Deterministically Isolating a Perfect Matching in Bipartite Planar Graphs

Joint work with

Raghav Kulkarni (U. Chicago)

Sambuddha Roy (IBM Research, Delhi)

Samir Datta

`sdatta@cmi.ac.in`

Chennai Mathematical Institute

Outline

- Introduction

Outline

- Introduction
- Historical Perspective

Outline

- Introduction
- Historical Perspective
- New Results

Outline

- Introduction
- Historical Perspective
- New Results
- Open Questions

Perfect Matching

Given an undirected graph $G = (V, E)$ and a weight function $w : E \to N$.

- Matching: A set of vertex disjoint edges

Perfect Matching

Given an undirected graph $G = (V, E)$ and a weight function $w : E \rightarrow N$.

- Matching: A set of vertex disjoint edges
  - Matched Vertex: vertex incident on matched edge

Perfect Matching

Given an undirected graph $G = (V, E)$ and a weight function $w : E \rightarrow N$.

- Matching: A set of vertex disjoint edges
  - Matched Vertex: vertex incident on matched edge
- Perfect Matching (pm): all vertices matched

Perfect Matching

Given an undirected graph $G = (V, E)$ and a weight function $w : E \to N$.

- Matching: A set of vertex disjoint edges
  - Matched Vertex: vertex incident on matched edge
- Perfect Matching (pm): all vertices matched
- Min-Weight Perfect Matching: pm with min weight

Computational Problems

Given an undirected graph G:

- Decision: Does $\exists$ pm?

Computational Problems

Given an undirected graph G:

- Decision: Does $\exists$ pm?
- Search: Find (the edges in) any one pm.

Computational Problems

Given an undirected graph G:

- Decision: Does $\exists$ pm?
- Search: Find (the edges in) any one pm.
- Counting: What is #pm's?

Computational Problems

Given an undirected graph G:

- Decision: Does $\exists$ pm?
- Search: Find (the edges in) any one pm.
- Counting: What is #pm's?
- Uniqueness: Is #pm's = 1?

Classes of Graphs

- General

Classes of Graphs

- General
- Bipartite: Two-colourable $\equiv$ no odd cycles

Classes of Graphs

- General
- Bipartite: Two-colourable $\equiv$ no odd cycles
- Planar: Embeddable on plane without edge intersection except at endpoints

Classes of Graphs

- General
- Bipartite: Two-colourable $\equiv$ no odd cycles
- Planar: Embeddable on plane without edge intersection except at endpoints
- Bipartite Planar

Classes of Graphs

- General
- Bipartite: Two-colourable $\equiv$ no odd cycles
- Planar: Embeddable on plane without edge intersection except at endpoints
- Bipartite Planar
- Small Genus: Embeddable on surface of small genus

Known Results

	General	Bipartite	Planar	Bip Planar
Decision	$P \cap RNC$	$P \cap RNC$	NC	NC (SPL)
Search	$P \cap RNC$	$P \cap RNC$	$P \cap RNC$	NC (SPL)
Counting	#P	#P	NC	NC
Uniqueness	P	NC	NC	NC (SPL)

(Some) Approaches

- Classical (and Sequential): e.g.

(Some) Approaches

- Classical (and Sequential): e.g.
  - Augmenting Paths

(Some) Approaches

- Classical (and Sequential): e.g.
  - Augmenting Paths
  - Contracting Blossoms

(Some) Approaches

- Classical (and Sequential): e.g.
  - Augmenting Paths
  - Contracting Blossoms
- Randomized Parallel:

(Some) Approaches

- Classical (and Sequential): e.g.
  - Augmenting Paths
  - Contracting Blossoms
- Randomized Parallel:
  - Karp-Upfal-Wigderson

(Some) Approaches

- Classical (and Sequential): e.g.
  - Augmenting Paths
  - Contracting Blossoms
- Randomized Parallel:
  - Karp-Upfal-Wigderson
  - Mulmuley-Vazirani-Vazirani

(Some) Approaches

- Classical (and Sequential): e.g.
  - Augmenting Paths
  - Contracting Blossoms
- Randomized Parallel:
  - Karp-Upfal-Wigderson
  - Mulmuley-Vazirani-Vazirani
  - Allender-Reinhardt-Zhou

(Some) Approaches

- Classical (and Sequential): e.g.
  - Augmenting Paths
  - Contracting Blossoms
- Randomized Parallel:
  - Karp-Upfal-Wigderson
  - Mulmuley-Vazirani-Vazirani
  - Allender-Reinhardt-Zhou
- Parallel (Topologically restricted graphs)

(Some) Approaches

- Classical (and Sequential): e.g.
  - Augmenting Paths
  - Contracting Blossoms
- Randomized Parallel:
  - Karp-Upfal-Wigderson
  - Mulmuley-Vazirani-Vazirani
  - Allender-Reinhardt-Zhou
- Parallel (Topologically restricted graphs)
  - Miller-Naor : Planar Bipartite

(Some) Approaches

- Classical (and Sequential): e.g.
  - Augmenting Paths
  - Contracting Blossoms
- Randomized Parallel:
  - Karp-Upfal-Wigderson
  - Mulmuley-Vazirani-Vazirani
  - Allender-Reinhardt-Zhou
- Parallel (Topologically restricted graphs)
  - Miller-Naor : Planar Bipartite
  - Mahajan-Vardarajan: Planar/Low Genus Bipartite

(Some) Approaches

- Classical (and Sequential): e.g.
  - Augmenting Paths
  - Contracting Blossoms
- Randomized Parallel:
  - Karp-Upfal-Wigderson
  - Mulmuley-Vazirani-Vazirani
  - Allender-Reinhardt-Zhou
- Parallel (Topologically restricted graphs)
  - Miller-Naor : Planar Bipartite
  - Mahajan-Vardarajan: Planar/Low Genus Bipartite
  - D-Kulkarni-Roy: Planar Bipartite

(Some) Approaches

- Classical (and Sequential): e.g.
  - Augmenting Paths
  - Contracting Blossoms
- Randomized Parallel:
  - Karp-Upfal-Wigderson
  - Mulmuley-Vazirani-Vazirani
  - Allender-Reinhardt-Zhou
- Parallel (Topologically restricted graphs)
  - Miller-Naor : Planar Bipartite
  - Mahajan-Vardarajan: Planar/Low Genus Bipartite
  - D-Kulkarni-Roy: Planar Bipartite
- Parallel (Uniqueness): Hoang-Mahajan-Thierauf

NC **Algo for Bipartite Planar Matching**

[Mahajan-Varadarajan]

- Fractional Solution via counting

NC **Algo for Bipartite Planar Matching**

[Mahajan-Varadarajan]

- Fractional Solution via counting
  - $x : e \mapsto x_e$ s.t.

NC **Algo for Bipartite Planar Matching**

[Mahajan-Varadarajan]

- Fractional Solution via counting
  - $x : e \mapsto x_e$ s.t.
  - $x_e = \frac{\#pm\ containing\ e}{\#pm}$

NC **Algo for Bipartite Planar Matching**

[Mahajan-Varadarajan]

- Fractional Solution via counting
  - $x : e \mapsto x_e$ s.t.
  - $x_e = \frac{\#pm\ containing\ e}{\#pm}$
  - For any vertex u : $\sum_{e=(u,v)\in E} x_e = 1$

NC **Algo for Bipartite Planar Matching**

[Mahajan-Varadarajan]

- Fractional Solution via counting
  - $x : e \mapsto x_e$ s.t.
  - $x_e = \frac{\#pm\ containing\ e}{\#pm}$
  - For any vertex u : $\sum_{e=(u,v)\in E} x_e = 1$
- Round the solution face-wise

NC Algo for Bipartite Planar Matching

[Mahajan-Varadarajan]

- Fractional Solution via counting
  - $x : e \mapsto x_e$ s.t.
  - $x_e = \frac{\#pm\ containing\ e}{\#pm}$
  - For any vertex u : $\sum_{e=(u,v)\in E} x_e = 1$
- Round the solution face-wise
  - Find the smallest $x = x_{e_0}$ for $e_0 \in F$

NC **Algo for Bipartite Planar Matching**

[Mahajan-Varadarajan]

- Fractional Solution via counting
  - $x : e \mapsto x_e$ s.t.
  - $x_e = \frac{\#pm\ containing\ e}{\#pm}$
  - For any vertex u : $\sum_{e=(u,v)\in E} x_e = 1$
- Round the solution face-wise
  - Find the smallest $x = x_{e_0}$ for $e_0 \in F$
  - $x_e \leftarrow x_e - x$ if even # of vertices between e and e_0.

NC Algo for Bipartite Planar Matching

[Mahajan-Varadarajan]

- Fractional Solution via counting
  - $x : e \mapsto x_e$ s.t.
  - $x_e = \frac{\#pm\ containing\ e}{\#pm}$
  - For any vertex u : $\sum_{e=(u,v)\in E} x_e = 1$
- Round the solution face-wise
  - Find the smallest $x = x_{e_0}$ for $e_0 \in F$
  - $x_e \leftarrow x_e - x$ if even # of vertices between e and e_0.
  - $x_e \leftarrow x_e + x$ otherwise.

