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

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

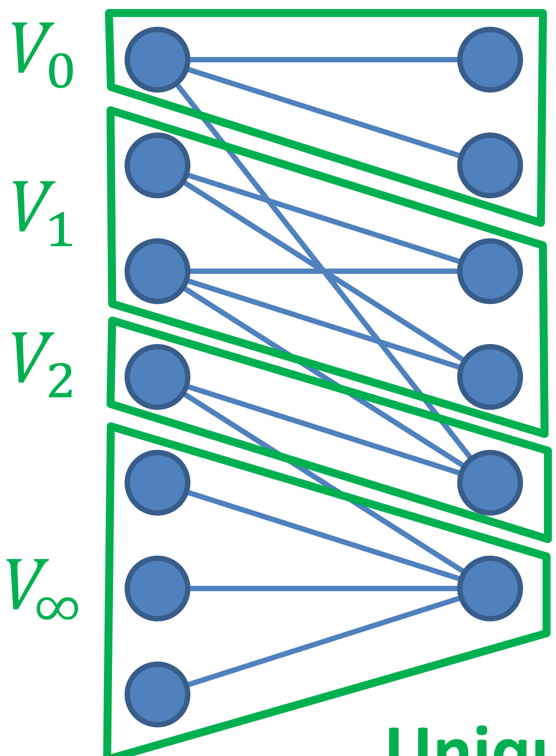
1. University of Tokyo, Japan.
2. TOYOTA Motor Corporation, Japan.
3. Osaka University, Japan.

Shonan Meeting 071 @Shonan April 12, 2016

Dulmage–**M**endelsohn Decomposition

[Dulmage–Mendelsohn 1958,59]

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



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

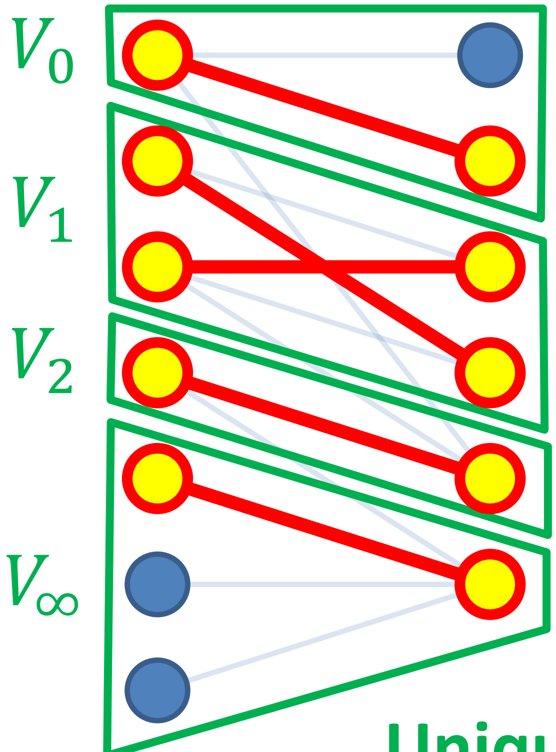
Unique Partition of Vertex Set

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 $V_\infty = \emptyset$

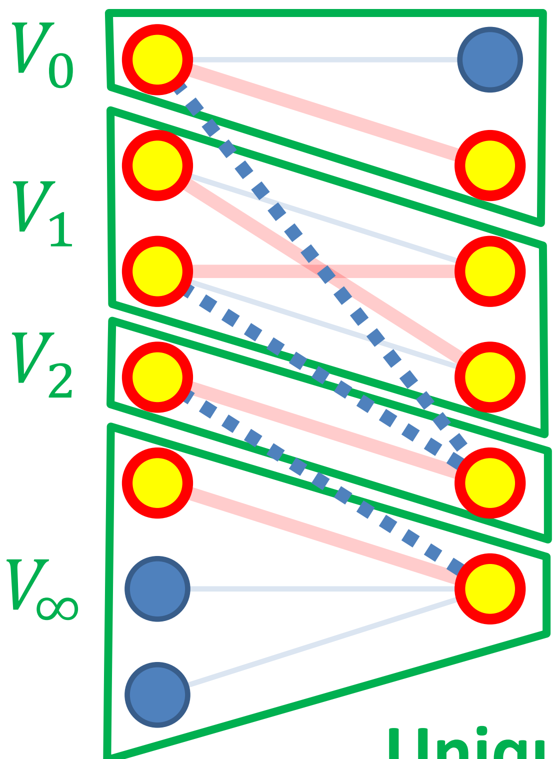
- $\forall$**Max. Matching** in G is a union of **Perfect Matchings** in $G[V_i]$

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.

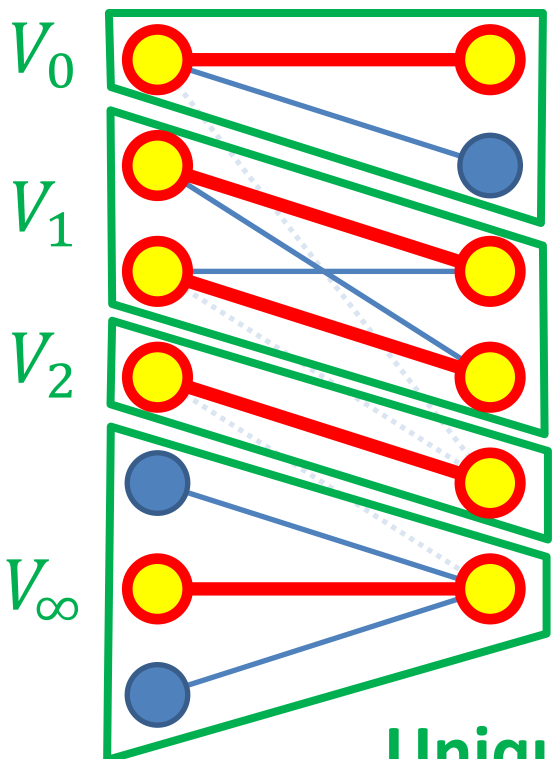
Unique Partition of Vertex Set

reflecting Structure of **Maximum Matchings**

Dulmage–**M**endelsohn Decomposition

[Dulmage–Mendelsohn 1958,59]

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



- $\forall$**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.

- $\forall e$: Edge in $G[V_i]$, $\exists$**Perfect Matching** in $G[V_i]$ using e

Unique Partition of Vertex Set

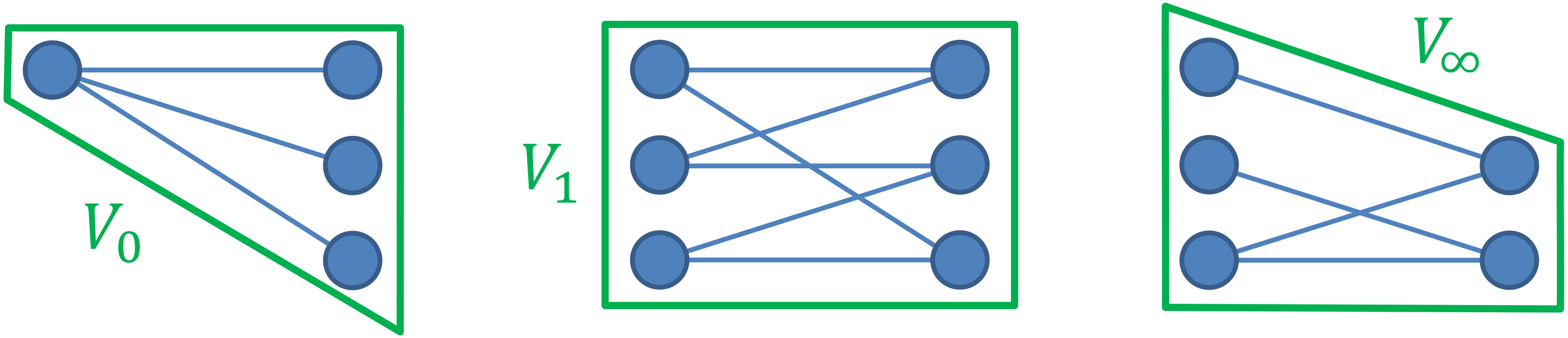
reflecting Structure of **Maximum Matchings**

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**

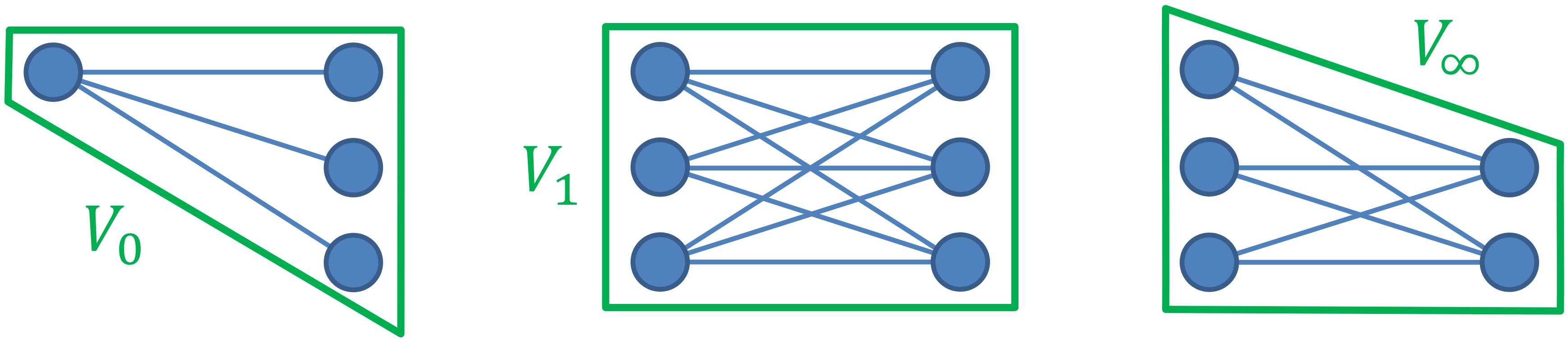
$\Downarrow$

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

DM-irreducibility

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

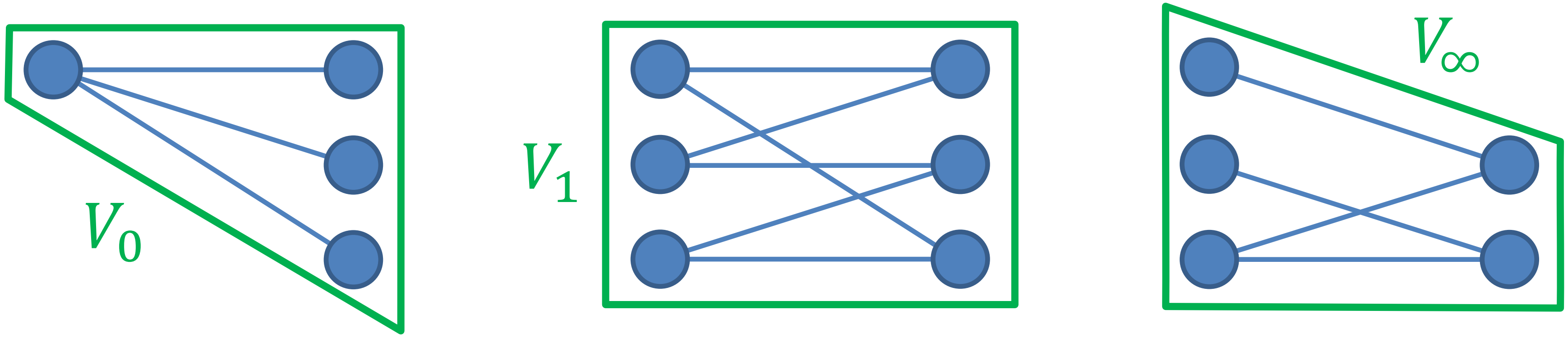
- Connected
- Every Edge is in some Perfect Matching



