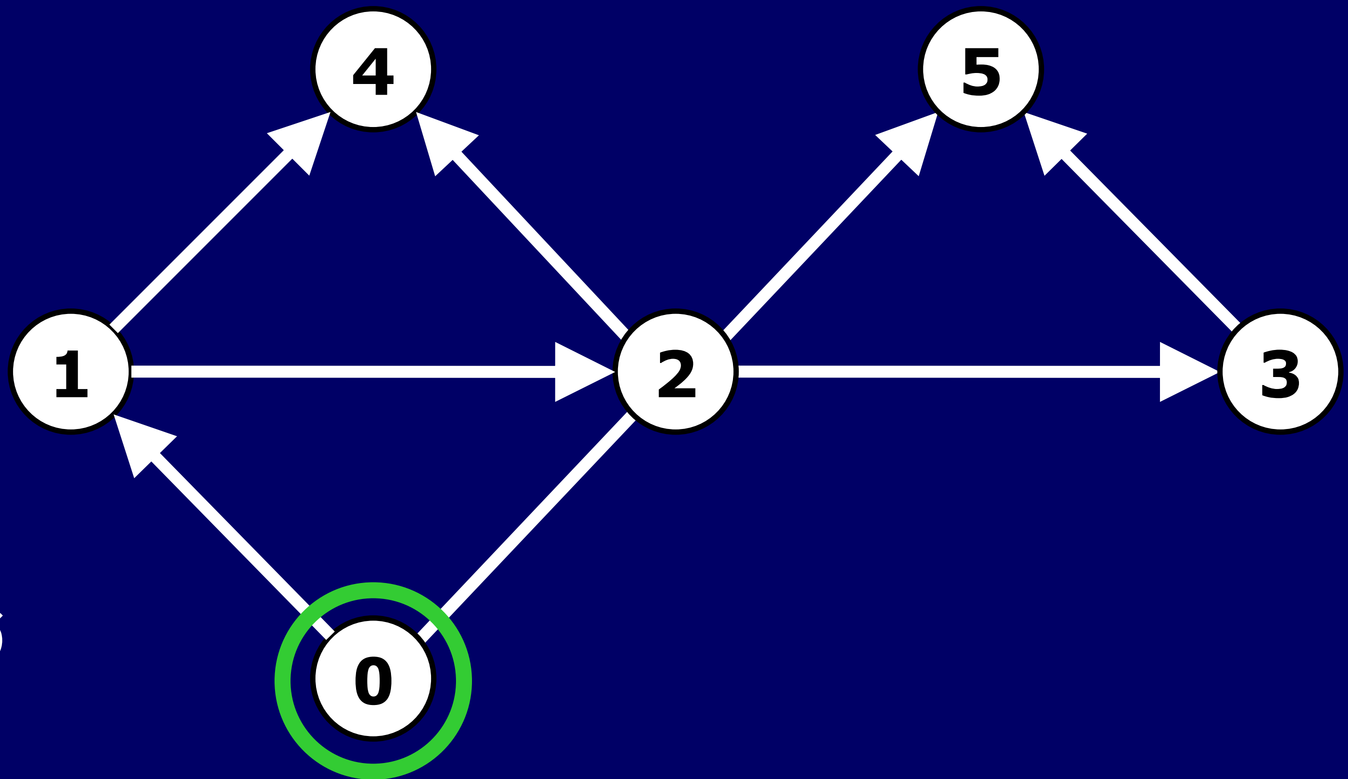
$H_{10}=\varnothing$

$L_{10}=\varnothing$

$F_{10}=\{(1,2),(1,4)\}$

Running Example

Back from recursion



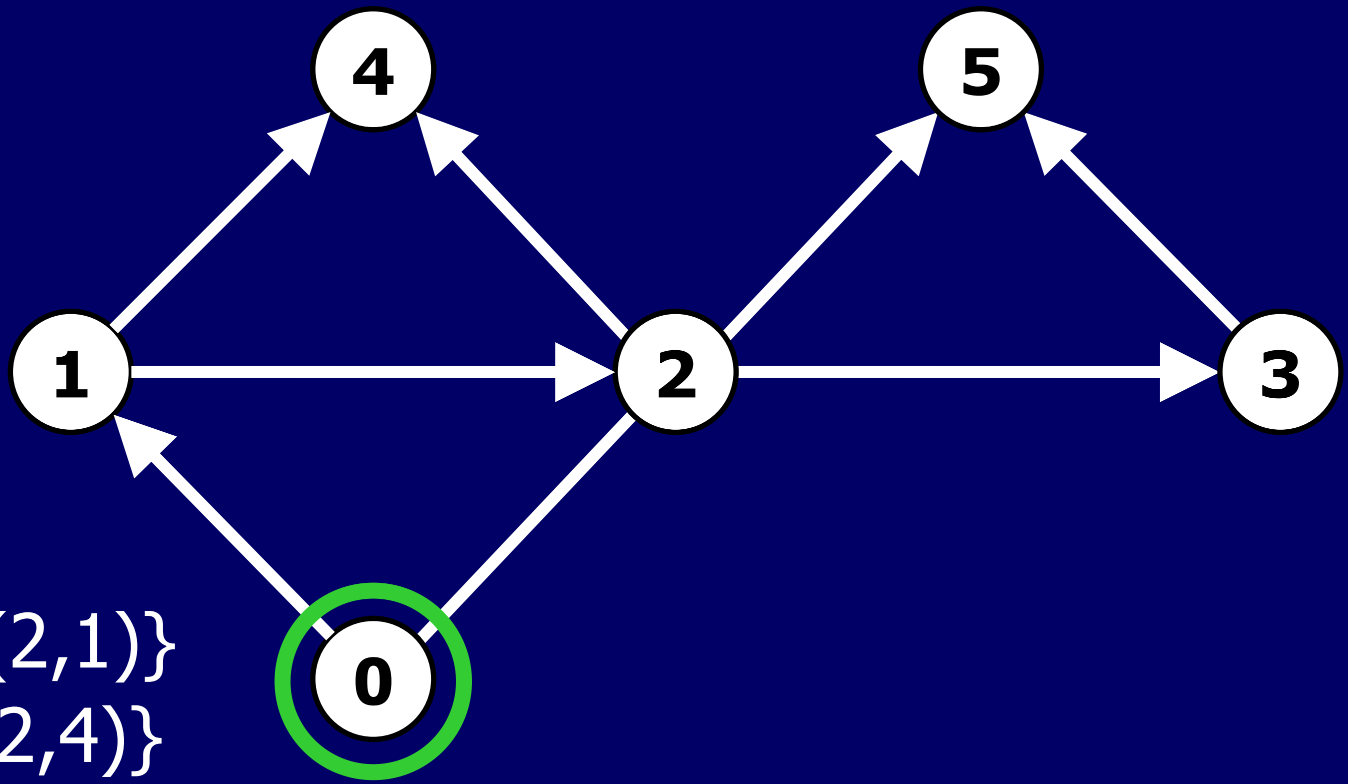
