Approximation Algorithms for the Partial Degree Bounded Edge Packing (PDBEP) Problem

Pawan Aurora

Department of Electrical Engineering and Computer Science
IISER Bhopal

STTP, MANIT Bhopal
October 16, 2016

The Graph Matching Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)

The Graph Matching Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $F \subseteq E$ s.t. $deg(u) \leq 1$ **AND** $deg(v) \leq 1$ for all $\{u, v\} \in F$

The Graph Matching Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $F \subseteq E$ s.t. $deg(u) \leq 1$ **AND** $deg(v) \leq 1$ for all $\{u, v\} \in F$
- Objective is to find a F of maximum cardinality

The Graph Matching Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $F \subseteq E$ s.t. $deg(u) \leq 1$ **AND** $deg(v) \leq 1$ for all $\{u, v\} \in F$
- Objective is to find a F of maximum cardinality
- There exist polynomial time algorithms to find the maximum matching in graph G

The PDBEP Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)

The PDBEP Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $F \subseteq E$ s.t. $deg(u) \leq 1$ **OR** $deg(v) \leq 1$ for all $\{u, v\} \in F$

The PDBEP Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $F \subseteq E$ s.t. $deg(u) \leq 1$ **OR** $deg(v) \leq 1$ for all $\{u, v\} \in F$
- Objective is to find a F of maximum cardinality

The PDBEP Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $F \subseteq E$ s.t. $deg(u) \leq 1$ **OR** $deg(v) \leq 1$ for all $\{u, v\} \in F$
- Objective is to find a F of maximum cardinality
- Introduced by Peng Zhang in 2012

Example

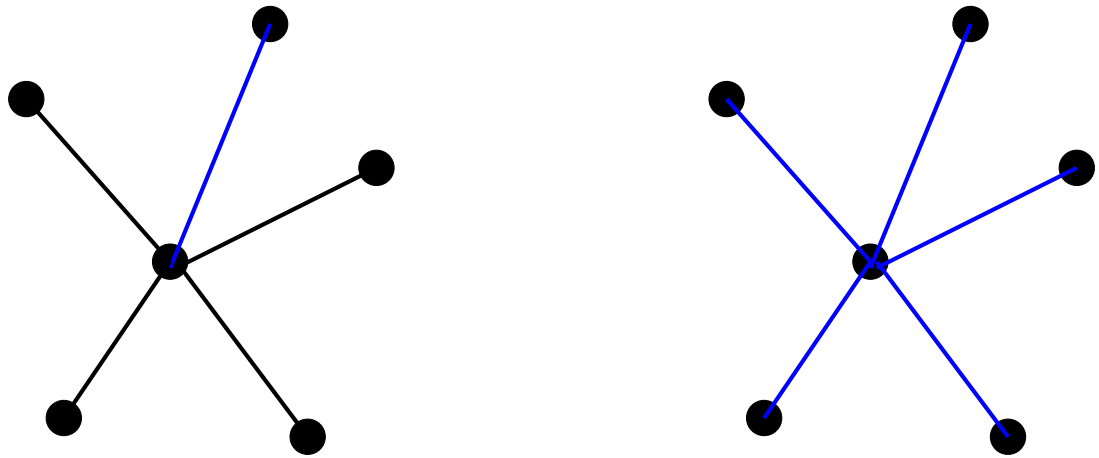


Figure: Matching Vs PDBEP

The PDBEP Problem

- Application in resource allocation

The PDBEP Problem

- Application in resource allocation
- Edges are jobs and vertices are resources

The PDBEP Problem

- Application in resource allocation
- Edges are jobs and vertices are resources
- Each job requires two resources

The PDBEP Problem

- Application in resource allocation
- Edges are jobs and vertices are resources
- Each job requires two resources
- A job can execute only if at least one of the two required resources is not shared with other jobs

The PDBEP Problem

- Application in resource allocation
- Edges are jobs and vertices are resources
- Each job requires two resources
- A job can execute only if at least one of the two required resources is not shared with other jobs
- Objective is to execute maximum number of jobs

The Dominating Set Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)

The Dominating Set Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $D \subseteq V$ s.t. $\forall\, u \in V \setminus D\ \exists\, \{u, v\} \in E$ for some $v \in D$

The Dominating Set Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $D \subseteq V$ s.t. $\forall\, u \in V \setminus D\ \exists\, \{u, v\} \in E$ for some $v \in D$
- Objective is to find a D of minimum cardinality

The Dominating Set Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $D \subseteq V$ s.t. $\forall\, u \in V \setminus D\ \exists\, \{u, v\} \in E$ for some $v \in D$
- Objective is to find a D of minimum cardinality
- A classical NP-hard problem

The Dominating Set Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $D \subseteq V$ s.t. $\forall\, u \in V \setminus D\ \exists\, \{u, v\} \in E$ for some $v \in D$
- Objective is to find a D of minimum cardinality
- A classical NP-hard problem
- Given a solution F to the PDBEP problem we can construct a solution to the Dominating set problem of size $|V| - |F|$