DM-irreducibility

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

- Connected
- Every Edge is in some Perfect Matching

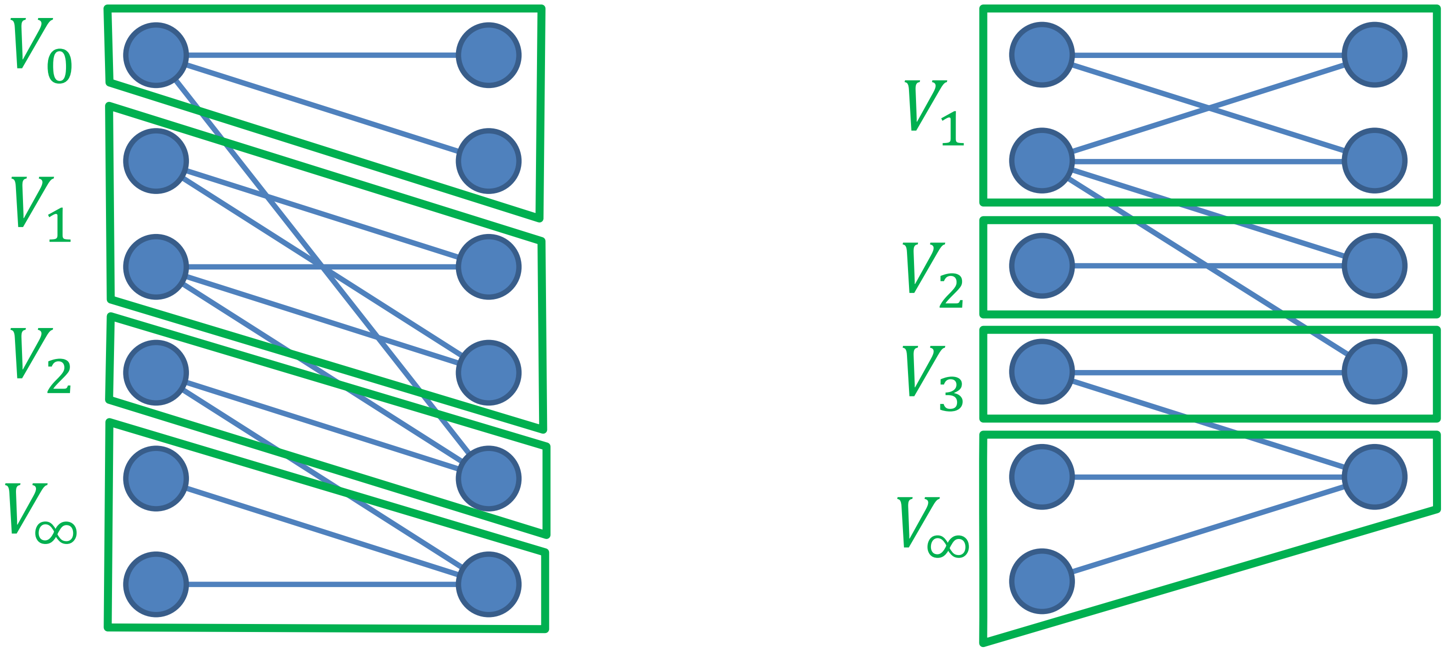


Complete $\overset{\Longrightarrow}{\not\Longleftarrow}$ **DM-irreducible**

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

Our Problem

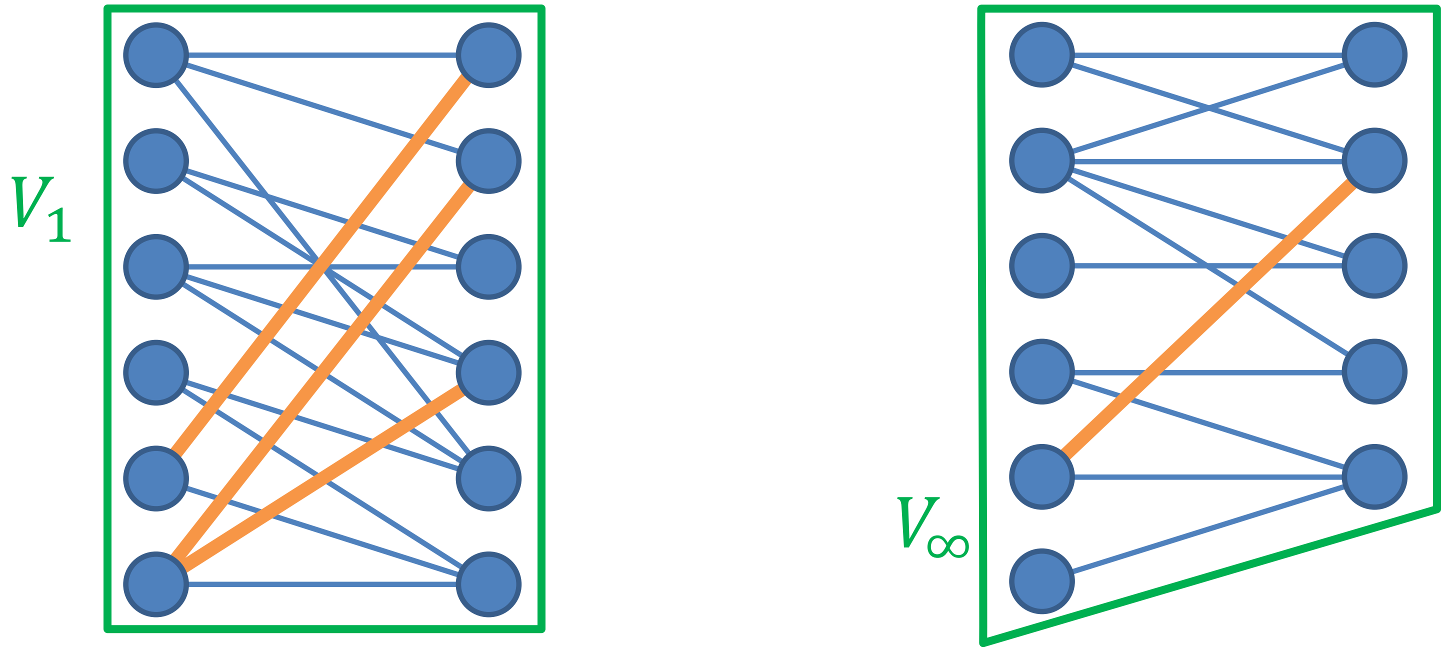
Given $G = (V^+, V^-; E)$: Bipartite Graph



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**

Our Result

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

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

Thm. This problem can be solved in polynomial time.

[I.–K.–Y. 2016]

Tools

- Finding a **Maximum Matching** in a **Bipartite Graph**
- Decomposition into **Strongly Connected Components**
- Making a Digraph **Strongly Connected** by Adding Edges
- Finding **Edge-Disjoint** s–t **Paths** in a Digraph

Outline

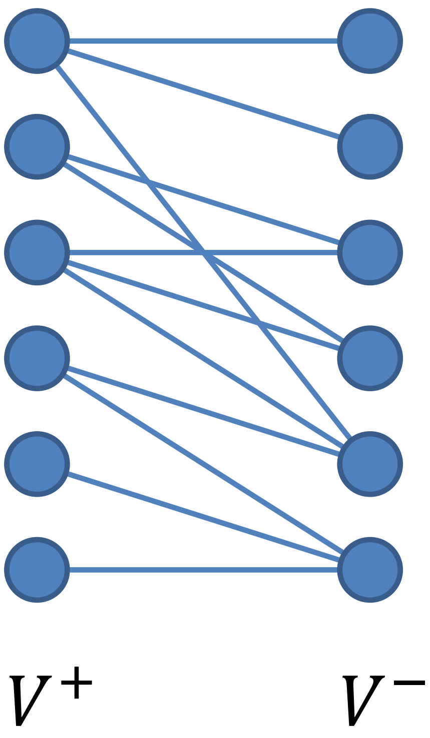
- **Preliminaries:** How to Compute DM-decomposition
  - Find a **Maximum Matching** in a Bipartite Graph
  - Decompose a Digraph into **Strongly Connected Components**
- **Result:** How to Make a Bipartite Graph DM-irreducible
  - Make a Digraph **Strongly Connected**
  - Find **Edge-Disjoint s–t Paths**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

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**

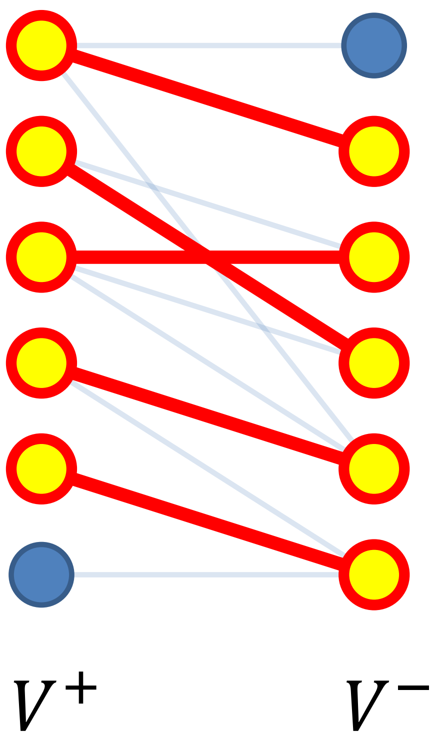
How to Compute DM-decomposition

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



How to Compute DM-decomposition

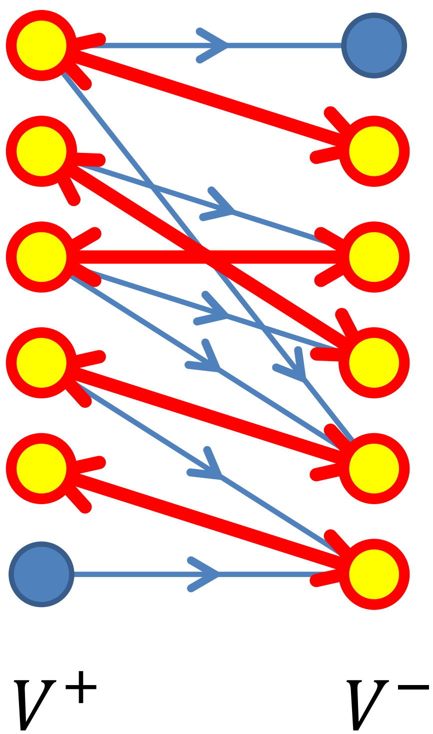
Given $G = (V^+, V^-; E)$: Bipartite Graph



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

How to Compute DM-decomposition

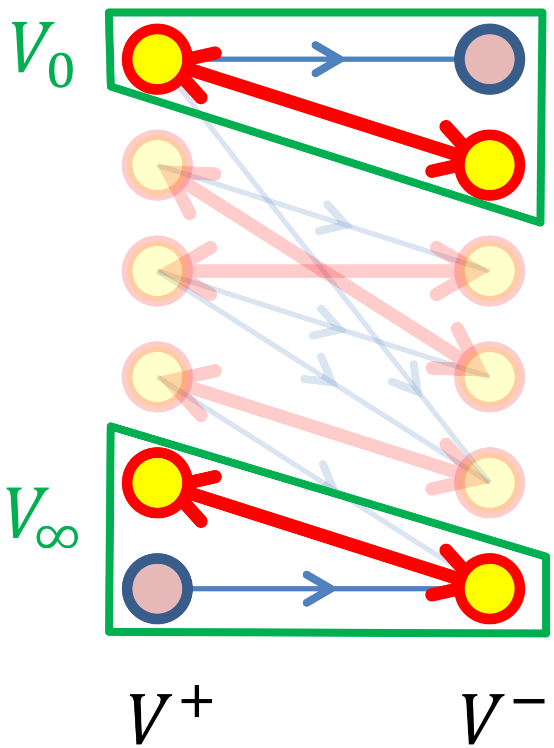
Given $G = (V^+, V^-; E)$: Bipartite Graph



- 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

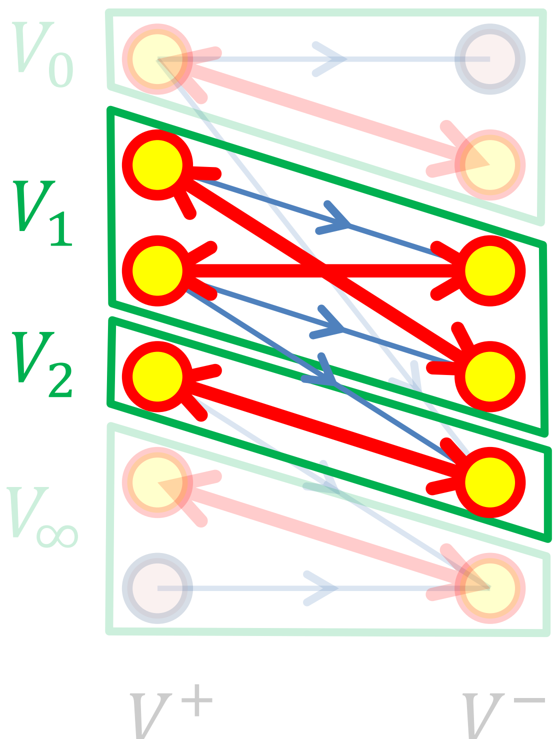
Given $G = (V^+, V^-; E)$: Bipartite Graph



- 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



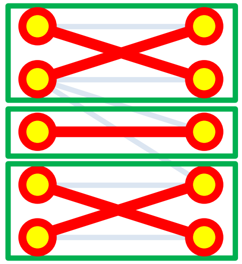
- Find a **Maximum Matching** M in G
- **Orient Edges** so that
  $M \implies$ Both Directions $\leftrightarrow$
  $E \setminus M \implies$ Left to Right $\rightarrow$
- V_0: **Reachable to** $V^- \setminus \partial^- M$
- V_∞: **Reachable from** $V^+ \setminus \partial^+ M$
- V_i: **Strongly Connected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

