How to Make a Bipartite Graph DM-irreducible by Adding Edges

Satoru Iwata[1], Jun Kato[2], **Yutaro Yamaguchi**[3]

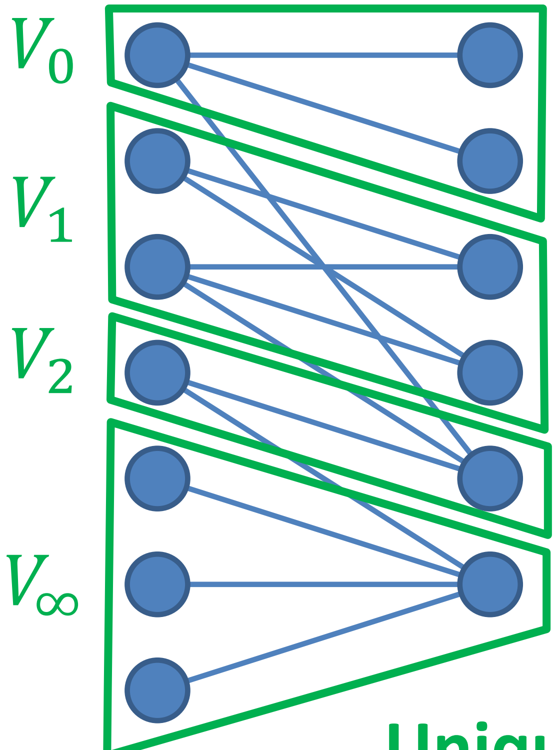
1. University of Tokyo, Japan.
2. TOYOTA Motor Corporation, Japan.
3. Osaka University, Japan.

JCCA2016 @Kyoto May 24, 2016

Dulmage–**M**endelsohn Decomposition

[Dulmage–Mendelsohn 1958,59]

Given $G = (V^+, V^-; E)$: Bipartite Graph



- $|V_0^+| < |V_0^-|$ or $V_0 = \emptyset$
- $|V_i^+| = |V_i^-| \quad (i \neq 0, \infty)$
- $|V_\infty^+| > |V_\infty^-|$ or $V_\infty = \emptyset$

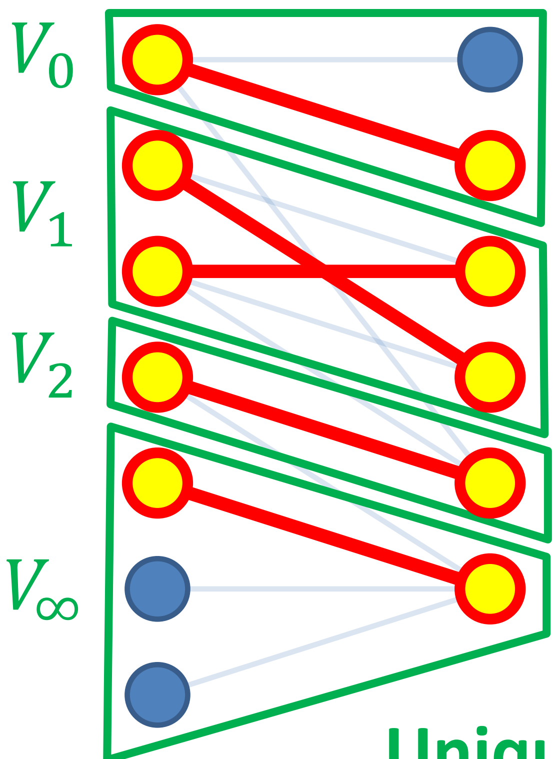
Unique Partition of Vertex Set

reflecting Structure of **Maximum Matchings**

Dulmage–**M**endelsohn Decomposition

[Dulmage–Mendelsohn 1958,59]

Given $G = (V^+, V^-; E)$: Bipartite Graph



- $|V_0^+| < |V_0^-|$ or $V_0 = \emptyset$
- $|V_i^+| = |V_i^-|$ $(i \neq 0, \infty)$
- $|V_\infty^+| > |V_\infty^-|$ or $V_\infty = \emptyset$

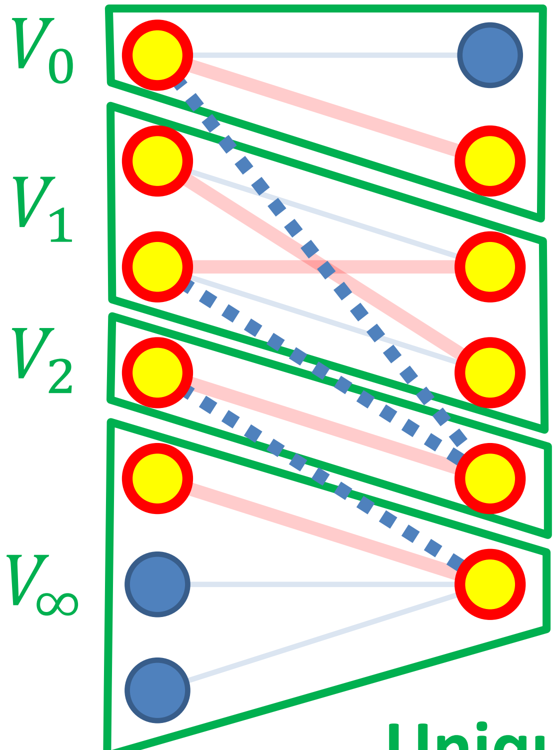
- $\forall$**Max. Matching** in G is a union of **Perfect Matchings** in $G[V_i]$

Unique Partition of Vertex Set
reflecting Structure of **Maximum Matchings**

Dulmage–**M**endelsohn Decomposition

[Dulmage–Mendelsohn 1958,59]

Given $G = (V^+, V^-; E)$: Bipartite Graph



- $\forall$**Max. Matching** in G is a union of **Perfect Matchings** in $G[V_i]$

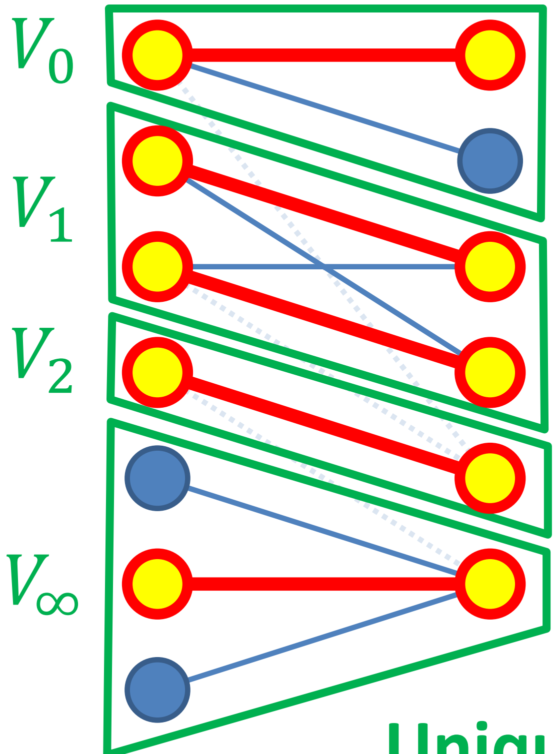
→ **Edges** between V_i and V_j $(i \neq j)$ can**NOT** be used.

Unique Partition of Vertex Set
reflecting Structure of **Maximum Matchings**

Dulmage–**M**endelsohn Decomposition

[Dulmage–Mendelsohn 1958,59]

Given $G = (V^+, V^-; E)$: Bipartite Graph



- ∀**Max. Matching** in G is a union of **Perfect Matchings** in $G[V_i]$
  → **Edges** between V_i and V_j $(i \neq j)$ can**NOT** be used.
- ∀e: Edge in $G[V_i]$,
  ∃**Perfect Matching** in $G[V_i]$ using e

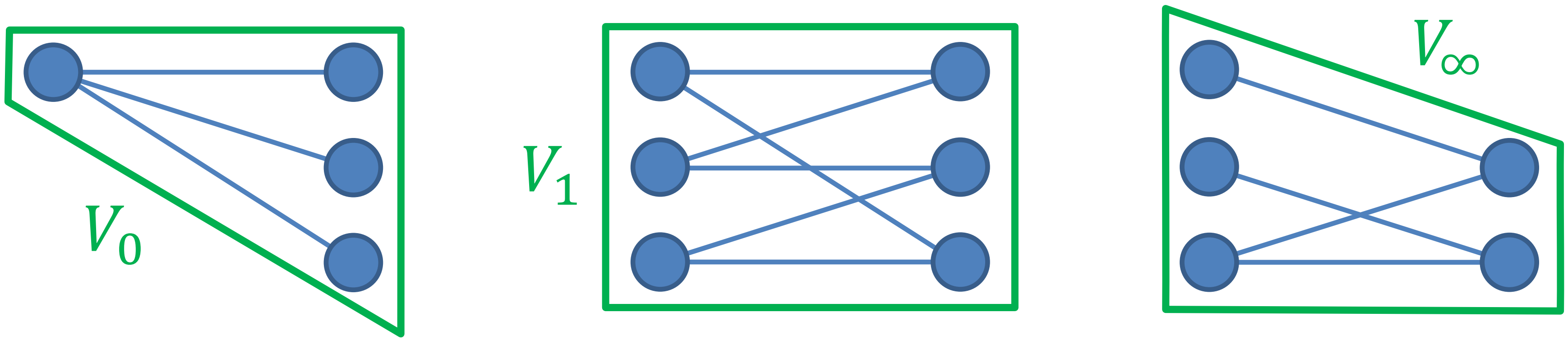
Unique Partition of Vertex Set
reflecting Structure of **Maximum Matchings**

DM-irreducibility

Def. A bipartite graph is **DM-irreducible**

$\Updownarrow$

The DM-decomposition consists of a single component



Obs. A bipartite graph G is **DM-irreducible**

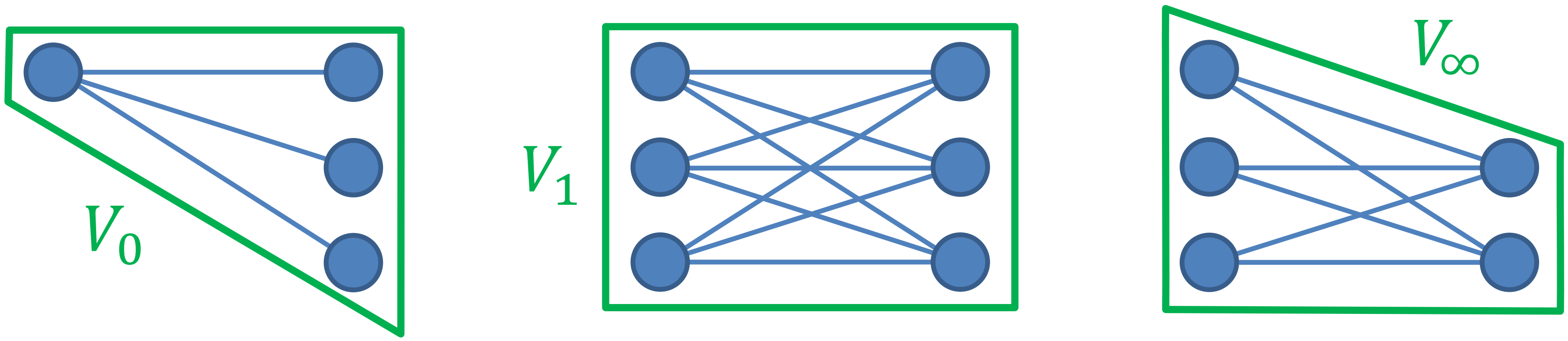
$\Downarrow$

$\forall e$: Edge in G, $\exists$Perfect Matching in G using e

DM-irreducibility

Obs. **Complete** bipartite graphs are **DM-irreducible**.

