Maximal Matching via Gaussian Elimination

Piotr Sankowski

Uniwersytet Warszawski

Outline

- Maximum Matchings
- Lovásza's Idea,
- Maximum matchings,
- Rabin and Vazirani,
- Gaussian Elimination,
- Simple $O(n^3)$ time algorithm,
- $O(n^\omega)$ time for bipartite graphs,
- $O(n^\omega)$ time for non-bipartite graphs – idea,
- Weighed matching in bipartite graphs.

Previous Results

- $O(m\sqrt{n})$ time for bipartite graphs — Hopcroft and Karp '73,
- $O(m\sqrt{n})$ time for general graphs — Micali and Vazirani '80,

For dense graphs this gives $O(n^{2.5})$ time.

Algebraic techniques:

- $O(n^{\omega}) = O(n^{2.38})$ testing and computing the size — Lovász '79,
- $O(n^{\omega+1}) = O(n^{3.38})$ finding — Rabin and Vazirani '89.

The Algebraic Matchings

New method based on Gaussian elimination.

Algebraic algorithms for finding maximum size matchings:

- simple $O(n^3)$ time,
- $O(n^\omega) = O(n^{2.38})$ time,
- for weighted graphs $O(Wn^\omega) = O(Wn^{2.38})$ time.

These algorithms are randomized Monte Carlo.

Fast Matrix Multiplication

Let ω be the matrix multiplication exponent.

Twierdzenie 1 (Coppersmith and Winograd '90)

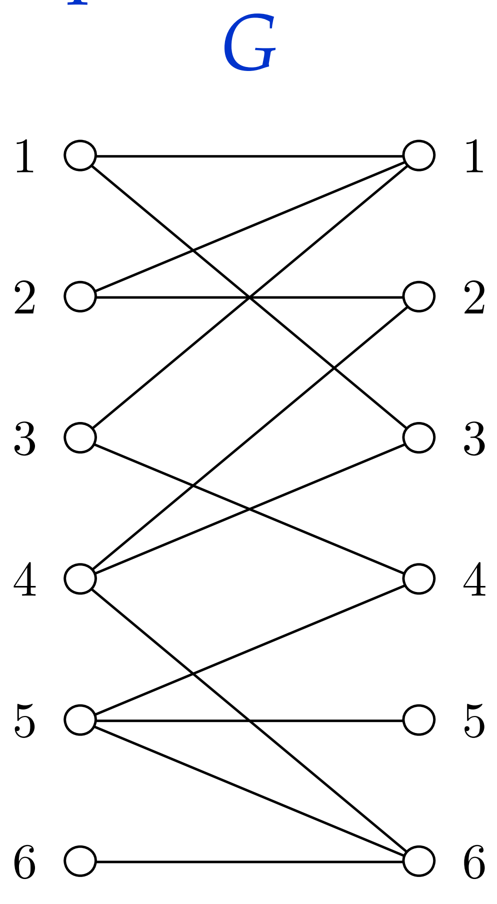
$$\omega < 2.376.$$

Twierdzenie 2 (Bunch and Hopcroft '74)
LU-factorization (Gaussian elimination) can be computed in $O(n^{\omega})$.

Twierdzenie 3 (Ibarra, Moran and Hui '82)
Maximum size nonsingular submatrix can be computed in $O(n^{\omega})$ time.

Symbolic Adjacency Matrix

The symbolic adjacency matrix of a bipartite graph:



$$\Longrightarrow \quad \tilde{A}(G) = \begin{pmatrix} x_{11} & 0 & x_{13} & 0 & 0 & 0 \\ x_{21} & x_{22} & 0 & 0 & 0 & 0 \\ x_{31} & 0 & 0 & x_{34} & 0 & 0 \\ 0 & x_{42} & x_{43} & 0 & 0 & x_{46} \\ 0 & 0 & 0 & x_{54} & x_{55} & x_{56} \\ 0 & 0 & 0 & 0 & 0 & x_{66} \end{pmatrix}$$

Symbolic Adjacency Matrix

$$\det \begin{pmatrix} x_{11} & 0 & x_{13} & 0 & 0 & 0 \\ x_{21} & x_{22} & 0 & 0 & 0 & 0 \\ x_{31} & 0 & 0 & x_{34} & 0 & 0 \\ 0 & x_{42} & x_{43} & 0 & 0 & x_{46} \\ 0 & 0 & 0 & x_{54} & x_{55} & x_{56} \\ 0 & 0 & 0 & 0 & 0 & x_{66} \end{pmatrix} =$$

$$= -x_{13}x_{21}x_{34}x_{42}x_{55}x_{66} - x_{11}x_{22}x_{34}x_{43}x_{55}x_{66}.$$

The monomials in the determinant correspond to perfect matchings in G.

Symbolic Adjacency Matrix

The determinant is given as:

$$\det(A) = \sum_{p \in \Pi_n} \sigma(p) \prod_{i=1}^{n} a_{i,p_i}.$$

Each nonzero term in this sum chooses for every vertex i a different vertex p_i.

The terms in this sum correspond to perfect matchings.

Symbolic Adjacency Matrix

Twierdzenie 4 *For a bipartite graph G, $\det \tilde{A}(G) \neq 0$ iff G has a perfect matching.*

Substitute random numbers into $\tilde{A}(G)$ and compute the determinant of $A(G)$ — *random adjacency matrix*.

With high probability $\det A(G) \neq 0$ iff $\det \tilde{A}(G) \neq 0$, because 'polynomials do not have many zeros' — this gives an efficient test by Zuppel-Schwartz lemma.

Lovász's Idea

An $O(n^\omega)$ time (Monte Carlo) algorithm testing whether graph G has a perfect matching:

substitute for variables in $\tilde{A}(G)$
    radom elements from $\mathcal{Z}_P$
let $A(G)$ be the resulting matrix
if det $A(G) <> 0$ **then**
    return "YES"
else
    return "NO"

Lovász's Idea

An $O(n^{\omega+2})$ time (Monte Carlo) finding a perfect matching in G:

$M := \emptyset$
for $e \in E$ **do**
  if $G - e$ has a perfect matching **then**
    remove e with its endpoints from G
    add e to M

Maximum Matching

Twierdzenie 5 (Lovász (79)) *Let m be a maximum matching size in G, then* $\text{rank}(\tilde{A}(G)) = m$.

The rank of $A(G)$ can be computed in $O(n^\omega)$ time.

Let M a matching in then from Tutte's theorem $\tilde{A}_{V(M),V(M)}(G)$ is nonsingular, i.e., $\text{rank}(\tilde{A}(G)) \geq m$.

Maximum Matching

Let $\tilde{A}_{X,Y}(G)$ be maximum size nonsingular submatrix of $\tilde{A}(G)$.

The determinant of $\tilde{A}_{X,Y}$ is non-zero.

In $\det(\tilde{A}_{X,Y})$ there exists a nonzero permutation p.

p gives a perfect matching of X and Y so $\operatorname{rank}(\tilde{A}(G)) \leq m$.

Rabin and Vazirani Algorithm

$A^{-1}_{i,j} = (-1)^{i+j} \det A^{j,i} / \det A$, where $A^{j,i}$ is the matrix A with j-th row and i-th column removed.

When G is bipartite then $A^{j,i} = A(G - \{u_j, v_i\})$.

The matrix $A(G)^{-1}$ codes which edges in G are *allowed*, i.e., belong to some perfect matching.

Rabin and Vazirani Algorithm

An $O(n^{\omega+1}) = O(n^{3.38})$ time (Monte Carlo) algorithm for finding a perfect matching in G:

$M := \emptyset$
while G is not empty **do**
    compute $A^{-1}(G)$
    find allowed edge $e \in E$
    remove e with its endpoints from G
    add e to M

Gaussian Elimination

Twierdzenie 6 (Elimination Theorem) *Niech*

$$A = \begin{pmatrix} a_{1,1} & v^T \\ u & B \end{pmatrix}, \quad A^{-1} = \begin{pmatrix} \hat{a}_{1,1} & \hat{v}^T \\ \hat{u} & \hat{B} \end{pmatrix},$$

where $\hat{a}_{1,1} \neq 0$. *Wtedy* $B^{-1} = \hat{B} - \hat{u}\hat{v}^T / \hat{a}_{1,1}$.

This is a single step of Gaussian elimination.

An $O(n^3)$ Time Algorithm

A Monte Carlo algorithm that finds a perfect matching in graph G in $O(n^3)$ time:

> $M := \emptyset$
> compute $A^{-1}(G)$
> **while** G non-empty **do**
>   fine arbitrary allowed edge $e \in E$
>   remove e with its endpoints from G
>   add e to M
>   update $A^{-1}(G)$
>   using Gaussian elimination

Lazy Updates

$$u_1 v_1^T + \ldots + u_k v_k^T = \left(\begin{array}{c|c|c} u_1 & \ldots & u_k \end{array} \right) \left(\begin{array}{c} v_1^T \\ \hline \vdots \\ \hline v_k^T \end{array} \right)$$

Elimination Without Pivots

The following algorithm performs Gaussian elimination without column or row pivoting in $O(n^\omega)$ time.

for $i := 1$ **to** n **do**
  lazily eliminate i-th row and i-th column
  let k be such that $2^k \mid i$, but $2^{k+1} \nmid i$
  update rows and columns
    with numbers $i+1, \ldots, i+2^k$

Elimination Without Pivots

In each step we need to multiply an $n \times 2^k$ matrix by an $2^k \times 2^k$ matrix in $(n/2^k)(2^k)^\omega = n2^{k(\omega-1)}$ time.

The given value k appears $n/2^k$ times, so the computations for this k require $n^2 2^{k(\omega-2)} = n^2(2^{\omega-2})^k$ time.

$$\sum_{k=0}^{\log n} n^2(2^{\omega-2})^k \leq Cn^2(2^{\omega-2})^{\log n} = Cn^2 n^{\omega-2} = Cn^\omega.$$

Matching Verification

Twierdzenie 7 *Inclusion-wise maximal allowed submatching M' of a matching M in bipartite graph G can be computed in $O(n^\omega)$ time (Monte Carlo).*

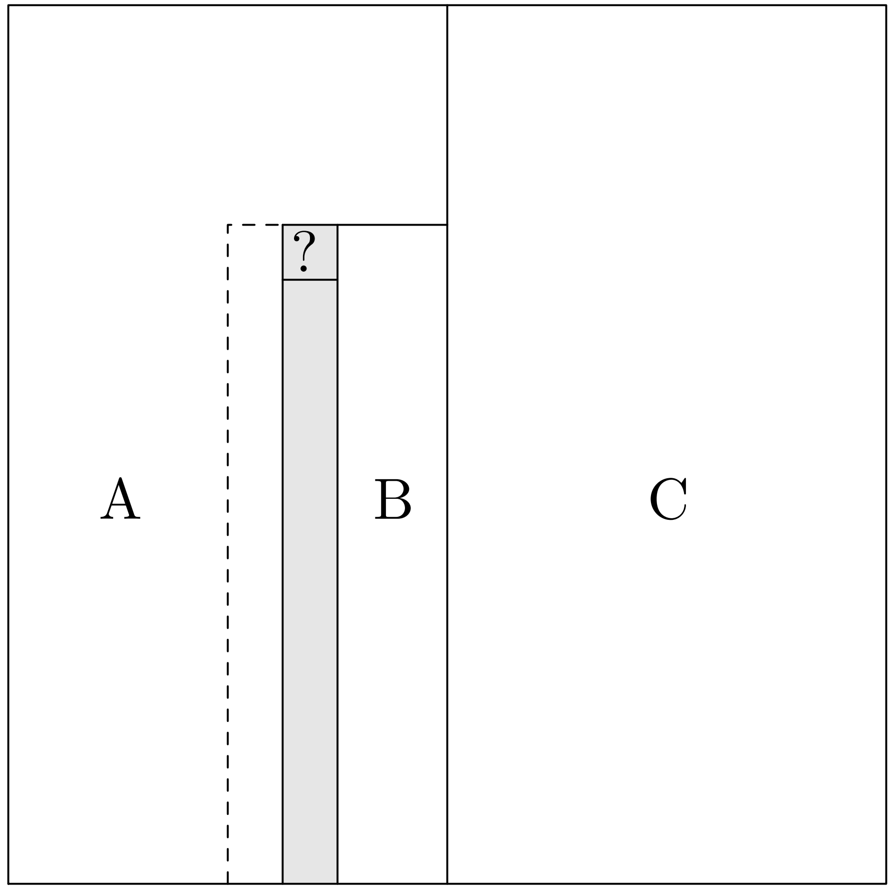
	u_1	u_2	u_3	u_4	$\dots$	u_{n-1}	u_n
v_1	$\surd$						
v_2		$\surd$					
v_3			$\times$				
v_4				$\times$			
$\dots$					$\ddots$		
v_{n-1}						$\surd$	
v_n							$\surd$

Elim. Without Column Pivots

Hopcroft-Bunch LU algorithm executes Gaussian elimination with row but without column pivots in $O(n^\omega)$ time.

for $i := 1$ **to** n **do**
  find row j such that $A_{i,j} \neq 0$
  lazily eliminate j-th row and i-th column
  let k be such that $2^k \mid i$ but $2^{k+1} \nmid i$
  update columns $i+1, \ldots, i+2^k$

Elim. Without Column Pivots



Bipartite Case

Twierdzenie 8 *A perfect matching in a bipartite graph can be found in* $O(n^{\omega})$ *time using modified Hopcroft-Bunch algorithm.*

Can this result be extended to weighted matching problem?

YES — the algorithm works in $O(Wn^{\omega})$ time.