Outline

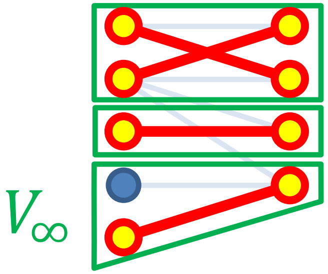
- **Preliminaries:** How to Compute DM-decomposition
  - Find a **Maximum Matching** in a Bipartite Graph
  - Decompose a Digraph into **Strongly Connected Components**
- **Result:** How to Make a Bipartite Graph DM-irreducible
  - Make a Digraph **Strongly Connected**
  - Find **Edge-Disjoint s–t Paths**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

Case Analysis

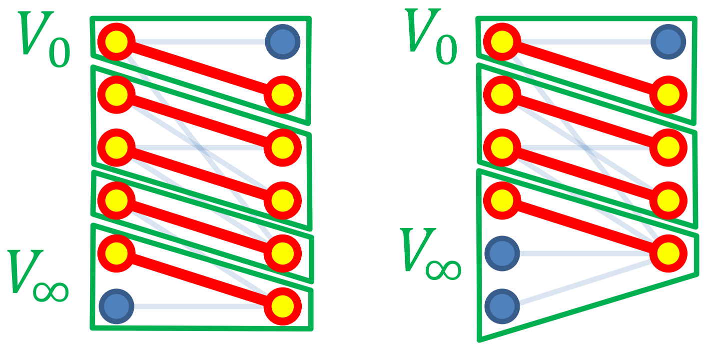
Case 1. When $V_0 = \emptyset = V_\infty$



Case 2. When $V_0 = \emptyset \neq V_\infty$

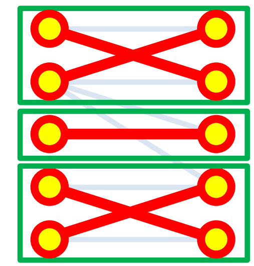


Case 3. When $V_0 \neq \emptyset \neq V_\infty$



Case Analysis

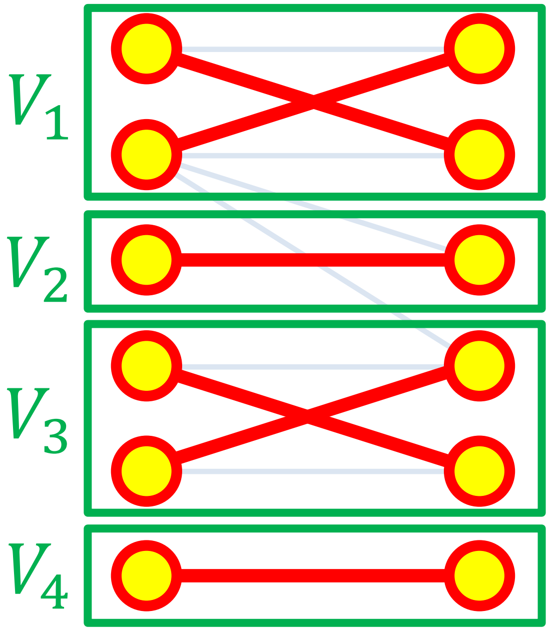
<u>Case 1.</u> When $V_0 = \emptyset = V_\infty$



<u>Case 2.</u> When $V_0 = \emptyset \neq V_\infty$

<u>Case 3.</u> When $V_0 \neq \emptyset \neq V_\infty$

Case 1. When $V_0 = \emptyset = V_\infty$



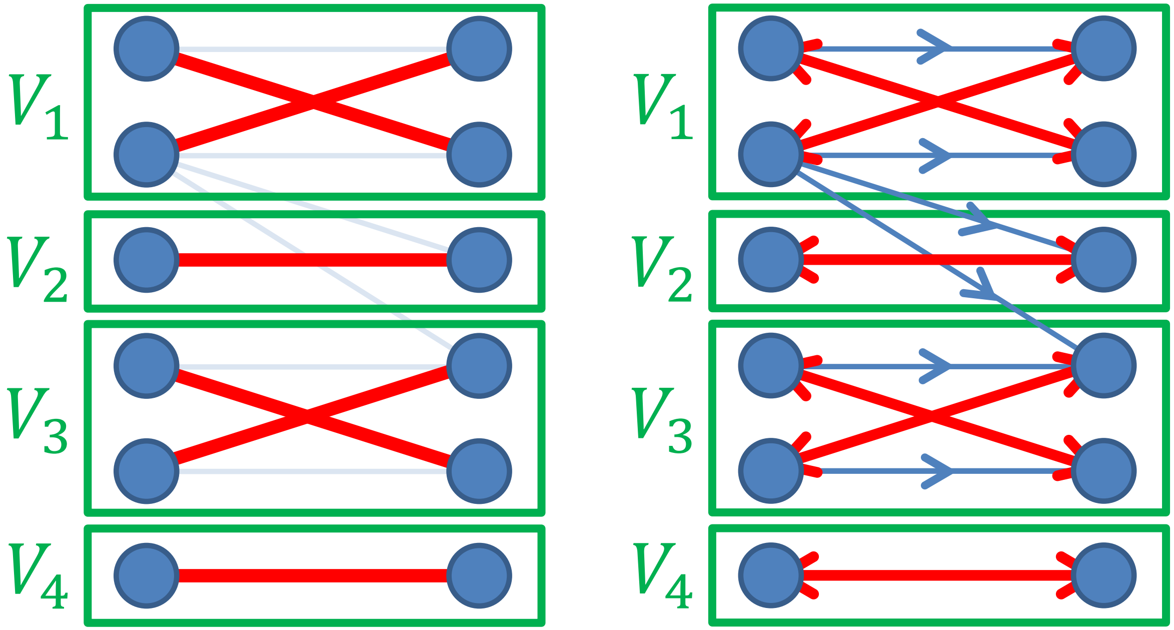
DM-decomposition

- $|V_i^+| = |V_i^-| \quad (i \neq 0, \infty)$
- $\forall$**Max. Matching** in G is a union of **Perfect Matchings** in $G[V_i]$

↓

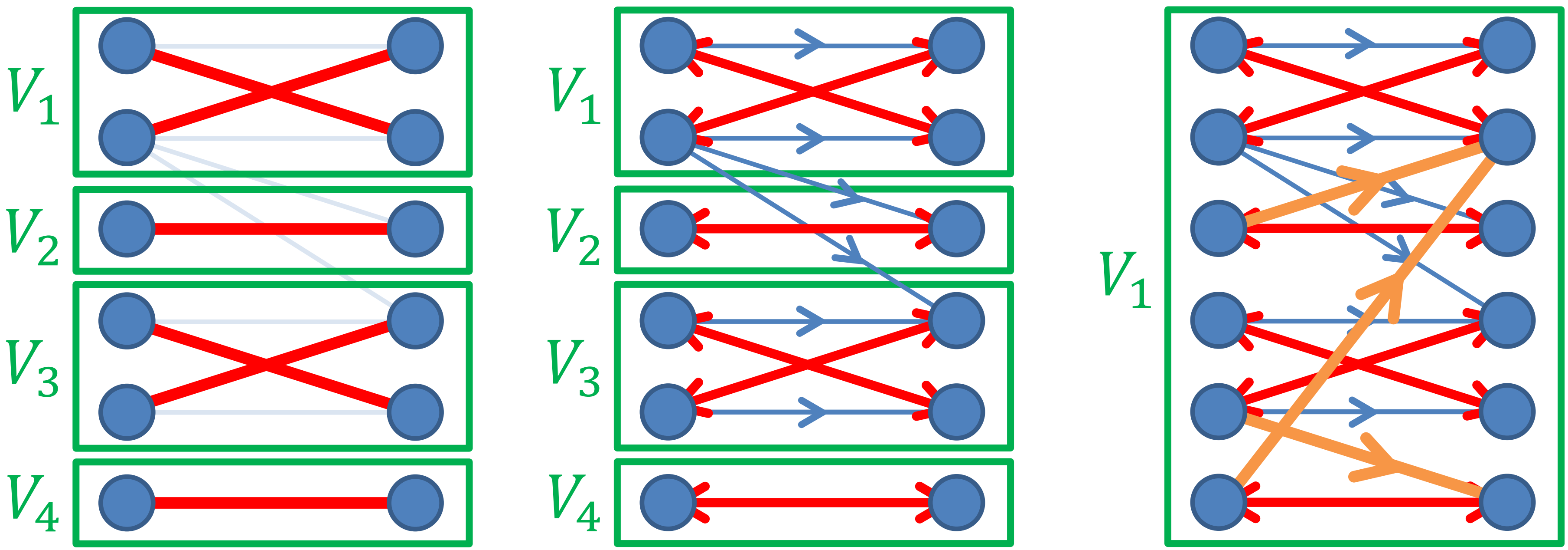
- $|V^+| = |V^-|$
- G has a **Perfect Matching**

Case 1. When $V_0 = \emptyset = V_\infty$



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

Case 1. When $V_0 = \emptyset = V_\infty$

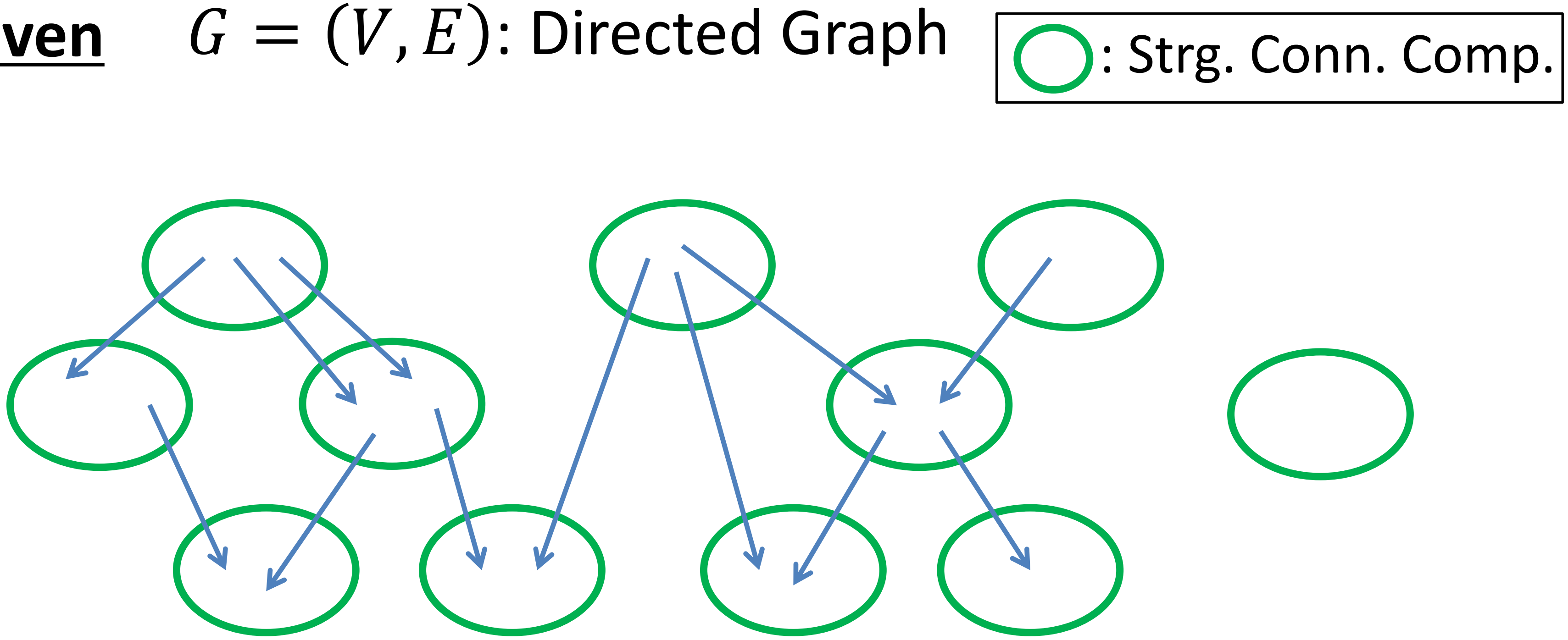


DM-decomposition $=$ Strg. Conn. Comps. $\rightarrow$ **Make it Strg. Conn. by Adding Edges**

<u>Obs.</u> <u>**DM-irreducibility**</u> is **Equivalent to** <u>**Strong Connectivity**</u> of the Oriented Graph