NC Algo for Bipartite Planar Matching

[Mahajan-Varadarajan]

- Fractional Solution via counting
  - $x : e \mapsto x_e$ s.t.
  - $x_e = \frac{\#pm\ containing\ e}{\#pm}$
  - For any vertex u : $\sum_{e=(u,v)\in E} x_e = 1$
- Round the solution face-wise
  - Find the smallest $x = x_{e_0}$ for $e_0 \in F$
  - $x_e \leftarrow x_e - x$ if even # of vertices between e and e_0.
  - $x_e \leftarrow x_e + x$ otherwise.
- Coalesce faces across edges with $x_e = 0$.

NC **Algo for Bipartite Planar Matching**

[Mahajan-Varadarajan]

- Fractional Solution via counting
  - $x : e \mapsto x_e$ s.t.
  - $x_e = \frac{\#pm\ containing\ e}{\#pm}$
  - For any vertex u : $\sum_{e=(u,v)\in E} x_e = 1$
- Round the solution face-wise
  - Find the smallest $x = x_{e_0}$ for $e_0 \in F$
  - $x_e \leftarrow x_e - x$ if even # of vertices between e and e_0.
  - $x_e \leftarrow x_e + x$ otherwise.
- Coalesce faces across edges with $x_e = 0$.
- Constant fraction of faces processed in parallel

NC **Algo for Bipartite Planar Graphs : II**

Needed Tools:

- #pm in Planar Graphs (Kastelyn/Little/Vazirani)

NC **Algo for Bipartite Planar Graphs : II**

Needed Tools:

- #pm in Planar Graphs (Kastelyn/Little/Vazirani)
- Maximal Independence Set in Planar Graphs (He/Luby)

NC **Algo for Bipartite Planar Graphs : II**

Needed Tools:

- #pm in Planar Graphs (Kastelyn/Little/Vazirani)
- Maximal Independence Set in Planar Graphs (He/Luby)

Extension to Small Genus Bipartite Graphs

NC Algo for Bipartite Planar Graphs : II

Needed Tools:

- #pm in Planar Graphs (Kastelyn/Little/Vazirani)
- Maximal Independence Set in Planar Graphs (He/Luby)

Extension to Small Genus Bipartite Graphs

- #pm in Small Genus Graphs (Galluccio-Loebl)

NC Algo for Bipartite Planar Graphs : II

Needed Tools:

- #pm in Planar Graphs (Kastelyn/Little/Vazirani)
- Maximal Independence Set in Planar Graphs (He/Luby)

Extension to Small Genus Bipartite Graphs

- #pm in Small Genus Graphs (Galluccio-Loebl)
- Maximal Independence Set in graphs (Luby)

RNC **Algo for Bipartite Matching**

[Mulmuley-Vazirani-Vazirani]

- Isolate:

RNC **Algo for Bipartite Matching**

[Mulmuley-Vazirani-Vazirani]

- Isolate:
  - Assign small random weights to edges

RNC **Algo for Bipartite Matching**

[Mulmuley-Vazirani-Vazirani]

- Isolate:
  - Assign small random weights to edges
  - min-weight pm unique (with high probability)

RNC Algo for Bipartite Matching

[Mulmuley-Vazirani-Vazirani]

- Isolate:
  - Assign small random weights to edges
  - min-weight pm unique (with high probability)
- Extract:

RNC **Algo for Bipartite Matching**

[Mulmuley-Vazirani-Vazirani]

- Isolate:
  - Assign small random weights to edges
  - min-weight pm unique (with high probability)
- Extract:
  - Determinant = Signed sum of matchings

RNC **Algo for Bipartite Matching**

[Mulmuley-Vazirani-Vazirani]

- Isolate:
  - Assign small random weights to edges
  - min-weight pm unique (with high probability)
- Extract:
  - Determinant = Signed sum of matchings
  - if $M_{ij} = 2^{w_{ij}}$ then

RNC **Algo for Bipartite Matching**

[Mulmuley-Vazirani-Vazirani]

- Isolate:
  - Assign small random weights to edges
  - min-weight pm unique (with high probability)
- Extract:
  - Determinant = Signed sum of matchings
  - if $M_{ij} = 2^{w_{ij}}$ then
  - $det(M)$ contains $\pm 2^w$ for each matching of weight w

RNC **Algo for Bipartite Matching**

[Mulmuley-Vazirani-Vazirani]

- Isolate:
  - Assign small random weights to edges
  - min-weight pm unique (with high probability)
- Extract:
  - Determinant = Signed sum of matchings
  - if $M_{ij} = 2^{w_{ij}}$ then
  - $det(M)$ contains $\pm 2^w$ for each matching of weight w
  - If $\exists$ unique min-weight matching of weight W, then,

RNC **Algo for Bipartite Matching**

[Mulmuley-Vazirani-Vazirani]

- Isolate:
  - Assign small random weights to edges
  - min-weight pm unique (with high probability)
- Extract:
  - Determinant = Signed sum of matchings
  - if $M_{ij} = 2^{w_{ij}}$ then
  - $det(M)$ contains $\pm 2^w$ for each matching of weight w
  - If $\exists$ unique min-weight matching of weight W, then,
  - $2^W | det(M)$ but not $2^{W+1} | det(M)$

RNC **Algo for Bipartite Matching**

[Mulmuley-Vazirani-Vazirani]

- Isolate:
  - Assign small random weights to edges
  - min-weight pm unique (with high probability)
- Extract:
  - Determinant = Signed sum of matchings
  - if $M_{ij} = 2^{w_{ij}}$ then
  - $det(M)$ contains $\pm 2^w$ for each matching of weight w
  - If $\exists$ unique min-weight matching of weight W, then,
  - $2^W | det(M)$ but not $2^{W+1} | det(M)$
  - $e \in$ unique min-wt matching iff W increases in $G - e$

UL/$poly$ Algo for Reachability

[Reinhardt-Allender]

- Universal Isolation:

UL/$poly$ Algo for Reachability

[Reinhardt-Allender]

- Universal Isolation:
  - Isolate min-weight $s-t$ paths instead of pm

UL$/poly$ **Algo for Reachability**

[Reinhardt-Allender]

- Universal Isolation:
  - Isolate min-weight $s - t$ paths instead of pm
  - Most wt fns isolate min-wt path for given graph

UL$/poly$ **Algo for Reachability**

[Reinhardt-Allender]

- Universal Isolation:
  - Isolate min-weight $s - t$ paths instead of pm
  - Most wt fns isolate min-wt path for given graph
  - Pick a poly sized set $\mathcal{W}$ of random weight functions

UL$/poly$ **Algo for Reachability**

[Reinhardt-Allender]

- Universal Isolation:
  - Isolate min-weight $s - t$ paths instead of pm
  - Most wt fns isolate min-wt path for given graph
  - Pick a poly sized set $\mathcal{W}$ of random weight functions
  - Whp $\forall G.|V(G)| = n.\exists w \in \mathcal{W}$ s.t. w isolates in G.

UL$/poly$ **Algo for Reachability**

[Reinhardt-Allender]

- Universal Isolation:
  - Isolate min-weight $s - t$ paths instead of pm
  - Most wt fns isolate min-wt path for given graph
  - Pick a poly sized set $\mathcal{W}$ of random weight functions
  - Whp $\forall G.|V(G)| = n.\exists w \in \mathcal{W}$ s.t. w isolates in G.
  - Therefore, $\exists \mathcal{W}_0$, one of which isolates for all graphs

UL$/poly$ Algo for Reachability

[Reinhardt-Allender]

- Universal Isolation:
  - Isolate min-weight $s - t$ paths instead of pm
  - Most wt fns isolate min-wt path for given graph
  - Pick a poly sized set $\mathcal{W}$ of random weight functions
  - Whp $\forall G.|V(G)| = n.\exists w \in \mathcal{W}$ s.t. w isolates in G.
  - Therefore, $\exists \mathcal{W}_0$, one of which isolates for all graphs
- Extraction:

UL/$poly$ **Algo for Reachability**

[Reinhardt-Allender]

- Universal Isolation:
  - Isolate min-weight $s - t$ paths instead of pm
  - Most wt fns isolate min-wt path for given graph
  - Pick a poly sized set $\mathcal{W}$ of random weight functions
  - Whp $\forall G.|V(G)| = n.\exists w \in \mathcal{W}$ s.t. w isolates in G.
  - Therefore, $\exists \mathcal{W}_0$, one of which isolates for all graphs
- Extraction:
  - Double Counting : extension of Inductive Counting.