The Dominating Set Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $D \subseteq V$ s.t. $\forall\, u \in V \setminus D\ \exists\, \{u, v\} \in E$ for some $v \in D$
- Objective is to find a D of minimum cardinality
- A classical NP-hard problem
- Given a solution F to the PDBEP problem we can construct a solution to the Dominating set problem of size $|V| - |F|$
  - A solution to the PDBEP problem is a set of node-disjoint stars

The Dominating Set Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $D \subseteq V$ s.t. $\forall\, u \in V \setminus D\ \exists\, \{u, v\} \in E$ for some $v \in D$
- Objective is to find a D of minimum cardinality
- A classical NP-hard problem
- Given a solution F to the PDBEP problem we can construct a solution to the Dominating set problem of size $|V| - |F|$
  - A solution to the PDBEP problem is a set of node-disjoint stars
  - The centers of these stars form a Dominating set

The Dominating Set Problem

- Given a graph $G = (V, E)$ (unweighted, undirected)
- Find $D \subseteq V$ s.t. $\forall\, u \in V \setminus D\ \exists\, \{u, v\} \in E$ for some $v \in D$
- Objective is to find a D of minimum cardinality
- A classical NP-hard problem
- Given a solution F to the PDBEP problem we can construct a solution to the Dominating set problem of size $|V| - |F|$
  - A solution to the PDBEP problem is a set of node-disjoint stars
  - The centers of these stars form a Dominating set
- Hence, PDBEP too is NP-hard [Zhang 2012]

2-Approximation [Zhang 2012]

- Construct a minimal dominating set D

2-Approximation [Zhang 2012]

- Construct a minimal dominating set D
  - For all $u \in V$ add u to D if u is not *dominated* by some $v \in D$

2-Approximation [Zhang 2012]

- Construct a minimal dominating set D
  - For all $u \in V$ add u to D if u is not *dominated* by some $v \in D$
  - If $|D| > |V|/2$ then $D = V \setminus D$

2-Approximation [Zhang 2012]

- Construct a minimal dominating set D
  - For all $u \in V$ add u to D if u is not *dominated* by some $v \in D$
  - If $|D| > |V|/2$ then $D = V \setminus D$
- $|D| \leq |V|/2$

2-Approximation [Zhang 2012]

- Construct a minimal dominating set D
  - For all $u \in V$ add u to D if u is not *dominated* by some $v \in D$
  - If $|D| > |V|/2$ then $D = V \setminus D$
- $|D| \leq |V|/2$
- From D construct a solution F to the PDBEP problem

2-Approximation [Zhang 2012]

- Construct a minimal dominating set D
  - For all $u \in V$ add u to D if u is not *dominated* by some $v \in D$
  - If $|D| > |V|/2$ then $D = V \setminus D$
- $|D| \leq |V|/2$
- From D construct a solution F to the PDBEP problem
  - Let $v_1, \ldots, v_{|D|}$ be an arbitrary ordering of the vertices in D

2-Approximation [Zhang 2012]

- Construct a minimal dominating set D
  - For all $u \in V$ add u to D if u is not *dominated* by some $v \in D$
  - If $|D| > |V|/2$ then $D = V \setminus D$
- $|D| \leq |V|/2$
- From D construct a solution F to the PDBEP problem
  - Let $v_1, \ldots, v_{|D|}$ be an arbitrary ordering of the vertices in D
  - Initialize $F = A = \emptyset$

2-Approximation [Zhang 2012]

- Construct a minimal dominating set D
  - For all $u \in V$ add u to D if u is not *dominated* by some $v \in D$
  - If $|D| > |V|/2$ then $D = V \setminus D$
- $|D| \leq |V|/2$
- From D construct a solution F to the PDBEP problem
  - Let $v_1, \ldots, v_{|D|}$ be an arbitrary ordering of the vertices in D
  - Initialize $F = A = \emptyset$
  - If $e = \{u, v_i\} \in E$ and $u \notin A$, $F = F \cup \{e\}, A = A \cup \{u\}$

2-Approximation [Zhang 2012]

- Construct a minimal dominating set D
  - For all $u \in V$ add u to D if u is not *dominated* by some $v \in D$
  - If $|D| > |V|/2$ then $D = V \setminus D$
- $|D| \leq |V|/2$
- From D construct a solution F to the PDBEP problem
  - Let $v_1, \ldots, v_{|D|}$ be an arbitrary ordering of the vertices in D
  - Initialize $F = A = \emptyset$
  - If $e = \{u, v_i\} \in E$ and $u \notin A$, $F = F \cup \{e\}, A = A \cup \{u\}$
- $|F| = |V| - |D|$

2-Approximation [Zhang 2012]

- Construct a minimal dominating set D
  - For all $u \in V$ add u to D if u is not *dominated* by some $v \in D$
  - If $|D| > |V|/2$ then $D = V \setminus D$
- $|D| \leq |V|/2$
- From D construct a solution F to the PDBEP problem
  - Let $v_1, \ldots, v_{|D|}$ be an arbitrary ordering of the vertices in D
  - Initialize $F = A = \emptyset$
  - If $e = \{u, v_i\} \in E$ and $u \notin A$, $F = F \cup \{e\}, A = A \cup \{u\}$
- $|F| = |V| - |D|$
- Clearly, $|F| \geq |V|/2$