How to Make a Digraph Strongly Connected

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



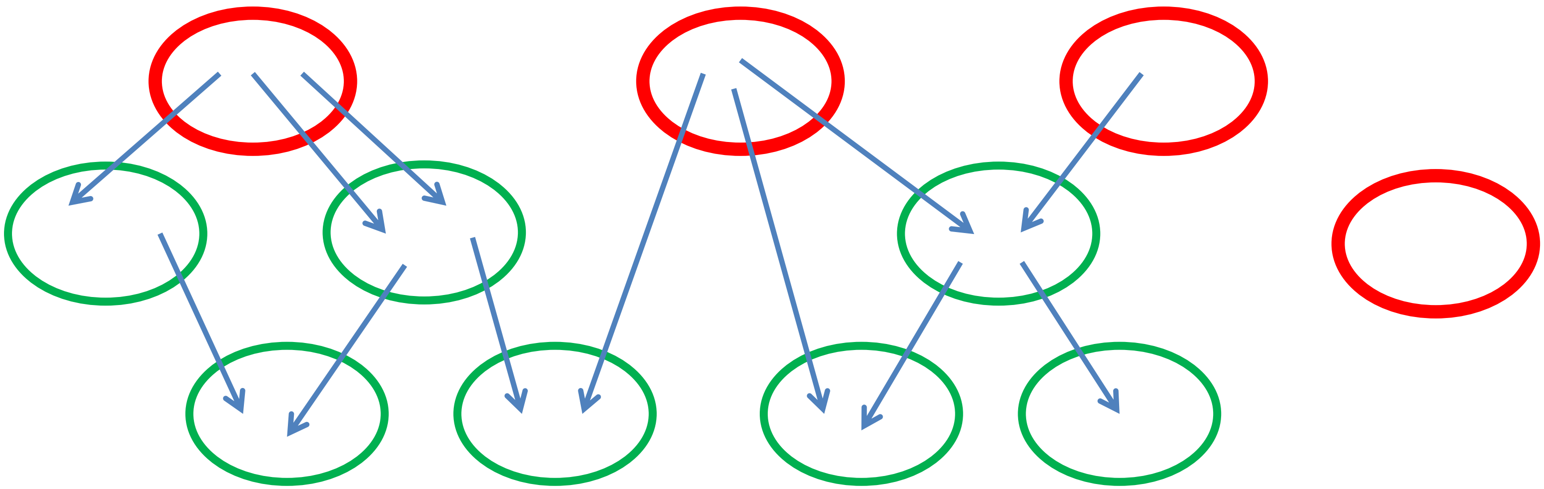
Find **Minimum Number of Additional Edges**
to Make G **Strongly Connected**

How to Make a Digraph Strongly Connected

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

: Strg. Conn. Comp.

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

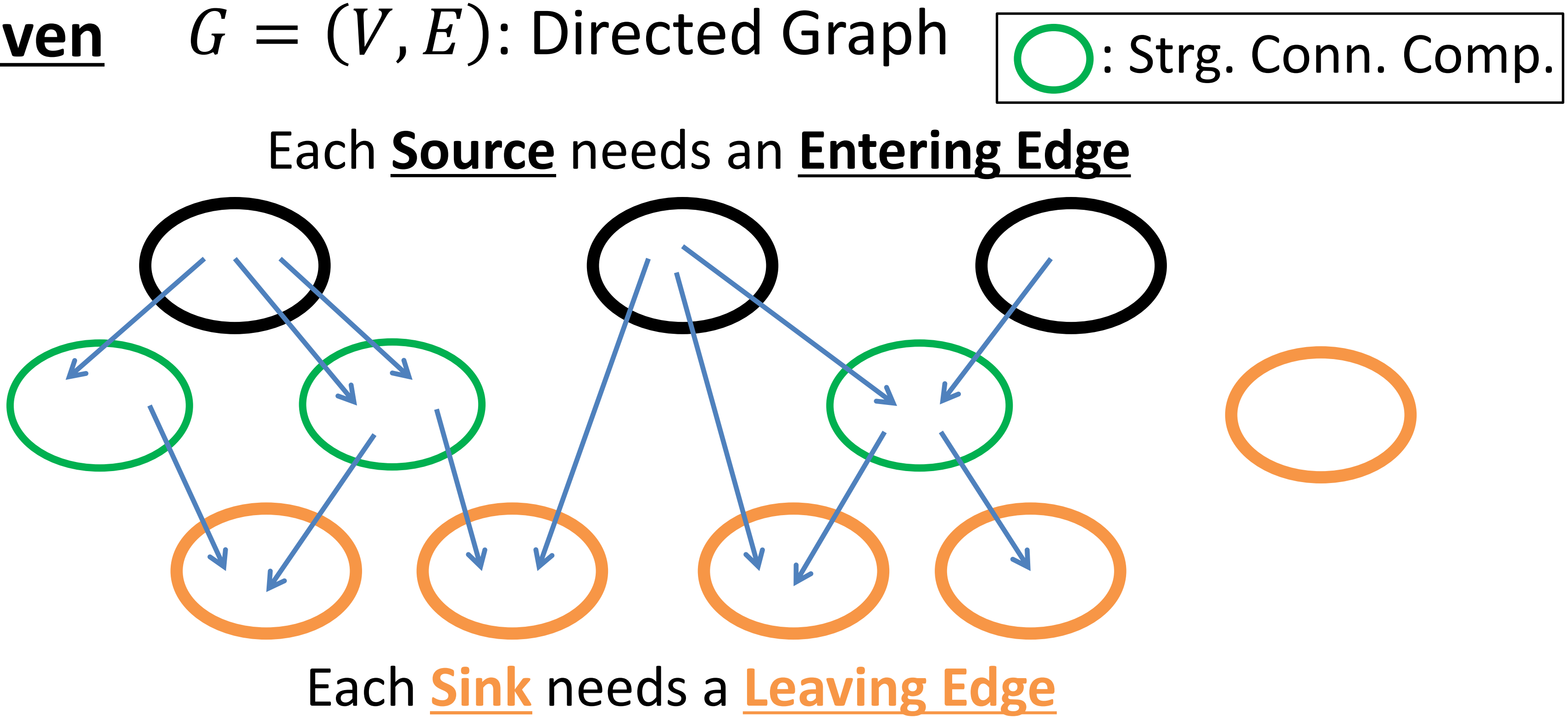


Find **Minimum Number of Additional Edges**

to Make G **Strongly Connected**

How to Make a Digraph Strongly Connected

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



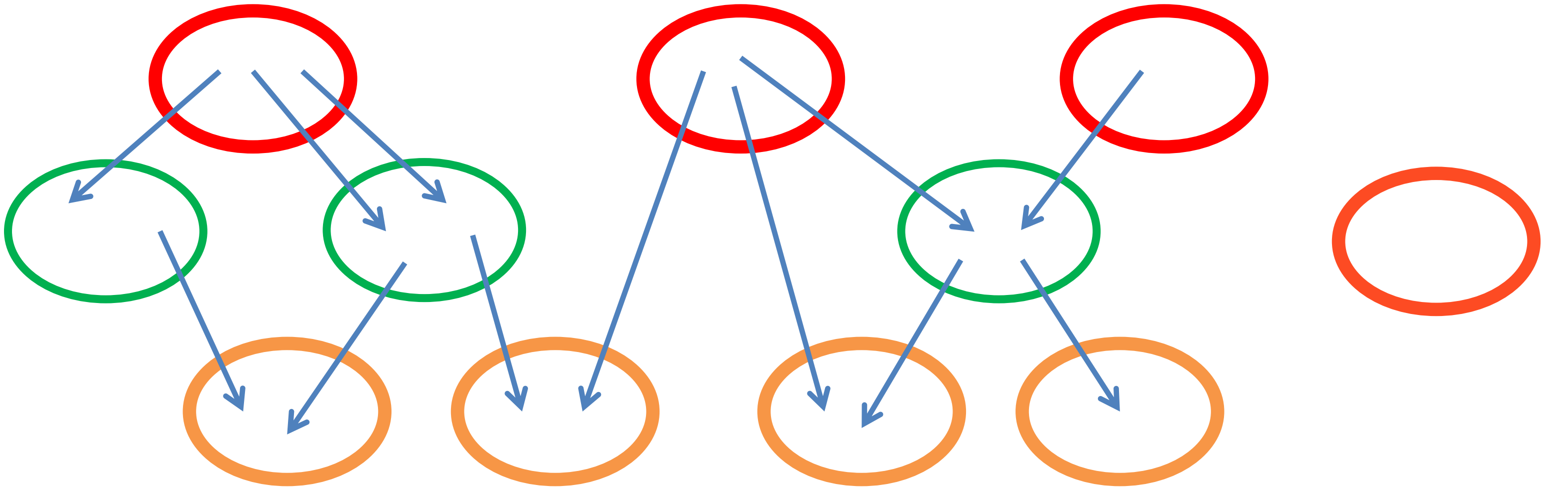
Find **Minimum Number of Additional Edges**
to Make G **Strongly Connected**

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**.
∃ **Polytime Algorithm** to find such Additional Edges.

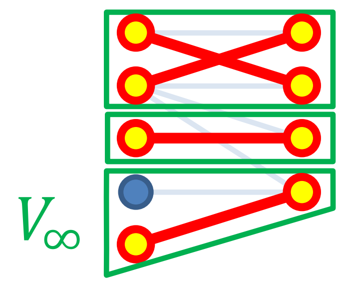
[Eswaran–Tarjan 1976]

→ Case 1 is **Polytime Solvable**.

Case Analysis

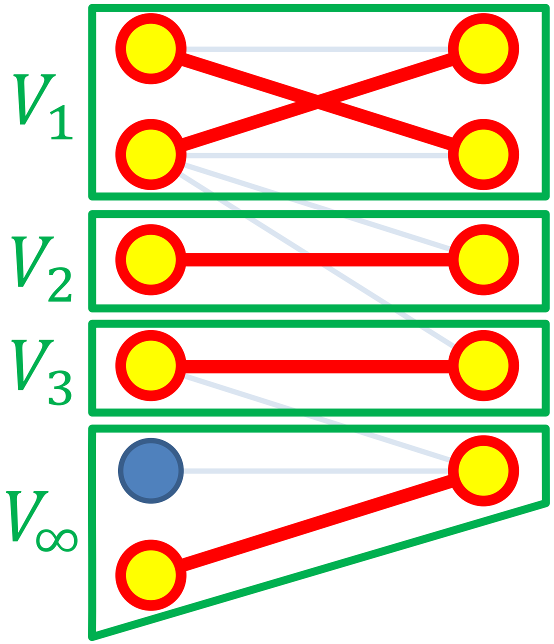
Case 1. When $V_0 = \emptyset = V_\infty$

Case 2. When $V_0 = \emptyset \neq V_\infty$



Case 3. When $V_0 \neq \emptyset \neq V_\infty$

Case 2. When $V_0 = \emptyset \neq V_\infty$



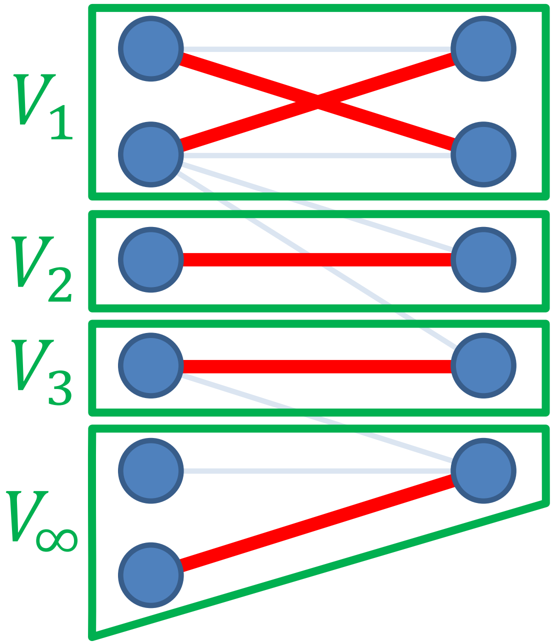
DM-decomposition

- $|V_i^+| = |V_i^-| \quad (i \neq 0, \infty)$
- $|V_\infty^+| > |V_\infty^-|$
- $\forall$**Max. Matching** in G is a union of **Perfect Matchings** in $G[V_i]$

↓

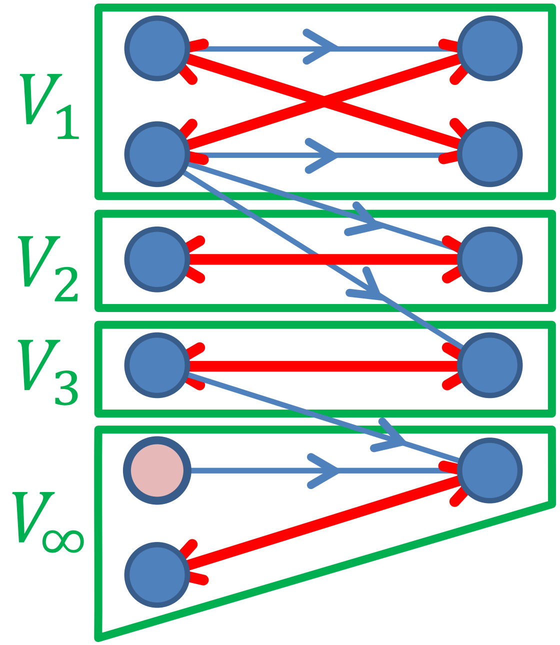
- $|V^+| > |V^-|$
- G has a **Perfect Matching**