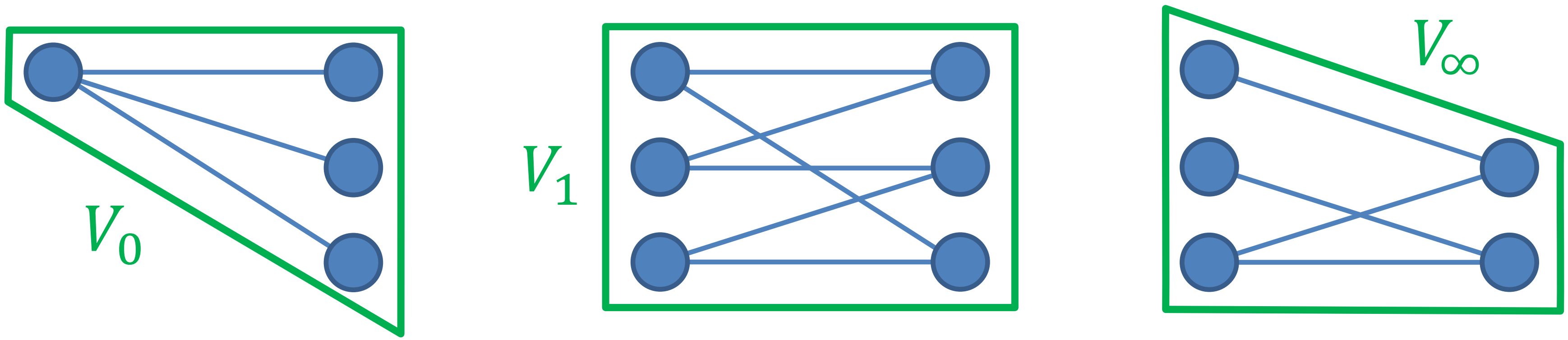
- Connected
- Every Edge is in some Perfect Matching



DM-irreducibility

<u>Obs.</u> **<u>Complete</u>** bipartite graphs are **<u>DM-irreducible</u>**.

- Connected
- Every Edge is in some Perfect Matching

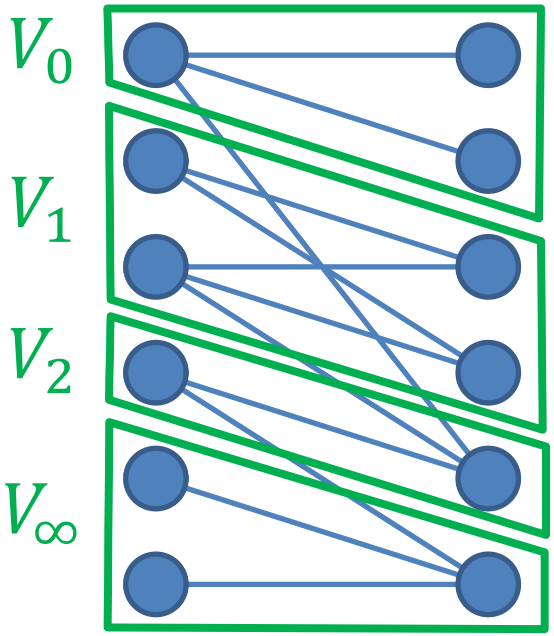


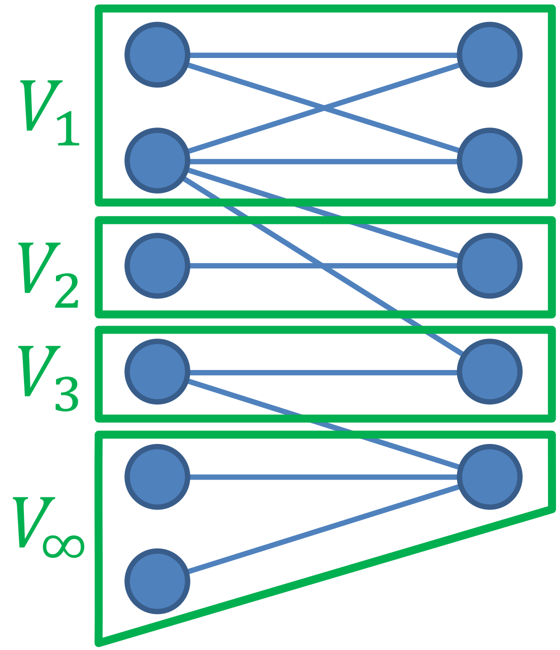
Complete $\overset{\Longrightarrow}{\not\Longleftarrow}$ **DM-irreducible**

How many additional edges are necessary to make a bipartite graph DM-irreducible?

Our Problem

Given $G = (V^+, V^-; E)$: Bipartite Graph

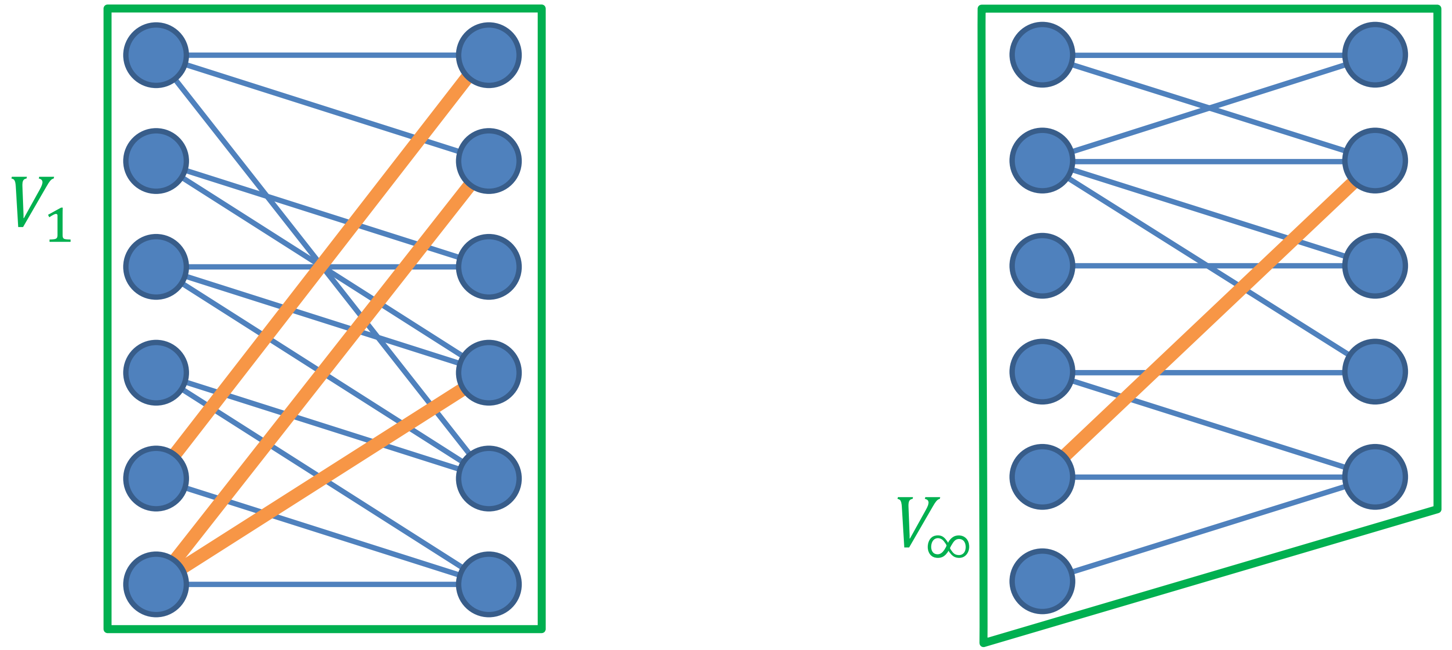




Find **Minimum Number of Additional Edges**
to Make G **DM-irreducible**

Our Problem

Given $G = (V^+, V^-; E)$: Bipartite Graph



Find **Minimum Number of Additional Edges**
to Make G **DM-irreducible**

Background

Covering a Bisupermodular Function by Directed Edges

- Min-Max Duality
- Polytime by Ellipsoid [Frank–Jordán 1995]
- Pseudopolytime Algo. [Végh–Benczúr 2008]

↑

Our Problem

↑

Making a Digraph Strongly Connected

- Min-Max Duality
- Linear-time Algo. [Eswaran–Tarjan 1976]

Our Results

Covering a Bisupermodular Function by Directed Edges

- Min-Max Duality
- Polytime by Ellipsoid

[Frank–Jordán 1995]

- Pseudopolytime Algo.

[Végh–Benczúr 2008]

↑

Our Problem

- Simple Polytime Algo.
- Constructive Proof for Min-Max

[I.–K.–Y. 2016]

↑

Making a Digraph Strongly Connected

- Min-Max Duality
- Linear-time Algo.

[Eswaran–Tarjan 1976]

Outline

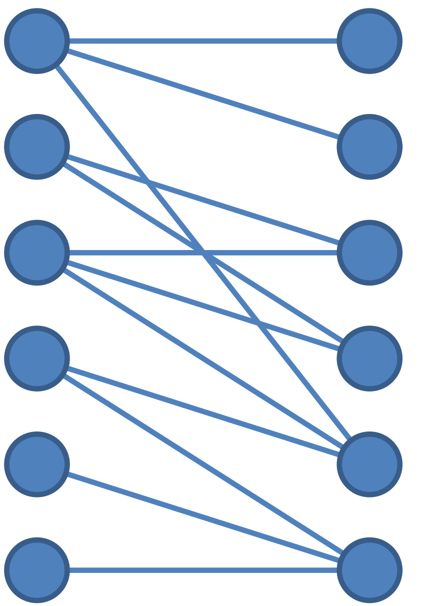
- **Preliminaries:** How to Compute DM-decomposition
  - Find a **Maximum Matching** in a Bipartite Graph
  - Decompose a Digraph into **Strongly Connected Components**
- **Result:** How to Make a Bipartite Graph DM-irreducible
  - Make a Digraph **Strongly Connected** [Eswaran–Tarjan 1976]
  - Find **Edge-Disjoint** s–t **Paths** in a Digraph
- **Conclusion**

Outline

- **Preliminaries:** How to Compute DM-decomposition
  - Find a **Maximum Matching** in a Bipartite Graph
  - Decompose a Digraph into **Strongly Connected Components**
- **Result:** How to Make a Bipartite Graph DM-irreducible
  - Make a Digraph **Strongly Connected** [Eswaran–Tarjan 1976]
  - Find **Edge-Disjoint** s–t **Paths** in a Digraph
- **Conclusion**