2-Approximation [Zhang 2012]

- Construct a minimal dominating set D
  - For all $u \in V$ add u to D if u is not *dominated* by some $v \in D$
  - If $|D| > |V|/2$ then $D = V \setminus D$
- $|D| \leq |V|/2$
- From D construct a solution F to the PDBEP problem
  - Let $v_1, \ldots, v_{|D|}$ be an arbitrary ordering of the vertices in D
  - Initialize $F = A = \emptyset$
  - If $e = \{u, v_i\} \in E$ and $u \notin A$, $F = F \cup \{e\}, A = A \cup \{u\}$
- $|F| = |V| - |D|$
- Clearly, $|F| \geq |V|/2$
- Since $OPT \leq |V|$, we get $|F| \geq OPT/2$

Generalizations of PDBEP

- Consider the problem for arbitrary degree constraints

Generalizations of PDBEP

- Consider the problem for arbitrary degree constraints
  - $c : V \rightarrow \mathbb{N}$

Generalizations of PDBEP

- Consider the problem for arbitrary degree constraints
  - $c : V \rightarrow \mathbb{N}$
  - Find $F \subseteq E$ s.t. $deg(u) \leq c(u)$ **OR** $deg(v) \leq c(v)$ for all $\{u, v\} \in F$

Generalizations of PDBEP

- Consider the problem for arbitrary degree constraints
  - $c : V \rightarrow \mathbb{N}$
  - Find $F \subseteq E$ s.t. $deg(u) \leq c(u)$ **OR** $deg(v) \leq c(v)$ for all $\{u, v\} \in F$
- Consider the problem for edge-weighted graphs

Generalizations of PDBEP

- Consider the problem for arbitrary degree constraints
  - $c : V \rightarrow \mathbb{N}$
  - Find $F \subseteq E$ s.t. $deg(u) \leq c(u)$ **OR** $deg(v) \leq c(v)$ for all $\{u, v\} \in F$
- Consider the problem for edge-weighted graphs
  - $w : E \rightarrow \mathbb{N}$

Generalizations of PDBEP

- Consider the problem for arbitrary degree constraints
  - $c : V \to \mathbb{N}$
  - Find $F \subseteq E$ s.t. $deg(u) \leq c(u)$ **OR** $deg(v) \leq c(v)$ for all $\{u, v\} \in F$
- Consider the problem for edge-weighted graphs
  - $w : E \to \mathbb{N}$
  - Objective is to maximize the sum of the weights of the edges

Generalizations of PDBEP

- Consider the problem for arbitrary degree constraints
  - $c : V \to \mathbb{N}$
  - Find $F \subseteq E$ s.t. $deg(u) \leq c(u)$ **OR** $deg(v) \leq c(v)$ for all $\{u, v\} \in F$
- Consider the problem for edge-weighted graphs
  - $w : E \to \mathbb{N}$
  - Objective is to maximize the sum of the weights of the edges
- Consider the problem with demands on edges

Generalizations of PDBEP

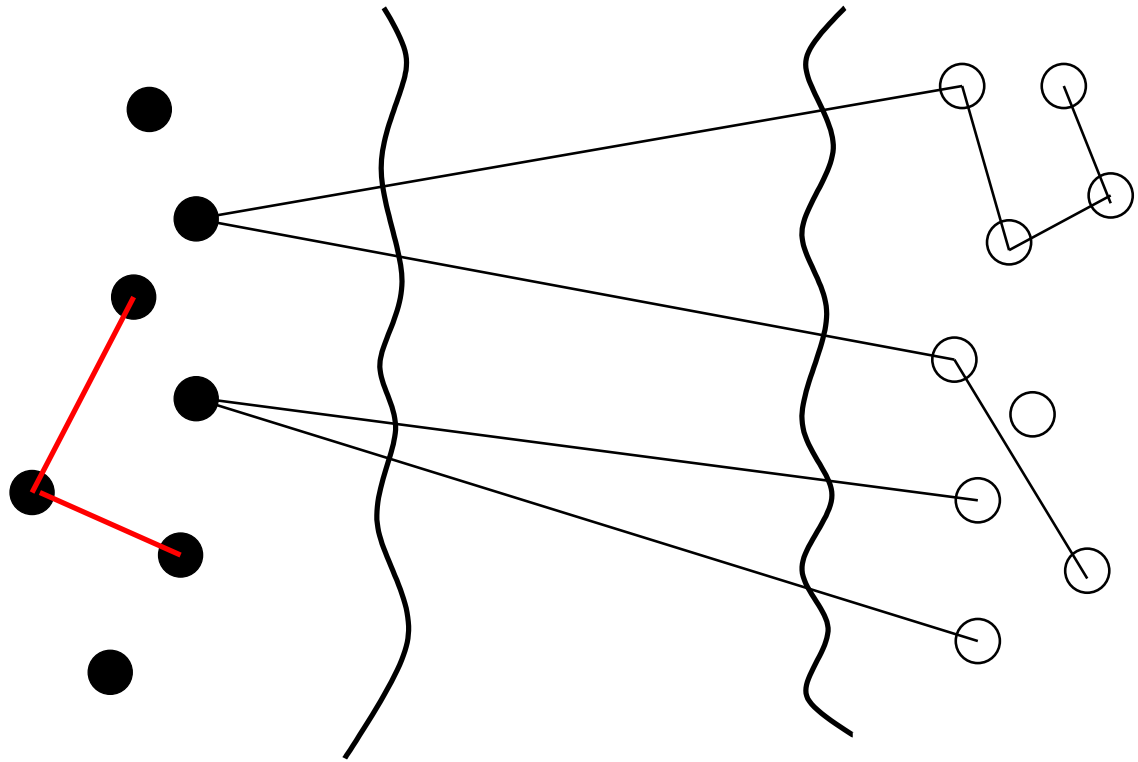
- Consider the problem for arbitrary degree constraints
  - $c : V \to \mathbb{N}$
  - Find $F \subseteq E$ s.t. $deg(u) \leq c(u)$ **OR** $deg(v) \leq c(v)$ for all $\{u, v\} \in F$
- Consider the problem for edge-weighted graphs
  - $w : E \to \mathbb{N}$
  - Objective is to maximize the sum of the weights of the edges
- Consider the problem with demands on edges
  - $d : E \to \mathbb{N}$