SPL$/poly$ Algo for Matching

[Allender-Reinhardt-Zhou]

- Universal Isolation: Similar to [Reinhardt-Allender]

SPL$/poly$ Algo for Matching

[Allender-Reinhardt-Zhou]

- Universal Isolation: Similar to [Reinhardt-Allender]
- Extraction:

SPL$/poly$ Algo for Matching

[Allender-Reinhardt-Zhou]

- Universal Isolation: Similar to [Reinhardt-Allender]
- Extraction:
  - Clever use of [Mahajan-Vinay] algo for determinant

Search = Isolation + Extraction

Search = Isolation + Extraction

- Randomized Isolation via Isolation Lemma

Search = Isolation + Extraction

- Randomized Isolation via Isolation Lemma
- Extraction:

Search = Isolation + Extraction

- Randomized Isolation via Isolation Lemma
- Extraction:
  - ([MVV]) Matrix Inversion

Search = Isolation + Extraction

- Randomized Isolation via Isolation Lemma
- Extraction:
  - ([MVV]) Matrix Inversion
  - ([RA]) Double Counting

Search = Isolation + Extraction

- Randomized Isolation via Isolation Lemma
- Extraction:
  - ([MVV]) Matrix Inversion
  - ([RA]) Double Counting
  - ([ARZ]) Using [Mahajan-Vinay]

Deterministic Isolation : I

Deterministic Isolation : I

Lemma ([Allender-D-Roy]) Given, an $n \times n$ complete Layered Grid Graph, one can assign small weights to its edges, s.t. any sub-digraph is min-unique.

Deterministic Isolation : I

Lemma ([Allender-D-Roy]) Given, an $n \times n$ complete Layered Grid Graph, one can assign small weights to its edges, s.t. any sub-digraph is min-unique.

Proof. Pair of paths enclosing simple region differ by area of region. $\square$

Deterministic Isolation : I

Lemma ([Allender-D-Roy]) Given, an $n \times n$ complete Layered Grid Graph, one can assign small weights to its edges, s.t. any sub-digraph is min-unique.

Proof. Pair of paths enclosing simple region differ by area of region. $\square$

Theorem ([ADR]) LGGR $\in$ UL

Deterministic Isolation : I

Lemma ([Allender-D-Roy]) Given, an $n \times n$ complete Layered Grid Graph, one can assign small weights to its edges, s.t. any sub-digraph is min-unique.

Proof. Pair of paths enclosing simple region differ by area of region. □

Theorem ([ADR]) LGGR $\in$ UL

Proof. Extraction as in [Reinhardt-Allender]. □

Deterministic Isolation : II

Lemma ([Bourke-Tewari-Vinodchandran]) Given, an $n \times n$ complete grid graph, one can assign small weights to its edges, s.t. any sub-digraph is min-unique.

Deterministic Isolation : II

Lemma ([Bourke-Tewari-Vinodchandran]) Given, an $n \times n$ complete grid graph, one can assign small weights to its edges, s.t. any sub-digraph is min-unique.

Proof. Every simple cycle has weight equal to area (with a sign). $\square$

Deterministic Isolation : II

Lemma ([Bourke-Tewari-Vinodchandran]) Given, an $n \times n$ complete grid graph, one can assign small weights to its edges, s.t. any sub-digraph is min-unique.

Proof. Every simple cycle has weight equal to area (with a sign). $\square$

Theorem ([BTV]) PlanarReachability $\in$ UL

Proof. Extraction as in [Reinhardt-Allender]. $\square$

Motivating the Problem

- ([Allender-Reinhardt-Zhou]) Extracting an isolated matching is easy.

Motivating the Problem

- ([Allender-Reinhardt-Zhou]) Extracting an isolated matching is easy.
- Deterministic Isolation possible in planar graphs (reachability).

Motivating the Problem

- ([Allender-Reinhardt-Zhou]) Extracting an isolated matching is easy.
- Deterministic Isolation possible in planar graphs (reachability).
- Can one deterministically isolate a perfect matching in planar graphs?

Statement of Result

Problem Given a bipartite planar graph, construct a Perfect Matching.

Statement of Result

Problem Given a bipartite planar graph, construct a Perfect Matching.

Theorem Bipartite Planar Matching is in SPL.

Statement of Result

Problem Given a bipartite planar graph, construct a Perfect Matching.

Theorem Bipartite Planar Matching is in SPL.

Overview of proof:

-

Statement of Result

Problem Given a bipartite planar graph, construct a Perfect Matching.

Theorem Bipartite Planar Matching is in SPL.

Overview of proof:

- ([D-Kulkarni-Limaye-Mahajan]) Embed G in grid, preserving Perfect Matchings (only for bipartite planar graphs)

Statement of Result

Problem Given a bipartite planar graph, construct a Perfect Matching.

Theorem Bipartite Planar Matching is in SPL.

Overview of proof:

- ([D-Kulkarni-Limaye-Mahajan]) Embed G in grid, preserving Perfect Matchings (only for bipartite planar graphs)
- Find weight function: isolates a matching in each grid graph

Statement of Result

Problem Given a bipartite planar graph, construct a Perfect Matching.

Theorem Bipartite Planar Matching is in SPL.

Overview of proof:

- ([D-Kulkarni-Limaye-Mahajan]) Embed G in grid, preserving Perfect Matchings (only for bipartite planar graphs)
- Find weight function: isolates a matching in each grid graph
- ([Allender-Reinhardt-Zhou]) Extract isolated matching

Reformulation of Problem

Problem Given a grid graph $G = (V, E)$, "efficiently" construct a weight function $w : E \to [0, 1, \ldots, |V|^{O(1)}]$ such that the minimum weight perfect matching is unique.

Reformulation of Problem

Problem Given a grid graph $G = (V, E)$, “efficiently” construct a weight function $w : E \rightarrow [0, 1, \ldots, |V|^{O(1)}]$ such that the minimum weight perfect matching is unique.

Observation The symmetric difference of two pm’s is a disjoint union of even cycles.

Reformulation of Problem

Problem Given a grid graph $G = (V, E)$, “efficiently” construct a weight function $w : E \to [0, 1, \ldots, |V|^{O(1)}]$ such that the minimum weight perfect matching is unique.

Observation The symmetric difference of two pm’s is a disjoint union of even cycles.

Definition $circ(C)$ is the difference in weights of alternating edges in cycle C.

Reformulation of Problem

Problem Given a grid graph $G = (V, E)$, "efficiently" construct a weight function $w : E \rightarrow [0, 1, \ldots, |V|^{O(1)}]$ such that the minimum weight perfect matching is unique.

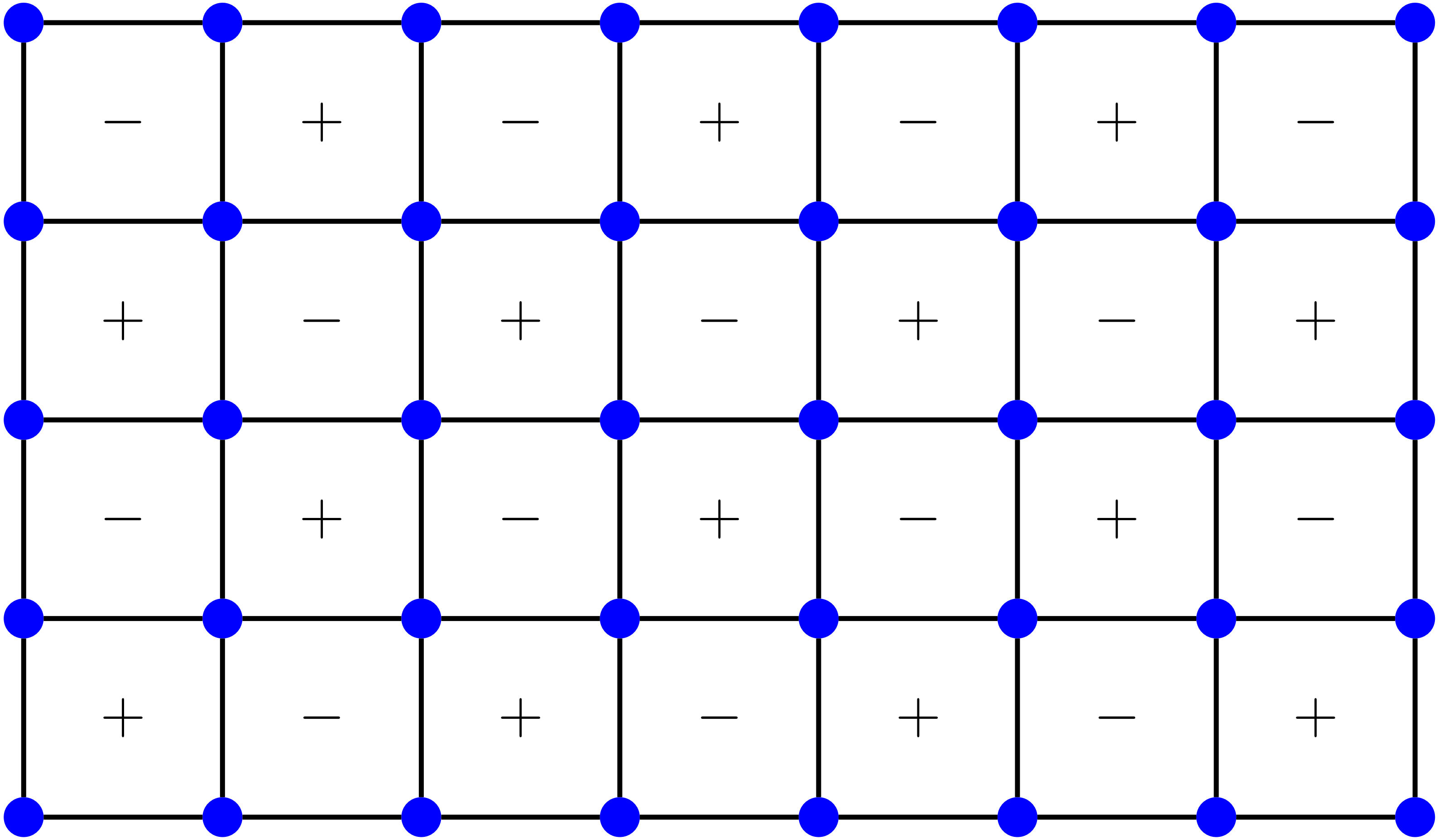
Observation The symmetric difference of two pm's is a disjoint union of even cycles.