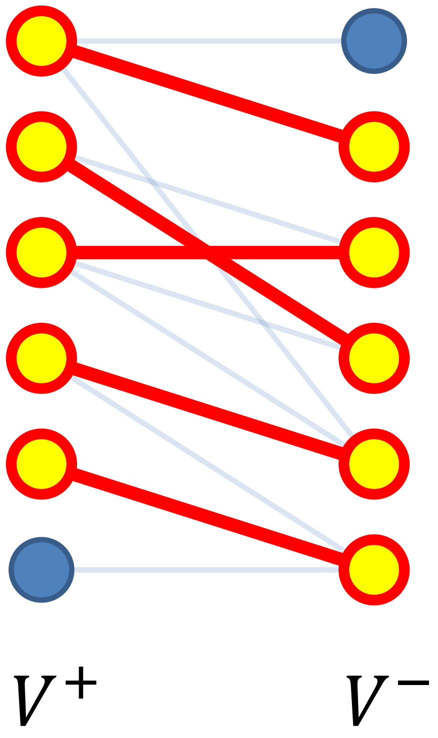
How to Compute DM-decomposition

<u>**Given**</u> $G = (V^+, V^-; E)$: Bipartite Graph



How to Compute DM-decomposition

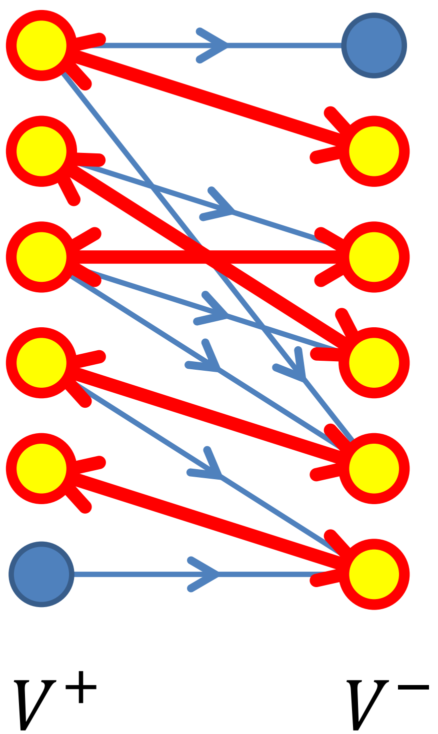
Given $G = (V^+, V^-; E)$: Bipartite Graph



- Find a **Maximum Matching** M in G

How to Compute DM-decomposition

Given $G = (V^+, V^-; E)$: Bipartite Graph



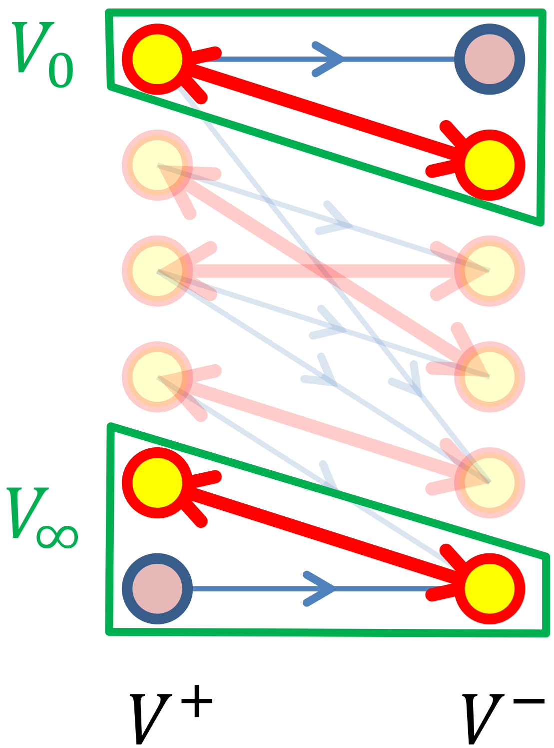
- Find a **Maximum Matching** M in G
- **Orient Edges** so that

  $M \implies$ Both Directions $\leftrightarrow$

  $E \setminus M \implies$ Left to Right $\rightarrow$

How to Compute DM-decomposition

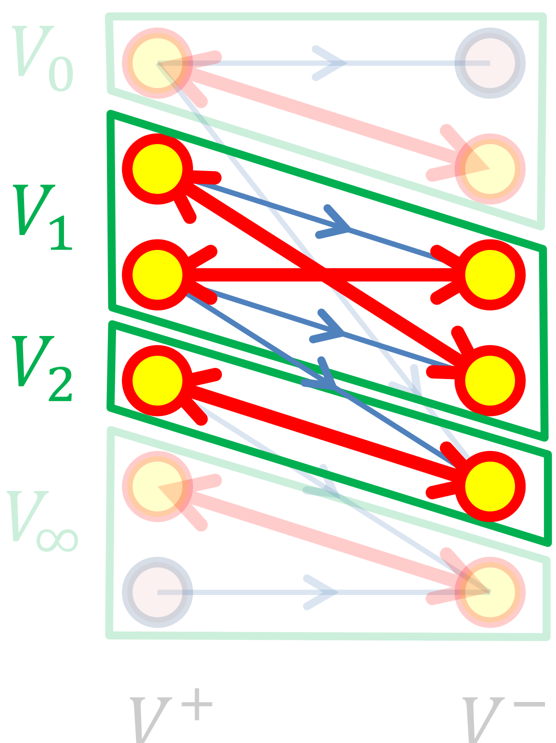
Given $G = (V^+, V^-; E)$: Bipartite Graph



- Find a **Maximum Matching** M in G
- **Orient Edges** so that
  $M \implies$ Both Directions $\leftrightarrow$
  $E \setminus M \implies$ Left to Right $\rightarrow$
- V_0: **Reachable to** $V^- \setminus \partial^- M$
- V_∞: **Reachable from** $V^+ \setminus \partial^+ M$

How to Compute DM-decomposition

Given $G = (V^+, V^-; E)$: Bipartite Graph



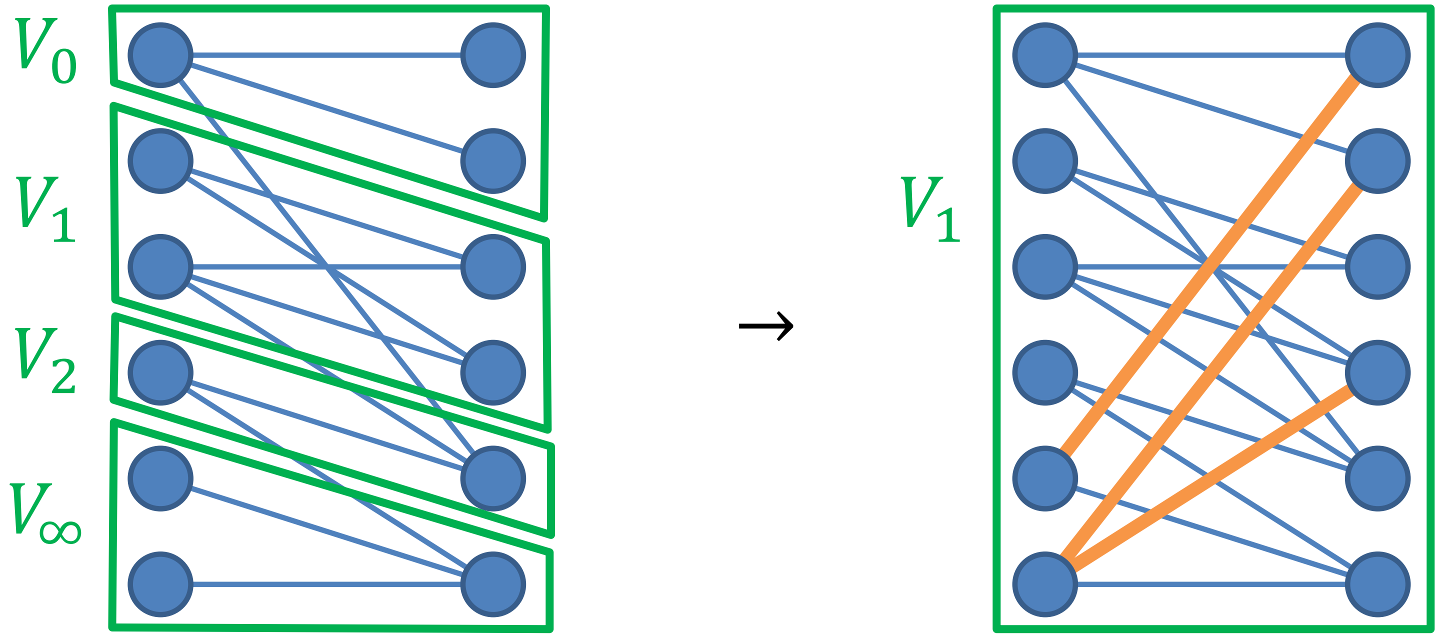
- Find a **Maximum Matching** M in G
- **Orient Edges** so that
  $M \implies$ Both Directions $\leftrightarrow$
  $E \setminus M \implies$ Left to Right $\rightarrow$
- V_0: **Reachable to** $V^- \setminus \partial^- M$
- V_∞: **Reachable from** $V^+ \setminus \partial^+ M$
- V_i: **Strongly Connected Component** of $G - V_0 - V_\infty$

Outline

- **Preliminaries:** How to Compute DM-decomposition
  - Find a **Maximum Matching** in a Bipartite Graph
  - Decompose a Digraph into **Strongly Connected Components**
- **Result:** How to Make a Bipartite Graph DM-irreducible
  - Make a Digraph **Strongly Connected** [Eswaran–Tarjan 1976]
  - Find **Edge-Disjoint s–t Paths** in a Digraph
- **Conclusion**

Our Problem (Reminder)

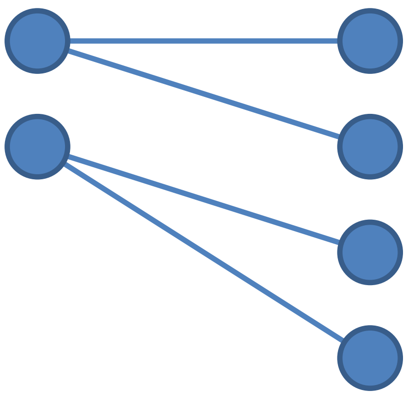
Given $G = (V^+, V^-; E)$: Bipartite Graph



Find **Minimum Number of Additional Edges**
to Make G **DM-irreducible**

Unbalanced Case → Balanced Case

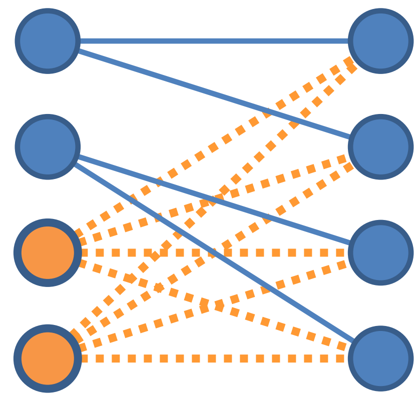
$|V^+| \neq |V^-|$



G

→

$|V^+| = |V^-|$

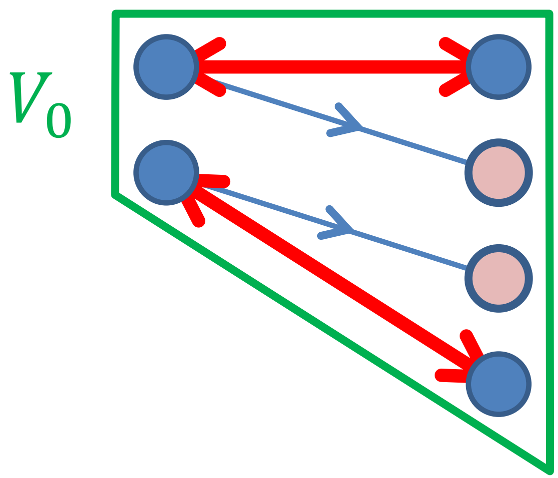


G'