Generalizations of PDBEP

- Consider the problem for arbitrary degree constraints
  - $c : V \to \mathbb{N}$
  - Find $F \subseteq E$ s.t. $deg(u) \leq c(u)$ **OR** $deg(v) \leq c(v)$ for all $\{u, v\} \in F$
- Consider the problem for edge-weighted graphs
  - $w : E \to \mathbb{N}$
  - Objective is to maximize the sum of the weights of the edges
- Consider the problem with demands on edges
  - $d : E \to \mathbb{N}$
  - Find $F \subseteq E$ s.t. $\sum_{e=\{w,u\} \in E} d(e) \leq c(u)$ **OR** $\sum_{e=\{w,v\} \in E} d(e) \leq c(v)$ for all $\{u, v\} \in F$

$c : V \rightarrow \mathbb{N}$ $d, w : E \rightarrow 1$ [A.,Singh,Mehta]

Upper Bound on OPT



$OPT \leq \sum_{w \in W} c(w) \leq \sum_{v \in V} c(v)$, $W = \{v \in V | deg(v) \leq c(v)\}$

$c : V \rightarrow \mathbb{N}$ $d, w : E \rightarrow 1$

Algorithm

- Iteratively delete violating edges

$c : V \rightarrow \mathbb{N}$ $d, w : E \rightarrow 1$

Algorithm

- Iteratively delete violating edges
- Output the set Y of surviving edges

$c : V \rightarrow \mathbb{N}$ $d, w : E \rightarrow 1$

Algorithm

- Iteratively delete violating edges
- Output the set Y of surviving edges

Analysis

$c : V \rightarrow \mathbb{N}$ $d, w : E \rightarrow 1$

Algorithm

- Iteratively delete violating edges
- Output the set Y of surviving edges

Analysis

- Observe that $deg_Y(v) \geq c(v)\ \forall\ v \in V$

$c : V \rightarrow \mathbb{N}$ $d, w : E \rightarrow 1$

Algorithm

- Iteratively delete violating edges
- Output the set Y of surviving edges

Analysis

- Observe that $deg_Y(v) \geq c(v)\ \forall\ v \in V$
- Hence $|Y| = \sum_v deg_Y(v)/2 \geq \sum_v c(v)/2 \geq OPT/2$

$c : V \rightarrow \mathbb{N}$ $d, w : E \rightarrow 1$

Algorithm

- Iteratively delete violating edges
- Output the set Y of surviving edges

Analysis

- Observe that $deg_Y(v) \geq c(v)\ \forall\ v \in V$
- Hence $|Y| = \sum_v deg_Y(v)/2 \geq \sum_v c(v)/2 \geq OPT/2$
- We have a 2-approximation

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$ [A.,Jena,Raman]

- Let $N_{max}(v)$ denote the set of edges of maximum weight incident on vertex v

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$ [A.,Jena,Raman]

- Let $N_{max}(v)$ denote the set of edges of maximum weight incident on vertex v
- Construct a set $E_{max} \subseteq E$ as follows:

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$ [A.,Jena,Raman]

- Let $N_{max}(v)$ denote the set of edges of maximum weight incident on vertex v
- Construct a set $E_{max} \subseteq E$ as follows:
  - Initialize $E^1_{max} = E^2_{max} = \emptyset$

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$ [A.,Jena,Raman]

- Let $N_{max}(v)$ denote the set of edges of maximum weight incident on vertex v
- Construct a set $E_{max} \subseteq E$ as follows:
  - Initialize $E^1_{max} = E^2_{max} = \emptyset$
  - Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$ [A.,Jena,Raman]

- Let $N_{max}(v)$ denote the set of edges of maximum weight incident on vertex v
- Construct a set $E_{max} \subseteq E$ as follows:
  - Initialize $E^1_{max} = E^2_{max} = \emptyset$
  - Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices
  - For $i = 1, \ldots, n$, if no edge from $N_{max}(v_i)$ has been chosen yet