Definition $circ(C)$ is the difference in weights of alternating edges in cycle C.

Problem Give a weighing scheme such that $circ(C) \neq 0$ for every ("nice") cycle in grid graphs.

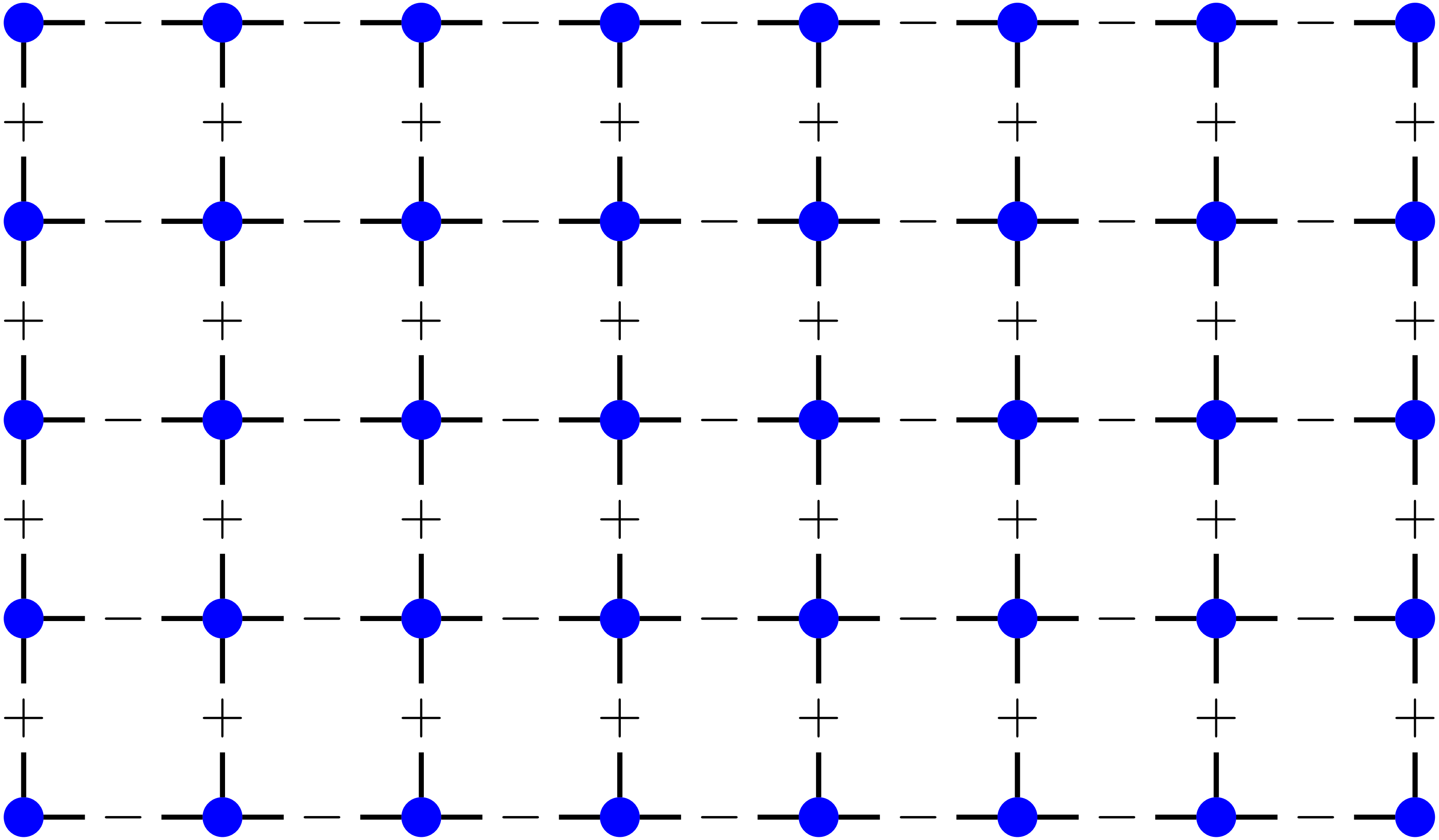
Signs of Blocks

Signs of Blocks



Signs of Edges

Signs of Edges



Block Circulation

- For a block B

$$circ(B) = \sum_{e \in B} sign(e) weight(e)$$

Block Circulation

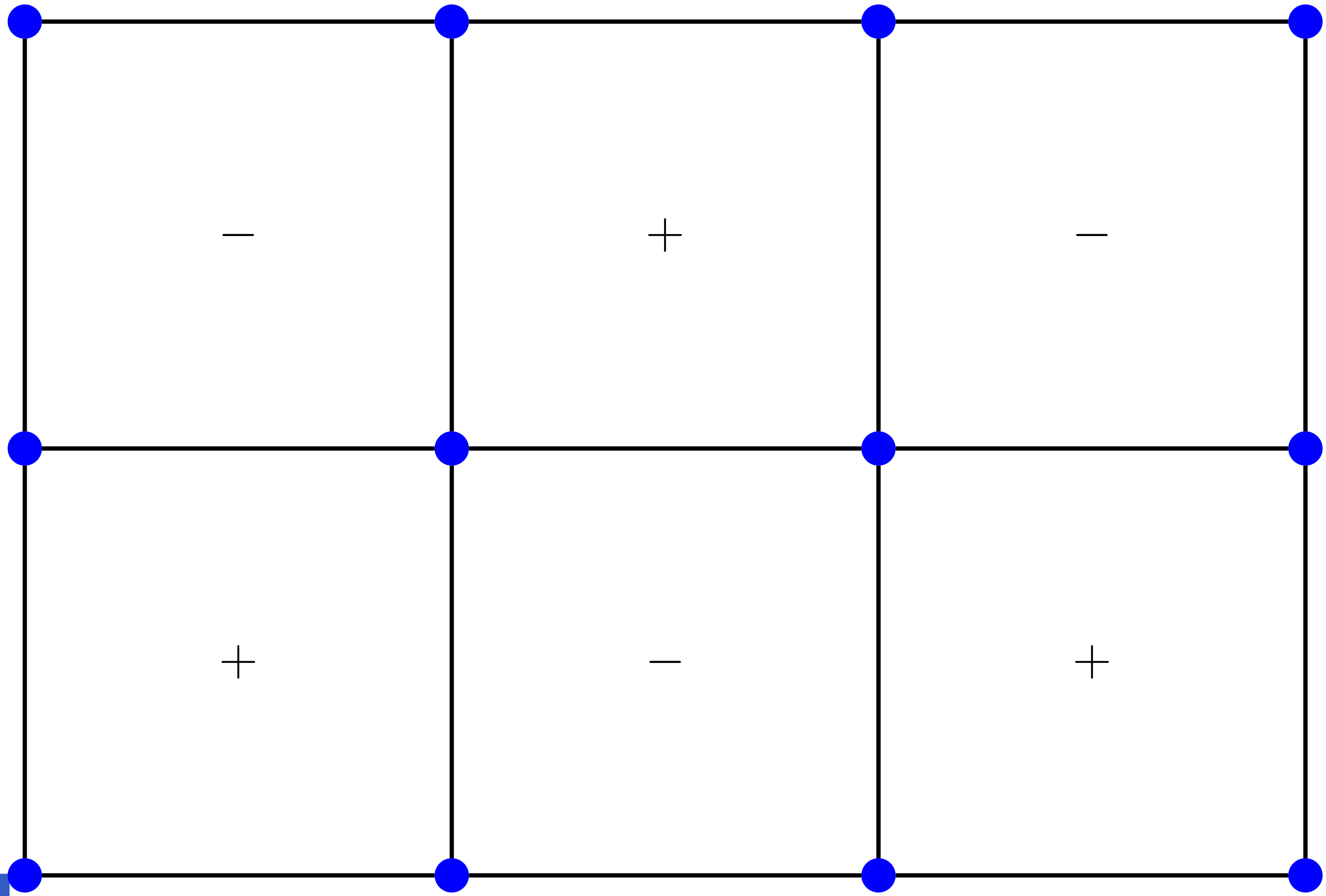
- For a block B

$$circ(B) = \sum_{e \in B} sign(e) weight(e)$$

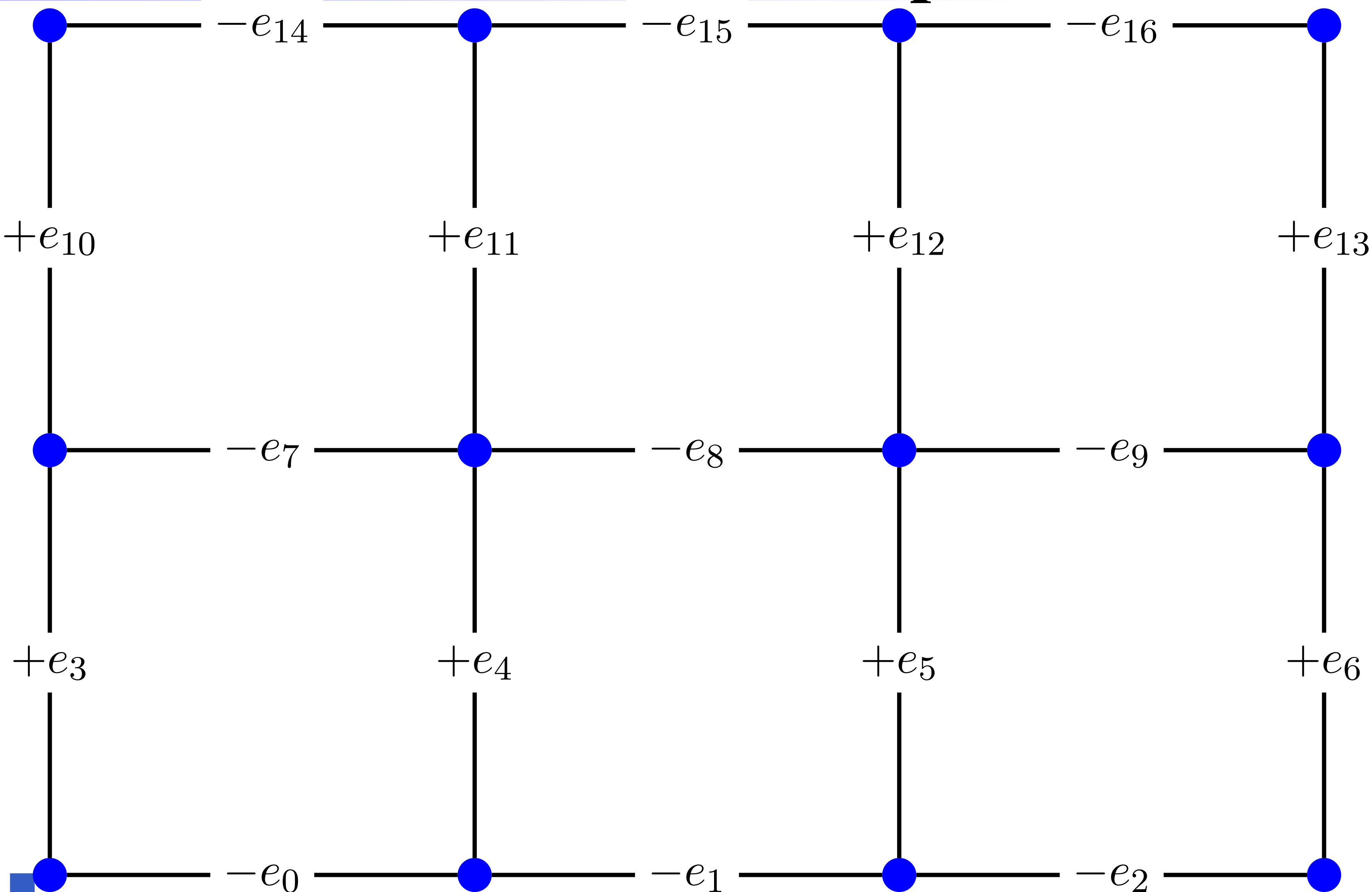
- For a cycle, $C = (e_0, e_1, \ldots, e_l)$ (e_0 leftmost-topmost edge)