$H_{10}=\varnothing$

$L_{10}=\varnothing$

$F_{10}=\{(1,2),(1,4)\}$

Running Example

Back from recursion



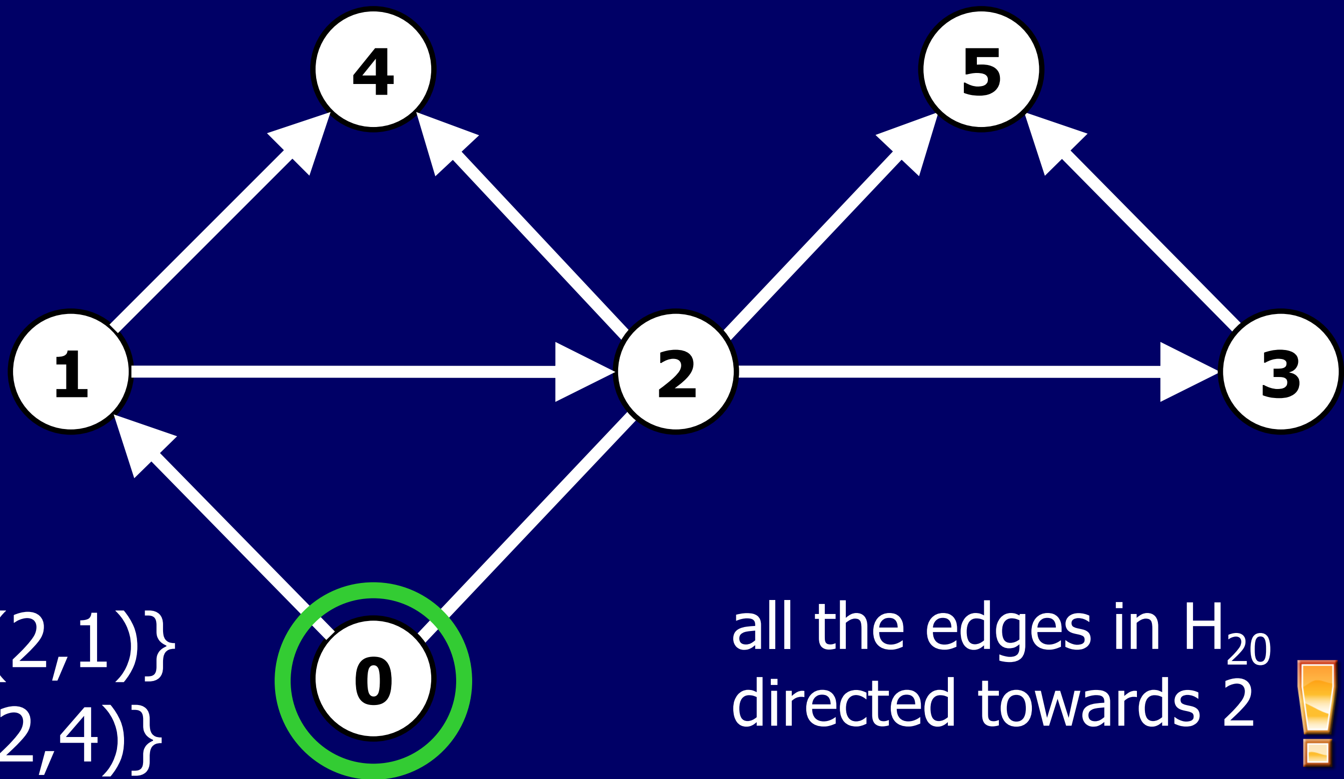
H_{20}={(2,1)}

L_{20}={(2,4)}