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$ [A.,Jena,Raman]

- Let $N_{max}(v)$ denote the set of edges of maximum weight incident on vertex v
- Construct a set $E_{max} \subseteq E$ as follows:
  - Initialize $E^1_{max} = E^2_{max} = \emptyset$
  - Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices
  - For $i = 1, \ldots, n$, if no edge from $N_{max}(v_i)$ has been chosen yet
    - Pick an arbitrary edge $e = \{v_i, v_j\}$ from $N_{max}(v_i)$

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$ [A.,Jena,Raman]

- Let $N_{max}(v)$ denote the set of edges of maximum weight incident on vertex v
- Construct a set $E_{max} \subseteq E$ as follows:
  - Initialize $E^1_{max} = E^2_{max} = \emptyset$
  - Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices
  - For $i = 1, \ldots, n$, if no edge from $N_{max}(v_i)$ has been chosen yet
    - Pick an arbitrary edge $e = \{v_i, v_j\}$ from $N_{max}(v_i)$
    - If $e \in N_{max}(v_j)$ and no edge from $N_{max}(v_j)$ has been chosen yet add e to E^2_{max}

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$ [A.,Jena,Raman]

- Let $N_{max}(v)$ denote the set of edges of maximum weight incident on vertex v
- Construct a set $E_{max} \subseteq E$ as follows:
  - Initialize $E^1_{max} = E^2_{max} = \emptyset$
  - Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices
  - For $i = 1, \ldots, n$, if no edge from $N_{max}(v_i)$ has been chosen yet
    - Pick an arbitrary edge $e = \{v_i, v_j\}$ from $N_{max}(v_i)$
    - If $e \in N_{max}(v_j)$ and no edge from $N_{max}(v_j)$ has been chosen yet add e to E^2_{max}
    - Else, add e to E^1_{max}

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$ [A.,Jena,Raman]

- Let $N_{max}(v)$ denote the set of edges of maximum weight incident on vertex v
- Construct a set $E_{max} \subseteq E$ as follows:
  - Initialize $E^1_{max} = E^2_{max} = \emptyset$
  - Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices
  - For $i = 1, \ldots, n$, if no edge from $N_{max}(v_i)$ has been chosen yet
    - Pick an arbitrary edge $e = \{v_i, v_j\}$ from $N_{max}(v_i)$
    - If $e \in N_{max}(v_j)$ and no edge from $N_{max}(v_j)$ has been chosen yet add e to E^2_{max}
    - Else, add e to E^1_{max}
  - $E_{max} = E^1_{max} \cup E^2_{max}$

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- The subgraph $G' = (V, E_{max})$ is acyclic

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- The subgraph $G' = (V, E_{max})$ is acyclic
  - Let $u_0, u_1, \ldots, u_k, u_0$ be a cycle in G'

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- The subgraph $G' = (V, E_{max})$ is acyclic
  - Let $u_0, u_1, \ldots, u_k, u_0$ be a cycle in G'
  - W.l.o.g. assume that the vertices are visited in the same order i.e., $u_0, u_1, \ldots, u_k$

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- The subgraph $G' = (V, E_{max})$ is acyclic
  - Let $u_0, u_1, \ldots, u_k, u_0$ be a cycle in G'
  - W.l.o.g. assume that the vertices are visited in the same order i.e., $u_0, u_1, \ldots, u_k$
  - All edges in the cycle must have equal weight

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- The subgraph $G' = (V, E_{max})$ is acyclic
  - Let $u_0, u_1, \ldots, u_k, u_0$ be a cycle in G'
  - W.l.o.g. assume that the vertices are visited in the same order i.e., $u_0, u_1, \ldots, u_k$
  - All edges in the cycle must have equal weight
  - Edge $\{u_{i-1}, u_i\} \in N_{max}(u_{i-1})$ as well as $\{u_{i-1}, u_i\} \in N_{max}(u_i)$

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- The subgraph $G' = (V, E_{max})$ is acyclic
  - Let $u_0, u_1, \ldots, u_k, u_0$ be a cycle in G'
  - W.l.o.g. assume that the vertices are visited in the same order i.e., $u_0, u_1, \ldots, u_k$
  - All edges in the cycle must have equal weight
  - Edge $\{u_{i-1}, u_i\} \in N_{max}(u_{i-1})$ as well as $\{u_{i-1}, u_i\} \in N_{max}(u_i)$
  - Edge $\{u_i, u_{i+1}\}$ will never get added to E_{max}

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- The subgraph $G' = (V, E_{max})$ is acyclic
  - Let $u_0, u_1, \ldots, u_k, u_0$ be a cycle in G'
  - W.l.o.g. assume that the vertices are visited in the same order i.e., $u_0, u_1, \ldots, u_k$
  - All edges in the cycle must have equal weight
  - Edge $\{u_{i-1}, u_i\} \in N_{max}(u_{i-1})$ as well as $\{u_{i-1}, u_i\} \in N_{max}(u_i)$
  - Edge $\{u_i, u_{i+1}\}$ will never get added to E_{max}
  - A contradiction