$$circ(C) = \sum_{i=0}^{l} (-1)^i weight(e_i)$$

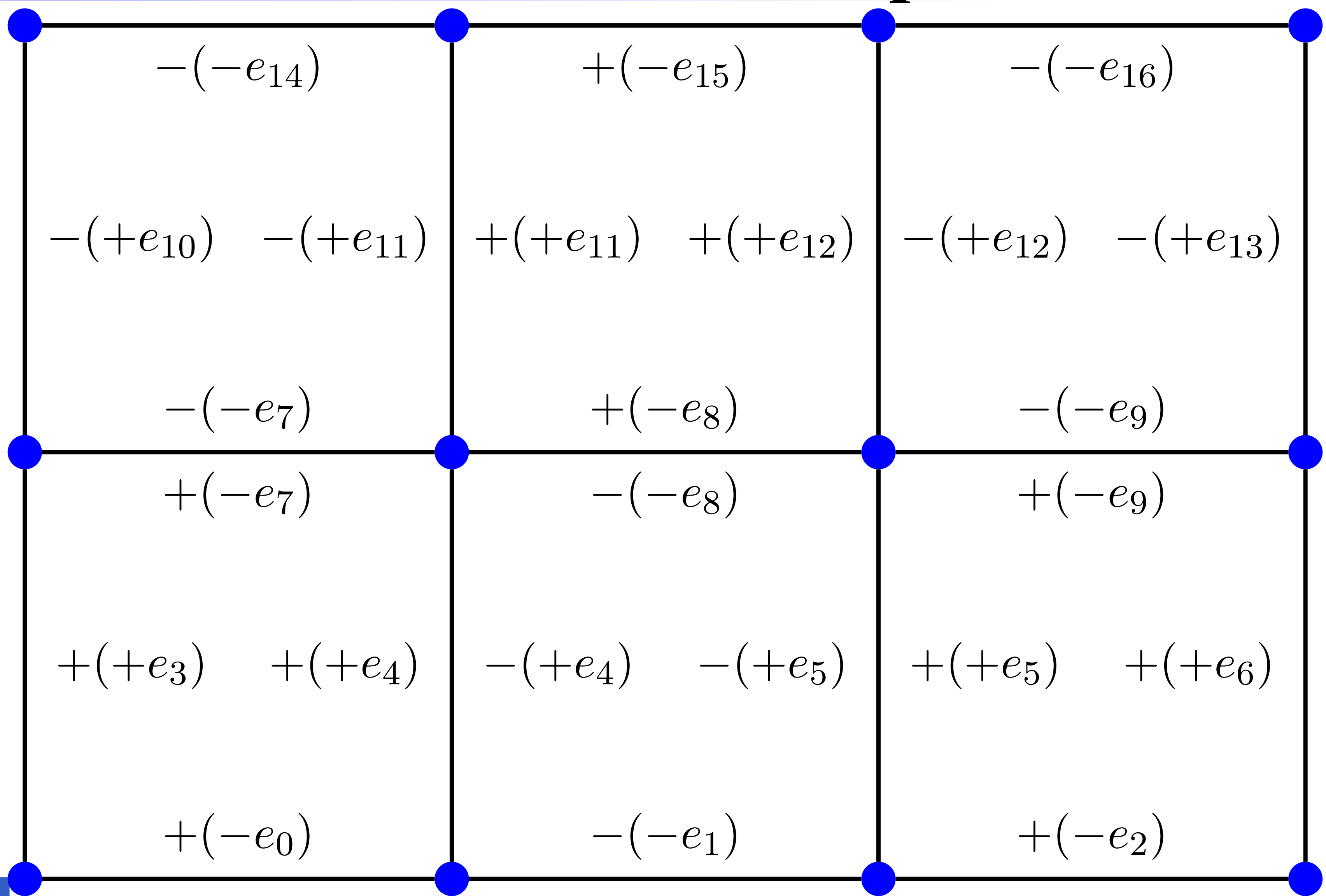
Block Circulations : Example



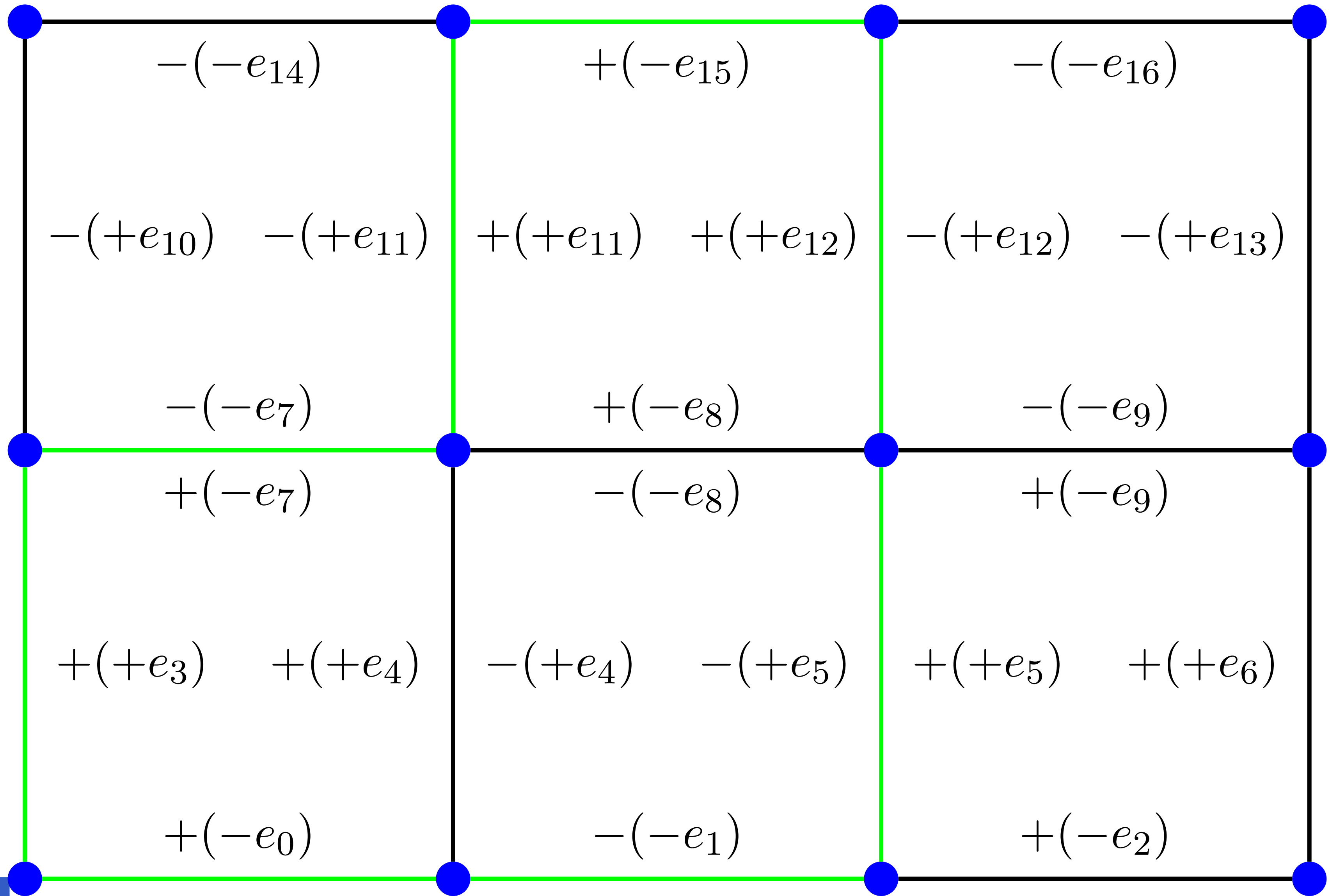
Block Circulations : Example



Block Circulations : Example



Block Circulations : Example



Circulation Lemma

Lemma

$$|circ(C)| = \left| \sum_{B \in interior(C)} sign(B)circ(B) \right|$$

Circulation Lemma

Lemma

$$|circ(C)| = \left| \sum_{B \in interior(C)} sign(B) circ(B) \right|$$

Observation If $circ(B) = sign(B)$ for each B, $circ(C) \neq 0$.

Circulation Lemma

Lemma

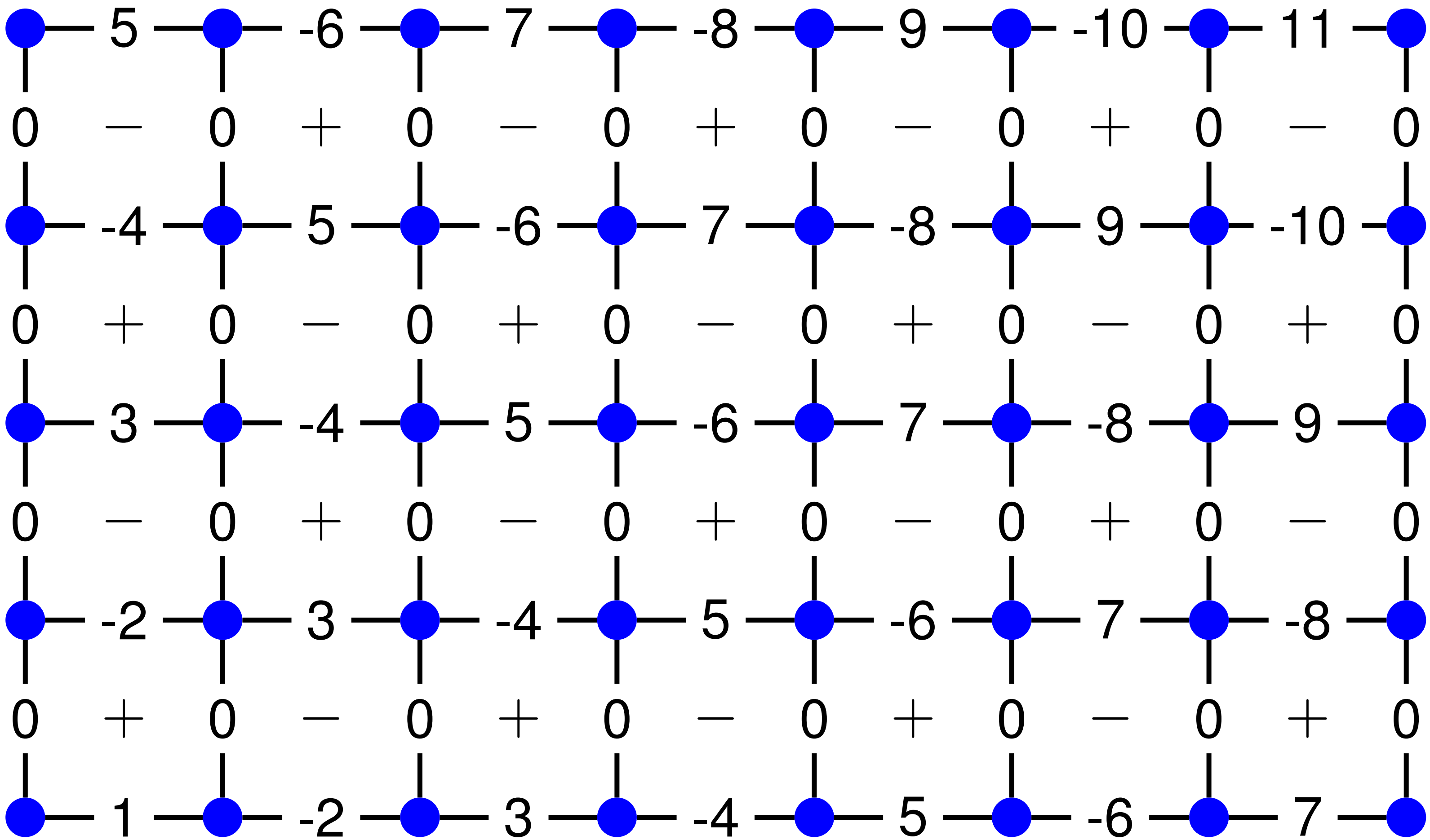
$$|circ(C)| = \left| \sum_{B \in interior(C)} sign(B)circ(B) \right|$$

Observation If $circ(B) = sign(B)$ for each B, $circ(C) \neq 0$.

Hence Done!

Weighing Scheme

Weighing Scheme



UPMs in Bipartite Graphs

- Identify edges in non-zero (or odd) number of pm's

UPMs in Bipartite Graphs

- Identify edges in non-zero (or odd) number of pm's
  - If graph is UPM, reduces to $det(A) \neq 0$ (or $\not\equiv 0 \mod 2$)

UPMs in Bipartite Graphs

- Identify edges in non-zero (or odd) number of pm's
  - If graph is UPM, reduces to $det(A) \neq 0$ (or $\not\equiv 0 \mod 2$)
- Verify that identified edges form upm.

UPMs in Bipartite Graphs

- Identify edges in non-zero (or odd) number of pm's
  - If graph is UPM, reduces to $det(A) \neq 0$ (or $\not\equiv 0 \mod 2$)
- Verify that identified edges form upm.
  - Verify that edges form a matching M.

UPMs in Bipartite Graphs

- Identify edges in non-zero (or odd) number of pm's
  - If graph is UPM, reduces to $det(A) \neq 0$ (or $\not\equiv 0 \mod 2$)
- Verify that identified edges form upm.
  - Verify that edges form a matching M.
  - $\exists$ an alternating cycle wrt M?