F_{20}={(2,5),(2,3)}

Running Example

Back from recursion

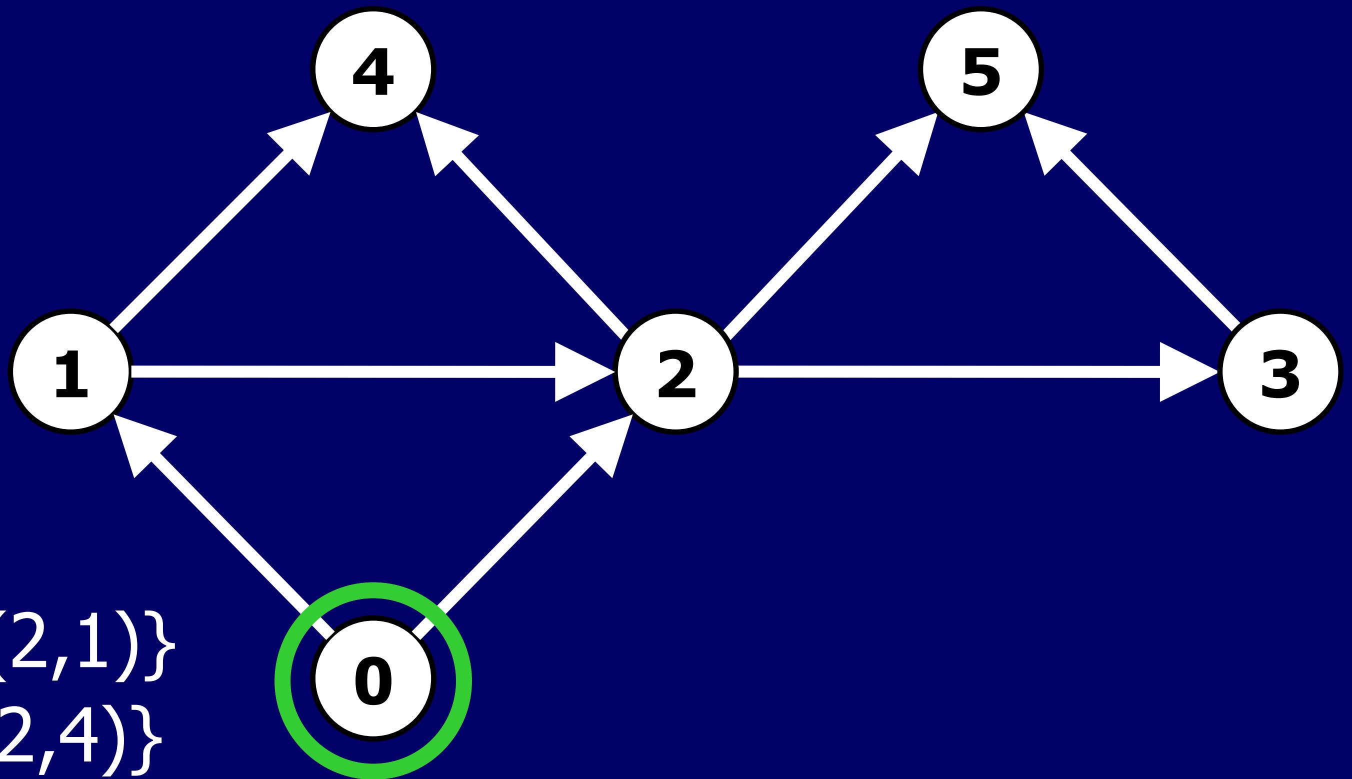


H_{20}={(2,1)}
L_{20}={(2,4)}
F_{20}={(2,5),(2,3)}

all the edges in H_{20} directed towards 2