Fact G is **DM-irreducible** $\Longleftrightarrow$ G' is **DM-irreducible**

Unbalanced Case → Balanced Case

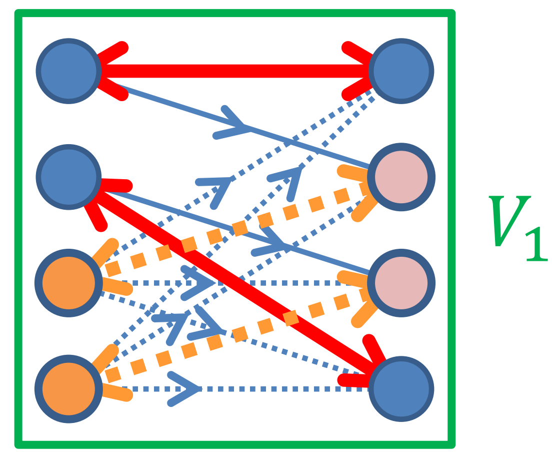
$|V^+| \neq |V^-|$



G

$\rightarrow$

$|V^+| = |V^-|$



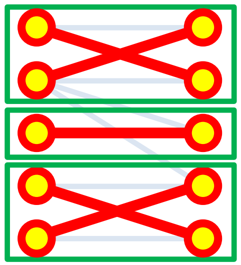
G'

Fact G is **DM-irreducible** $\Longleftrightarrow$ G' is **DM-irreducible**

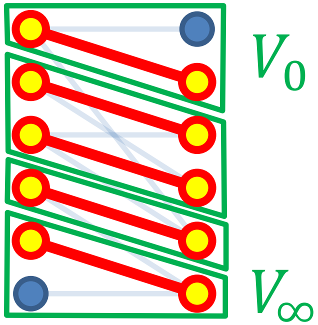
Case Analysis

Assumption $G = (V^+, V^-; E)$ is **Balanced**

Case 1. When G has a perfect matching.



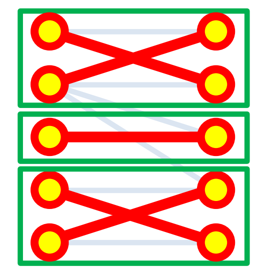
Case 2. When G has NO perfect matching.



Case Analysis

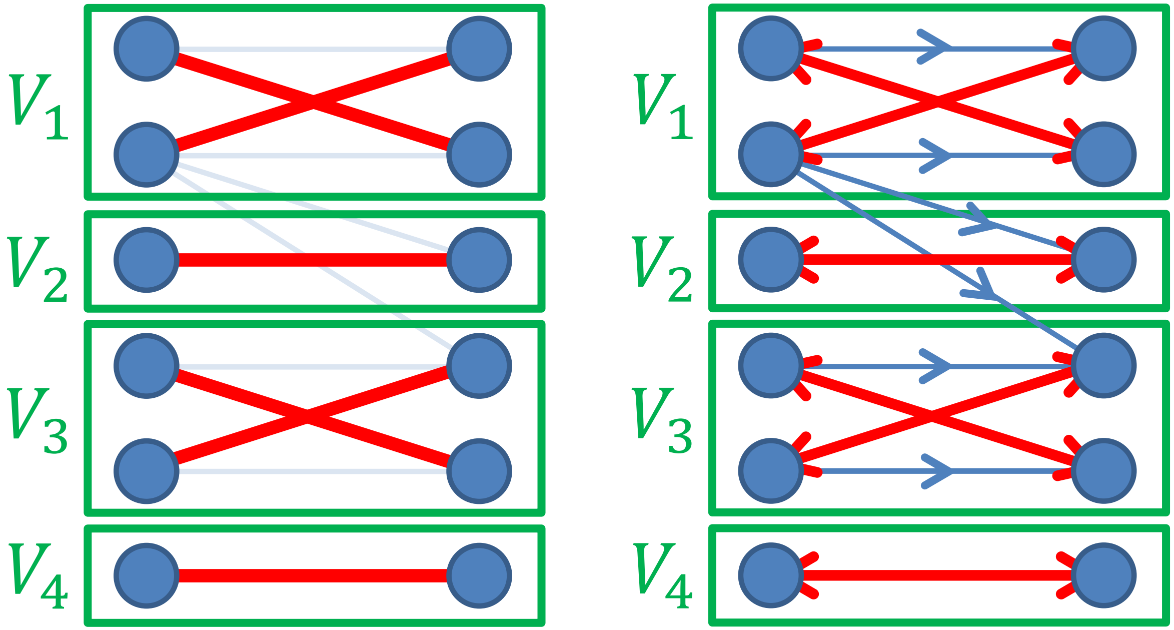
Assumption $G = (V^+, V^-; E)$ is **Balanced**

Case 1. When G has a perfect matching.



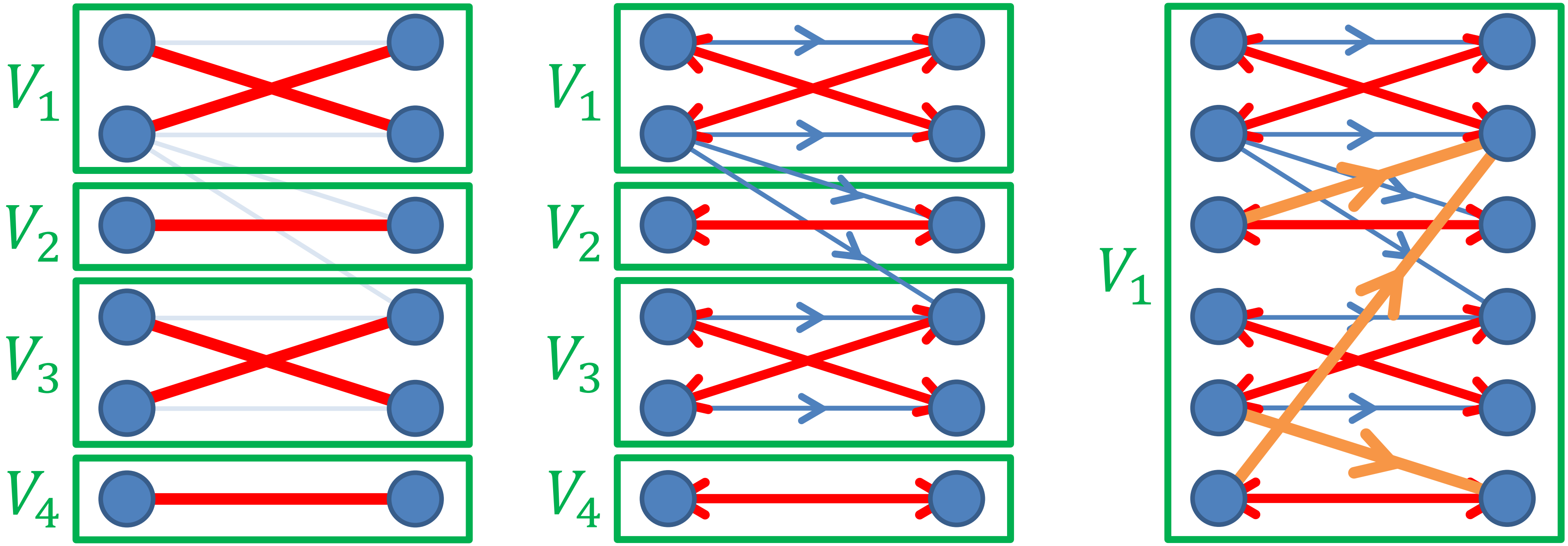
Case 2. When G has NO perfect matching.

Case 1. Perfectly Matchable



DM-decomposition $=$ **Strg. Conn. Comps.**

Case 1. Perfectly Matchable



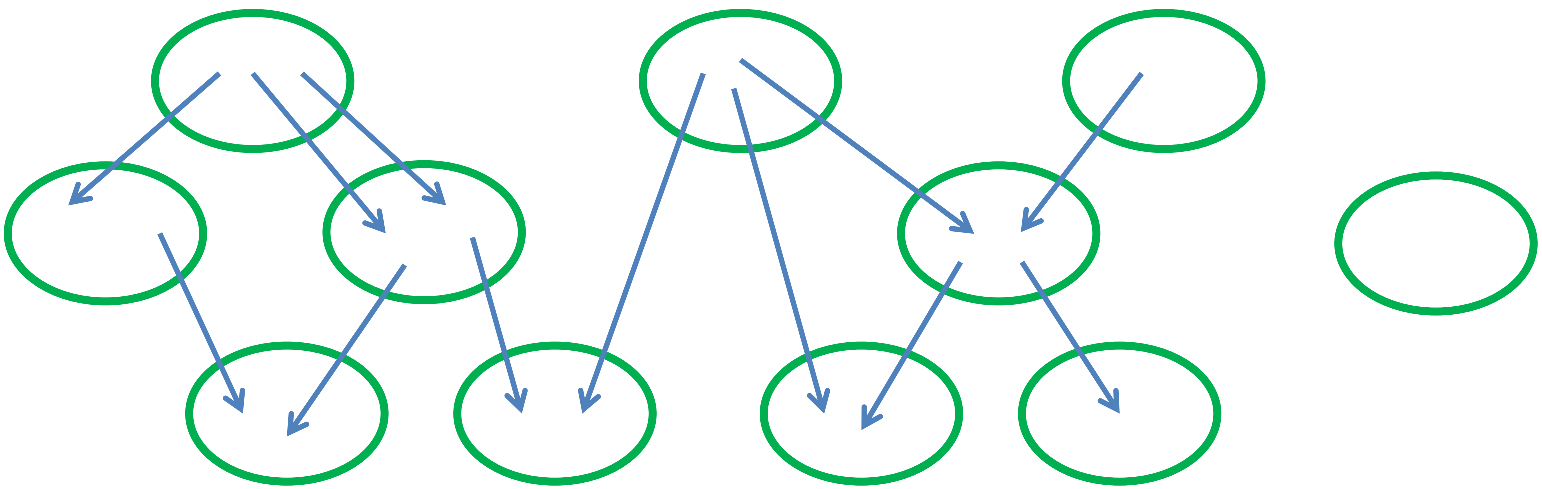
DM-decomposition $=$ **Strg. Conn. Comps.** → **Make it Strg. Conn. by Adding Edges**

Obs. **DM-irreducibility** is **Equivalent to Strong Connectivity** of the Oriented Graph