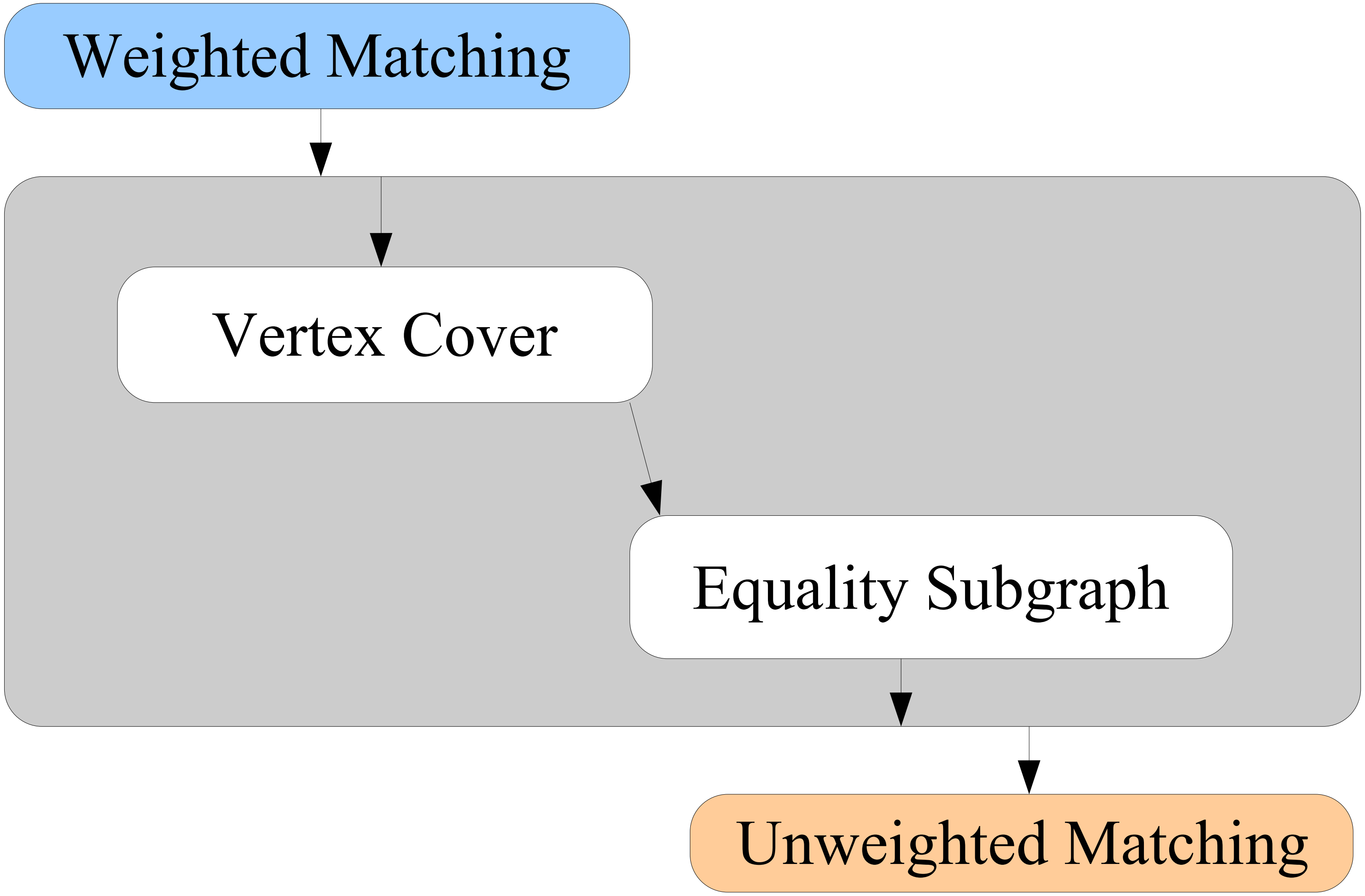
Idea

Weighted Matching

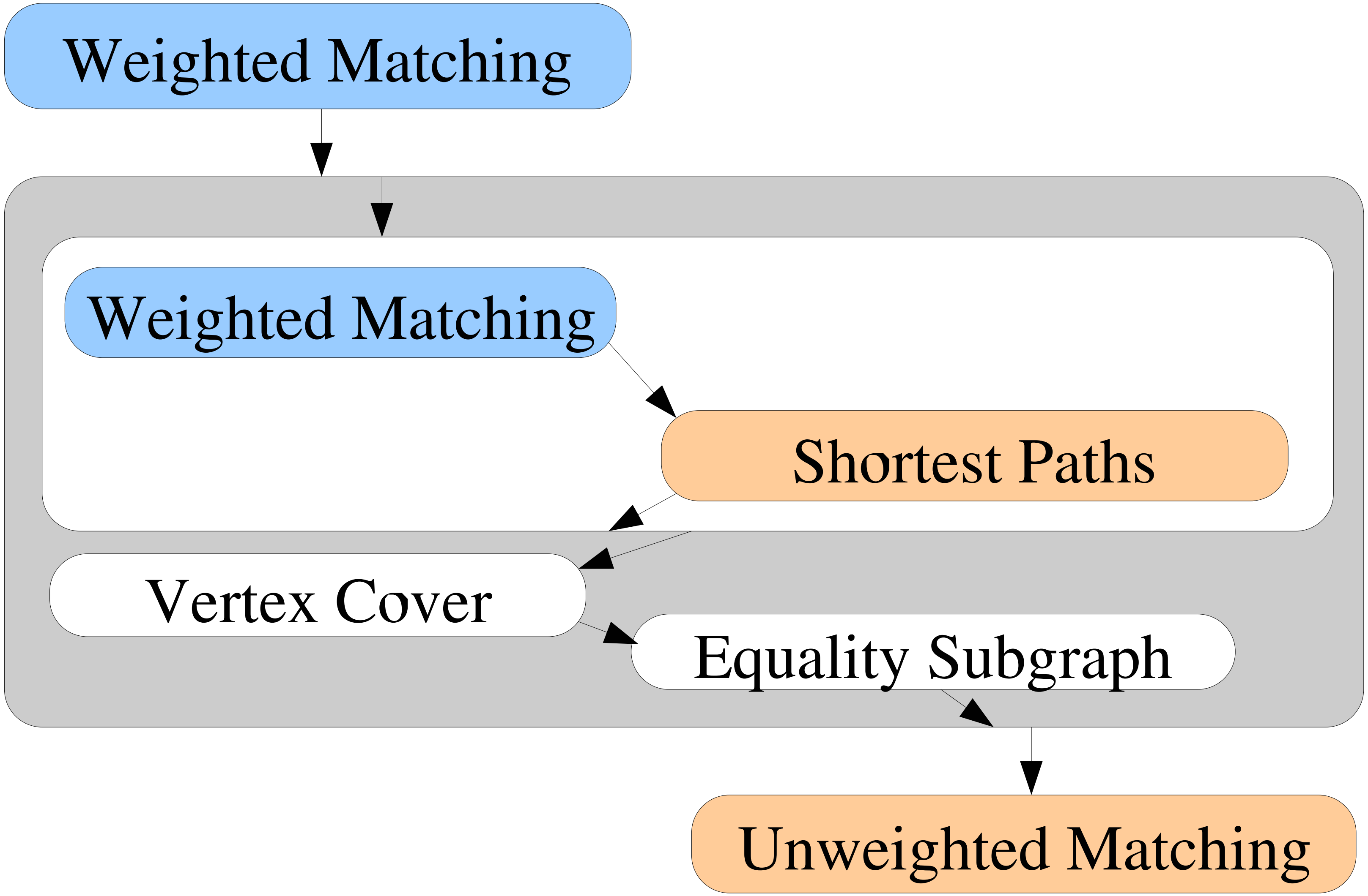
Reduction in $O(Wn^{\omega})$ time

Unweighted Matching
$O(n^{\omega})$ time

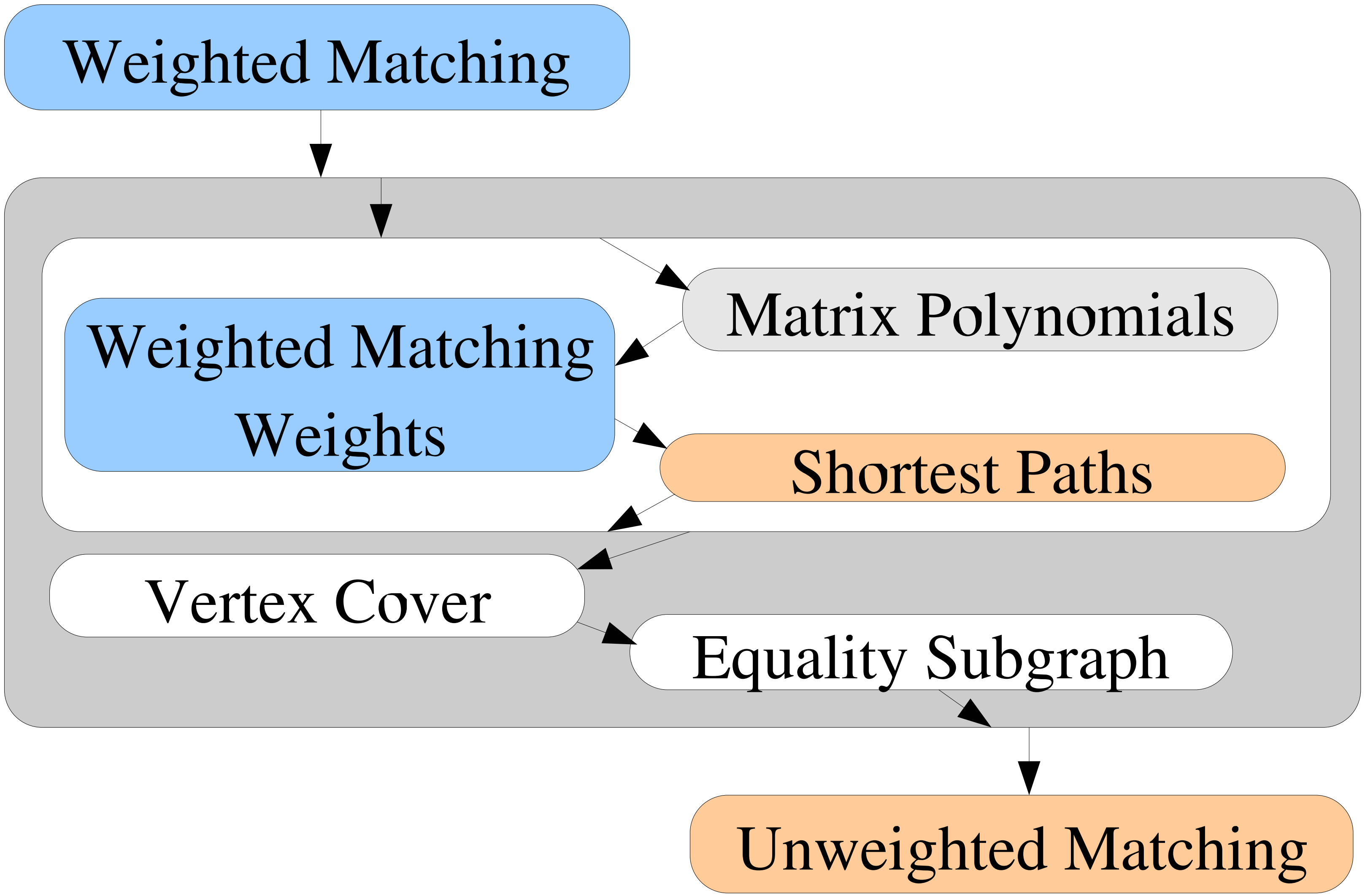
Idea


Weighted Matching
Vertex Cover
Equality Subgraph
Unweighted Matching

Idea

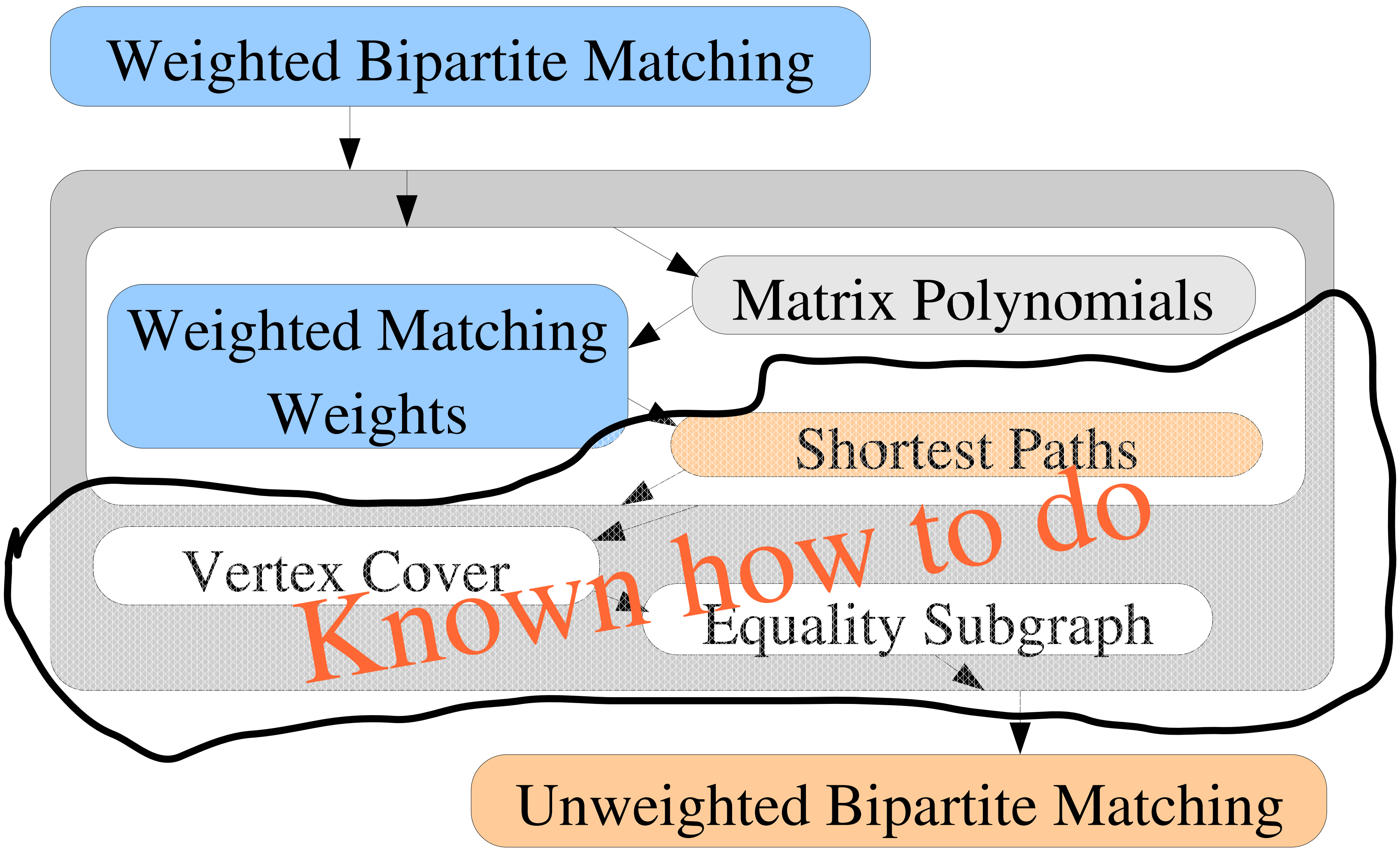

Weighted Matching
Weighted Matching
Shortest Paths
Vertex Cover
Equality Subgraph
Unweighted Matching

Idea



Weighted Matching
Matrix Polynomials
Weighted Matching
Weights
Shortest Paths
Vertex Cover
Equality Subgraph
Unweighted Matching

Bipartite Case



Some Definitions

A *weighted bipartite n-vertex graph* G is a tuple $G = (U, V, E, w)$, where

- $U = \{1, \ldots, n\}$ and $V = \{n+1, \ldots, 2n\}$ denote vertex sets,
- $E \subseteq U \times V$ denotes the edge set,
- the function $w : E \to \mathcal{Z}_+$ ascribes weights to the edges.

Some Definitions

In the *maximum weighted bipartite matching problem* we seek

- a perfect matching M in a weighted bipartite graph G,
- with maximum total weight $w(M) = \sum_{e \in M} w(e)$.

The size of the maximum weighted matching problem is given by n and W – the maximum weight in w.