Running Example

Back from recursion

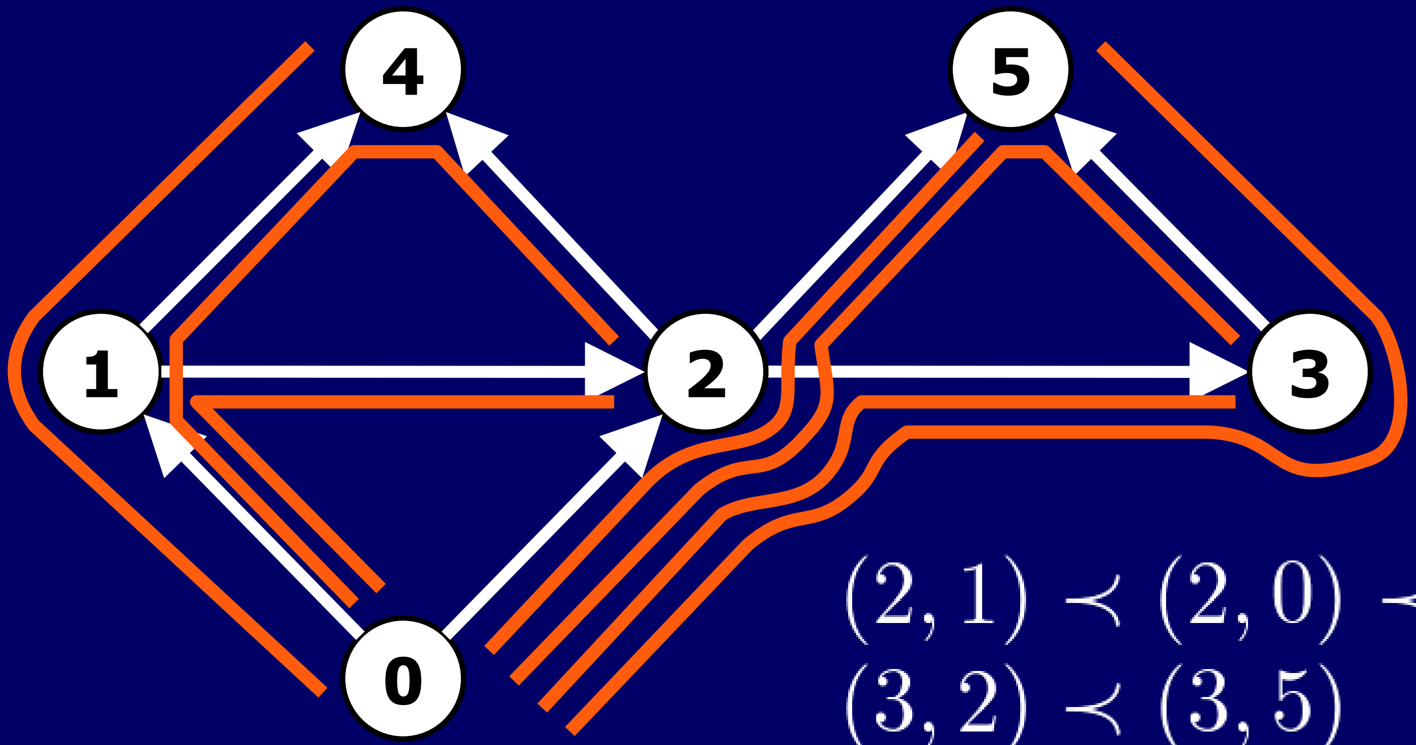


H_{20}={(2,1)}

L_{20}={(2,4)}

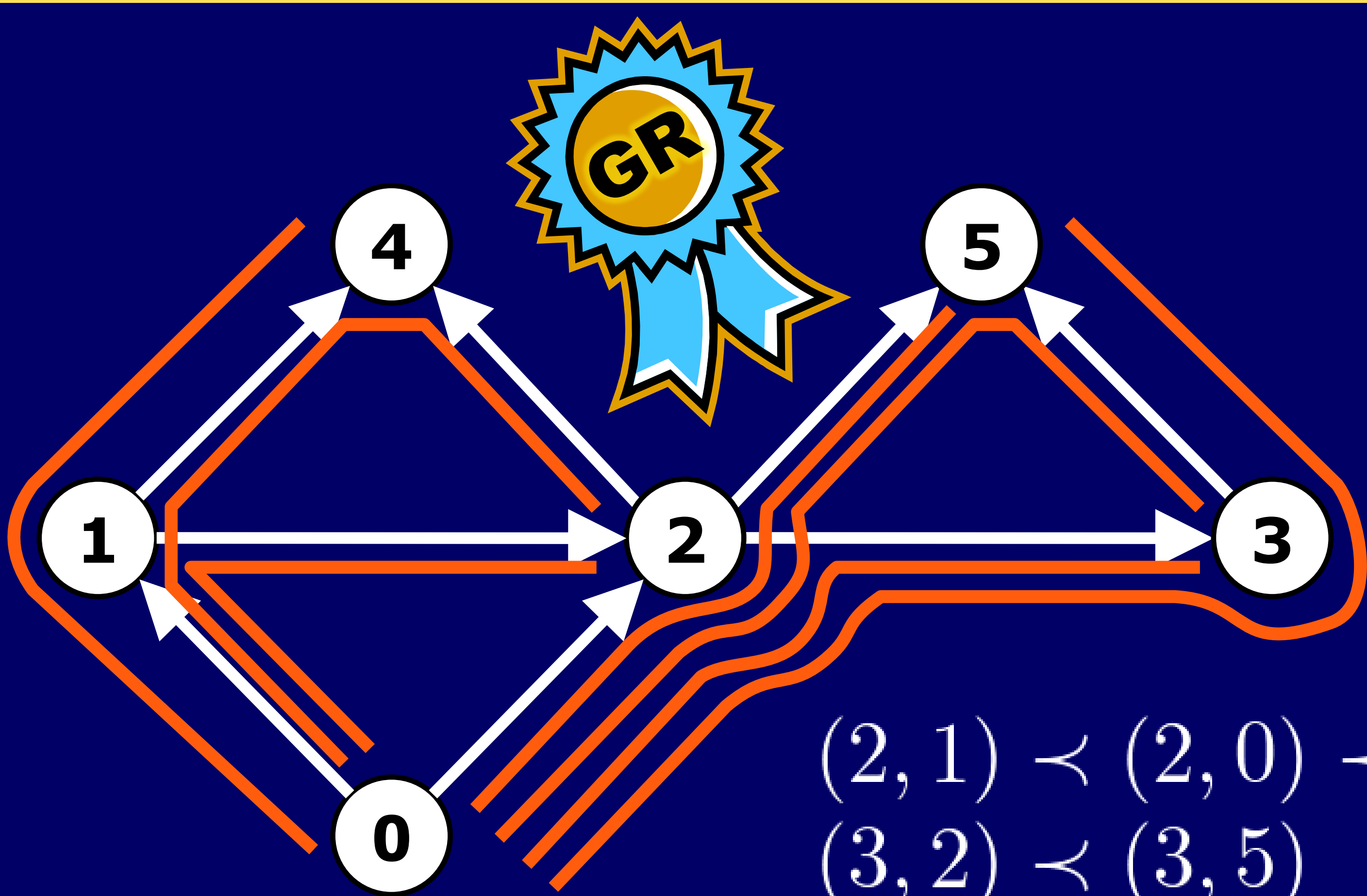
F_{20}={(2,5),(2,3)}

Running Example



$$(2,1) \prec (2,0) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$

Running Example



$$(2,1) \prec (2,0) \prec (2,4)$$
$$(3,2) \prec (3,5)$$
$$(5,2) \prec (5,3)$$