How to Make a Digraph Strongly Connected

Given $G = (V, E)$: Directed Graph

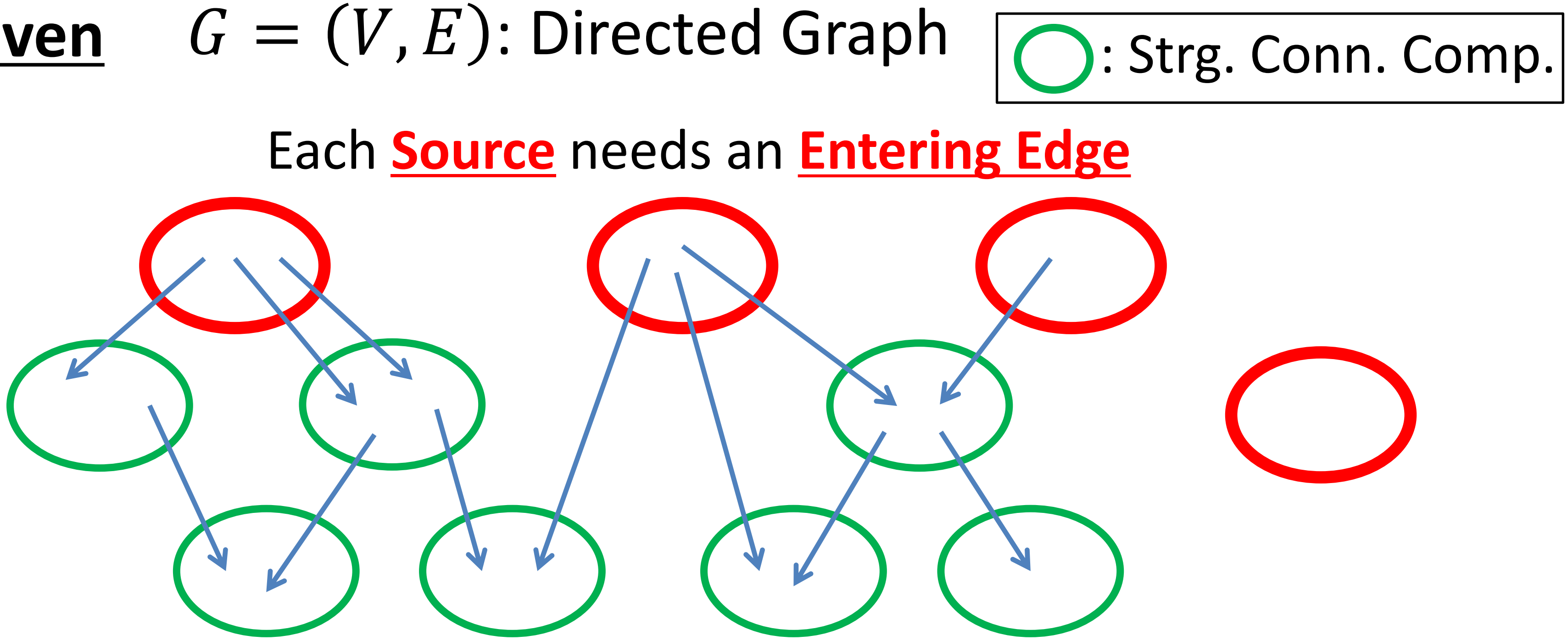
: Strg. Conn. Comp.



Find **Minimum Number of Additional Edges**
to Make G **Strongly Connected**

How to Make a Digraph Strongly Connected

Given $G = (V, E)$: Directed Graph



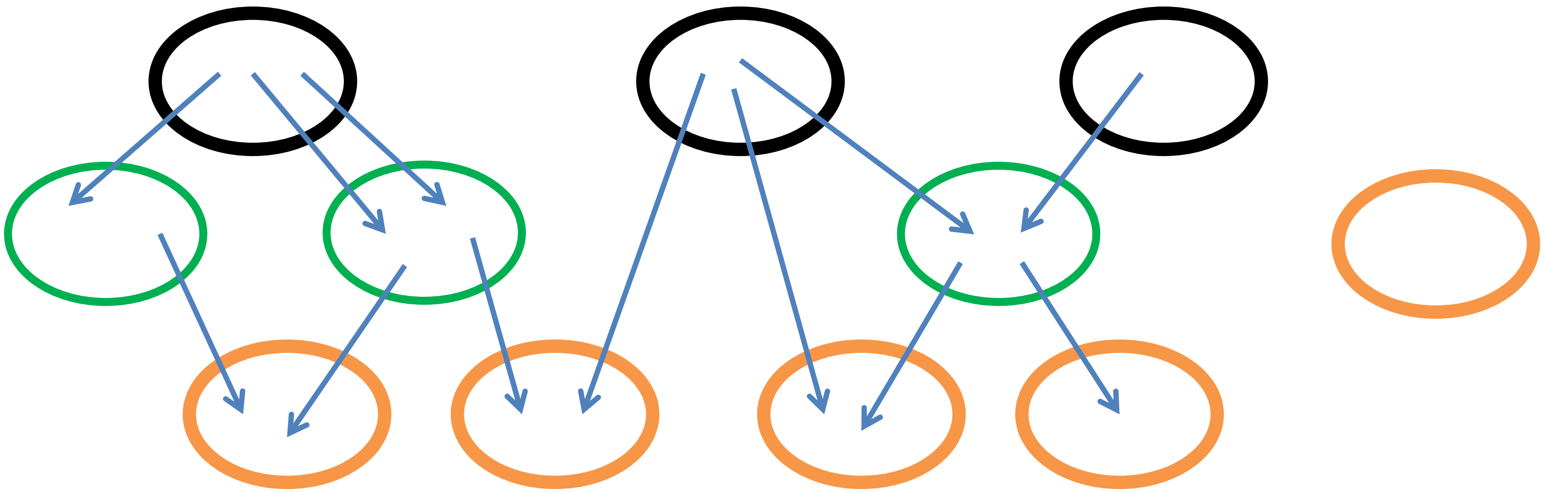
Find **Minimum Number of Additional Edges** to Make G **Strongly Connected**

How to Make a Digraph Strongly Connected

Given $G = (V, E)$: Directed Graph

(green circle): Strg. Conn. Comp.

Each **Source** needs an **Entering Edge**



Each **Sink** needs a **Leaving Edge**

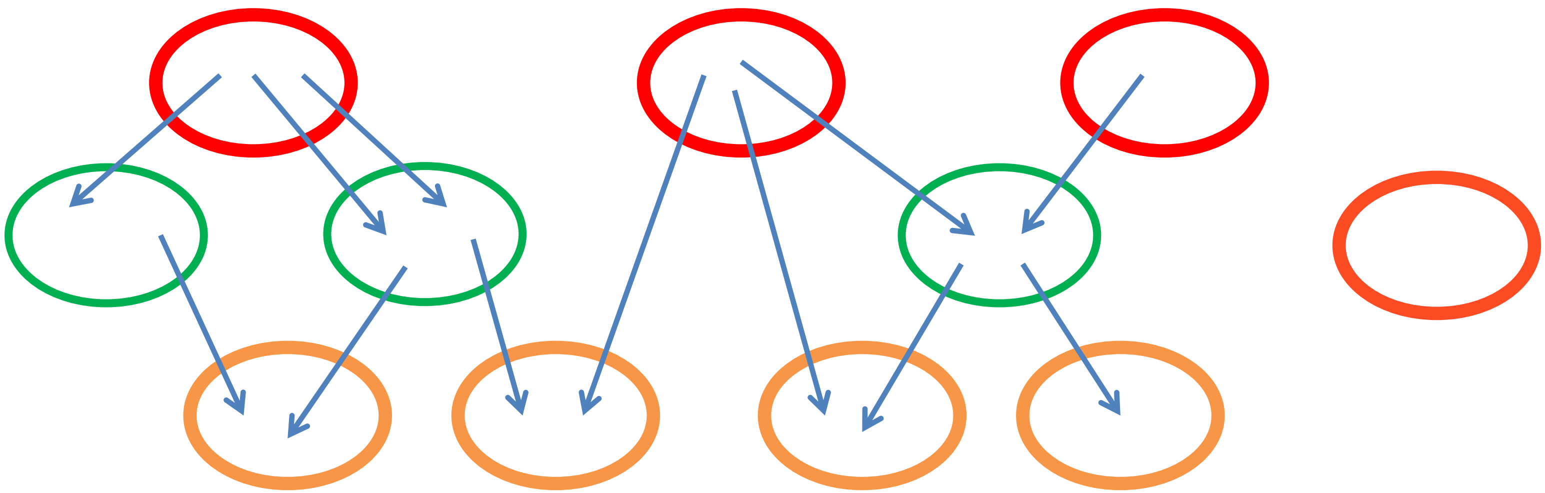
Find **Minimum Number of Additional Edges** to Make G **Strongly Connected**

How to Make a Digraph Strongly Connected

Given $G = (V, E)$: Directed Graph NOT Strg. Conn.

Find **Minimum Number of Additional Edges** to Make G **Strongly Connected**

Obs. $\max\{$# of **Sources**, # of **Sinks**$\}$ edges are **Necessary**.



How to Make a Digraph Strongly Connected

Given $G = (V, E)$: Directed Graph NOT Strg. Conn.

Find **Minimum Number of Additional Edges** to Make G **Strongly Connected**

Obs. $\max\{$# of **Sources**, # of **Sinks**$\}$ edges are **Necessary**.

Thm. $\max\{$# of **Sources**, # of **Sinks**$\}$ edges are **Sufficient**.
One can find such Additional Edges in **Linear Time**.

[Eswaran–Tarjan 1976]

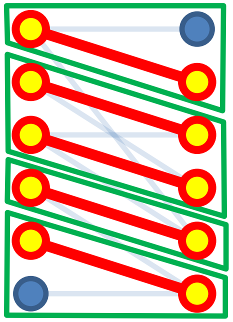
→ Case 1 is Solved in **Linear Time**.