UPMs in Bipartite Graphs

- Identify edges in non-zero (or odd) number of pm's
  - If graph is UPM, reduces to $det(A) \neq 0$ (or $\not\equiv 0 \mod 2$)
- Verify that identified edges form upm.
  - Verify that edges form a matching M.
  - $\exists$ an alternating cycle wrt M?
  - $\equiv$ reachability in auxiliary digraph

UPMs in Bipartite Graphs

- Identify edges in non-zero (or odd) number of pm's
  - If graph is UPM, reduces to $det(A) \neq 0$ (or $\not\equiv 0 \mod 2$)
- Verify that identified edges form upm.
  - Verify that edges form a matching M.
  - $\exists$ an alternating cycle wrt M?
  - $\equiv$ reachability in auxiliary digraph
  - Arc (u, w) if $\exists v.(u, v)$ matched $\wedge (v, w)$ unmatched

UPMs in Outerplanar Graphs

- Remove edges enclosing odd number of vertices

UPMs in Outerplanar Graphs

- Remove edges enclosing odd number of vertices
- Reduce $\oplus$pm to $\oplus$SpanningTree in some planar graph

UPMs in Outerplanar Graphs

- Remove edges enclosing odd number of vertices
- Reduce $\oplus$pm to $\oplus$SpanningTree in some planar graph
  - connect new vertex to all odd degree vertices

UPMs in Outerplanar Graphs

- Remove edges enclosing odd number of vertices
- Reduce $\oplus$pm to $\oplus$SpanningTree in some planar graph
  - connect new vertex to all odd degree vertices
  - Laplacian = Adjacency Matrix mod 2

UPMs in Outerplanar Graphs

- Remove edges enclosing odd number of vertices
- Reduce $\oplus$pm to $\oplus$SpanningTree in some planar graph
  - connect new vertex to all odd degree vertices
  - Laplacian = Adjacency Matrix mod 2
  - Apply Matrix Tree Theorem

UPMs in Outerplanar Graphs

- Remove edges enclosing odd number of vertices
- Reduce $\oplus$pm to $\oplus$SpanningTree in some planar graph
  - connect new vertex to all odd degree vertices
  - Laplacian = Adjacency Matrix mod 2
  - Apply Matrix Tree Theorem
- Compute $\oplus$SpanningTree in L ([BravermanKulkarniRoy])

UPMs in Outerplanar Graphs

- Remove edges enclosing odd number of vertices
- Reduce $\oplus$pm to $\oplus$SpanningTree in some planar graph
  - connect new vertex to all odd degree vertices
  - Laplacian = Adjacency Matrix mod 2
  - Apply Matrix Tree Theorem
- Compute $\oplus$SpanningTree in L ([BravermanKulkarniRoy])
- Verify that identified edges form a upm.

UPMs in Outerplanar Graphs

- Remove edges enclosing odd number of vertices
- Reduce $\oplus$pm to $\oplus$SpanningTree in some planar graph
  - connect new vertex to all odd degree vertices
  - Laplacian = Adjacency Matrix mod 2
  - Apply Matrix Tree Theorem
- Compute $\oplus$SpanningTree in L ([BravermanKulkarniRoy])
- Verify that identified edges form a upm.
  - Auxiliary digraph is outerplanar

UPMs in Outerplanar Graphs

- Remove edges enclosing odd number of vertices
- Reduce $\oplus$pm to $\oplus$SpanningTree in some planar graph
  - connect new vertex to all odd degree vertices
  - Laplacian = Adjacency Matrix mod 2
  - Apply Matrix Tree Theorem
- Compute $\oplus$SpanningTree in L ([BravermanKulkarniRoy])
- Verify that identified edges form a upm.
  - Auxiliary digraph is outerplanar
  - Reachability in outerplanar digraphs in L([Allender-Barrington-Chakraborty-D-Roy])

Consequences

- UPMtesting in Bipartite Planar Graphs $\in$ SPL

Consequences

- UPMtesting in Bipartite Planar Graphs $\in$ SPL
- Constructing min-weight perfect matching in Bipartite Planar Graphs (with polynomially bounded weights) in SPL

Consequences

- UPMtesting in Bipartite Planar Graphs $\in$ SPL
- Constructing min-weight perfect matching in Bipartite Planar Graphs (with polynomially bounded weights) in SPL
- Constructing a min-weight spanning tree in planar graphs in SPL(Temperley's bijection)

Consequences

- UPMtesting in Bipartite Planar Graphs $\in$ SPL
- Constructing min-weight perfect matching in Bipartite Planar Graphs (with polynomially bounded weights) in SPL
- Constructing a min-weight spanning tree in planar graphs in SPL(Temperley's bijection)
- Constructing a Perfect Matching in Outerplanar Graphs in L.

Open Questions

- Constructing Planar Matching in NC?

Open Questions

- Constructing Planar Matching in NC?
- Bipartite Matching in NC?

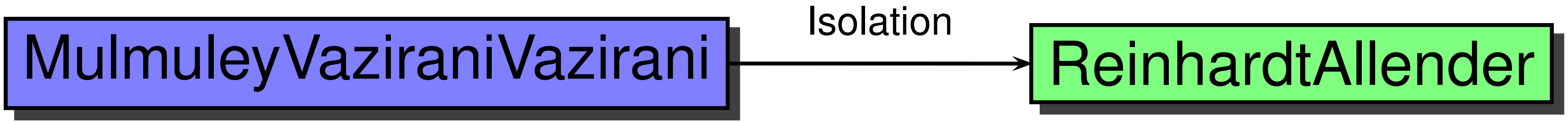
Open Questions

- Constructing Planar Matching in NC?
- Bipartite Matching in NC?
- Extract isolated max-weight perfect matching in NC?