Some Definitions

A *weighted cover* is a choice of labels $y(1), \ldots, y(2n)$ such that

$$y(i) + y(j) \geq w(ij),$$

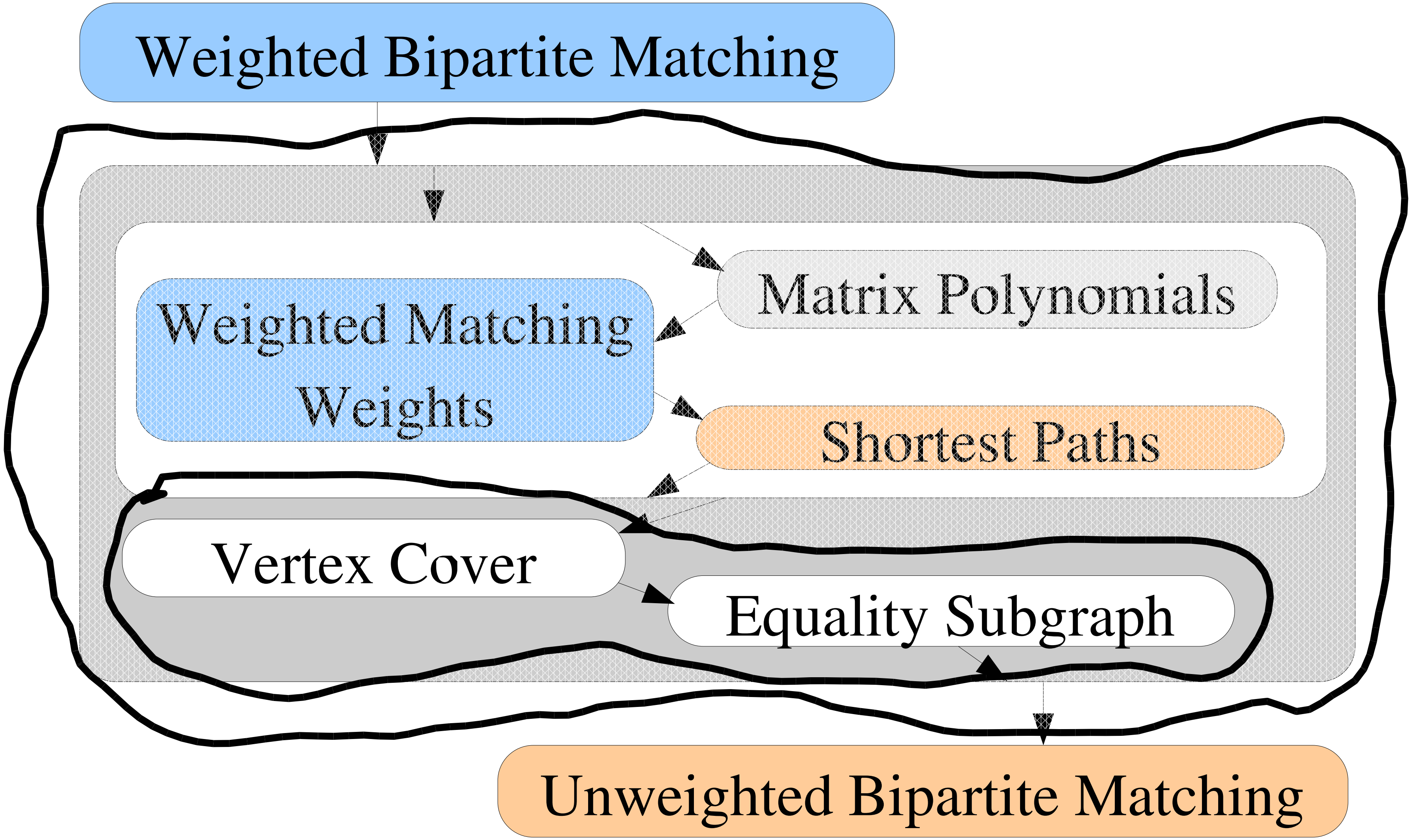
for all i, j.

The *minimum weighted cover problem* is that of finding a cover of minimum cost.

Egerváry Theorem

Twierdzenie 9 (Egerváry '31)
Let $G = (U, V, E, w)$ be a weighted bipartite graph.
The maximum weight of a perfect matching of G is equal to weight of the minimum weighted cover of G.

From Cover to Matching



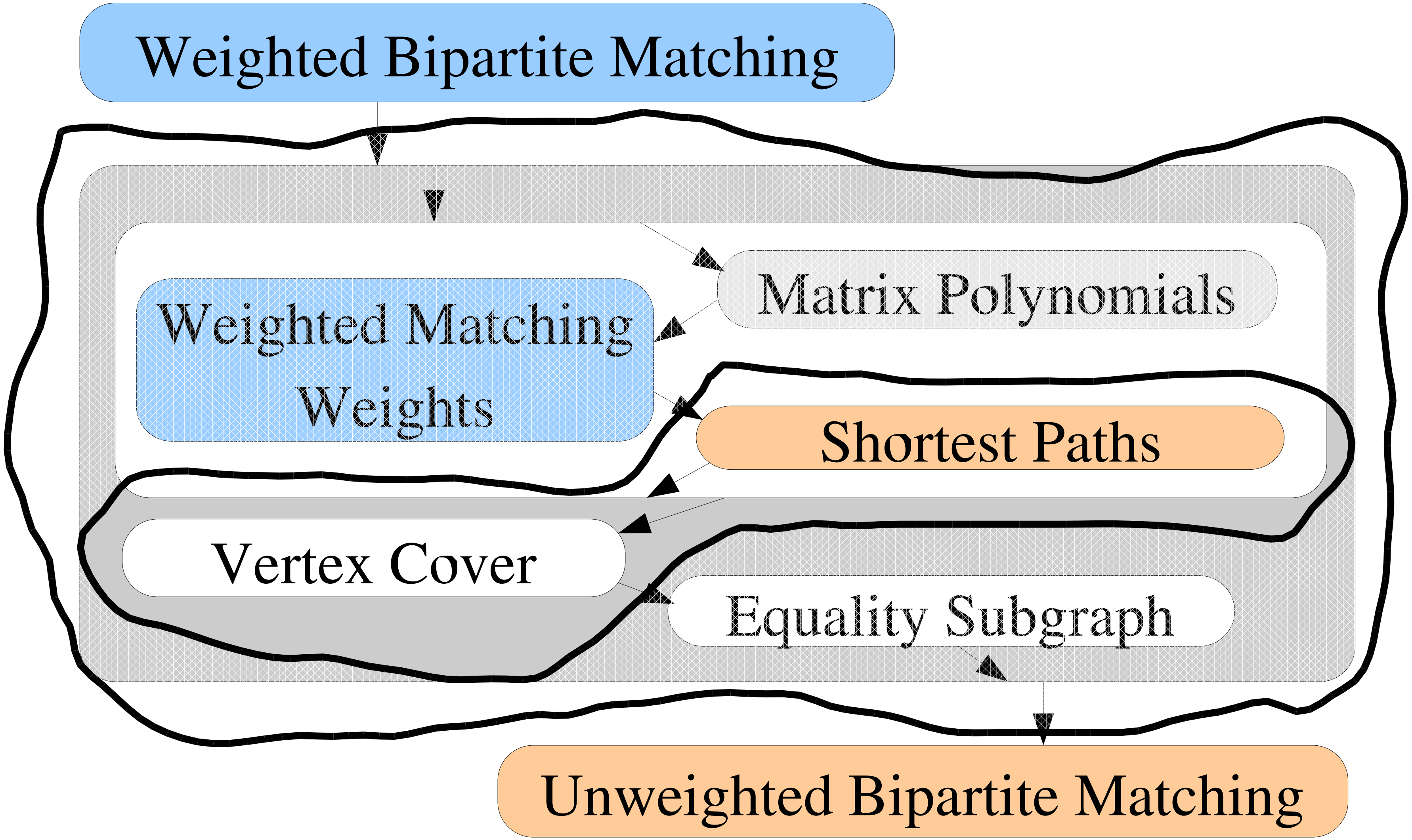
From Cover to Matching

The *equality graph* G_p for p and G is defined as

- $G_p = (U, V, E')$,
- $E' = \{uv : uv \in E$ and $p(u) + p(v) = w(uv)\}$.

Lemat 10
Consider a weighted bipartite graph G and a minimum weighted vertex cover p of G.
The matching M is a perfect matchings in G_p iff it is a maximum weighted perfect matchings in G.

From Paths to Cover



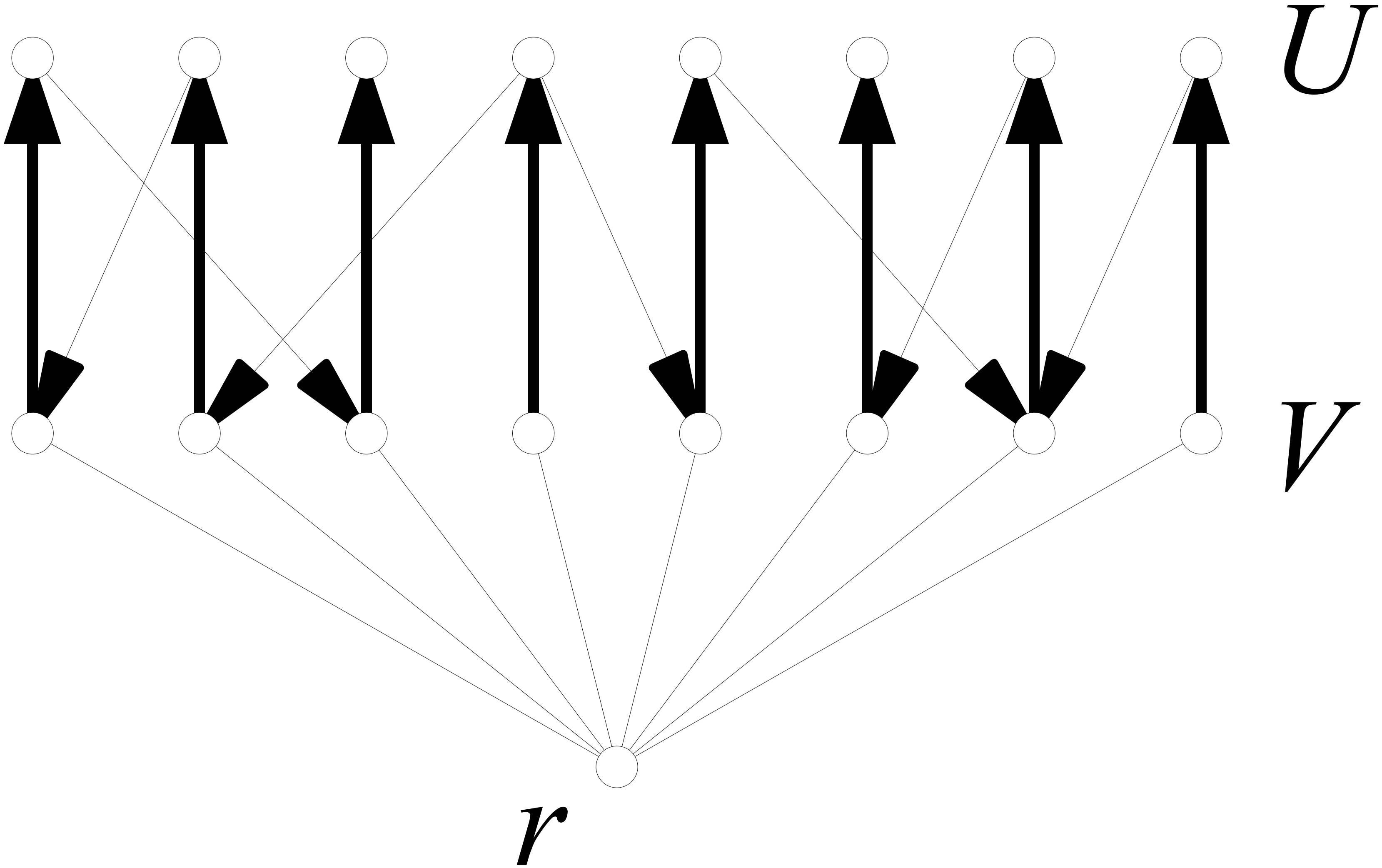
From Paths to Cover

Let M be a maximum weighted matching in a weighted bipartite graph $G = (U, V, E, w)$.

1. construct a directed weighted graph $D = (U \cup V \cup \{r\}, A, w_d)$,
2. for all $uv \in E$, $u \in U$ and $v \in V$, add an edge (u, v) to A, $w_d((u, v)) := -w(uv)$,
3. for all $uv \in M$, $u \in U$ and $v \in V$, add an edge (v, u) to A, $w_d((v, u)) := w_{uv}$,

No negative-weight cycles in D.

From Paths to Cover



From Paths to Cover

1. add zero weight edges (r, v) for each $v \in V$,
2. compute distances in D from r,
3. set $y_u := \text{dist}(r, u)$ for $u \in U$,
4. set $y_v := -\text{dist}(r, v)$ for $v \in V$.

Lemat 11 *The y found by the above algorithm is a minimum weighted vertex cover in G.*

From Paths to Cover

$\text{dist}(r,u)$ is a potential function

$$w_d((u,v)) \geq \text{dist}(r,v) - \text{dist}(r,u)$$

For $uv \in E$ we have an edge (u,v) in D and

$$w_d((u,v)) \geq \text{dist}(r,v) - \text{dist}(r,u),$$

$$-w(uv) \geq -y(v) - y(u).$$

$$w(uv) \leq y(v) + y(u).$$

Thus y is a vertex cover.

From Paths to Cover

$\mathrm{dist}(r,u)$ is a potential function

$$w_d((u,v)) \geq \mathrm{dist}(r,v) - \mathrm{dist}(r,u)$$

For $uv \in M$ we have an edge (v,u) in D and

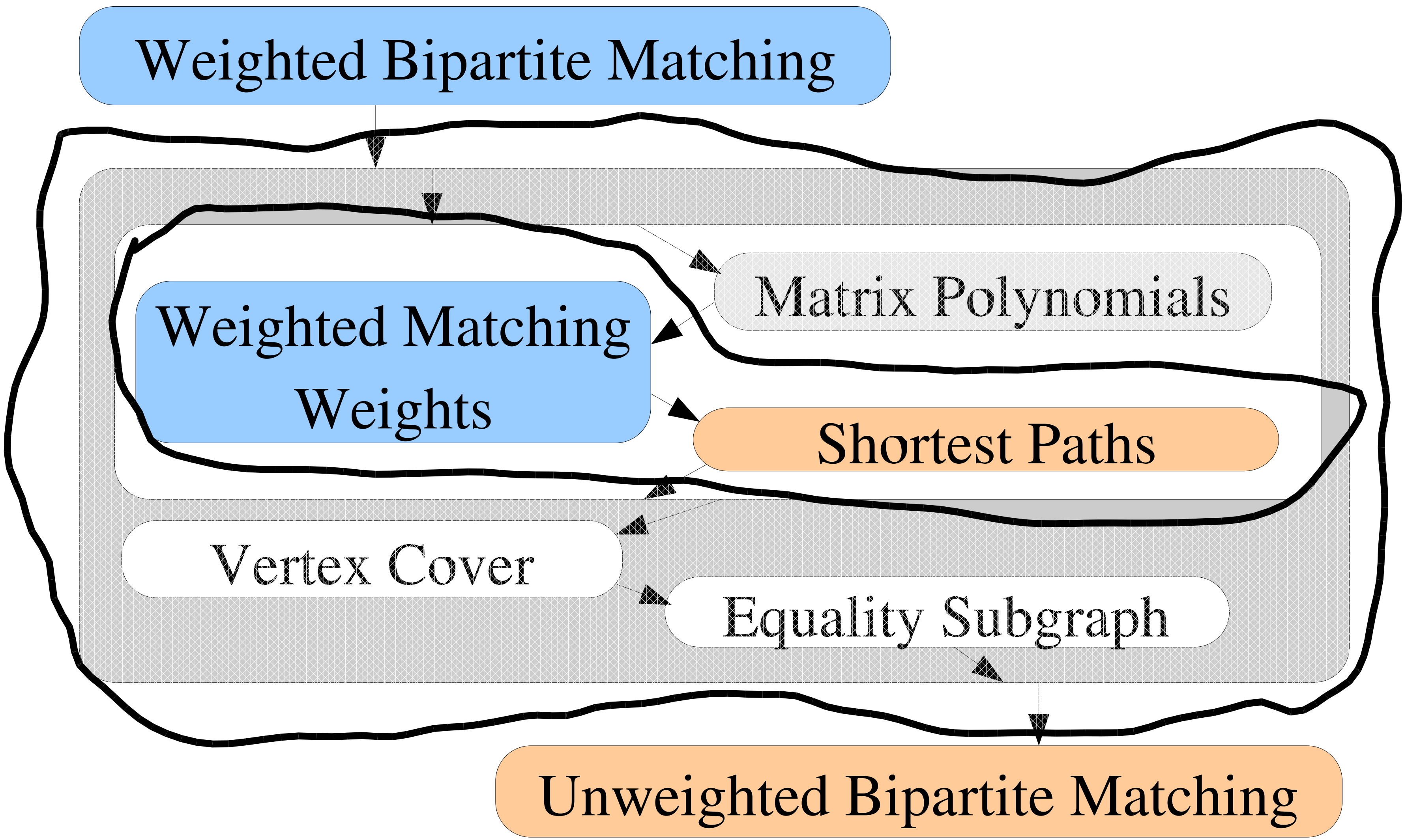
$$w_d((v,u)) \geq \mathrm{dist}(r,u) - \mathrm{dist}(r,v),$$

$$w(uv) \geq y(u) + y(v).$$

Summing up the above inequality for all edges in M we obtain that $w(y) \leq w(M)$.