Case Analysis

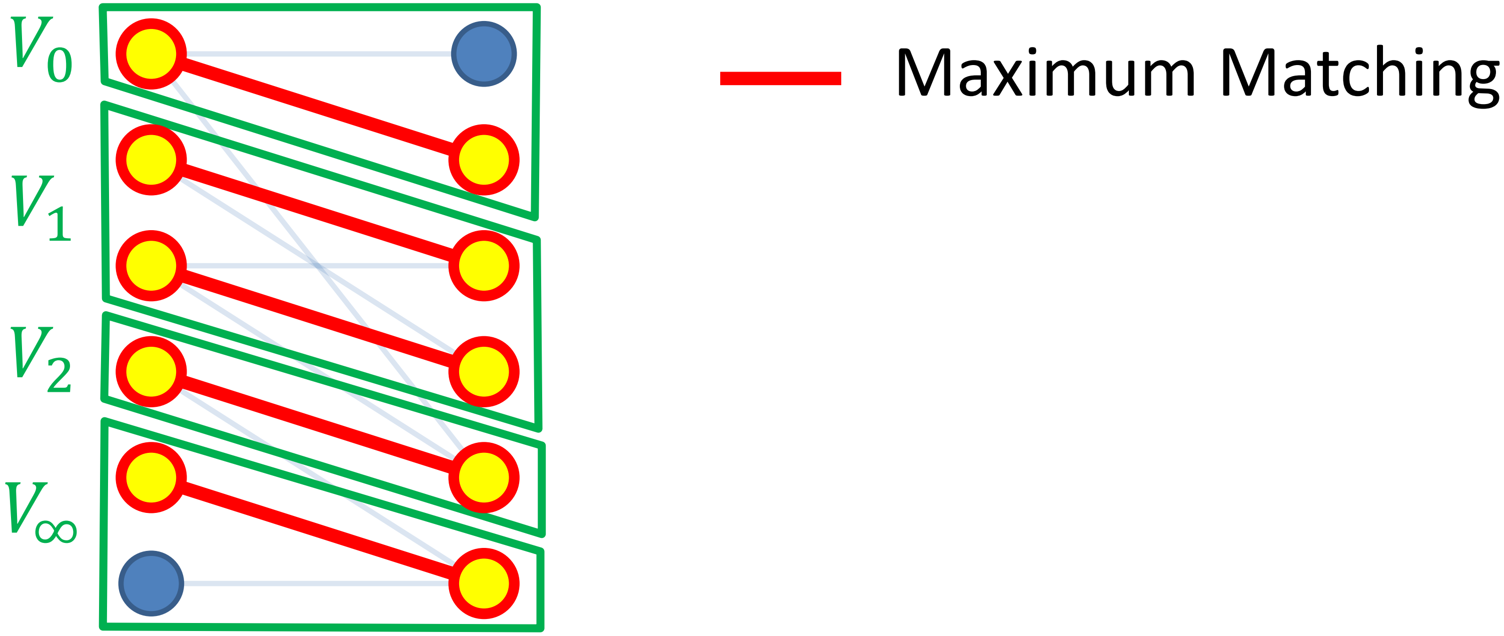
Assumption $G = (V^+, V^-; E)$ is **Balanced**

Case 1. When G has a perfect matching.

Case 2. When G has NO perfect matching.

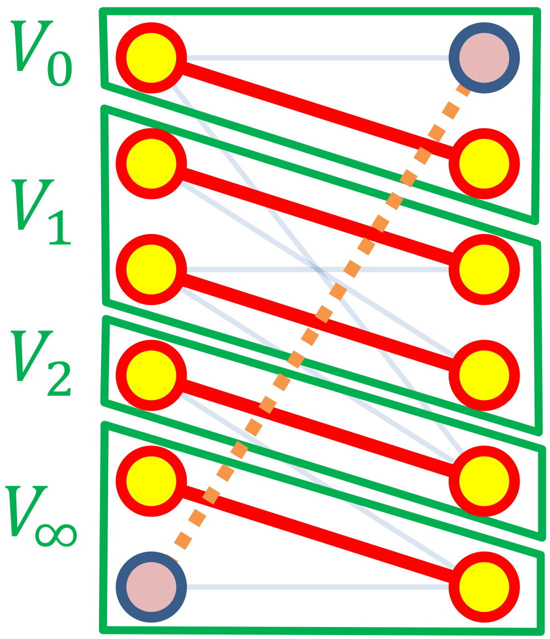


Case 2. NO Perfect Matching

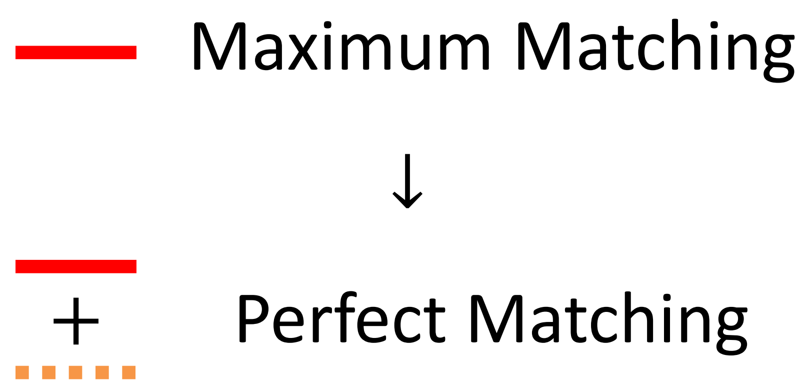


DM-decomposition

Case 2. NO Perfect Matching



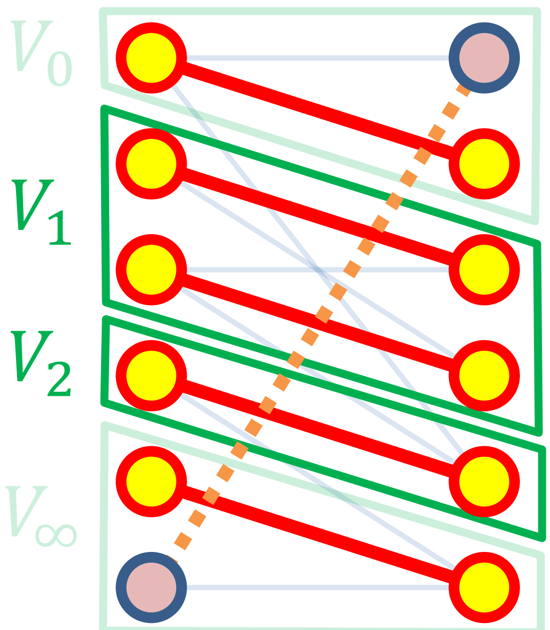
DM-decomposition



<u>Idea</u>

Connect Exposed Vertices to **Reduce to Case 1**

Case 2. NO Perfect Matching



DM-decomposition

Maximum Matching

↓

+ Perfect Matching

↓

Each V_i $(i \neq 0, \infty)$ remains as it was

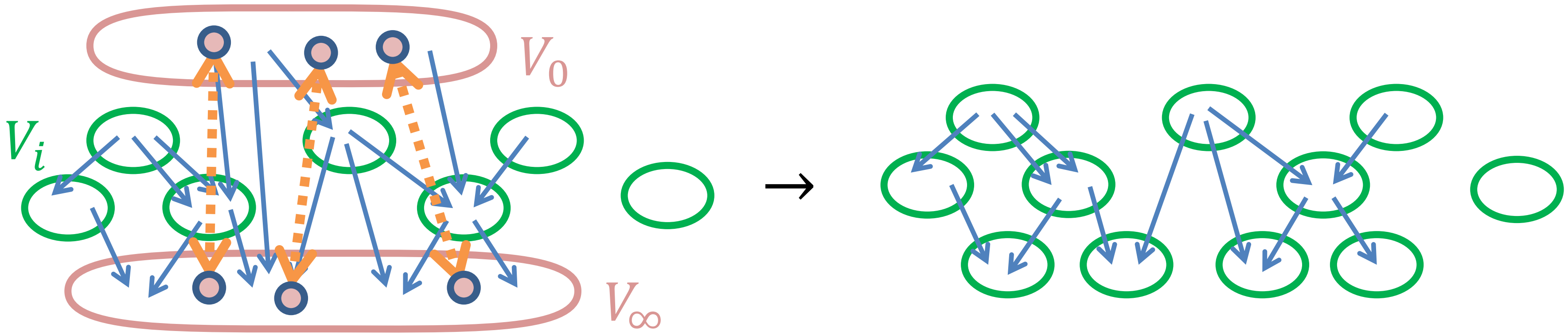
Idea

Connect Exposed Vertices to **Reduce to Case 1**

From the Viewpoint of Oriented Graphs

Idea

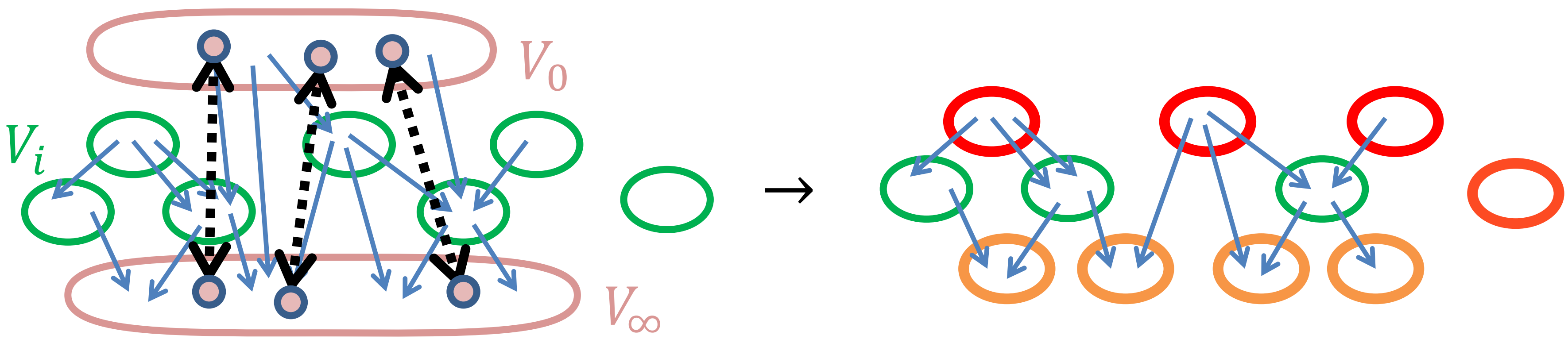
Connect Exposed Vertices to **Reduce to Case 1**



From the Viewpoint of Oriented Graphs

Idea

Connect Exposed Vertices to Reduce to Case 1



$|V^+| - |M|$

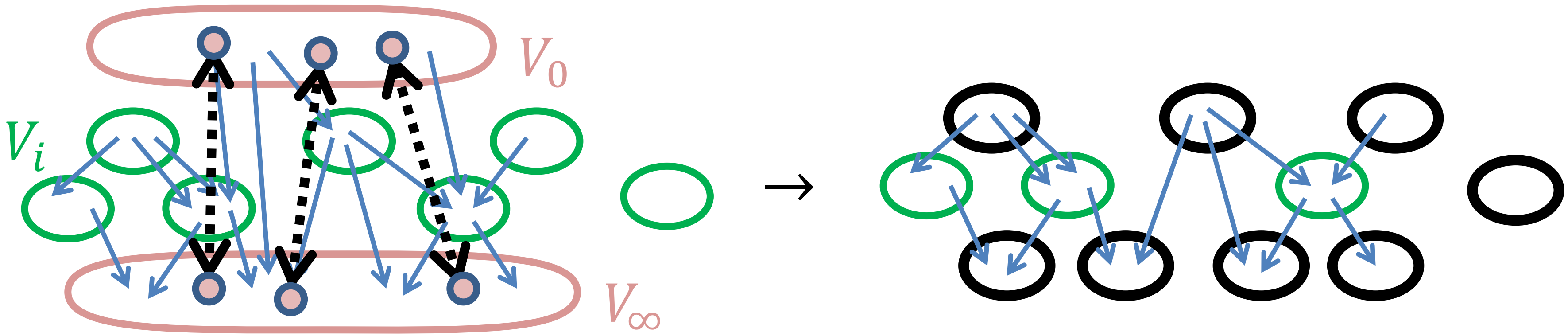
max{# of **Sources**, # of **Sinks**}

of **Additional Edges**

From the Viewpoint of Oriented Graphs

<u>Idea</u>

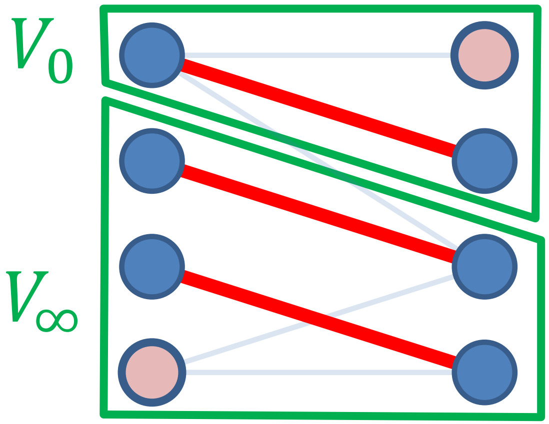
Connect Exposed Vertices to **Reduce to Case 1**