Case 2. When $V_0 = \emptyset \neq V_\infty$



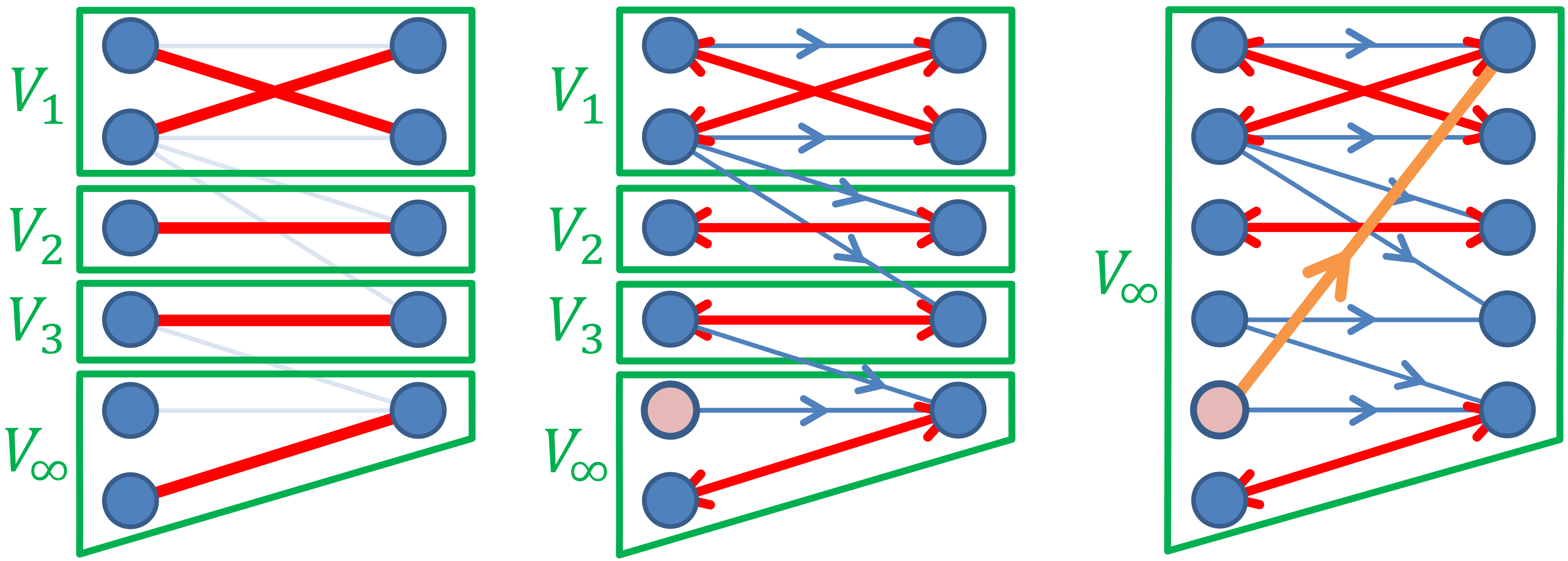
DM-decomposition

$=$



Reachability from
Exposed Vertices
$+$
Strg. Conn. Comps.
of the Rest

Case 2. When $V_0 = \emptyset \neq V_\infty$



DM-decomposition

$=$

Reachability from Exposed Vertices

$+$

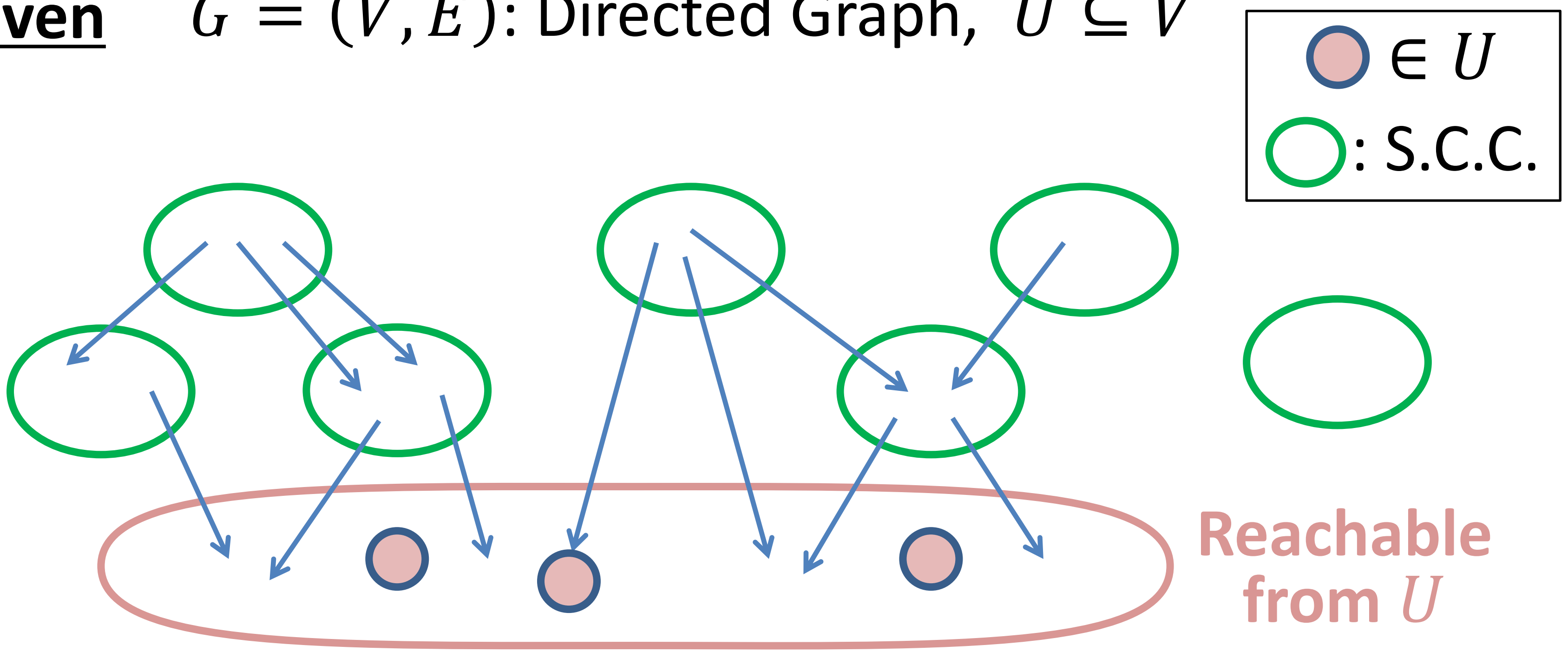
Strg. Conn. Comps. of the Rest

$\rightarrow$

Make ALL Vertices Reachable from Exposed Vertices by Adding Edges

How to Achieve such Reachability

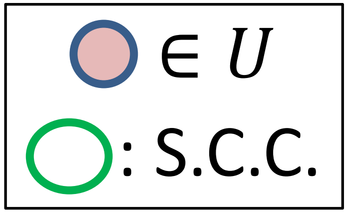
Given $G = (V, E)$: Directed Graph, $U \subseteq V$



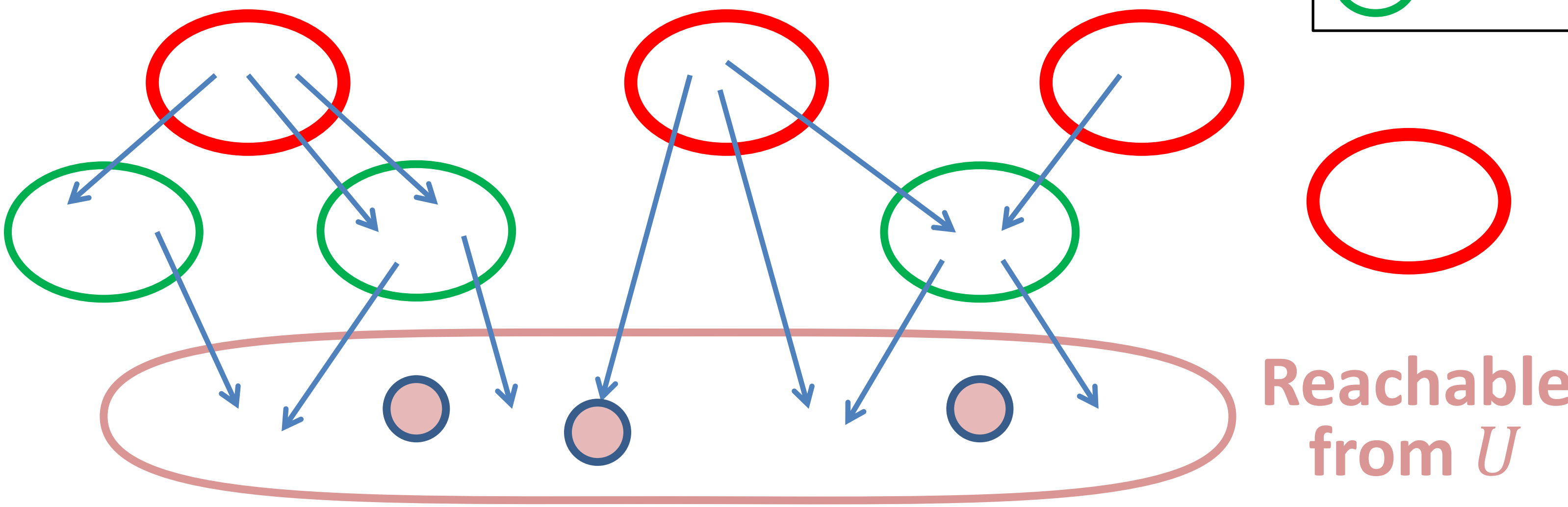
Find **Minimum Number of Additional Edges**
to Make **ALL Vertices** **Reachable from U**

How to Achieve such Reachability

<u>Given</u> $G = (V, E)$: Directed Graph, $U \subseteq V$



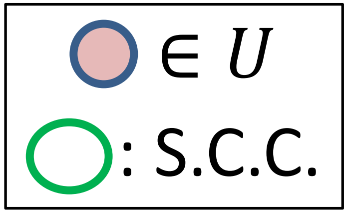
Each **<u>Source</u>** needs an **<u>Entering Edge</u>**



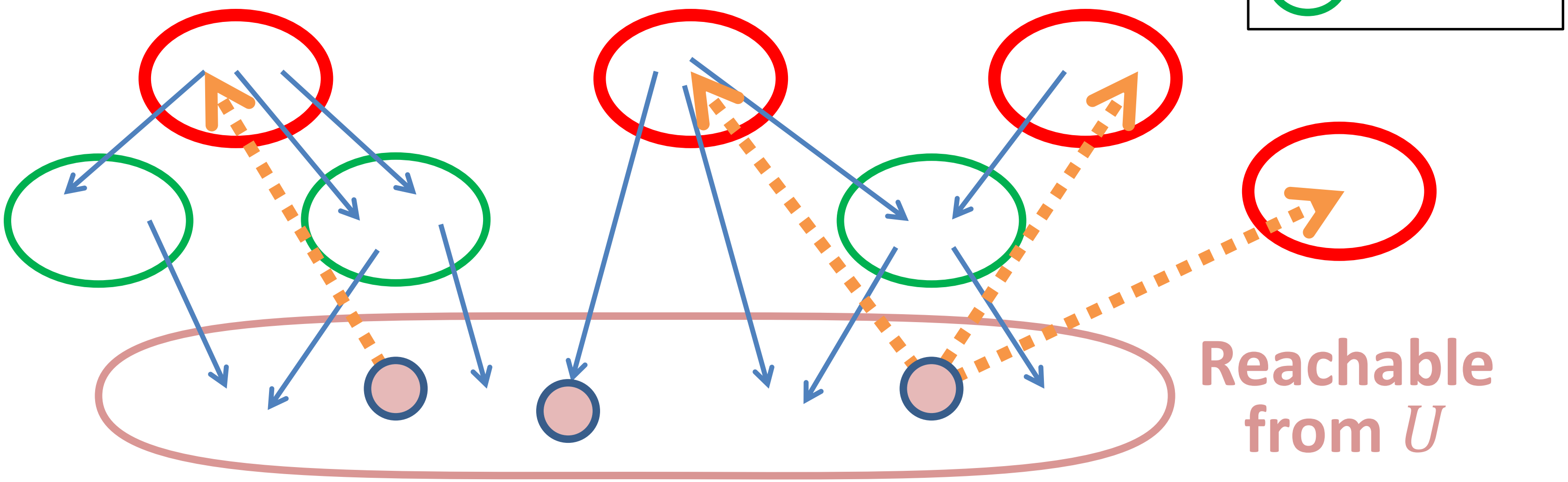
<u>Find</u> **Minimum Number of Additional Edges** to Make **ALL Vertices** **<u>Reachable from U</u>**