Thus y is minimum.

From Matching Weights to Paths



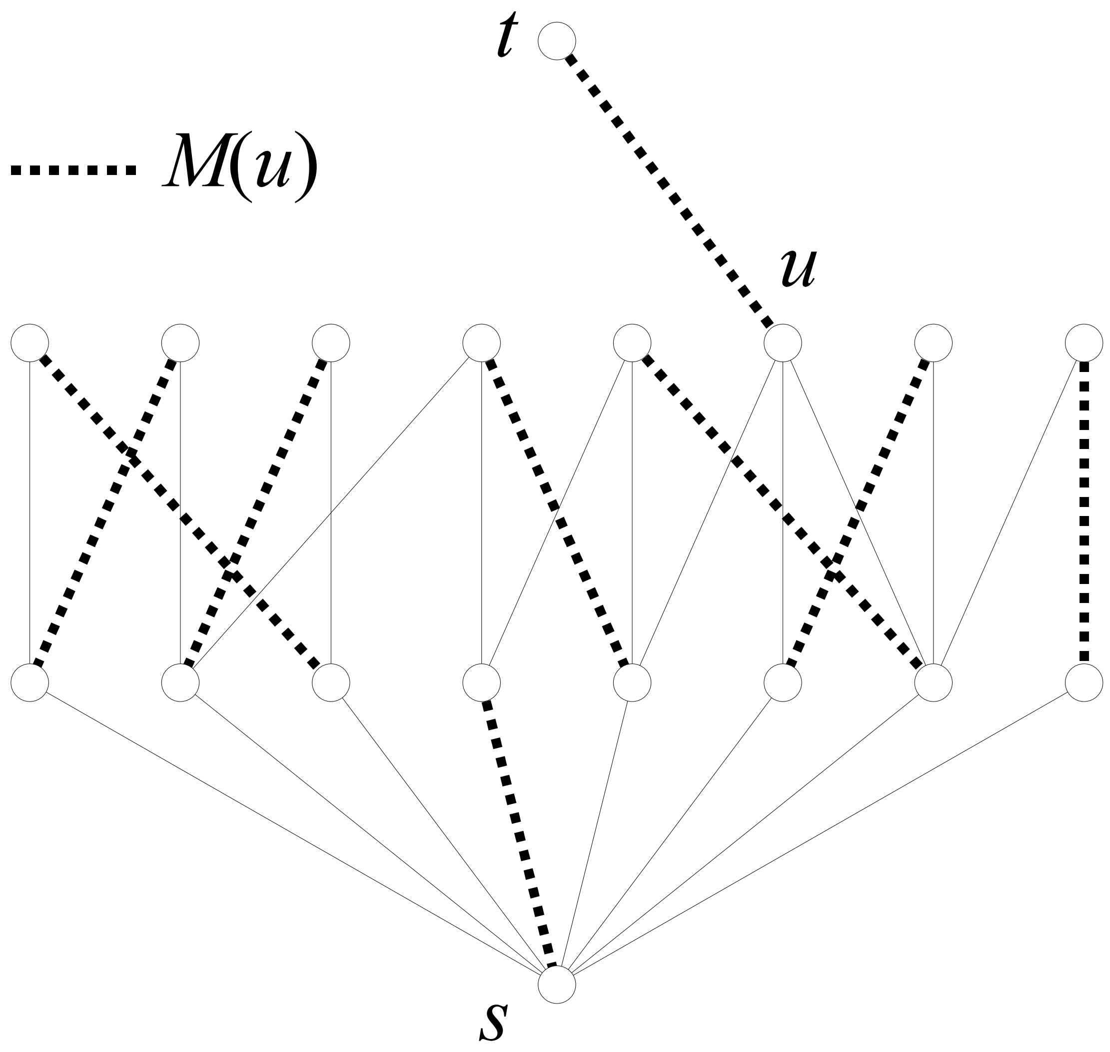
From Matching Weights to Paths

Consider a weighted bipartite graph $G = (U, V, E, w)$.

- add a new vertex s to U,
- add a new vertex t to V,
- connect s with all vertices from V with zero weight edges,
- connect the vertex t with the vertex u in U.

Let us denote by $G(u)$ the resulting graph and by $M(u)$ the maximum weighted perfect matching in this graph.

From Matching Weights to Paths



From Matching Weights to Paths

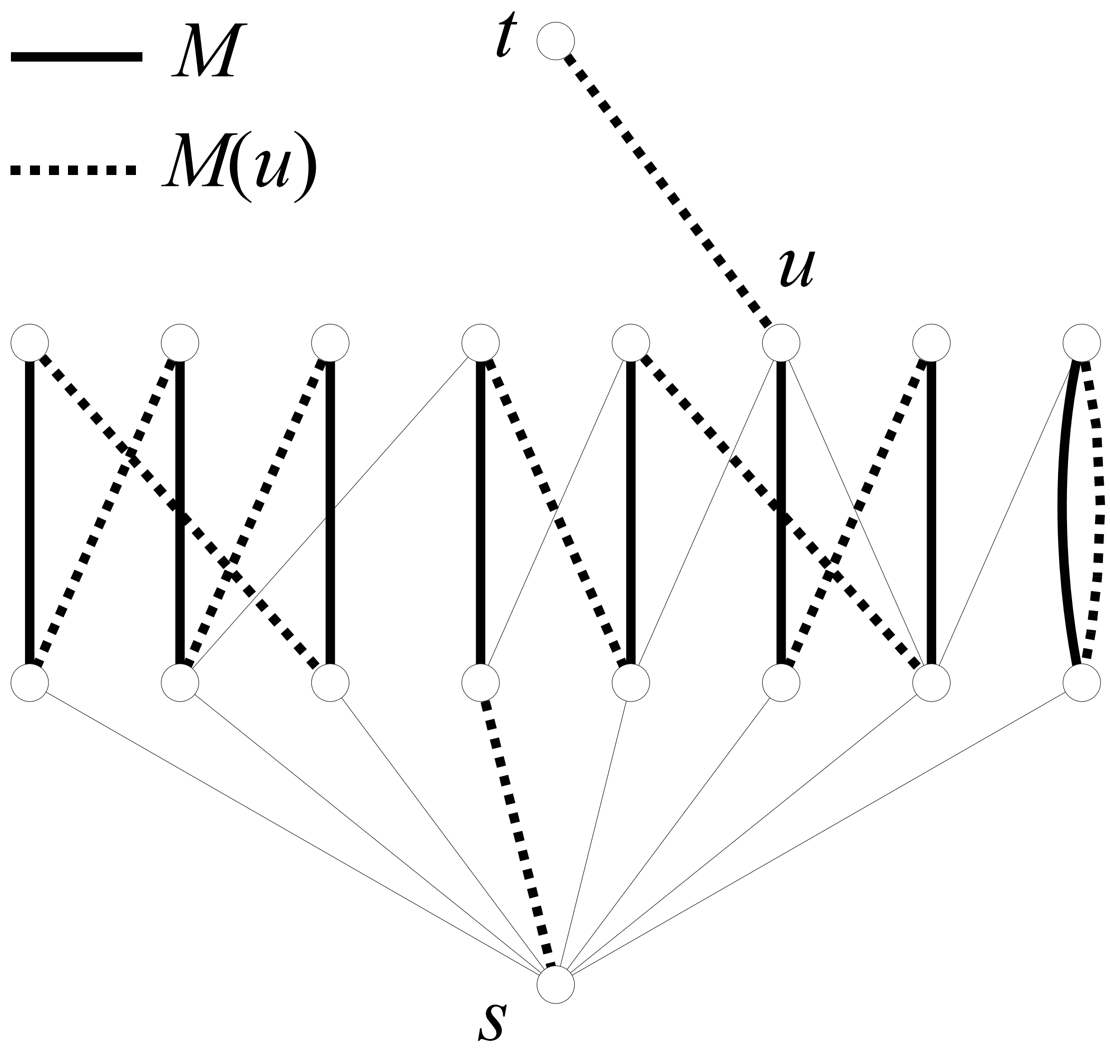
Lemat 12 *The distances in D satisfy*

$$\mathrm{dist}(r, u) := w(M) - w(M(u)) \text{ for } u \in U.$$

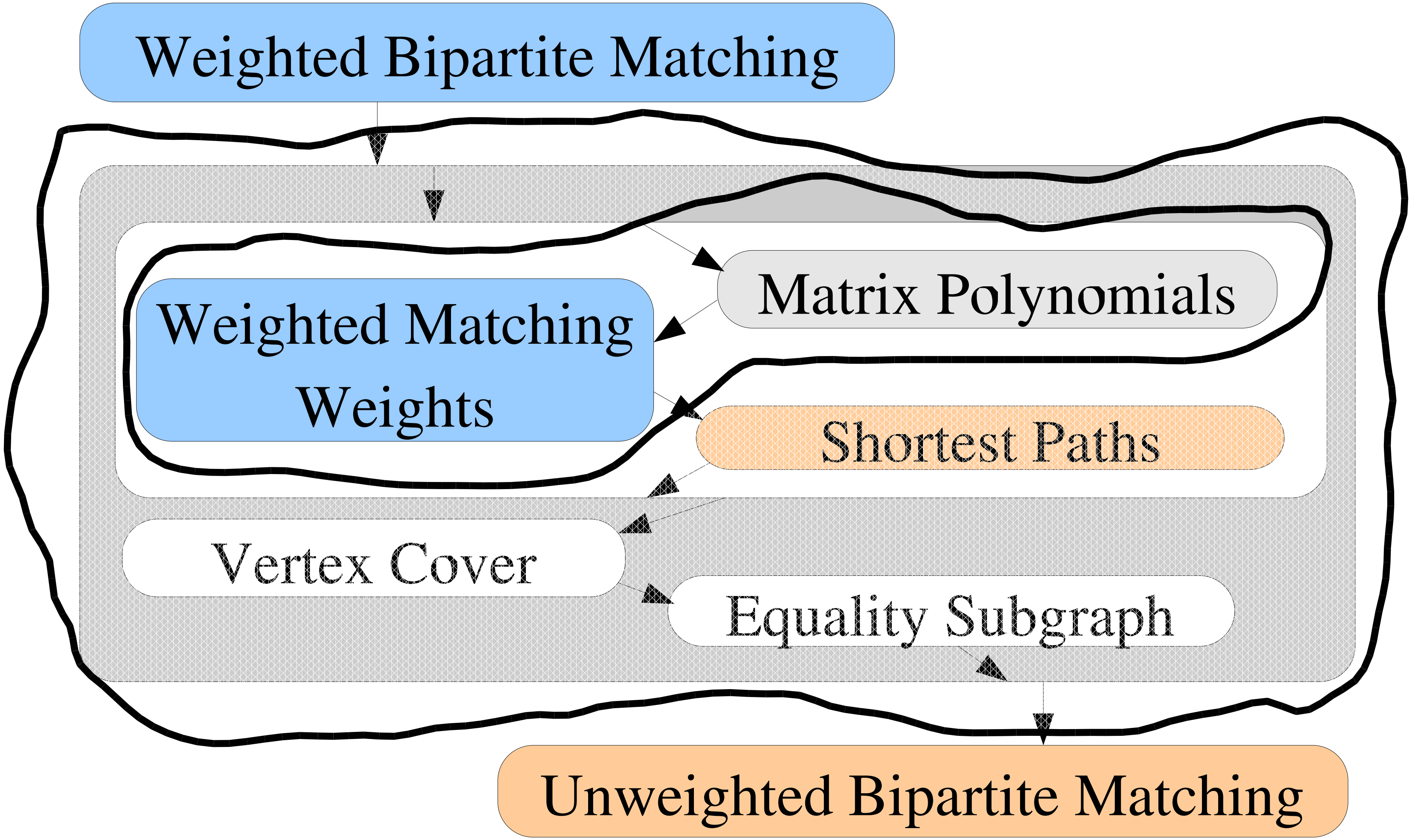
Consider the matchings $M(u)$ and M,

- direct all edges in M from V to U,
- direct all edges in $M(u)$ from U to V,
- we obtain a directed path p from s to t in D,
- and a set $\mathcal{C}$ of even length alternating cycles.

From Matching Weights to Paths



From Matrices to Matching Weights



From Matrices to Matchings

For $G = (U, V, E, w)$, define a $n \times n$ matrix $\tilde{B}(G, x)$ by

$$\tilde{B}(G, x)_{i,j} = x^{w(ij)} z_{i,j},$$

where $z_{i,j}$ are distinct variables corresponding to edges in G.

Lemat 13 (Karp, Upfal and Wigderson '86)
The degree of x in $\det(\tilde{B}(G, x))$ *is the weight of the maximum weight perfect matching in G.*

From Matrices to Matchings

Twierdzenie 14 (Storjohann '03)
Let $A \in K[x]^{n\times n}$ *be a polynomial matrix of degree* W *and* $b \in K[x]^{n\times 1}$ *be a polynomial vector of degree* W*, then*

- *determinant* $\det(A)$*,*
- *rational system solution* $A^{-1}b$*,*

can be computed in $\tilde{O}(n^\omega W)$ *operations in* K*.*

Wniosek 15 *The weight of the maximum weighted bipartite perfect matching can be computed in* $\tilde{O}(Wn^\omega)$ *time, with high probability.*

From Matrices to Matchings

Define $(n+1) \times (n+1)$ matrix $\hat{B}(G,x)$ by

$$\hat{B}(G,x) = \left[\begin{array}{ccc|c} & & & 0 \\ & \tilde{B}(G,x) & & \vdots \\ & & & 0 \\ \hline 1 & \cdots & 1 & 0 \end{array}\right].$$

We have

$$\operatorname{adj}(\hat{B}(G,x))_{n+1,i}) = \det(\hat{B}(G,x)^{i,n+1})) =$$

where $A^{i,j}$ is the matrix A with i-th row and j-th column removed.

From Matrices to Matchings

We have

$$\operatorname{adj}(\hat{B}(G,x))_{n+1,i}) = \det(\hat{B}(G,x)^{i,n+1})) =$$

$$= \det(\tilde{B}(G(i),x)),$$

From KUW Lemma we get that

$$\deg_x(\operatorname{adj}(\hat{B}(G,x))_{n+1,i})) = w(M(i)).$$

From Matrices to Matchings

Choose a prime number p of length $\Theta(\log n)$. Substitute random numbers from $\{1, \ldots, p\}$ for $z_{i,j}$ in $\hat{B}(G, x)$ to obtain $B(x)$.