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- The subgraph $G' = (V, E_{max})$ is acyclic
  - Let $u_0, u_1, \ldots, u_k, u_0$ be a cycle in G'
  - W.l.o.g. assume that the vertices are visited in the same order i.e., $u_0, u_1, \ldots, u_k$
  - All edges in the cycle must have equal weight
  - Edge $\{u_{i-1}, u_i\} \in N_{max}(u_{i-1})$ as well as $\{u_{i-1}, u_i\} \in N_{max}(u_i)$
  - Edge $\{u_i, u_{i+1}\}$ will never get added to E_{max}
  - A contradiction
- G' is a forest

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

Upper Bound on OPT

- Let $e_{max}(v)$ denote an edge in $N_{max}(v)$

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

Upper Bound on OPT

- Let $e_{max}(v)$ denote an edge in $N_{max}(v)$
- Clearly, $OPT \leq \sum_{v \in V} w(e_{max}(v))$

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

Upper Bound on OPT

- Let $e_{max}(v)$ denote an edge in $N_{max}(v)$
- Clearly, $OPT \leq \sum_{v \in V} w(e_{max}(v))$
- Note that any edge $e = \{u, v\}$ added to E^2_{max} belongs to both $N_{max}(u)$ and $N_{max}(v)$

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

Upper Bound on OPT

- Let $e_{max}(v)$ denote an edge in $N_{max}(v)$
- Clearly, $OPT \leq \sum_{v \in V} w(e_{max}(v))$
- Note that any edge $e = \{u, v\}$ added to E^2_{max} belongs to both $N_{max}(u)$ and $N_{max}(v)$
- Hence, $\sum_{v \in V} w(e_{max}(v)) = \sum_{e \in E^1_{max}} w(e) + 2 \sum_{e \in E^2_{max}} w(e)$

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

Upper Bound on OPT

- Let $e_{max}(v)$ denote an edge in $N_{max}(v)$
- Clearly, $OPT \leq \sum_{v \in V} w(e_{max}(v))$
- Note that any edge $e = \{u, v\}$ added to E^2_{max} belongs to both $N_{max}(u)$ and $N_{max}(v)$
- Hence, $\sum_{v \in V} w(e_{max}(v)) = \sum_{e \in E^1_{max}} w(e) + 2 \sum_{e \in E^2_{max}} w(e)$
- That gives, $OPT \leq w(E^1_{max}) + 2w(E^2_{max})$

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- Label the vertices in G' with distance from the root

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- Label the vertices in G' with distance from the root
- Partition V into two sets X and Y where X consists of odd-labeled vertices, and Y consists of even-labeled vertices

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- Label the vertices in G' with distance from the root
- Partition V into two sets X and Y where X consists of odd-labeled vertices, and Y consists of even-labeled vertices
- The set of edges between X and Y consist of all the edges in E_{max}

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- Label the vertices in G' with distance from the root
- Partition V into two sets X and Y where X consists of odd-labeled vertices, and Y consists of even-labeled vertices
- The set of edges between X and Y consist of all the edges in E_{max}
- Orient an edge $e = \{u, v\}$ in E^1_{max} towards u if e was added to E^1_{max} when processing u, else orient it towards v

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- Label the vertices in G' with distance from the root
- Partition V into two sets X and Y where X consists of odd-labeled vertices, and Y consists of even-labeled vertices
- The set of edges between X and Y consist of all the edges in E_{max}
- Orient an edge $e = \{u, v\}$ in E^1_{max} towards u if e was added to E^1_{max} when processing u, else orient it towards v
- Orient the edges in E^2_{max} towards both the end-points

$c : V \rightarrow 1$ $w : E \rightarrow \mathbb{N}$

- Label the vertices in G' with distance from the root
- Partition V into two sets X and Y where X consists of odd-labeled vertices, and Y consists of even-labeled vertices
- The set of edges between X and Y consist of all the edges in E_{max}
- Orient an edge $e = \{u, v\}$ in E^1_{max} towards u if e was added to E^1_{max} when processing u, else orient it towards v
- Orient the edges in E^2_{max} towards both the end-points
- Return $\max\{\overrightarrow{\delta}(X), \overrightarrow{\delta}(Y)\} \geq (w(E^1_{\max}) + 2w(E^2_{\max}))/2 \geq OPT/2$

$c : V \to \mathbb{N}$ $w, d : E \to \mathbb{N}$ [A.,Jena,Raman]

The Knapsack Problem

- Given a knapsack of size $B \in \mathbb{Z}^+$ and a set $S = \{a_1, \ldots, a_n\}$ of objects

$c : V \to \mathbb{N}$ $w, d : E \to \mathbb{N}$ [A.,Jena,Raman]

The Knapsack Problem

- Given a knapsack of size $B \in \mathbb{Z}^+$ and a set $S = \{a_1, \ldots, a_n\}$ of objects
- Each object has a size $s(a_i) \in \mathbb{Z}^+$ and a profit $p(a_i) \in \mathbb{Z}^+$

$c : V \to \mathbb{N}$ $w, d : E \to \mathbb{N}$ [A.,Jena,Raman]