How to Achieve such Reachability

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



Sufficient!! Each **Source** needs an **Entering Edge**



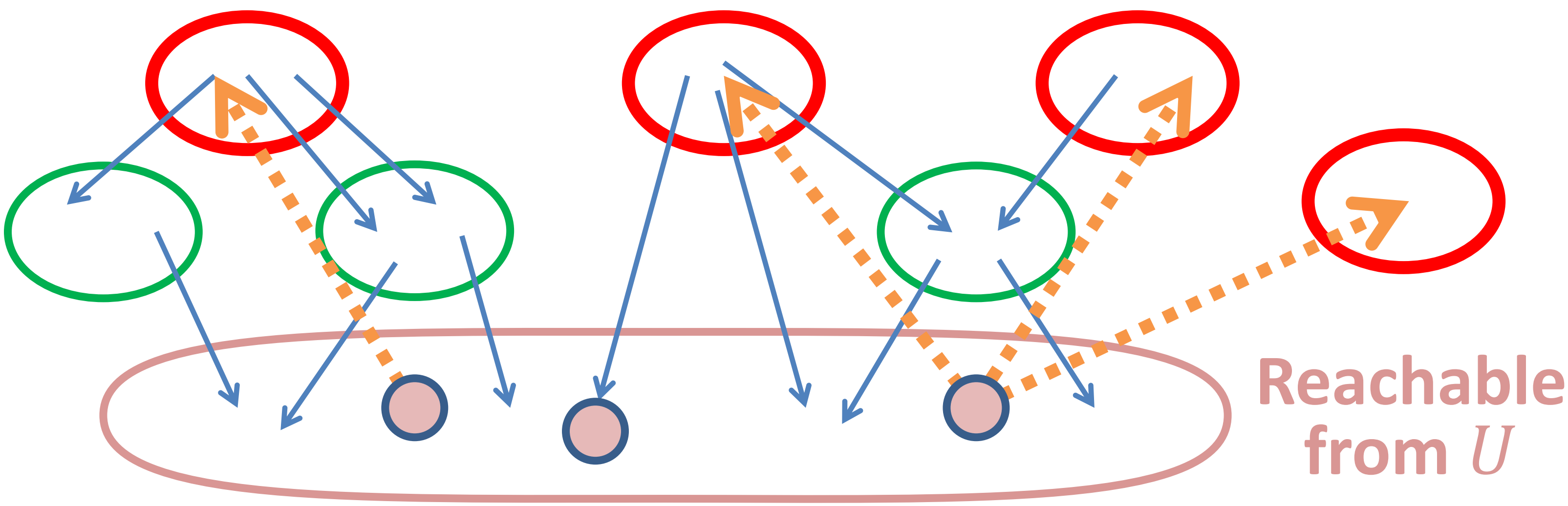
Find **Minimum Number of Additional Edges** to Make **ALL Vertices** **Reachable from U**

How to Make a Digraph Strongly Connected

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

Find **Minimum Number of Additional Edges** to Make **ALL Vertices** **Reachable from U**

Obs. (# of **Sources**) edges are **Necessary and Sufficient.**



Summary of Cases 1 and 2

Case 1. $|V^+| = |V^-|$ and G has a Perfect Matching

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

Case 2. $|V^+| > |V^-|$ and G has a Perfect Matching

$\mathrm{OPT} = ($# of **Sources NOT Reachable from** $V_\infty)$

Case 2'. $|V^+| < |V^-|$ and G has a Perfect Matching

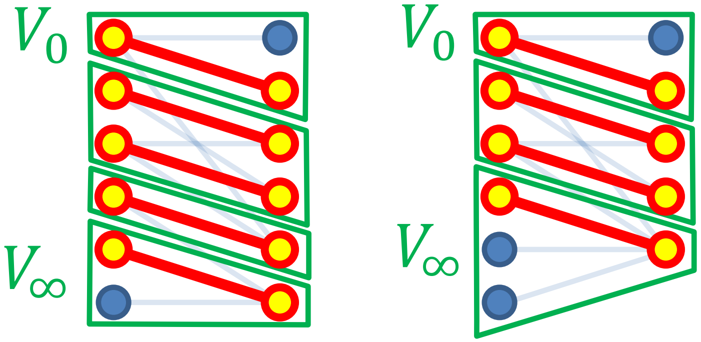
$\mathrm{OPT} = ($# of **Sinks NOT Reachable to** $V_0)$

Case Analysis

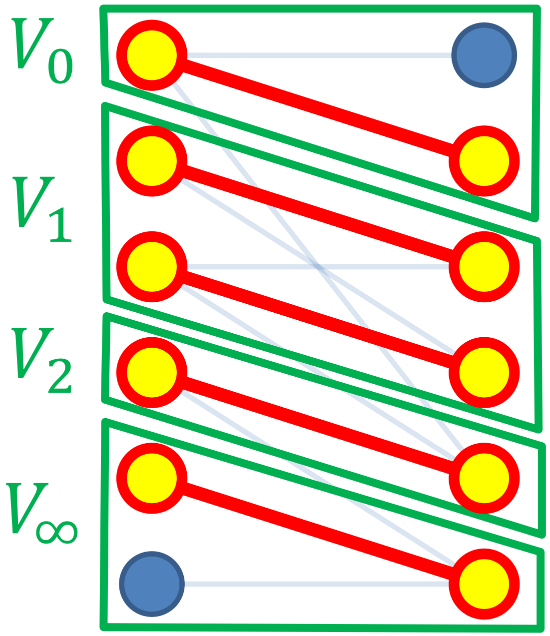
Case 1. When $V_0 = \emptyset = V_\infty$

Case 2. When $V_0 = \emptyset \neq V_\infty$

Case 3. When $V_0 \neq \emptyset \neq V_\infty$



Case 3. When $V_0 \neq \emptyset \neq V_\infty$



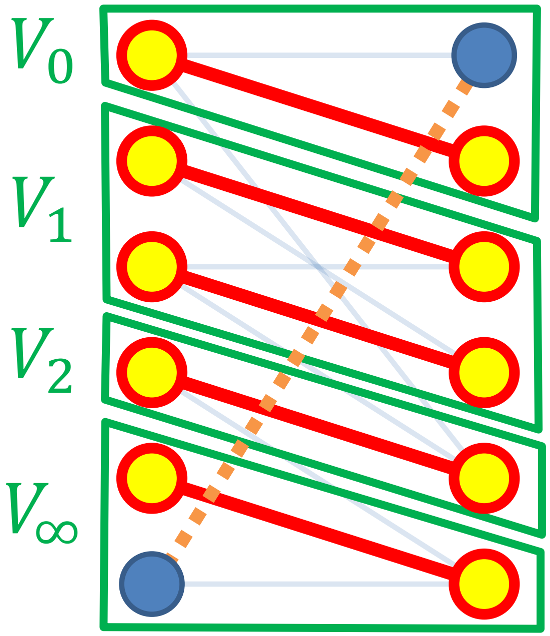
DM-decomposition

- $|V_0^+| < |V_0^-|$
- $|V_i^+| = |V_i^-| \quad (i \neq 0, \infty)$
- $|V_\infty^+| > |V_\infty^-|$
- $\forall$**Max. Matching** in G is a union of **Perfect Matchings** in $G[V_i]$

↓

G has NO **Perfect Matching**

Case 3. When $V_0 \neq \emptyset \neq V_\infty$



DM-decomposition

- $|V_0^+| < |V_0^-|$
- $|V_i^+| = |V_i^-| \quad (i \neq 0, \infty)$
- $|V_\infty^+| > |V_\infty^-|$
- ∀**Max. Matching** in G is a union of **Perfect Matchings** in $G[V_i]$

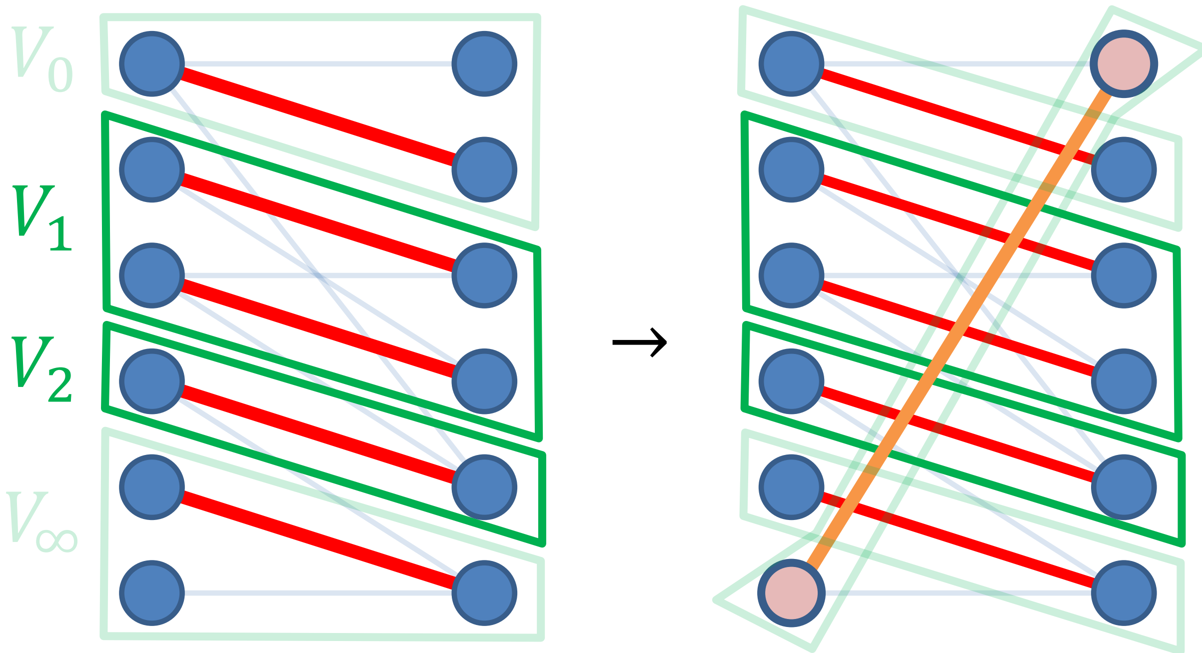
↓

G has NO **Perfect Matching**

<u>Idea</u>

Adding Edges to **Reduce to Cases 1,2** (∃**Perfect Matching**)

Key Observation



Connecting
Exposed Vertices
↓
∃New Max. Matching
including **the Original**
↓
Each V_i $(i \neq 0, \infty)$
remains as it was

DM-decomposition

Idea

Adding Edges to **Reduce to Cases 1,2** (∃**Perfect Matching**)

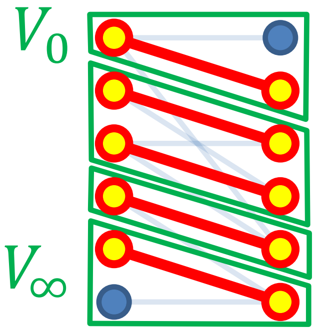
Case Analysis

Case 1. When $V_0 = \emptyset = V_\infty$

Case 2. When $V_0 = \emptyset \neq V_\infty$

Case 3. When $V_0 \neq \emptyset \neq V_\infty$

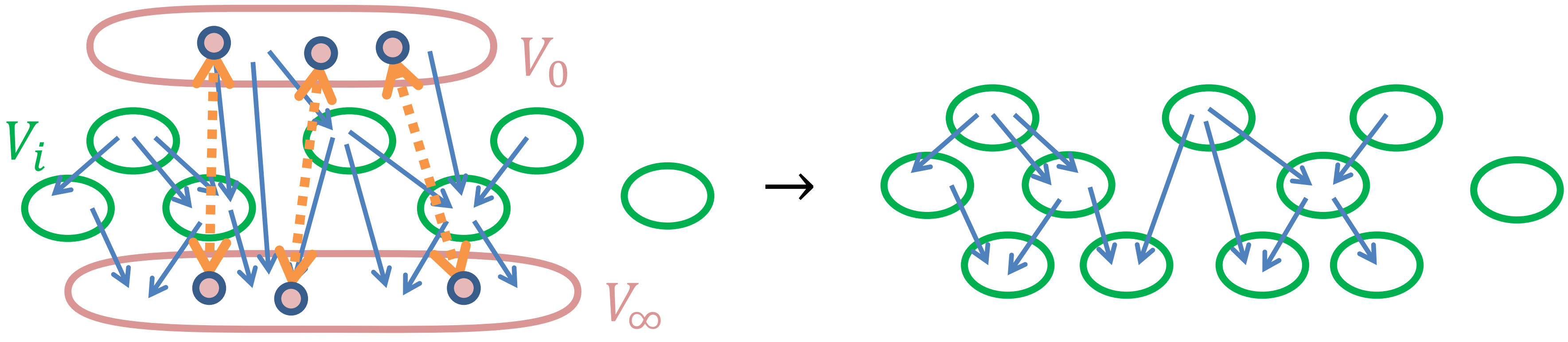
Case 3.1. $|V^+| = |V^-|$



Case 3.1. When $|V^+| = |V^-|$

Idea

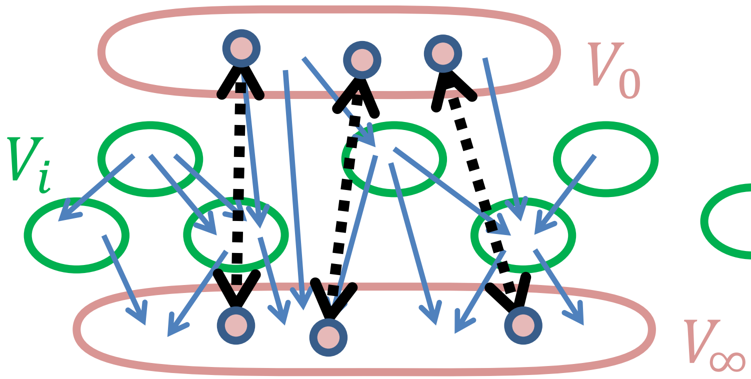
Adding Edges to Reduce to Case 1 ($\exists$**Perfect Matching**)
between Exposed Vertices
in a **Max. Matching** M in G