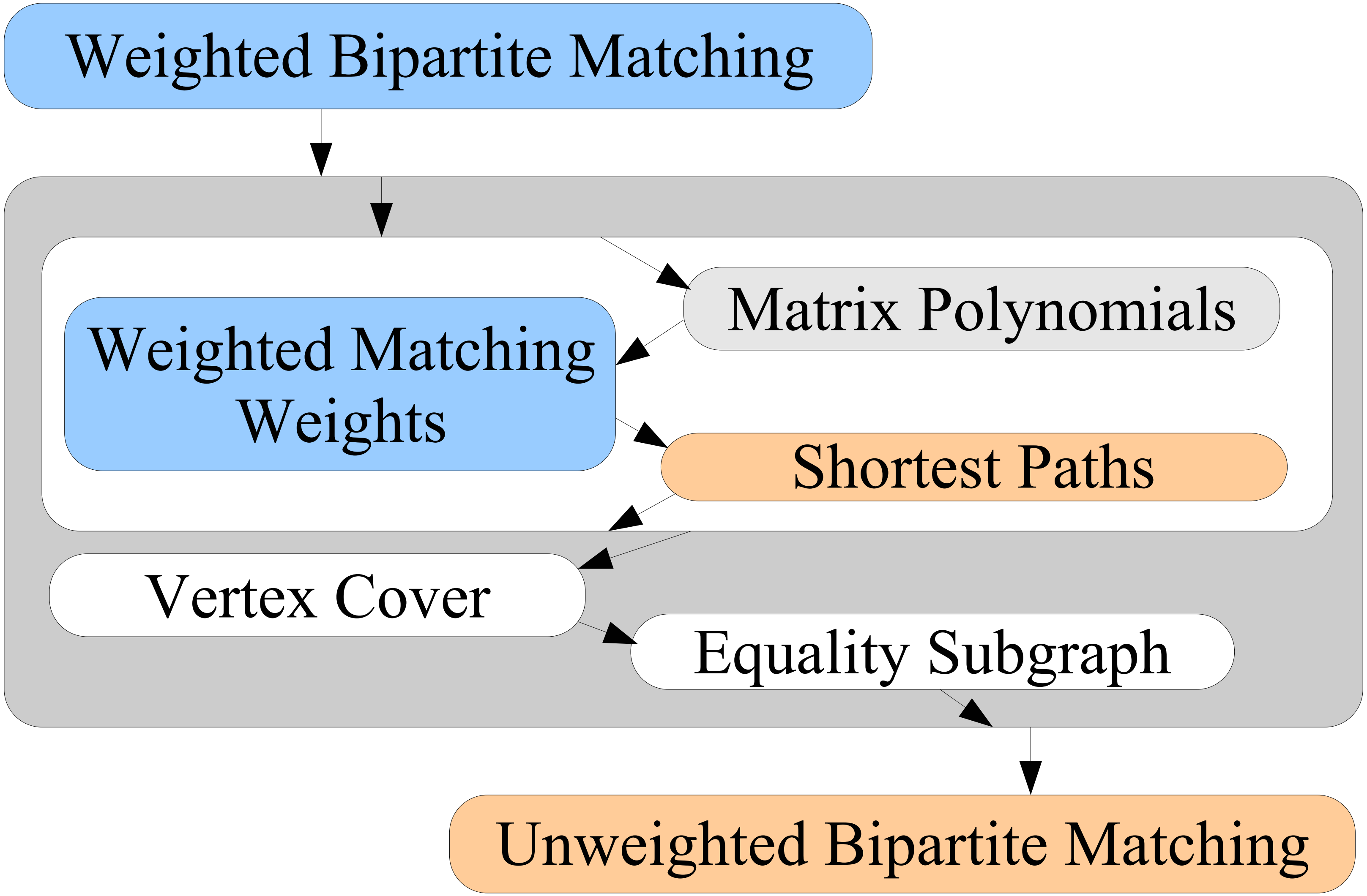
Compute with Storjohann's Theorem

$$v = \mathrm{adj}(B(x))_{n+1,i}) = (\mathrm{adj}(B(x))e_{n+1})_i =$$

$$= \det(B(x)) \left(B(x)^{-1} e_{n+1}\right)_i$$

With high probability $\deg_x(v_i) = w(M(i))$.

Weighted Bipartite Matching



Back to SSSP

We can get back to single source shortest paths with the following lemma.

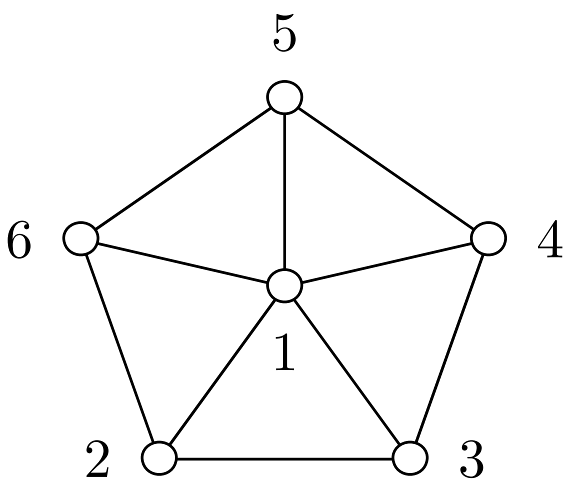
Lemat 16 (Gabow 1983) *An $f(n, m, W)$ time algorithm for maximum weighted perfect matchings implies an $f(n, m, W)$ time algorithm for SSSP with negative weights.*

Yet another algorithm for the SSSP problem working in $O(n^\omega W)$ time.

Symbolic Adjacency Matrix

Symbolic adjacency matrix of a non-bipartite graph is given as:

G $\qquad\qquad$ $\tilde{A}(G)$



$$\Longrightarrow \begin{pmatrix} 0 & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} \\ -x_{12} & 0 & x_{23} & 0 & 0 & x_{26} \\ -x_{13} & -x_{23} & 0 & x_{34} & 0 & 0 \\ -x_{14} & 0 & -x_{34} & 0 & x_{45} & 0 \\ -x_{15} & 0 & 0 & -x_{45} & 0 & x_{56} \\ -x_{16} & -x_{62} & 0 & 0 & -x_{56} & 0 \end{pmatrix}$$

Symbolic Adjacency Matrix

Twierdzenie 17 (Tutte (1947)) $\det \tilde{A}(G) \neq 0$ *iff* *G has a perfect matching.*

The determinant is given as:

$$\det(A) = \sum_{p \in \Pi_n} \sigma(p) \prod_{i=1}^{n} a_{i,p_i}.$$

The non-zero term in this sum corresponds to covering G with directed cycles.

Symbolic Adjacency Matrix

Even length cycle covers give matchings.

Consider a cycle cover p that contains a odd length cycle C.

Contribution of p cancels with a contribution of p', where C is oriented in opposite direction, because p' has:

- the same variables,
- the same parity,
- opposite sign of elements on C.

Maximum Matchings

Twierdzenie 18 (Lovász (79)) *Let m be the size of maximum matching in G, then* $\mathrm{rank}(\tilde{A}(G)) = 2m$.

The rank of $\tilde{A}(G)$ can be computed in $O(n^{\omega})$ time.

Let M be a matching then form Tutte theorem $\tilde{A}_{V(M),V(M)}(G)$ is non-singular — $\mathrm{rank}(\tilde{A}(G)) \geq 2m$.

Maximum Matchings

Let $\tilde{A}_{X,Y}(G)$ be maximum size nonsingular submatrix $\tilde{A}(G)$.

Let p be nonzero term of $\det(\tilde{A}_{X,Y}(G))$.

Let p' be nonzero term in $\det(\tilde{A}_{Y,X}(G))$ — antisymmetry.

The sum $p \cup p'$ gives an even length cycle cover of at least $|X|$ vertices in G — $\text{rank}(\tilde{A}(G)) \leq 2m$

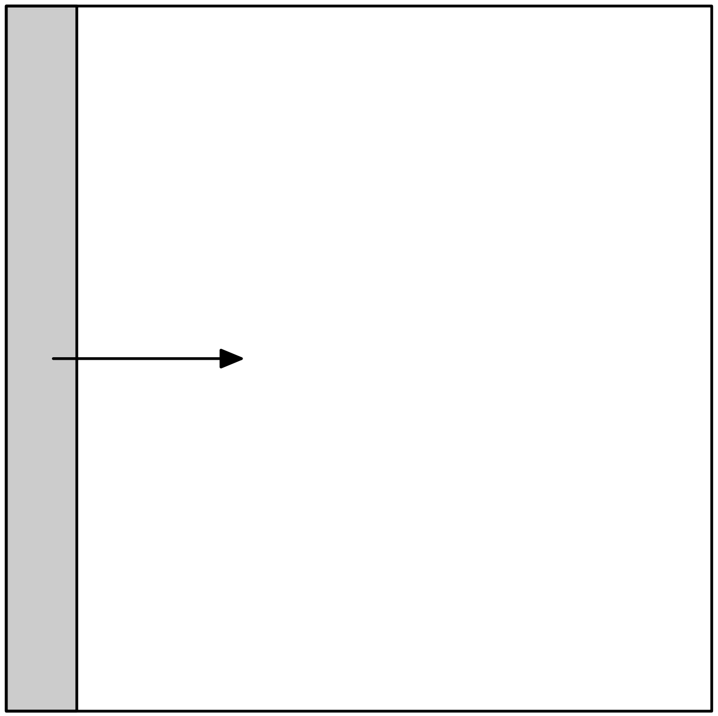
Matching Verification

Twierdzenie 19 *Inclusion-wise maximal allowed submatching M' of given matching M in G can be computed in $O(n^{\omega})$ time (Monte Carlo).*

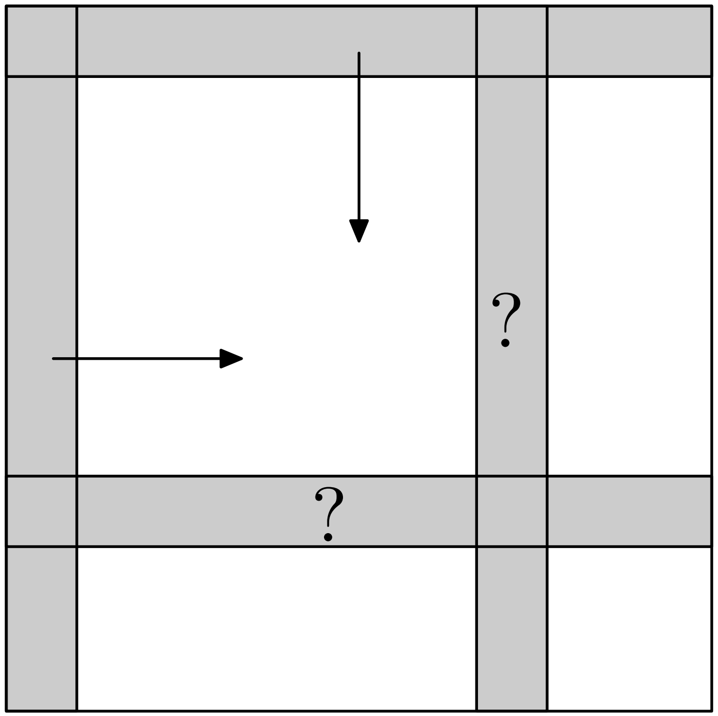
	v_1	v_2	v_3	v_4	$\ldots$	v_{n-1}	v_n
v_2	$\surd$						
v_1		$\surd$					
v_4			$\times$				
v_3				$\times$			
$\ldots$					$\ddots$		
v_{n-1}						$\surd$	
v_n							$\surd$

General Graphs

bipartite graphs



non-bipartite graphs



In non-bipartite graphs lazy updates are harder, so we will take different approach.

General Graphs

Algorithm for finding perfect matchings in general graphs:

find inclusion-wise maximal matching M in G
find maximal allowed submatching M' of M
match M' and remove it from G
if $|M'| \geq n/8$ **then**
    find perfect matching in G
else
    split G into smaller graphs
    find perfect matching in each of them

Canonical Decomposition

Elementarny graph is a graph that contains a perfect matching such that the set of allowed edges forms a connected subgraph.

Let us consider only elementary graphs.

For elementary G let $\equiv_G$ be the following relation:

$u \equiv_G v$ iff $G - \{u, v\}$ has no perfect matching.

The $\equiv_G$ relation can be read of from $A(G)^{-1}$.

Canonical Decomposition

Twierdzenie 20 $\equiv_G$ *is an equivalence relation.*

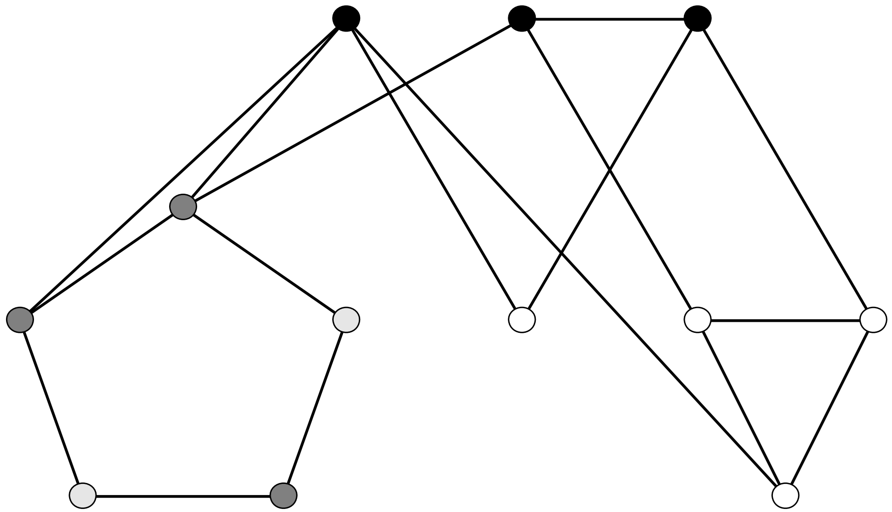
Canonical decomposition of G, is denoted by $P(G)$ and equals $V(G)/\equiv_G$.

Canonical Decomposition

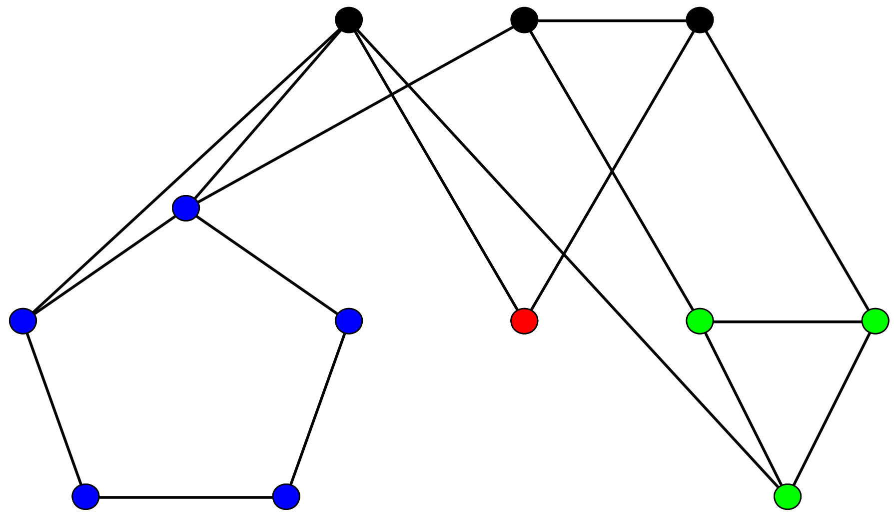
Twierdzenie 21 (Decomposition Theorem) *Let G be elementary, $S \in P(G)$, $|S| \geq 2$, and let C be some connected component of $G - S$. Wtedy:*

1. *The bipartite graph G'_S obtained from G by contracting every component in $G - S$ to a vertex and removing edges in S, is elementary;*
2. *The component C is* factor-critical*;*
3. *The graph C' obtained from $G[V(C) \cup S]$ by contracting S to single vertex v_c, is elementary;*
4. $P(C') = \{\{v_c\}\} \cup \{T \cap V(C) | T \in P(G)\}$.