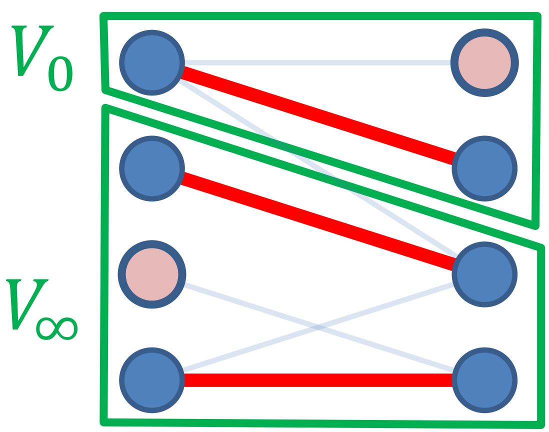
$|V^+| - |M|$ | $\max\{$# of **Sources**, # of **Sinks**$\}$

Const. | # of **Additional Edges** | **Depending on M**

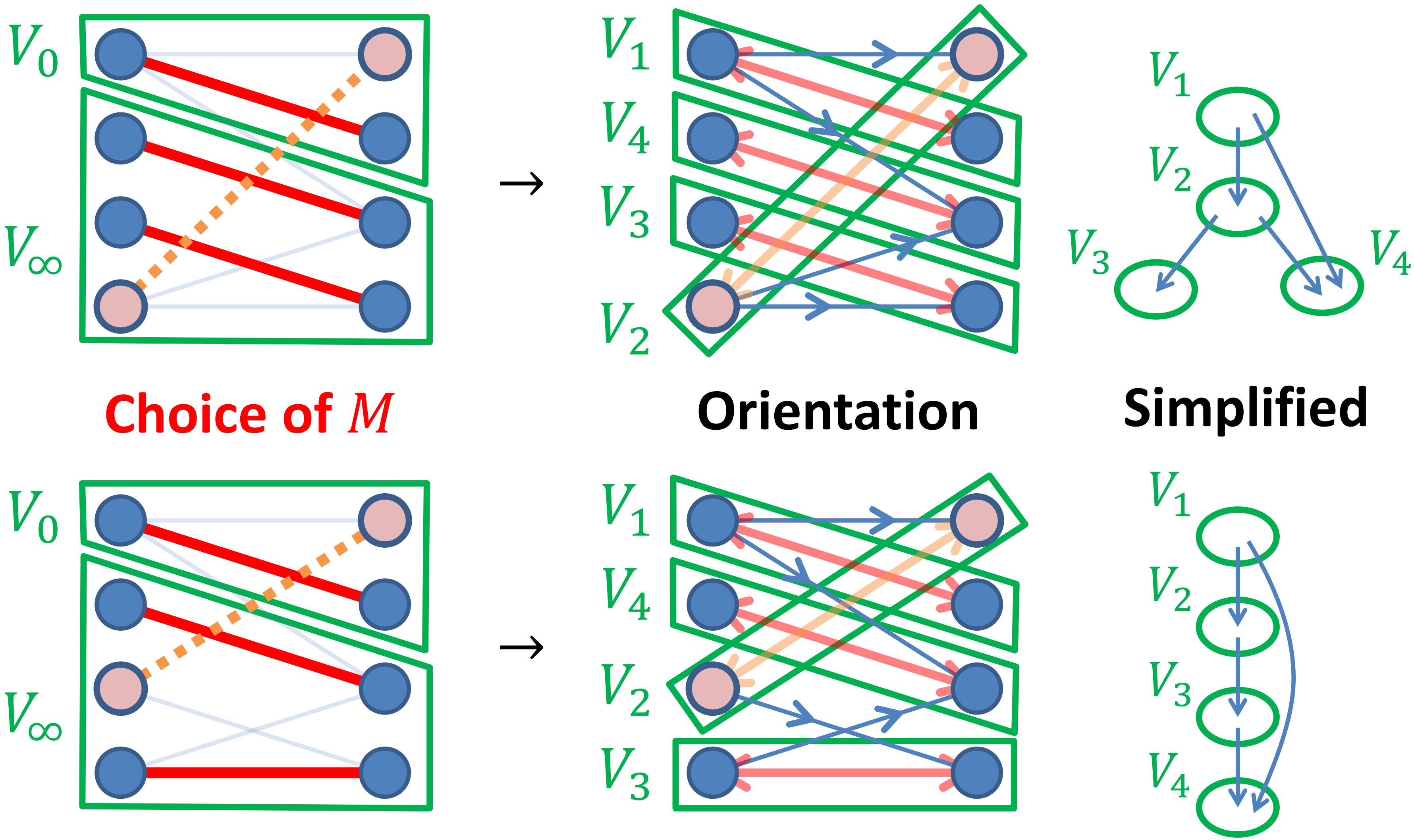
Sources and Sinks in Resulting Digraph



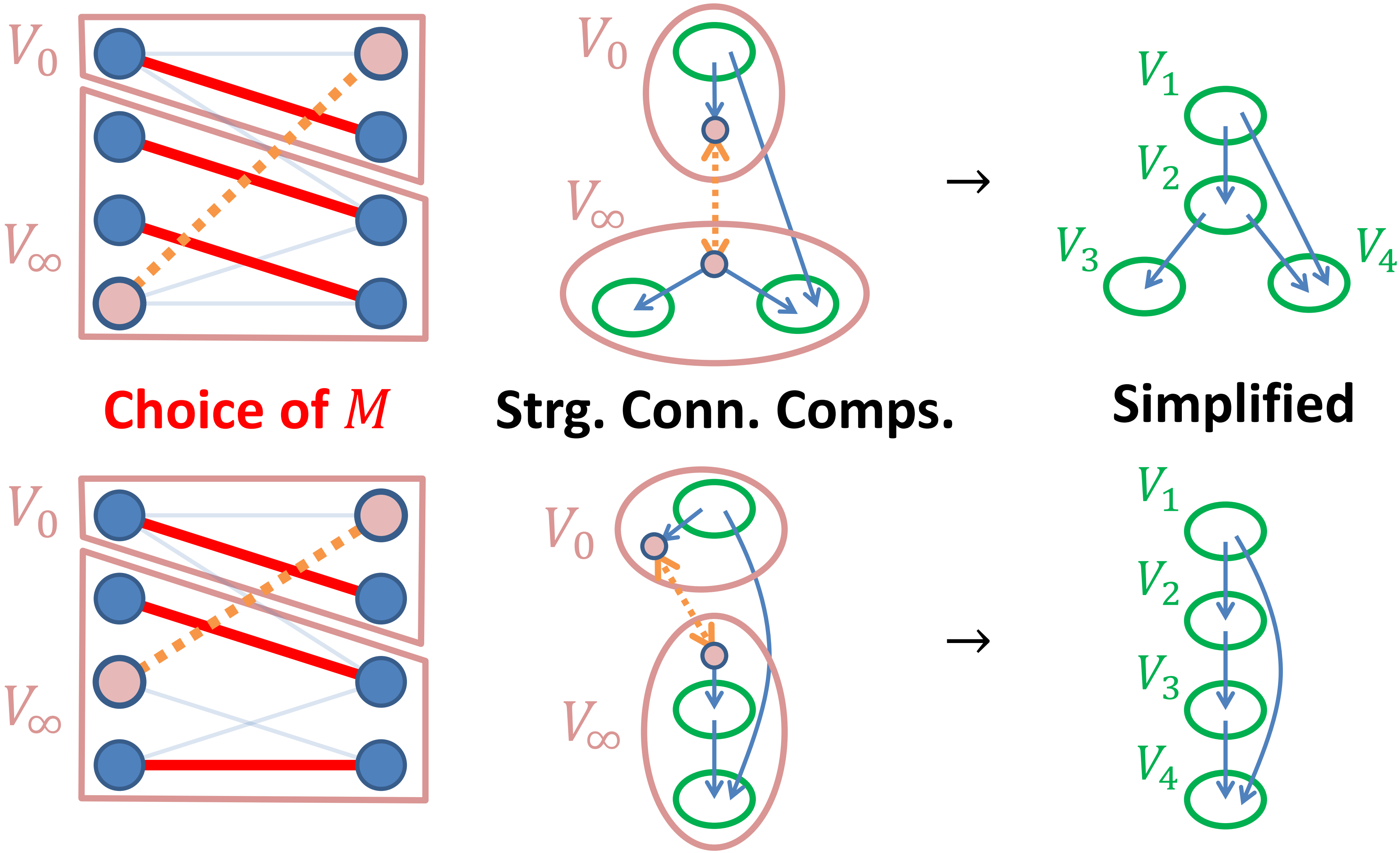
Choice of M



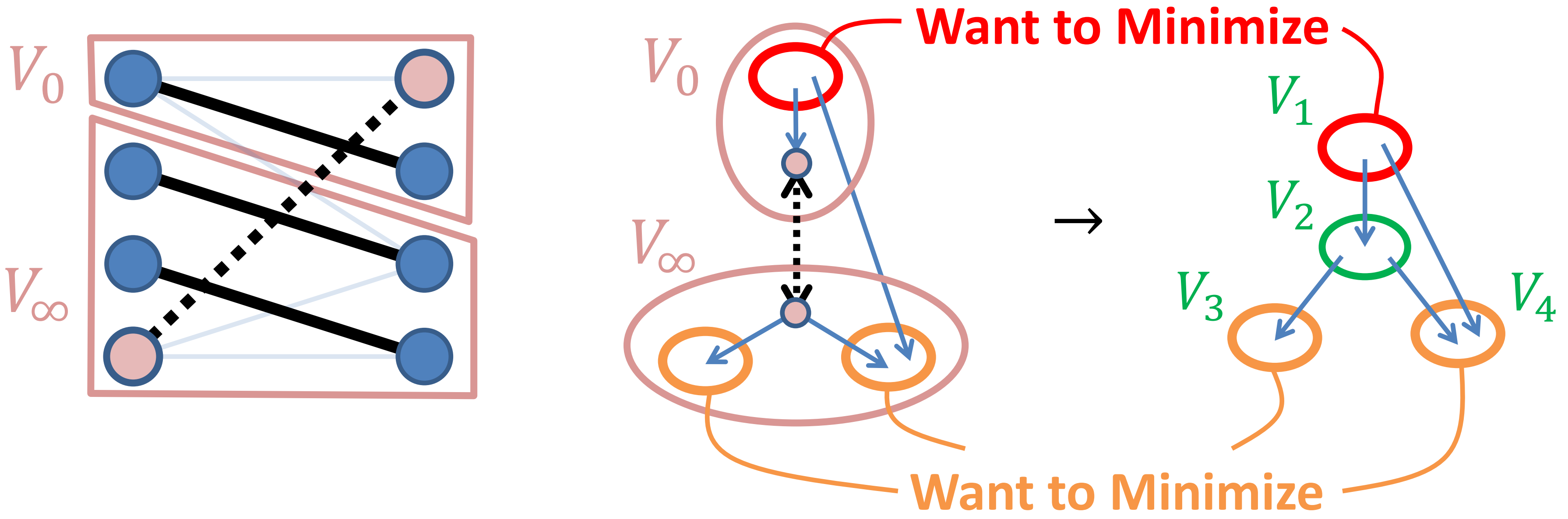
Sources and Sinks in Resulting Digraph



Sources and Sinks in Resulting Digraph



Sources and Sinks in Resulting Digraph

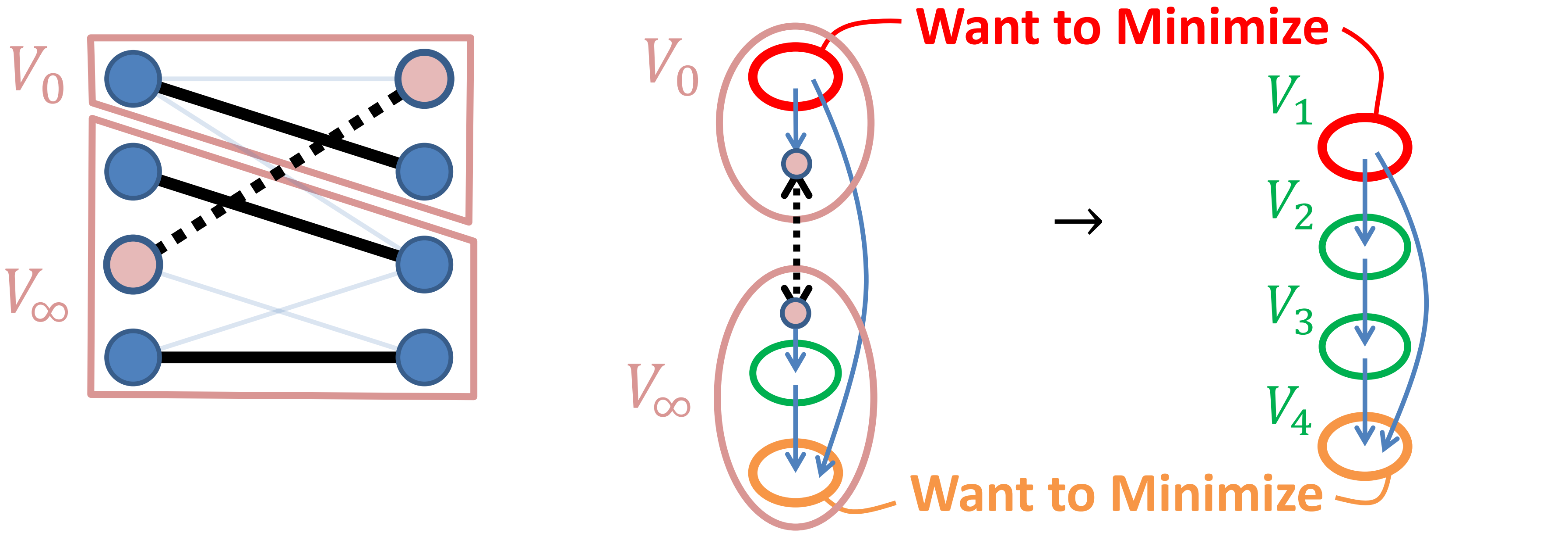


Obs.

(# of **Resulting Sources**) = (# of **Sources in V_0**) + const.

(# of **Resulting Sinks**) = (# of **Sinks in V_∞**) + const.

Sources and Sinks in Resulting Digraph



Obs.

(# of **Sources in V_0**) and (# of **Sinks in V_∞**) vary **Indep.** by choices of **Perfect Matchings** in $G[V_0]$ and $G[V_\infty]$.

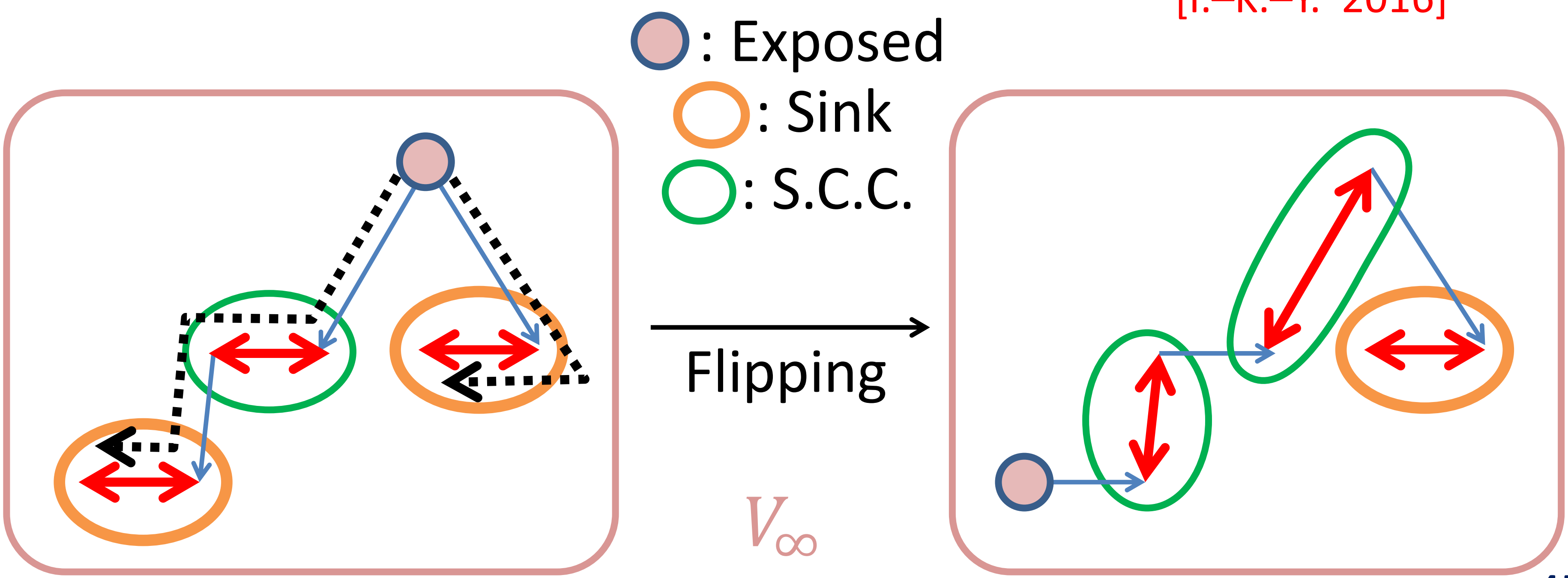
How to Minimize (# of Sinks in V_∞)

Lem. (# of Sinks in V_∞) is **NOT Minimized**

$\Updownarrow$

∃**Edge-disjoint Paths** from ∃● to ∃○$_1$, ○$_2$

[I.–K.–Y. 2016]



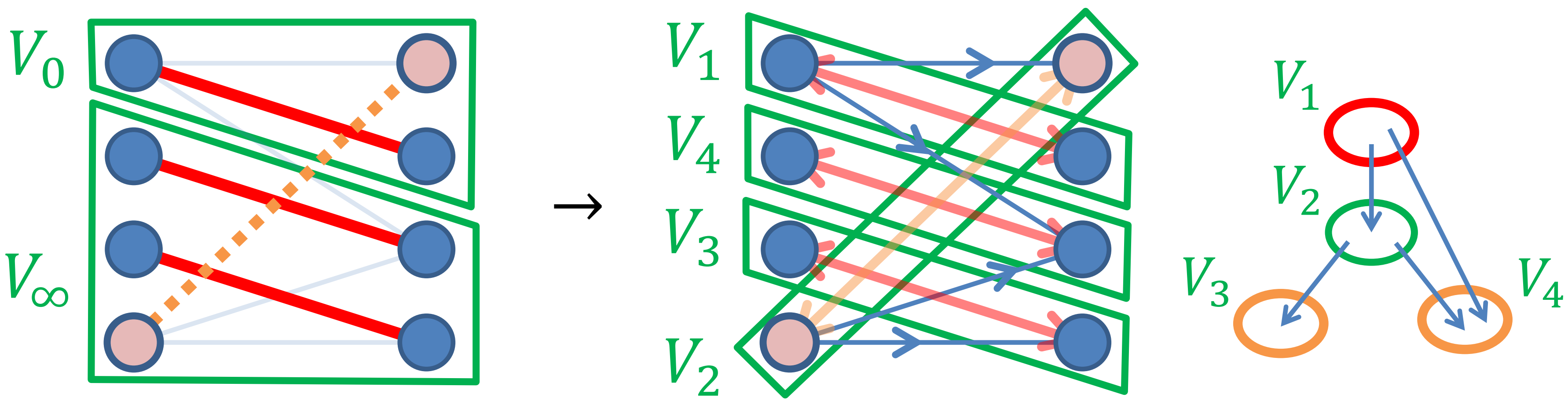
Summary of Case 2

Case 2. G has NO Perfect Matching

- Connect Exposed Vertices to Make **Perfect Matching**

  → Reduce to Case 1

$\text{OPT} = \max\{$# of **Sources**, # of **Sinks**$\}$



Summary of Case 2

Case 2. G has NO Perfect Matching

- Connect Exposed Vertices to Make **Perfect Matching**

  $\rightarrow$ Reduce to Case 1