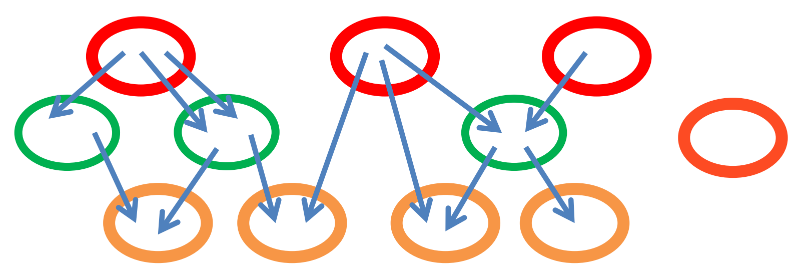


Case 3.1. When $|V^+| = |V^-|$

Idea

Adding Edges to Reduce to Case 1 ($\exists$**Perfect Matching**)
between Exposed Vertices
in a **Max. Matching** M in G





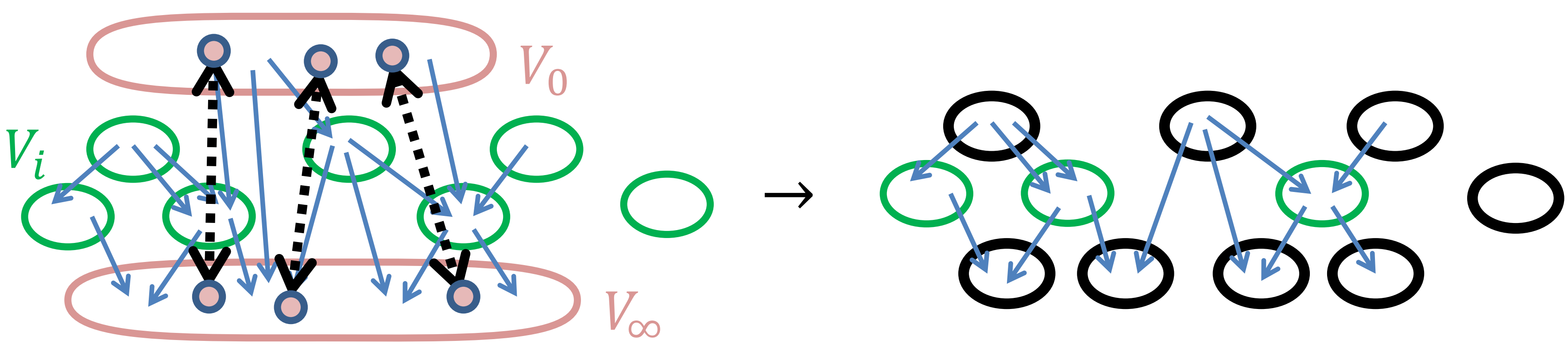
$|V^+| - |M|$ max{# of **Sources**, # of **Sinks**}

of **Additional Edges**

Case 3.1. When $|V^+| = |V^-|$

Idea

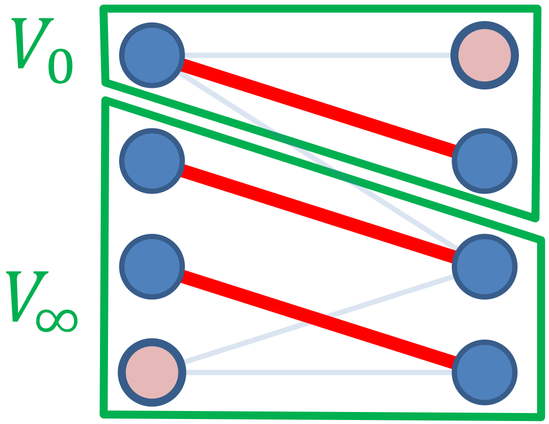
Adding Edges to Reduce to Case 1 (∃**Perfect Matching**)
between Exposed Vertices
in a **Max. Matching** M in G



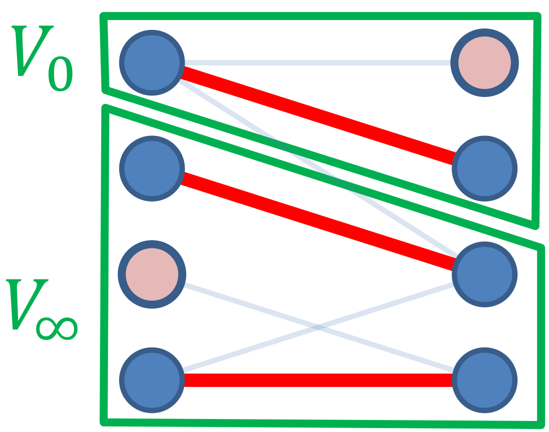
$|V^+| - |M|$ max{# of **Sources**, # of **Sinks**}