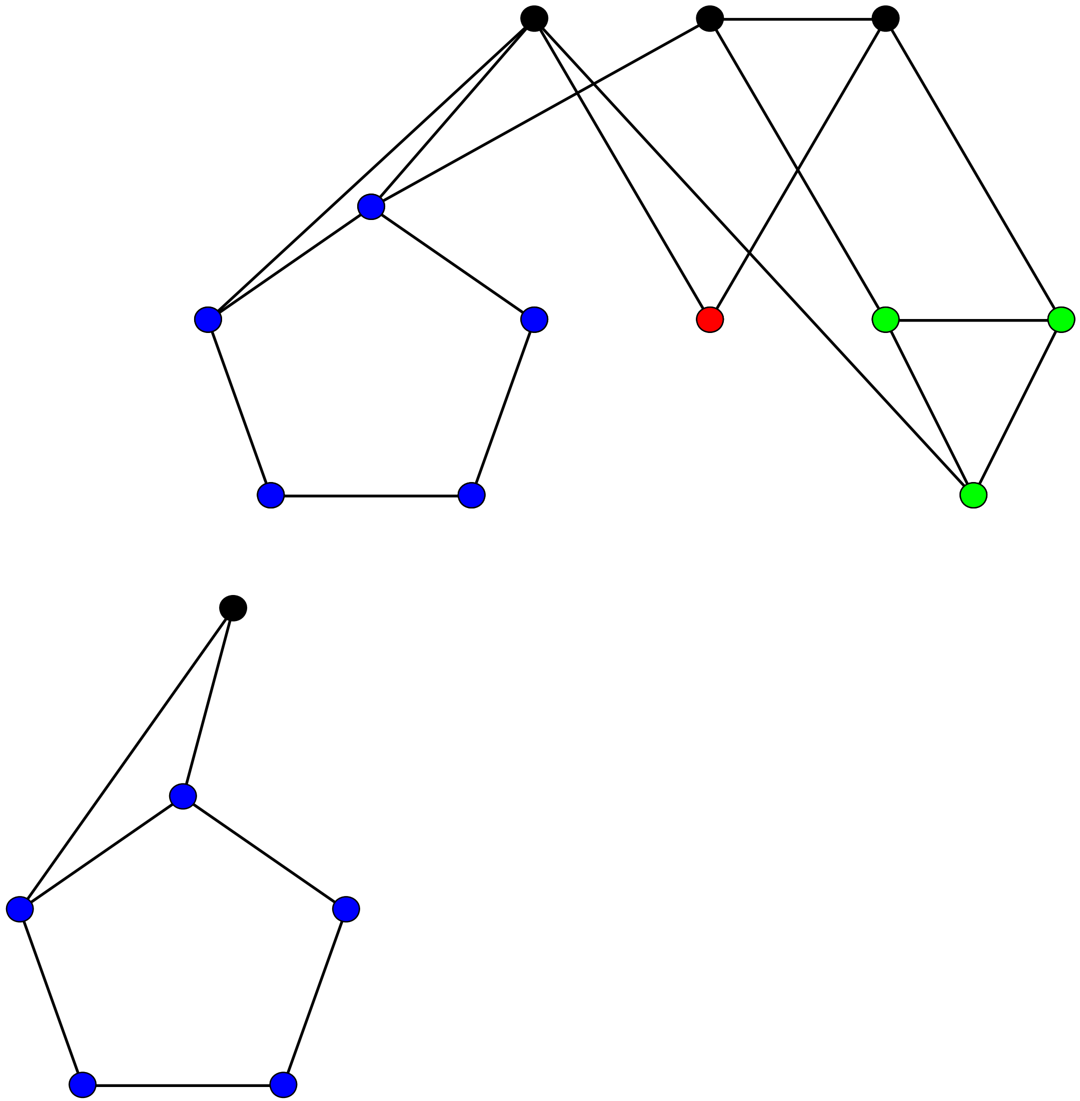
Canonical Decomposition



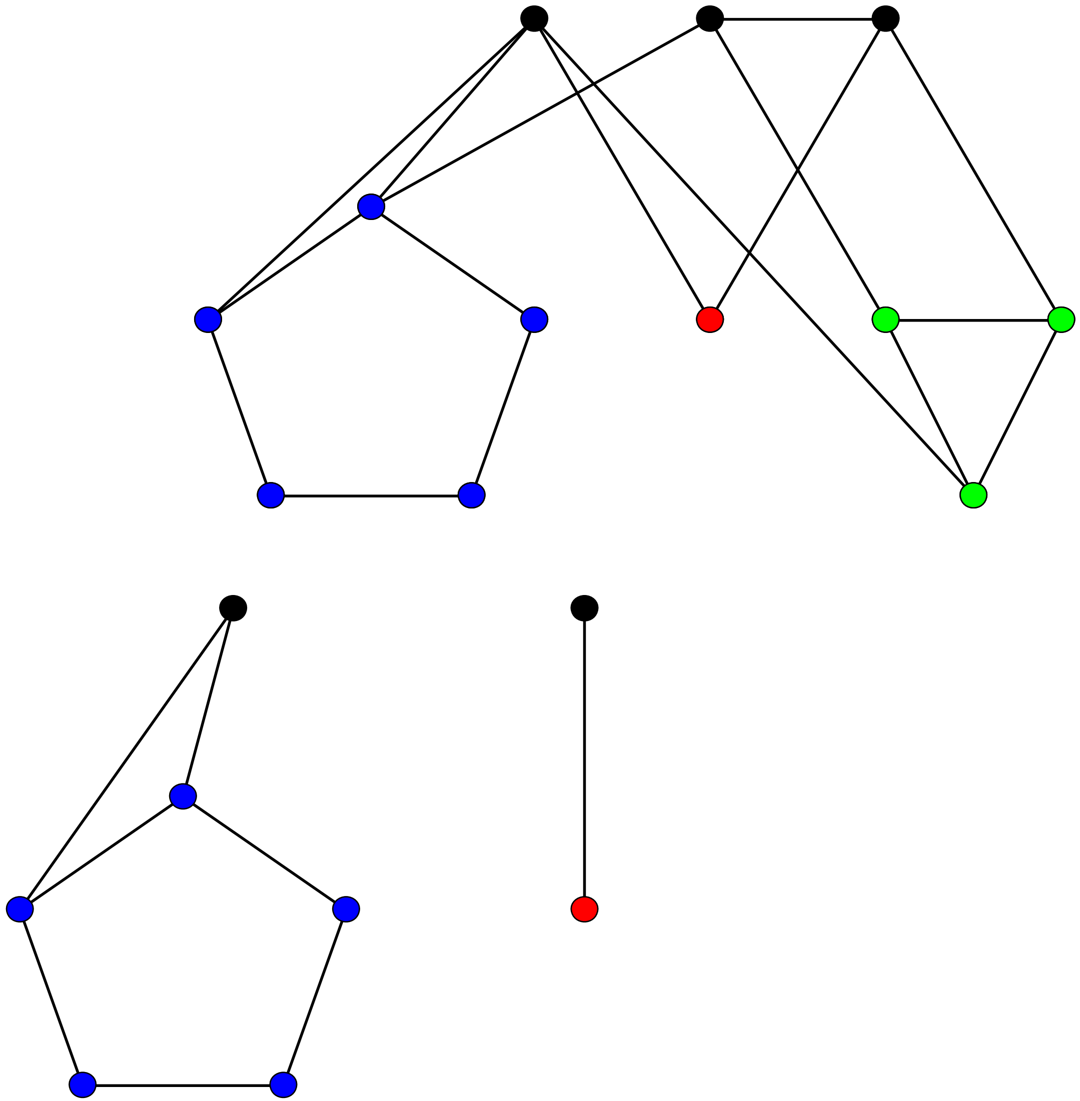
Canonical Decomposition



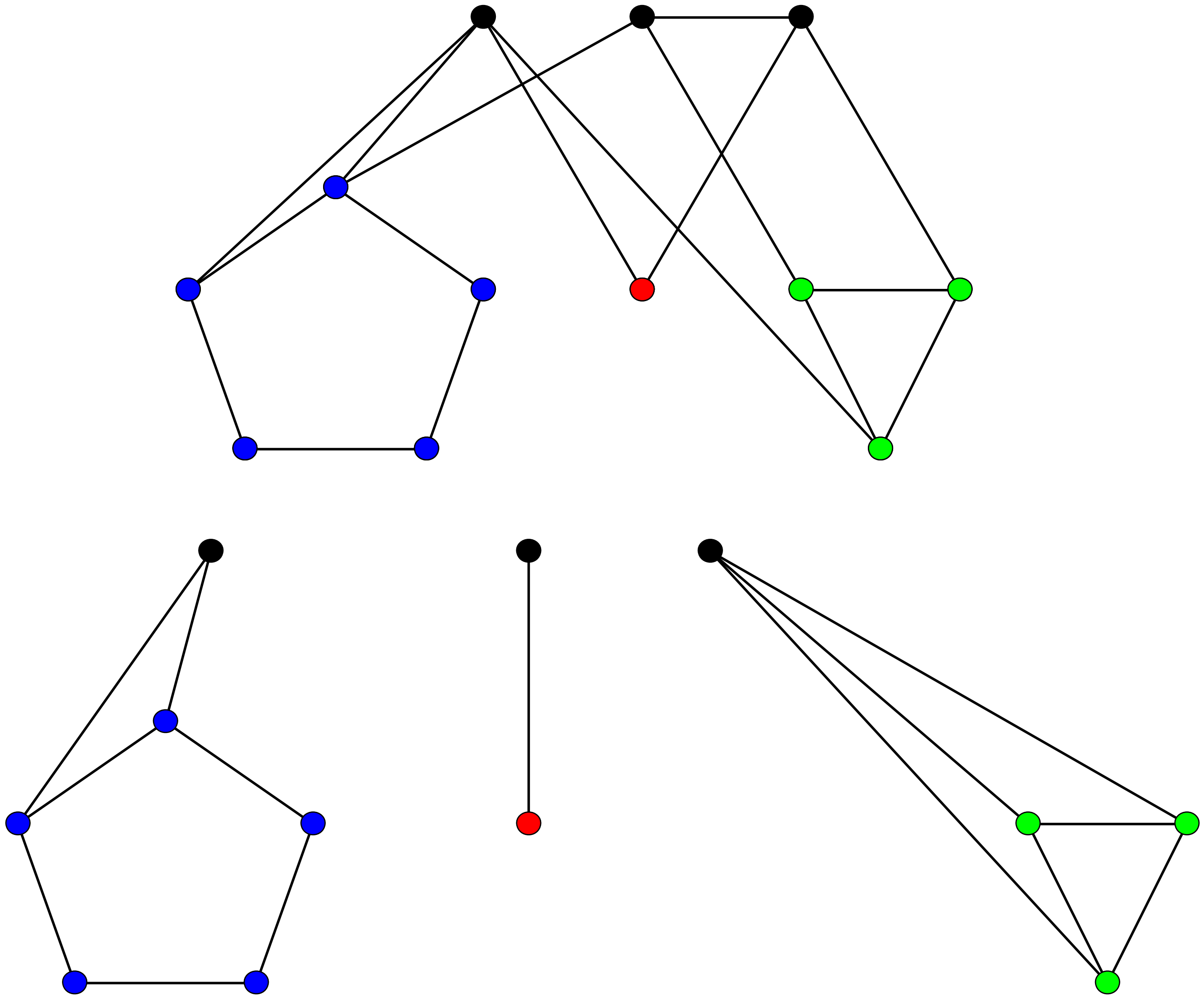
Canonical Decomposition



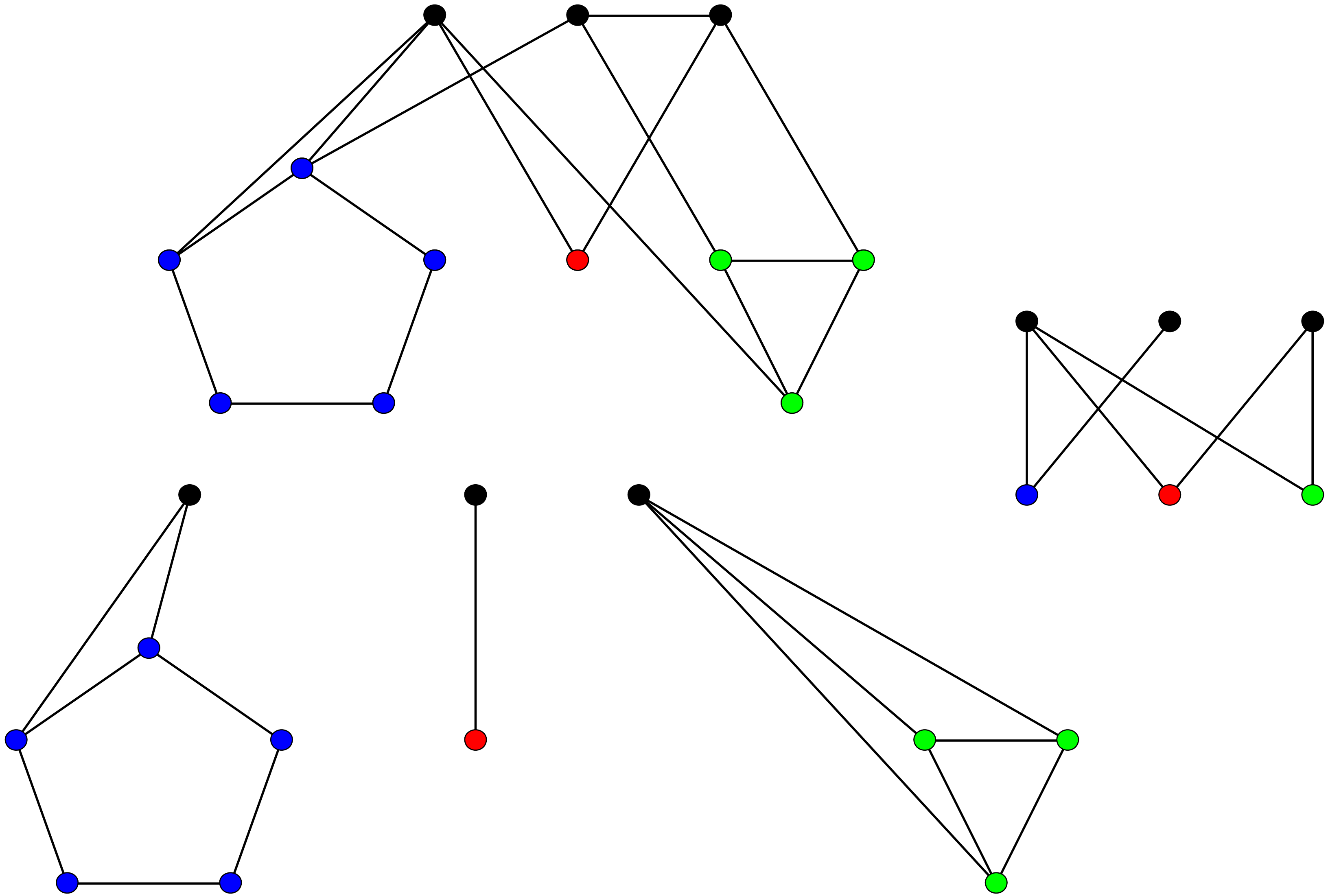
Canonical Decomposition



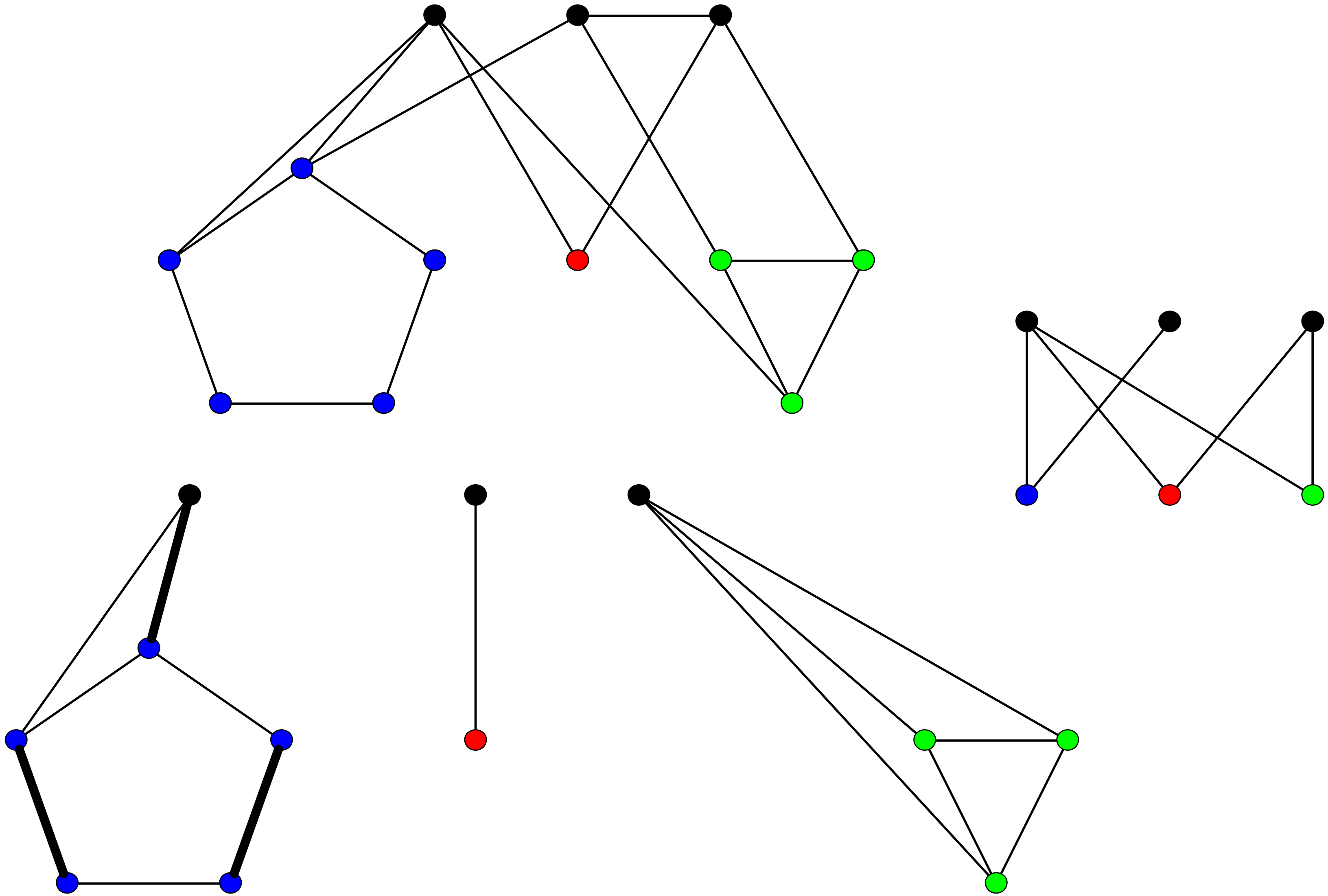