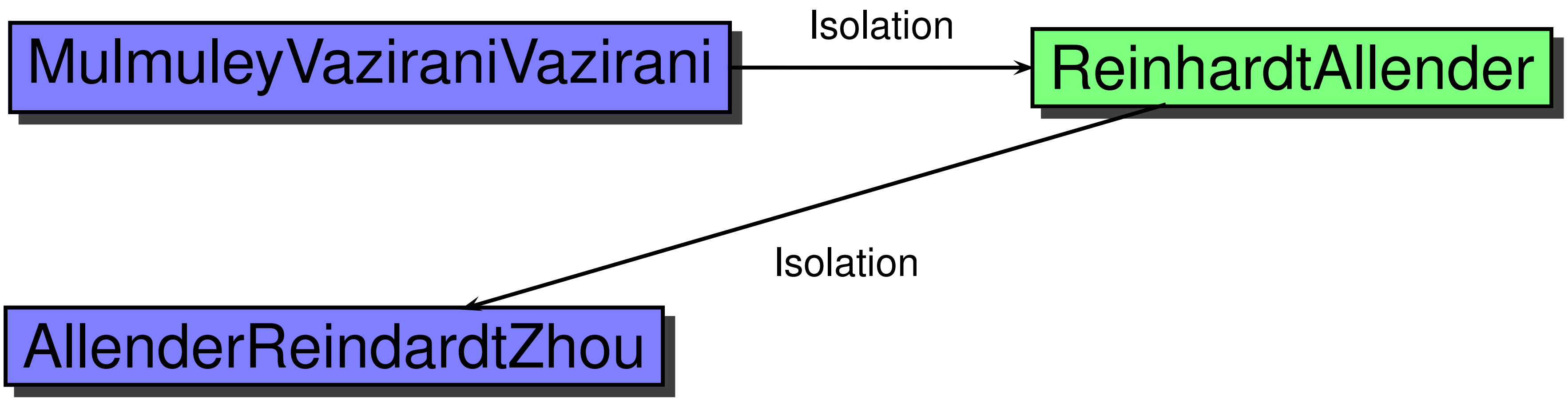
From Matching to Reachability and Back

MulmuleyVaziraniVazirani

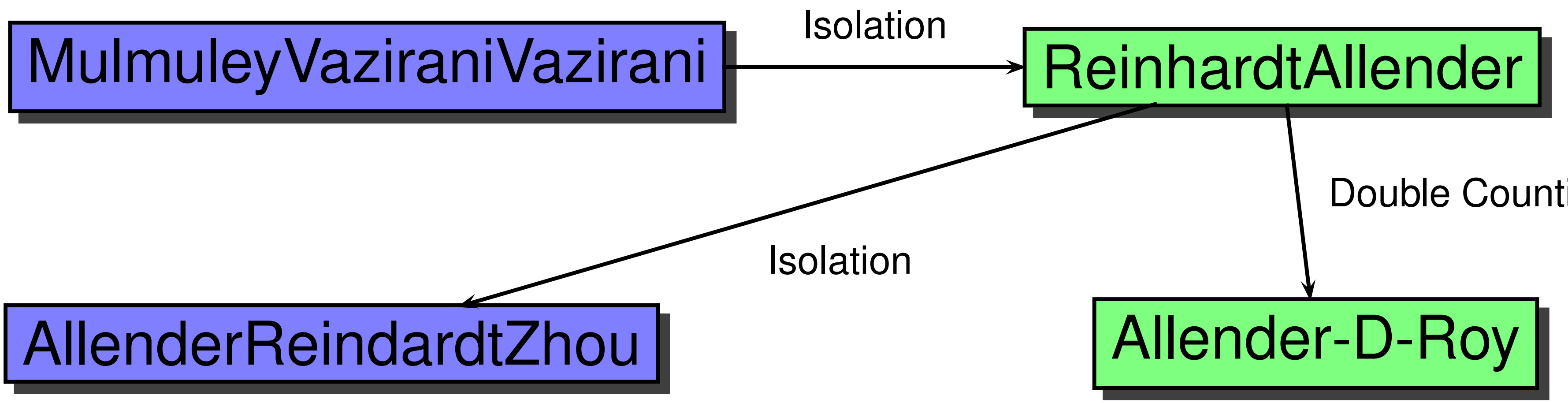
From Matching to Reachability and Back



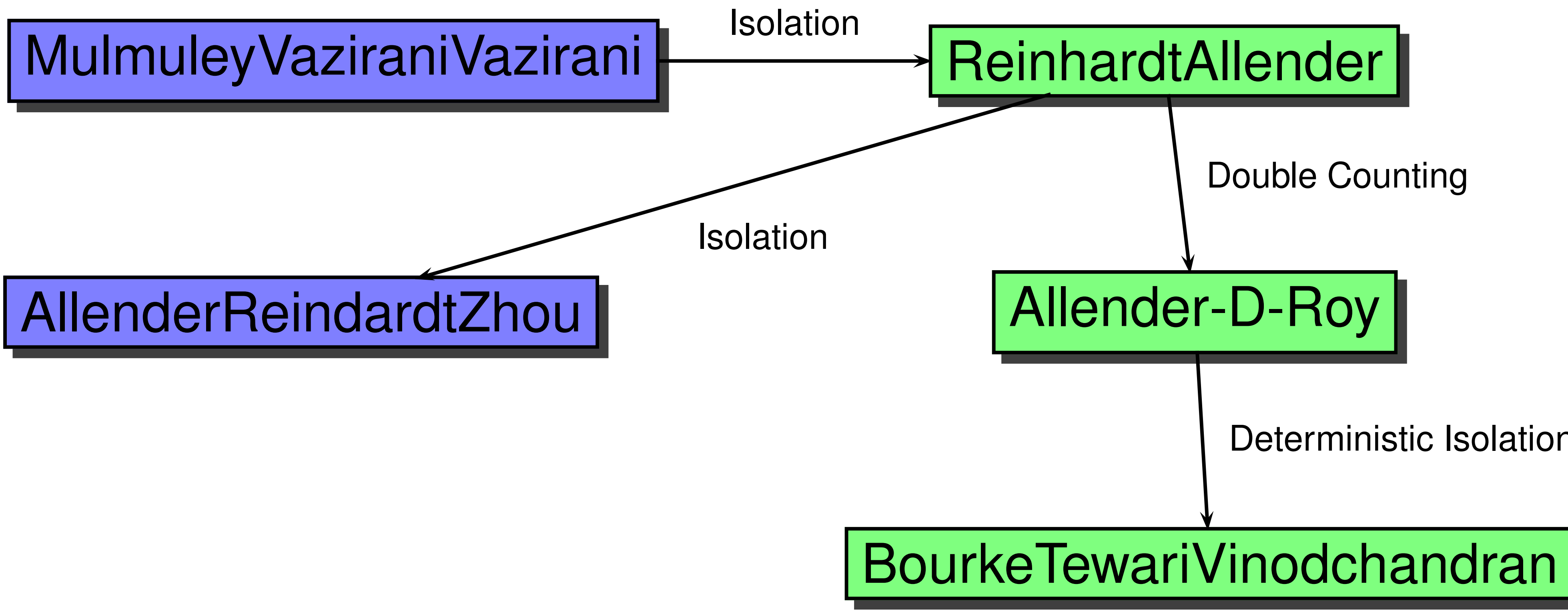
From Matching to Reachability and Back