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

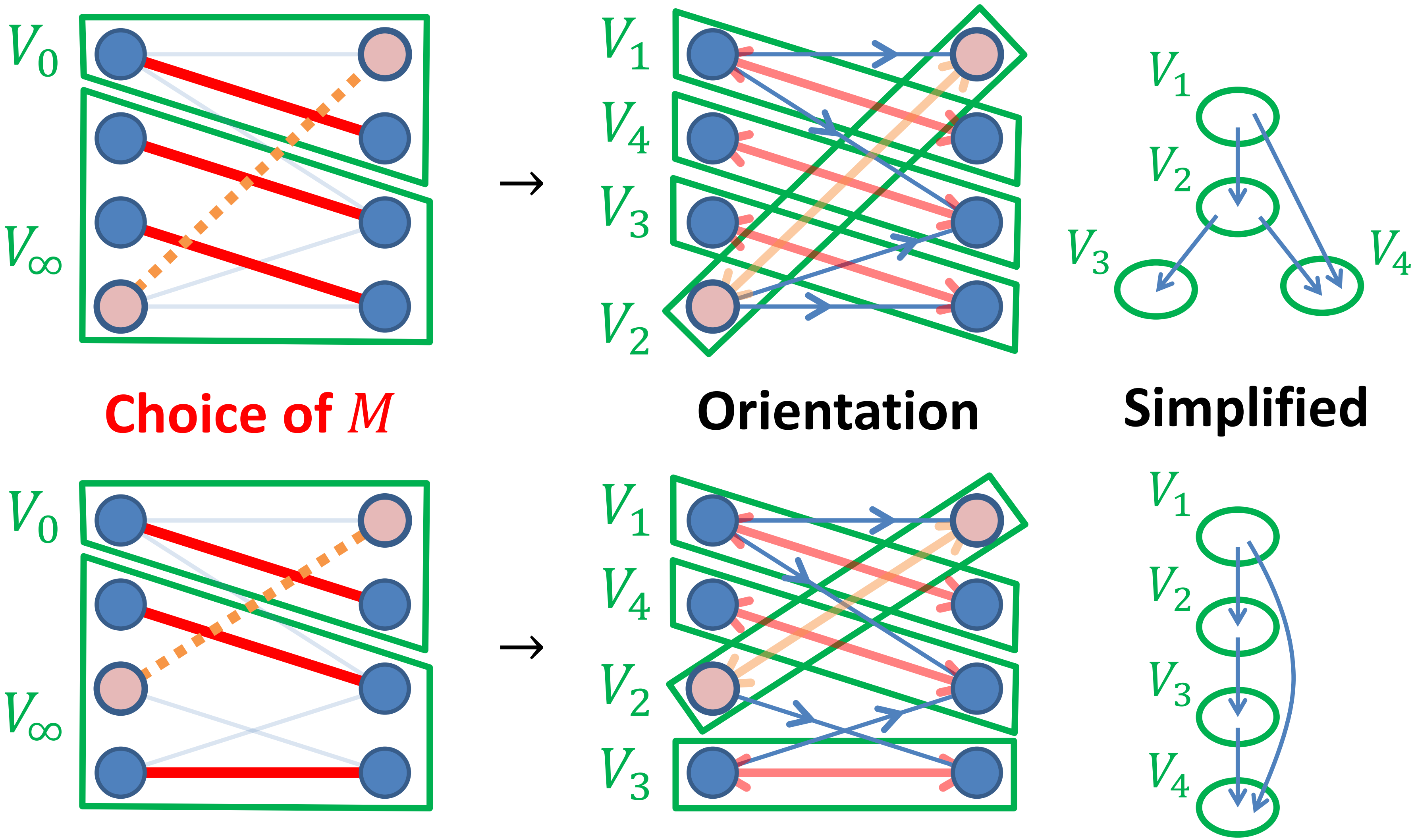
Sources and Sinks in Resulting Digraph



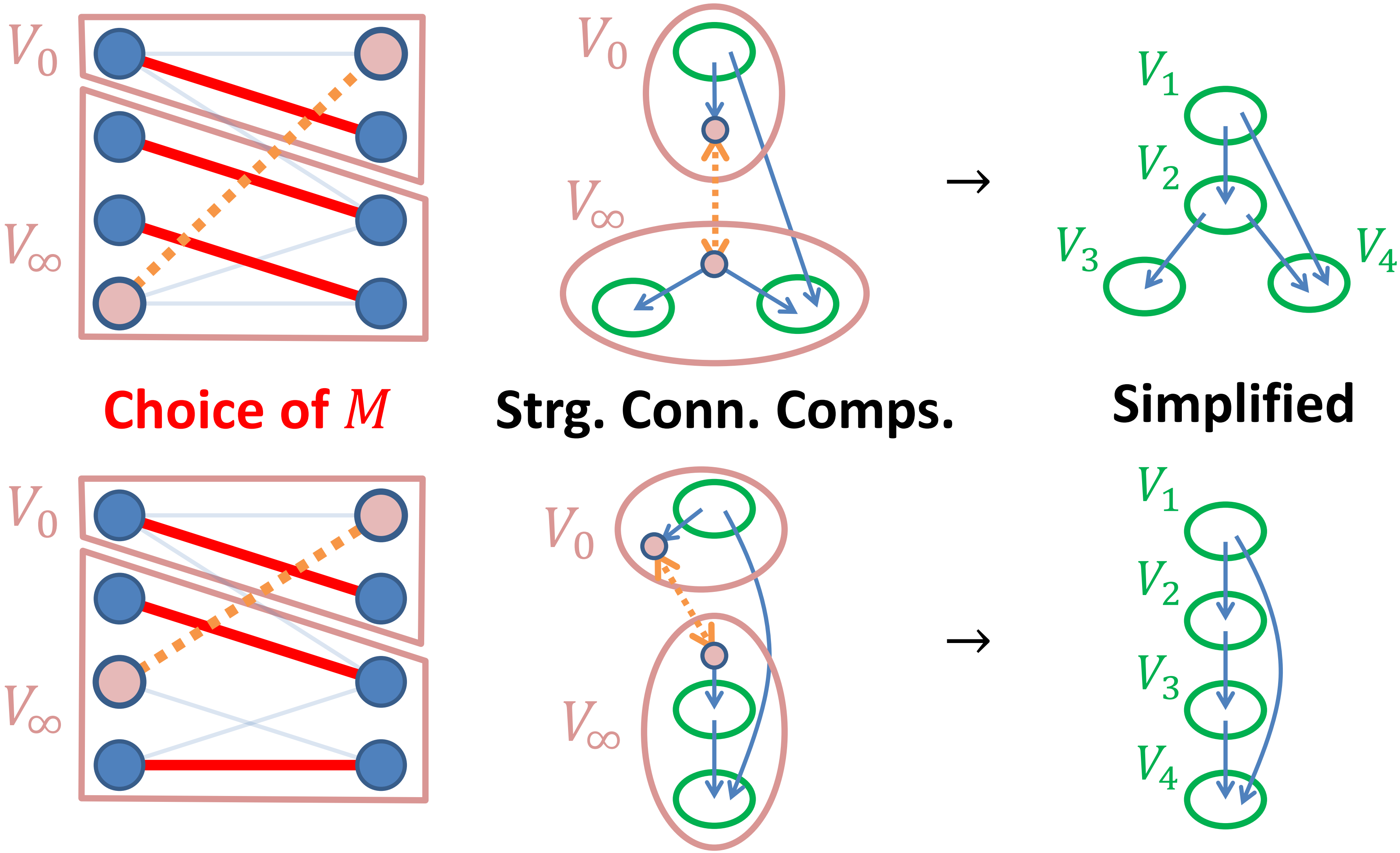
Choice of M



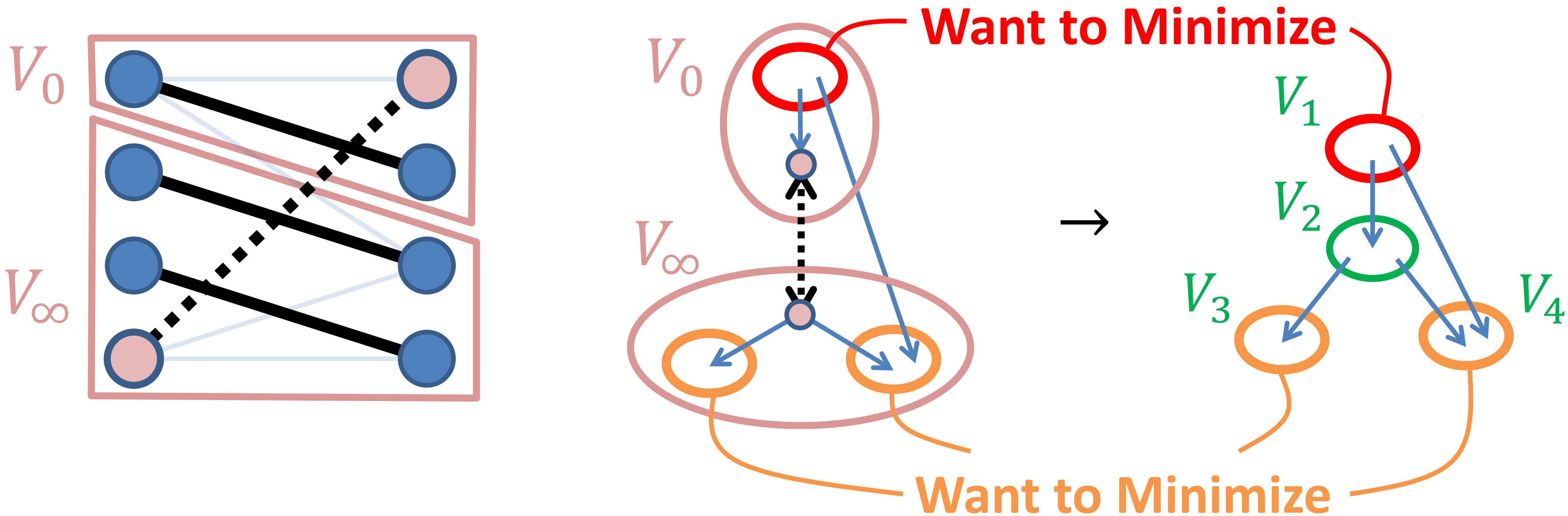
Sources and Sinks in Resulting Digraph



Sources and Sinks in Resulting Digraph



Sources and Sinks in Resulting Digraph

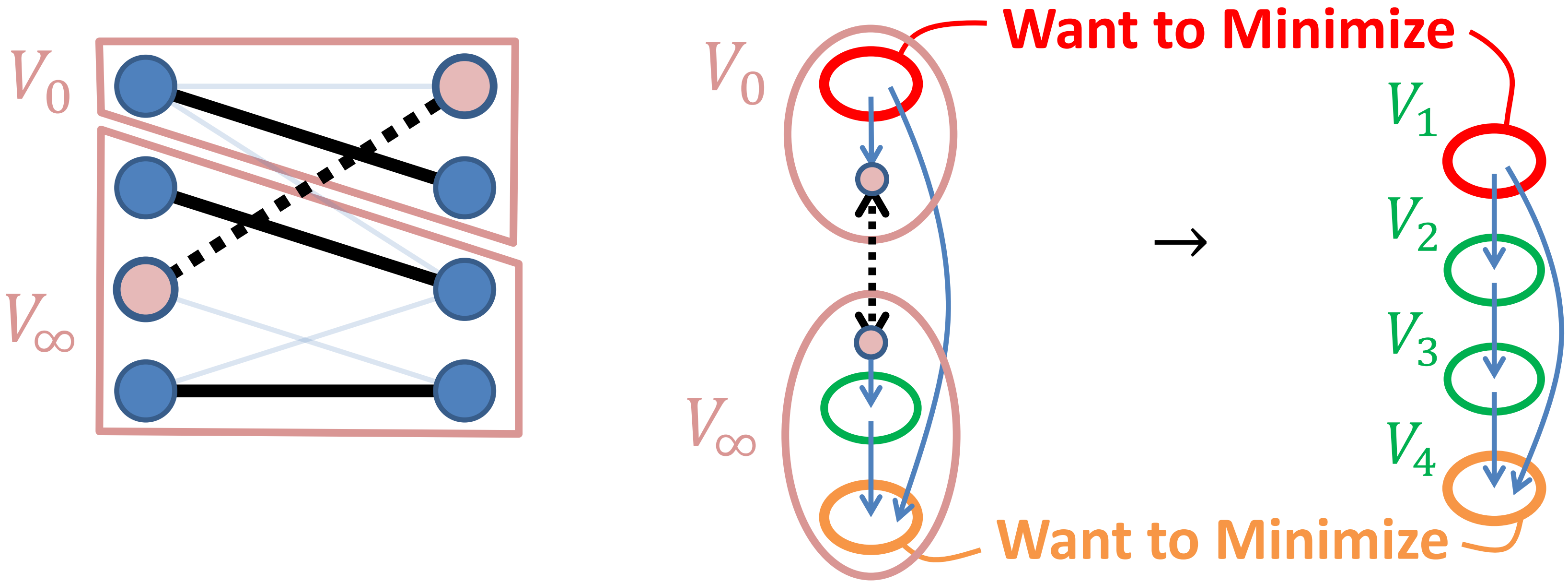


Obs.

(# 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 $G[V_\infty]$.

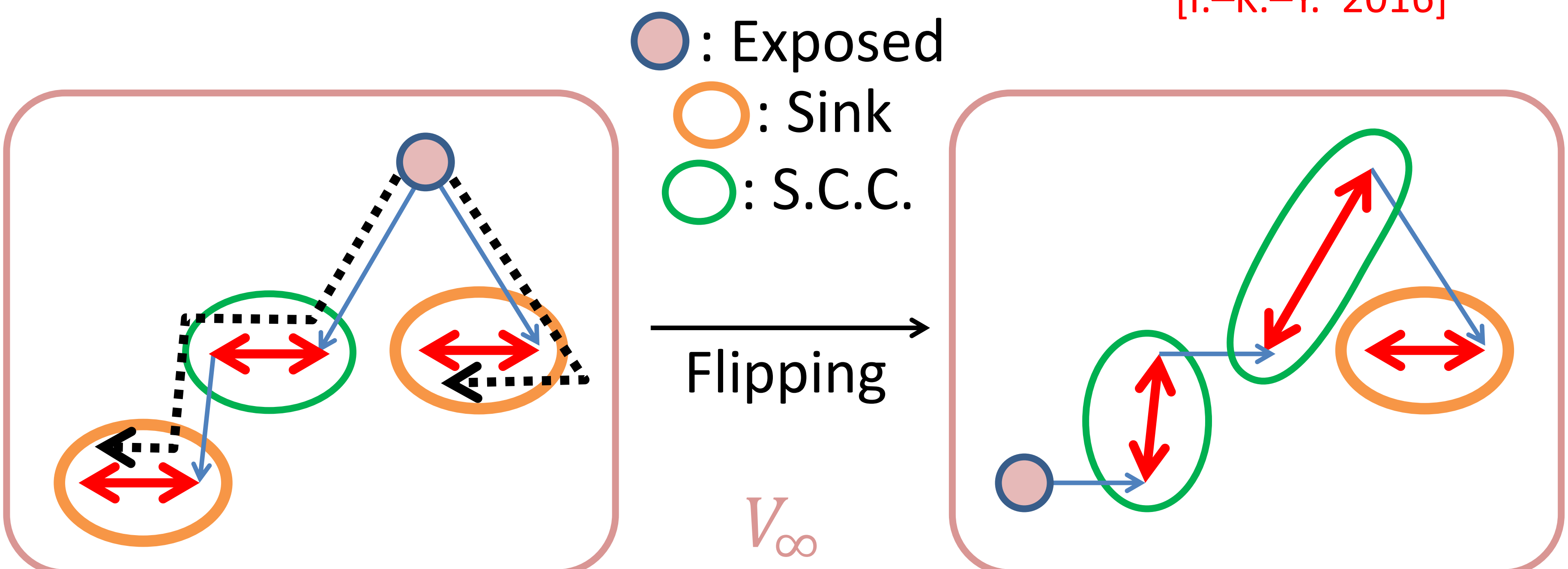
How to Minimize (# of **Sinks in V_∞**)

<u>Lem.</u> (# of **Sinks in V_∞**) is **NOT Minimized**

$\Updownarrow$

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

[I.–K.–Y. 2016]



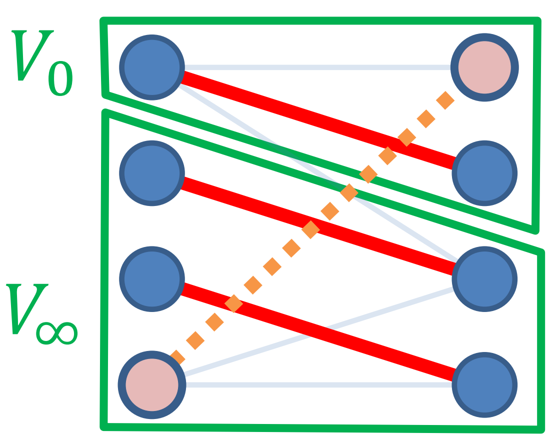
Summary of Cases 3.1

<u>Case 3.1.</u> $|V^+| = |V^-|$ and G has NO Perfect Matching

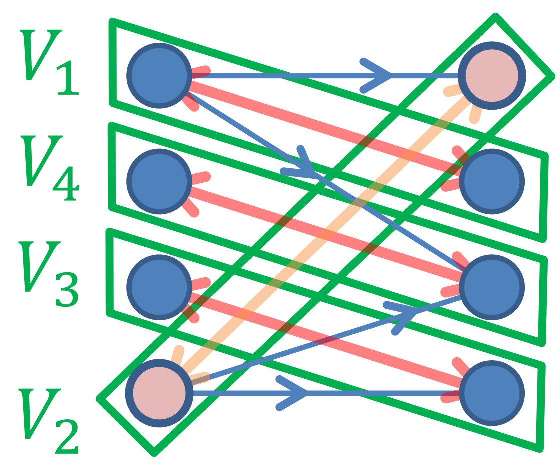
- Connect <u>Exposed Vertices</u> to Make **Perfect Matching**

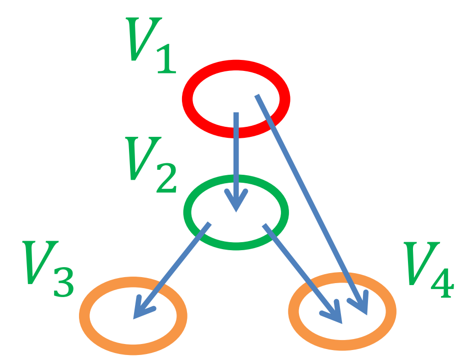
  $\rightarrow$ Reduce to Case 1

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



$\rightarrow$





Summary of Cases 3.1

Case 3.1. $|V^+| = |V^-|$ and G has NO Perfect Matching

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

  $\rightarrow$ 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.

Summary of Cases 3.1

Case 3.1. $|V^+| = |V^-|$ and G has NO Perfect Matching

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

  $\rightarrow$ 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]

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

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

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

Thm. This problem can be solved in polynomial time.

[I.–K.–Y. 2016]

Tools

- Finding a **Maximum Matching** in a **Bipartite Graph**
- Decomposition into **Strongly Connected Components**
- Making a Digraph **Strongly Connected** by Adding Edges
- Finding **Edge-Disjoint** s–t **Paths** in a Digraph