Canonical Decomposition



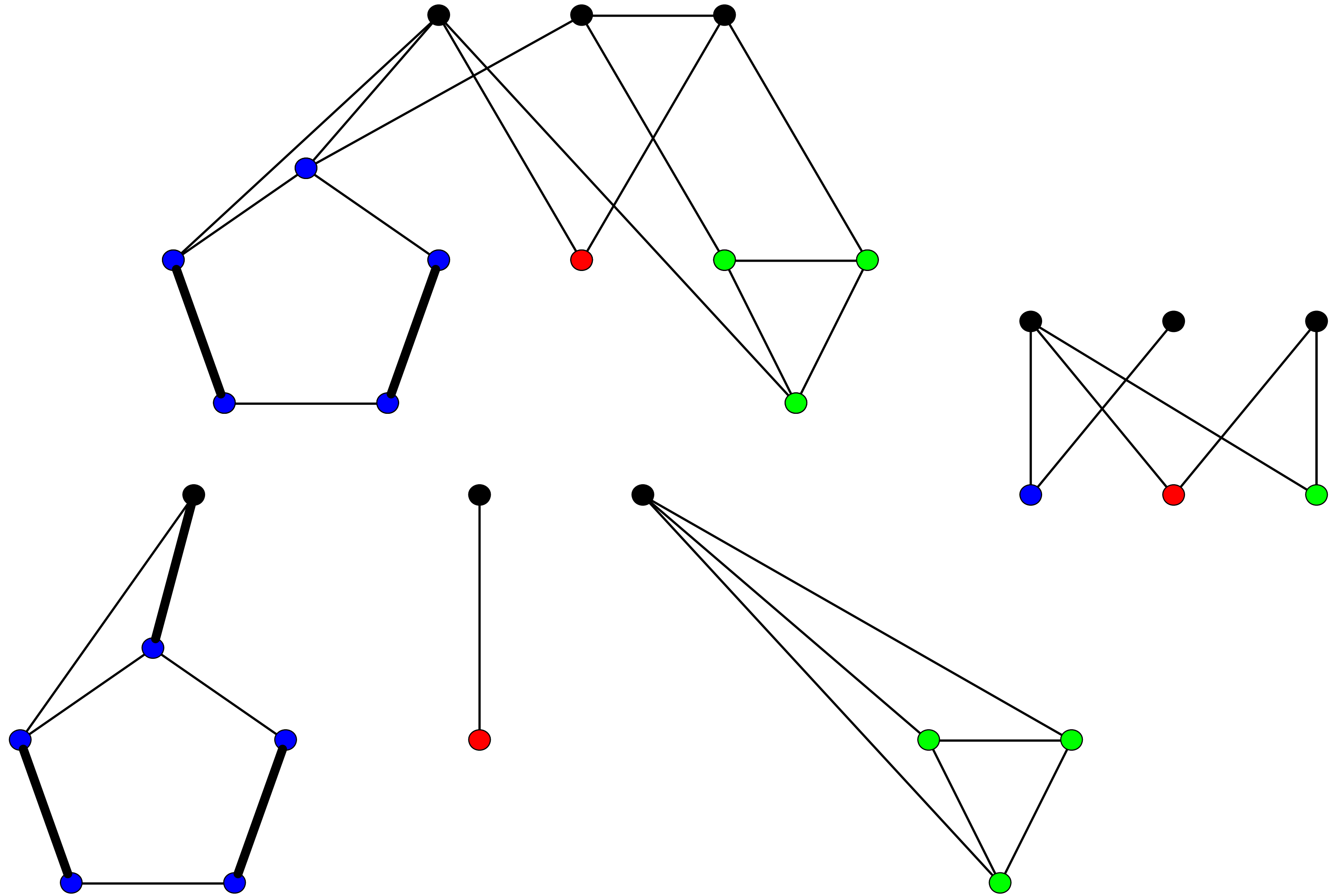
Canonical Decomposition



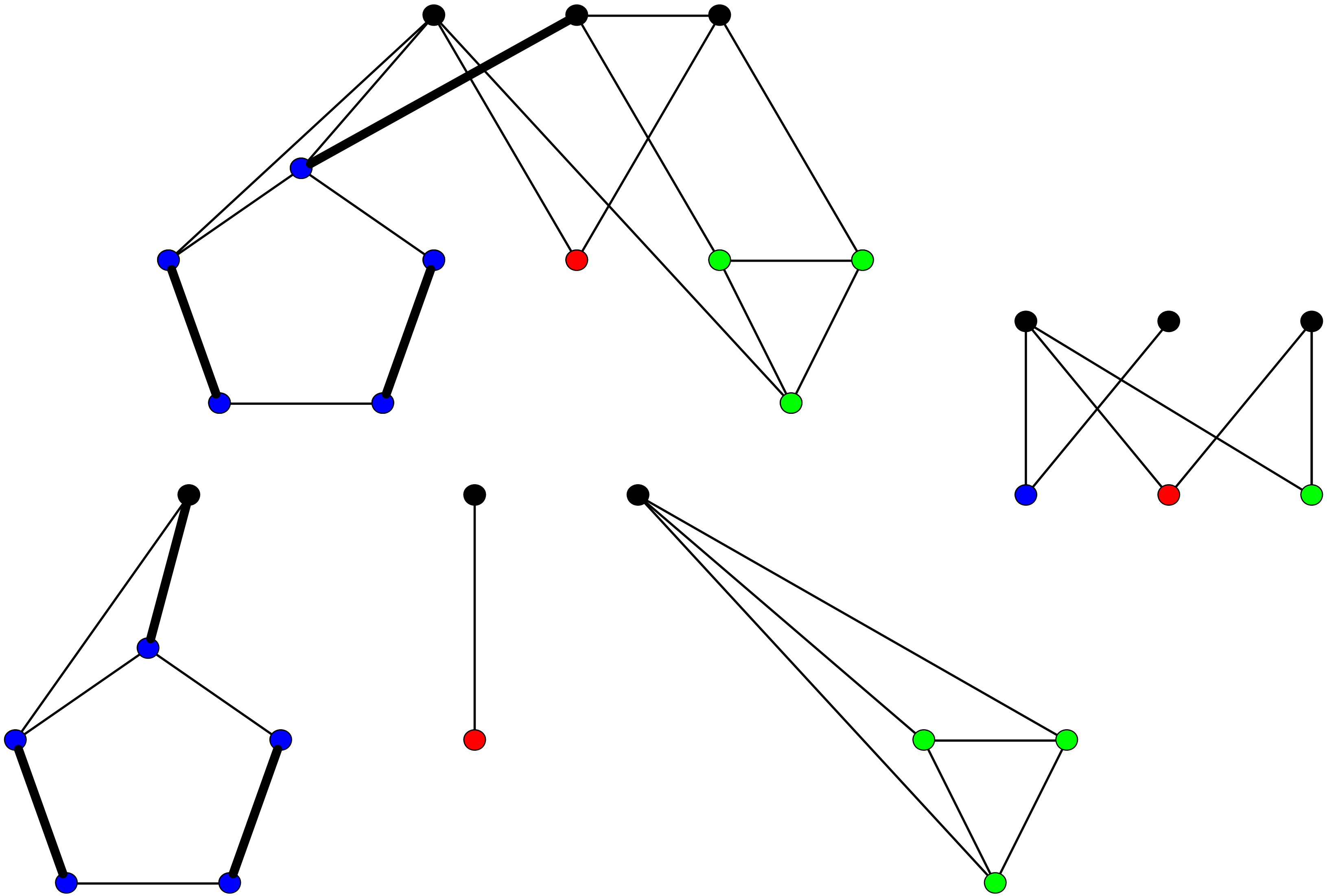
Canonical Decomposition



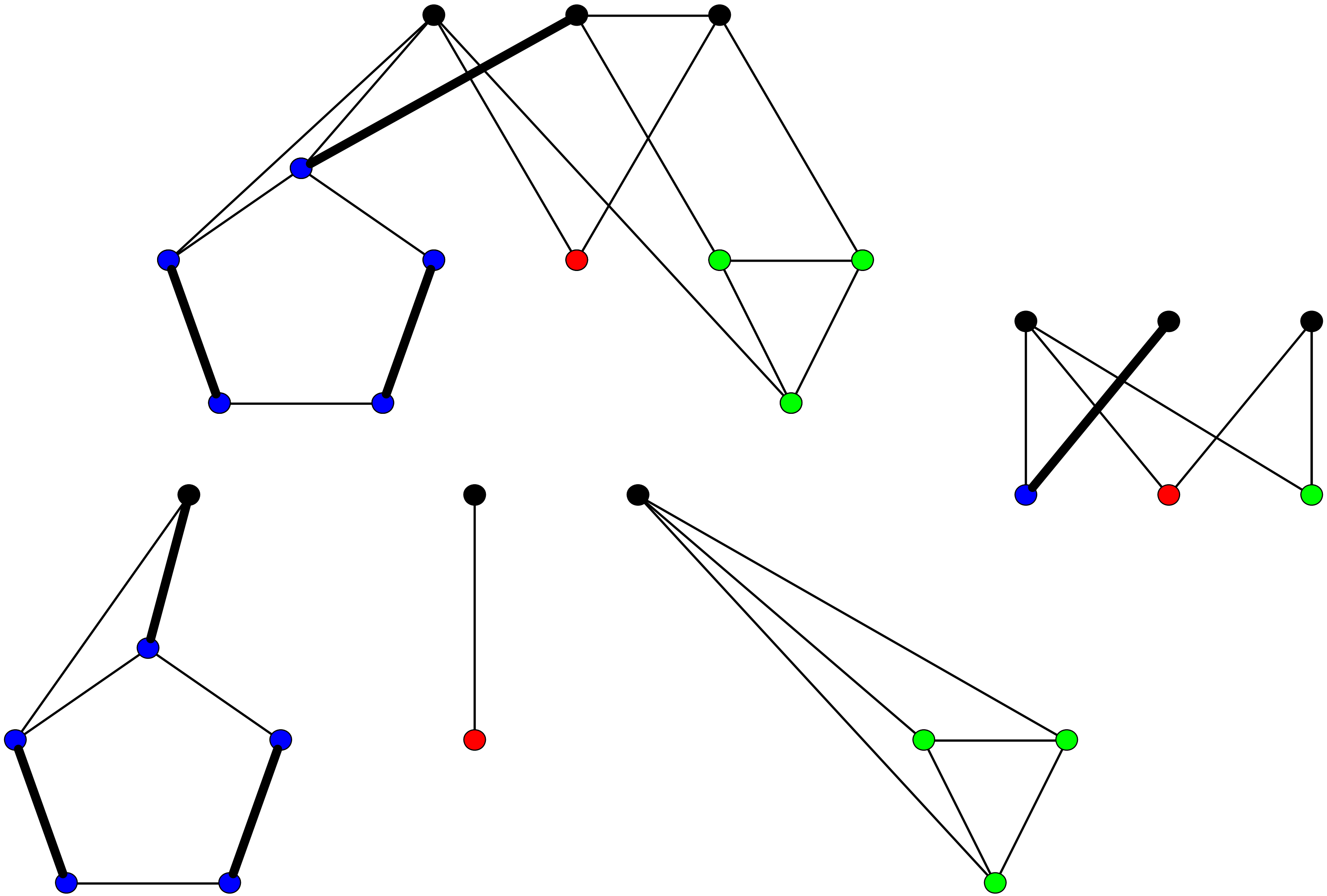
Canonical Decomposition



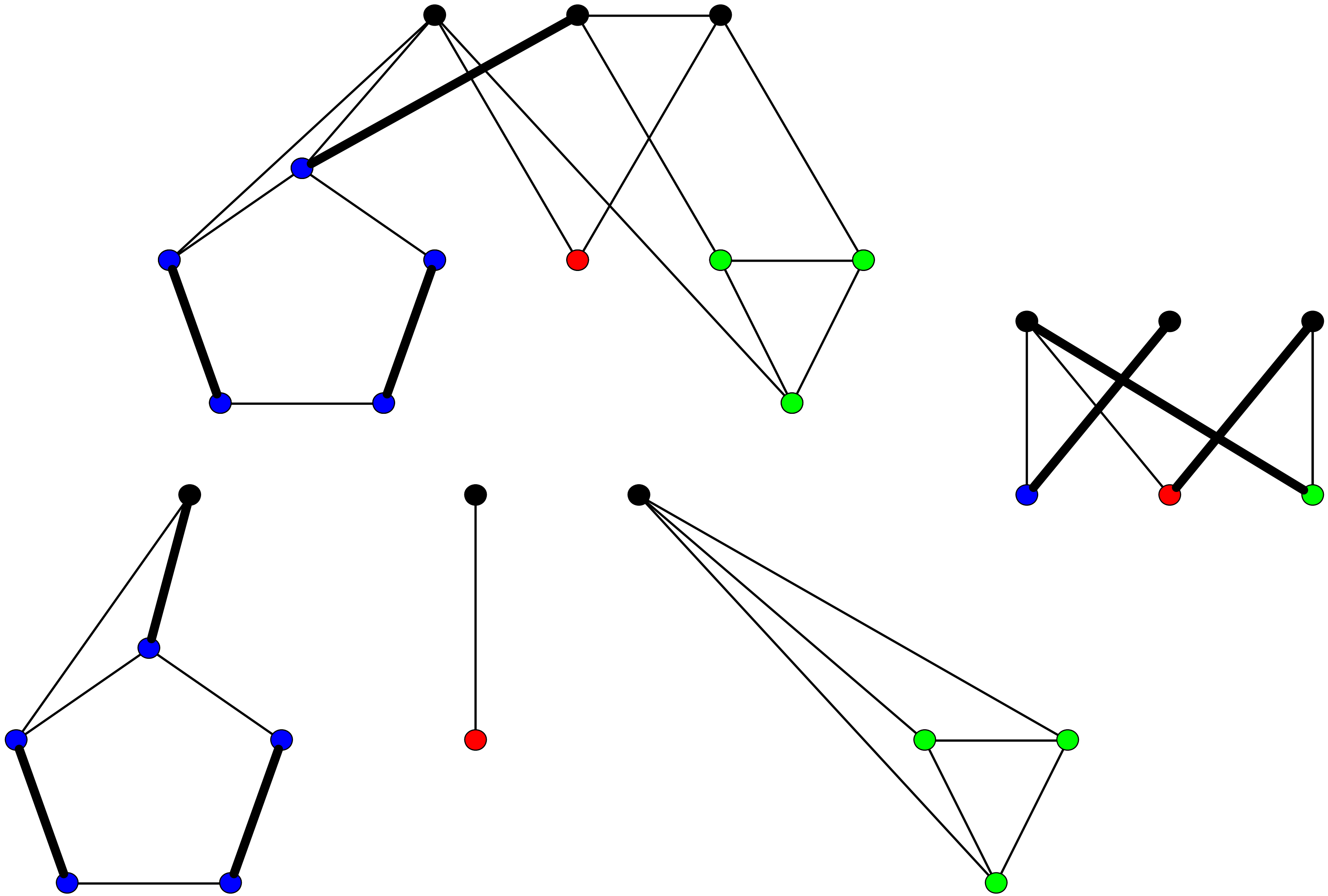