The Knapsack Problem

- Given a knapsack of size $B \in \mathbb{Z}^+$ and a set $S = \{a_1, \ldots, a_n\}$ of objects
- Each object has a size $s(a_i) \in \mathbb{Z}^+$ and a profit $p(a_i) \in \mathbb{Z}^+$
- The goal is to find the optimal subset of objects whose total size is bounded by B and has the maximum possible total profit

$c : V \to \mathbb{N}$ $w, d : E \to \mathbb{N}$ [A.,Jena,Raman]

The Knapsack Problem

- Given a knapsack of size $B \in \mathbb{Z}^+$ and a set $S = \{a_1, \ldots, a_n\}$ of objects
- Each object has a size $s(a_i) \in \mathbb{Z}^+$ and a profit $p(a_i) \in \mathbb{Z}^+$
- The goal is to find the optimal subset of objects whose total size is bounded by B and has the maximum possible total profit
- It is a famous NP-hard problem

$c : V \to \mathbb{N}$ $w, d : E \to \mathbb{N}$ [A.,Jena,Raman]

The Knapsack Problem

- Given a knapsack of size $B \in \mathbb{Z}^+$ and a set $S = \{a_1, \ldots, a_n\}$ of objects
- Each object has a size $s(a_i) \in \mathbb{Z}^+$ and a profit $p(a_i) \in \mathbb{Z}^+$
- The goal is to find the optimal subset of objects whose total size is bounded by B and has the maximum possible total profit
- It is a famous NP-hard problem
- There exists an algorithm that can find a solution that is at least $(1 - \epsilon)OPT$ in time polynomial in n and $1/\epsilon$

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- We solve a Knapsack problem independently at every vertex

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- We solve a Knapsack problem independently at every vertex
- The set S is the set of edges incident on a vertex v

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- We solve a Knapsack problem independently at every vertex
- The set S is the set of edges incident on a vertex v
- The size of the knapsack is $c(v)$

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- We solve a Knapsack problem independently at every vertex
- The set S is the set of edges incident on a vertex v
- The size of the knapsack is $c(v)$
- The size of an object corresponds to the demand $d(e)$ for an edge e incident on v

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- We solve a Knapsack problem independently at every vertex
- The set S is the set of edges incident on a vertex v
- The size of the knapsack is $c(v)$
- The size of an object corresponds to the demand $d(e)$ for an edge e incident on v
- Profit in this case is the weight of an edge $w(e)$ incident on v

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- We solve a Knapsack problem independently at every vertex
- The set S is the set of edges incident on a vertex v
- The size of the knapsack is $c(v)$
- The size of an object corresponds to the demand $d(e)$ for an edge e incident on v
- Profit in this case is the weight of an edge $w(e)$ incident on v
- So we want to maximize the total weight of the edges that can be packed such that the total demand does not exceed the capacity

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices
- Let $\mathcal{A}_i$ be the solution to the Knapsack problem at vertex v_i

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices
- Let $\mathcal{A}_i$ be the solution to the Knapsack problem at vertex v_i
- Orient the edges in $\mathcal{A}_i$ towards v_i

$c : V \to \mathbb{N}$ $w, d : E \to \mathbb{N}$

- Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices
- Let $\mathcal{A}_i$ be the solution to the Knapsack problem at vertex v_i
- Orient the edges in $\mathcal{A}_i$ towards v_i
- We get a directed multi-graph (V, E')

$c : V \to \mathbb{N}$ $w, d : E \to \mathbb{N}$

- Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices
- Let $\mathcal{A}_i$ be the solution to the Knapsack problem at vertex v_i
- Orient the edges in $\mathcal{A}_i$ towards v_i
- We get a directed multi-graph (V, E')
- Clearly, $\overrightarrow{\delta}(X)$ for any $X \subseteq V$ forms a feasible solution to the PDBEP problem

$c : V \to \mathbb{N}$ $w, d : E \to \mathbb{N}$

- Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices
- Let $\mathcal{A}_i$ be the solution to the Knapsack problem at vertex v_i
- Orient the edges in $\mathcal{A}_i$ towards v_i
- We get a directed multi-graph (V, E')
- Clearly, $\overrightarrow{\delta}(X)$ for any $X \subseteq V$ forms a feasible solution to the PDBEP problem
- Using a simple randomized algorithm we can compute a DICUT of expected weight $w(E')/4$

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Let OPT be the value of an optimal solution to our problem

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Let OPT be the value of an optimal solution to our problem
- Let $F \subseteq E$ be the edges in the optimal solution

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Let OPT be the value of an optimal solution to our problem
- Let $F \subseteq E$ be the edges in the optimal solution
- Let OPT_i be the total weight of the edges in F that are incident on v_i if the sum total of the demands on these edges does not exceed the capacity at v_i

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Let OPT be the value of an optimal solution to our problem
- Let $F \subseteq E$ be the edges in the optimal solution
- Let OPT_i be the total weight of the edges in F that are incident on v_i if the sum total of the demands on these edges does not exceed the capacity at v_i
- Else, $OPT_i = 0$