$$\mathrm{OPT} = \max\{\text{\# of } \textbf{Sources}, \text{\# of } \textbf{Sinks}\}$$

- Minimize (# of **Sources in** V_0) and (# of **Sinks in** V_∞), **in Advance**, by finding Edge-disjoint Paths repeatedly.

Summary of Case 2

Case 2. G has NO Perfect Matching

- Connect Exposed Vertices to Make **Perfect Matching**

  → Reduce to Case 1

  $\mathrm{OPT} = \max\{$# of **Sources**, # of **Sinks**$\}$

- Minimize (# of **Sources in** V_0) and (# of **Sinks in** V_∞), **in Advance**, by finding Edge-disjoint Paths repeatedly.

Thm. One can find an optimal solution by this strategy.

[I.–K.–Y. 2016]

→ Constructive Proof for **Min-Max Duality**

Outline

- **Preliminaries:** How to Compute DM-decomposition
  - Find a **Maximum Matching** in a Bipartite Graph
  - Decompose a Digraph into **Strongly Connected Components**
- **Result:** How to Make a Bipartite Graph DM-irreducible
  - Make a Digraph **Strongly Connected**
  - Find **Edge-Disjoint s–t Paths** in a Digraph
- **Conclusion**

Conclusion

- We propose a simple **Polytime Algorithm** for finding a minimum number of **Additional Edges** to make a Bipartite Graph **DM-irreducible**
- Our Algorithm is based on two elementary techniques:
  - Find **Edge-disjoint** s–t **Paths** in a **Directed Graph**
  - Make a Digraph **Strongly Connected** by Adding Edges
- The Halting Condition of Our Algorithm implies **Min-Max Duality** extending [Eswaran–Tarjan 1976]