Canonical Decomposition



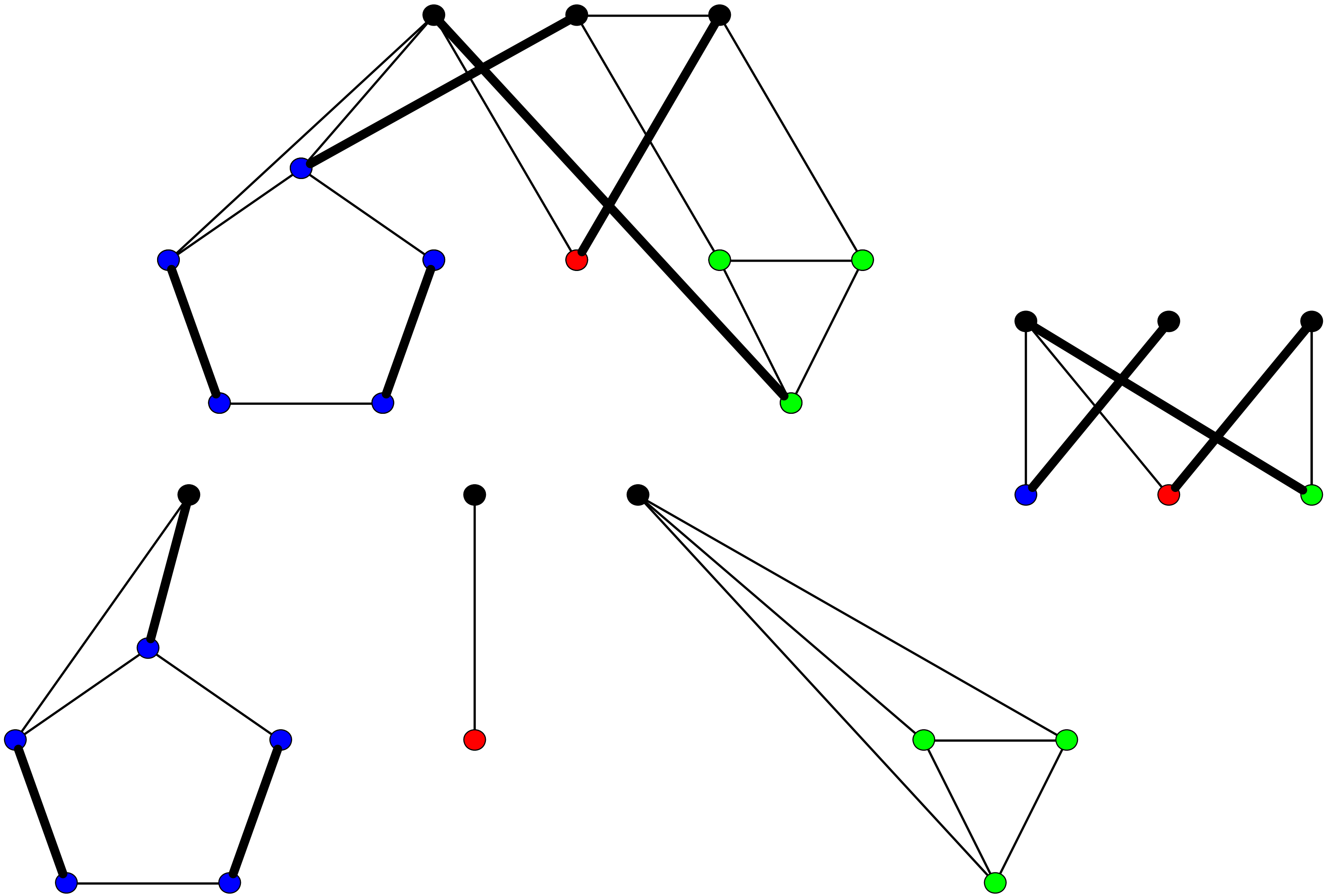
Canonical Decomposition



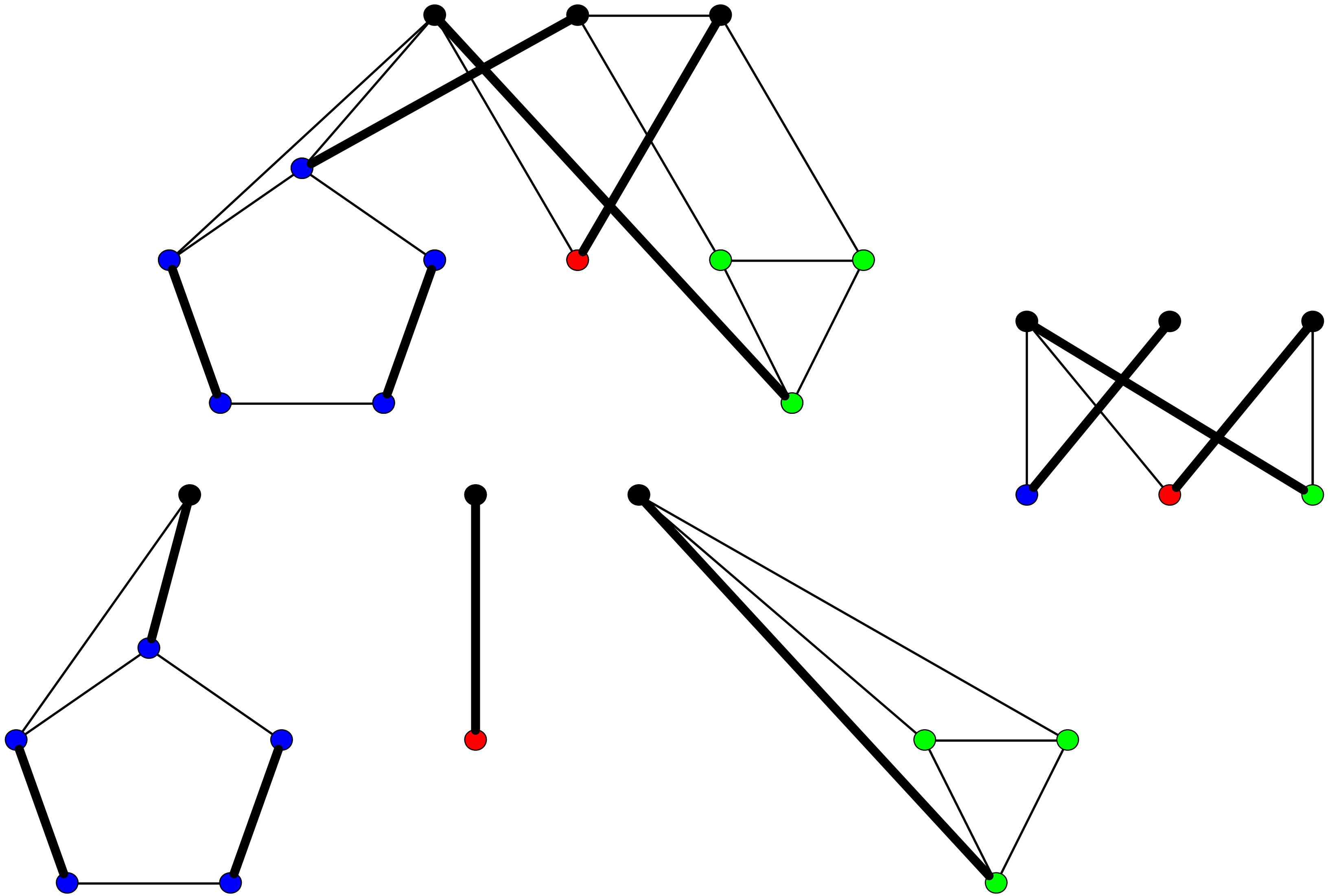
Canonical Decomposition



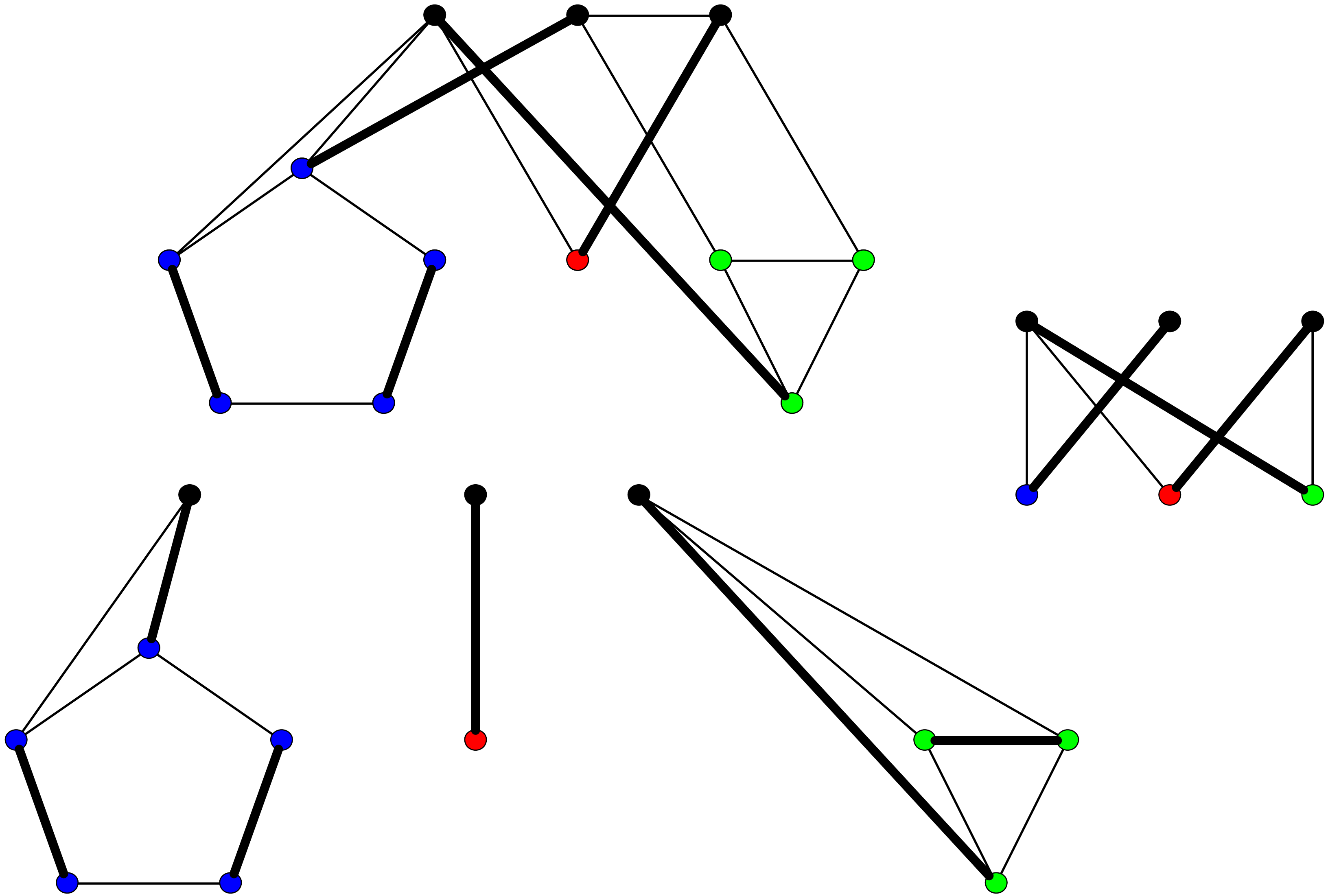
Canonical Decomposition



Canonical Decomposition



Canonical Decomposition



Canonical Decomposition