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Let OPT be the value of an optimal solution to our problem
- Let $F \subseteq E$ be the edges in the optimal solution
- Let OPT_i be the total weight of the edges in F that are incident on v_i if the sum total of the demands on these edges does not exceed the capacity at v_i
- Else, $OPT_i = 0$
- We have, $\sum_{i=1}^{n} OPT_i \geq OPT$

$c : V \to \mathbb{N}$ $w, d : E \to \mathbb{N}$

- Let OPT be the value of an optimal solution to our problem
- Let $F \subseteq E$ be the edges in the optimal solution
- Let OPT_i be the total weight of the edges in F that are incident on v_i if the sum total of the demands on these edges does not exceed the capacity at v_i
- Else, $OPT_i = 0$
- We have, $\sum_{i=1}^{n} OPT_i \geq OPT$
- Also, $w(\mathcal{A}_i) \geq (1 - \epsilon)OPT_K^i$, where OPT_K^i is the value of the optimal solution to the Knapsack problem at vertex v_i

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Let OPT be the value of an optimal solution to our problem
- Let $F \subseteq E$ be the edges in the optimal solution
- Let OPT_i be the total weight of the edges in F that are incident on v_i if the sum total of the demands on these edges does not exceed the capacity at v_i
- Else, $OPT_i = 0$
- We have, $\sum_{i=1}^{n} OPT_i \geq OPT$
- Also, $w(\mathcal{A}_i) \geq (1 - \epsilon) OPT_K^i$, where OPT_K^i is the value of the optimal solution to the Knapsack problem at vertex v_i
- $w(E') = \sum_{i=1}^{n} w(\mathcal{A}_i) \geq (1-\epsilon) \sum_{i=1}^{n} OPT_K^i \geq (1-\epsilon) \sum_{i=1}^{n} OPT_i \geq (1 - \epsilon) OPT$

$c : V \to \mathbb{N}$ $w, d : E \to \mathbb{N}$

- Let OPT be the value of an optimal solution to our problem
- Let $F \subseteq E$ be the edges in the optimal solution
- Let OPT_i be the total weight of the edges in F that are incident on v_i if the sum total of the demands on these edges does not exceed the capacity at v_i
- Else, $OPT_i = 0$
- We have, $\sum_{i=1}^{n} OPT_i \geq OPT$
- Also, $w(\mathcal{A}_i) \geq (1-\epsilon)OPT_K^i$, where OPT_K^i is the value of the optimal solution to the Knapsack problem at vertex v_i
- $w(E') = \sum_{i=1}^{n} w(\mathcal{A}_i) \geq (1-\epsilon)\sum_{i=1}^{n} OPT_K^i \geq (1-\epsilon)\sum_{i=1}^{n} OPT_i \geq (1-\epsilon)OPT$
- This leads to a $(4+\epsilon)$-approximate solution

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Consider the scenario when G is a bipartite graph

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Consider the scenario when G is a bipartite graph
- We create the directed multi-graph $(U \cup V, E')$ with $w(E') \geq (1 - \epsilon)OPT$

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Consider the scenario when G is a bipartite graph
- We create the directed multi-graph $(U \cup V, E')$ with $w(E') \geq (1 - \epsilon)OPT$
- Clearly, the set of incoming edges to U (similarly V) forms a feasible solution

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Consider the scenario when G is a bipartite graph
- We create the directed multi-graph $(U \cup V, E')$ with $w(E') \geq (1 - \epsilon)OPT$
- Clearly, the set of incoming edges to U (similarly V) forms a feasible solution
- Both the feasible solutions together cover all the edges in E'

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Consider the scenario when G is a bipartite graph
- We create the directed multi-graph $(U \cup V, E')$ with $w(E') \geq (1 - \epsilon)OPT$
- Clearly, the set of incoming edges to U (similarly V) forms a feasible solution
- Both the feasible solutions together cover all the edges in E'
- The maximum of the two solutions has weight at least $(1 - \epsilon)OPT/2$

$c : V \rightarrow \mathbb{N}$ $w, d : E \rightarrow \mathbb{N}$

- Consider the scenario when G is a bipartite graph
- We create the directed multi-graph $(U \cup V, E')$ with $w(E') \geq (1 - \epsilon)OPT$
- Clearly, the set of incoming edges to U (similarly V) forms a feasible solution
- Both the feasible solutions together cover all the edges in E'
- The maximum of the two solutions has weight at least $(1 - \epsilon)OPT/2$
- We get a $(2 + \epsilon)$-approximate solution

Conclusion

- We have given constant factor approximation algorithms for several variants of the PDBEP problem

Conclusion

- We have given constant factor approximation algorithms for several variants of the PDBEP problem
- We know that the problem is APX-hard, so a PTAS is ruled out

Conclusion

- We have given constant factor approximation algorithms for several variants of the PDBEP problem
- We know that the problem is APX-hard, so a PTAS is ruled out
- Does there exist a 2-approximation algorithm for the most general case?

Conclusion

- We have given constant factor approximation algorithms for several variants of the PDBEP problem
- We know that the problem is APX-hard, so a PTAS is ruled out
- Does there exist a 2-approximation algorithm for the most general case?
- We will need better upper bounds

Thank you!
Questions?