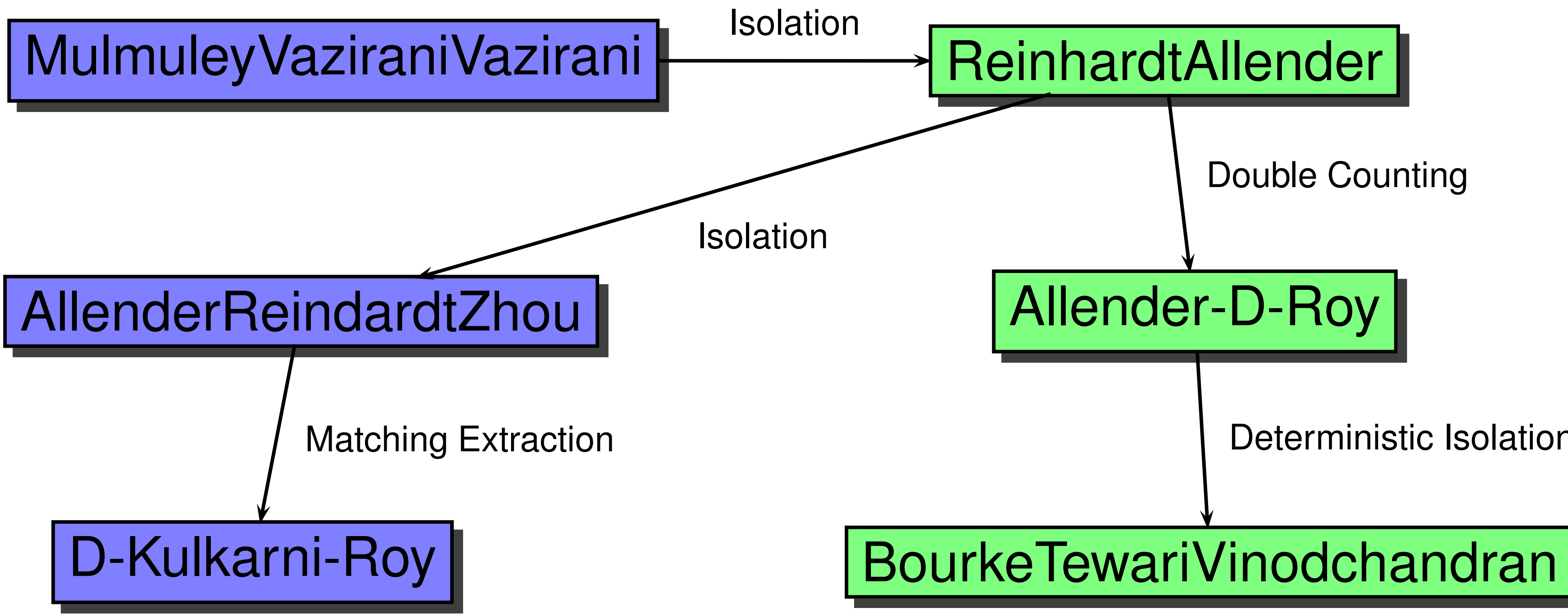
From Matching to Reachability and Back



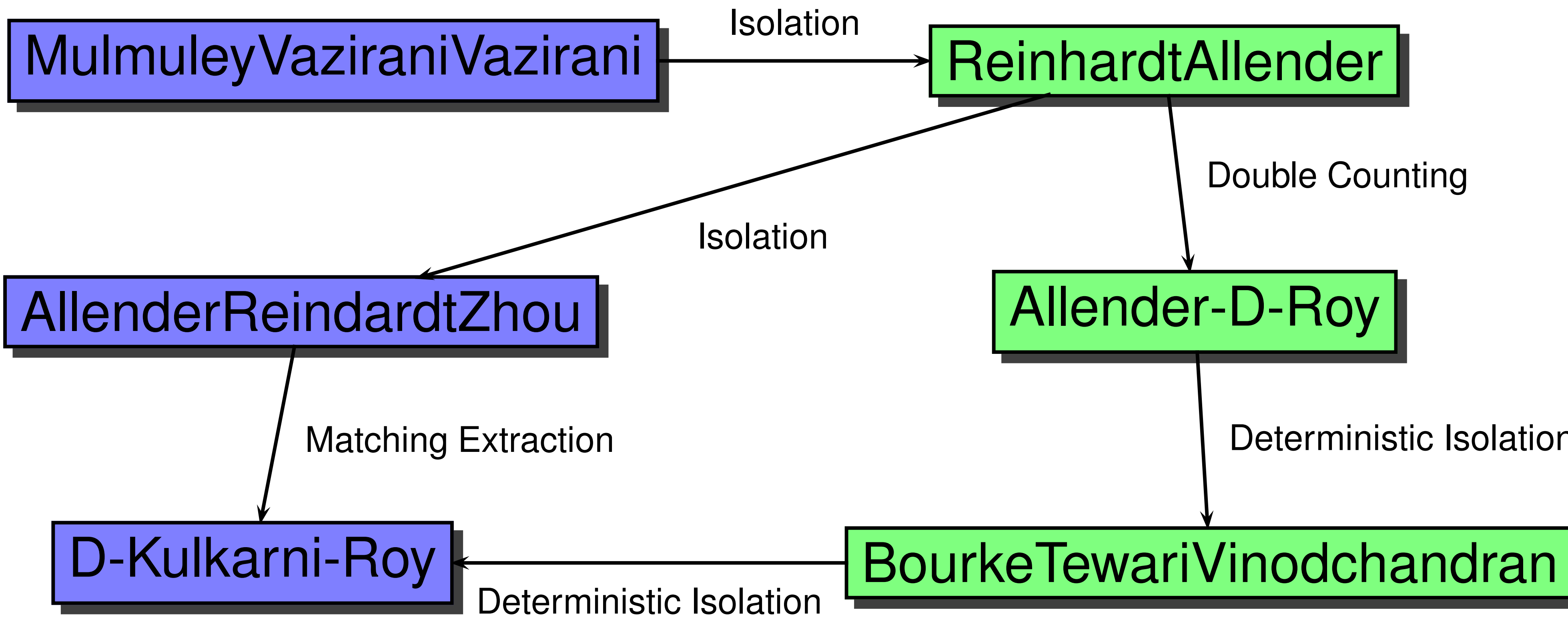
From Matching to Reachability and Back



From Matching to Reachability and Back



From Matching to Reachability and Back



Thank You!