ALLOCATION ALGORITHMS FOR NETWORKS WITH SCARCE RESOURCES

Kanthi Kiran Sarpatwar

Dissertation Defense: Feb 13, 2015

Committee

Samir Khuller (Advisor)
MohammadTaghi Hajiaghayi
Peter Keleher
Mark A. Shayman
Aravind Srinivasan

Motivation

Theme

How do we effectively handle bottleneck resources in networks?

Data Storage

Resource Replication Problems.

Computational Resources

Container Selection Problem.

Energy

Connected Dominating Set Problem.

Motivation

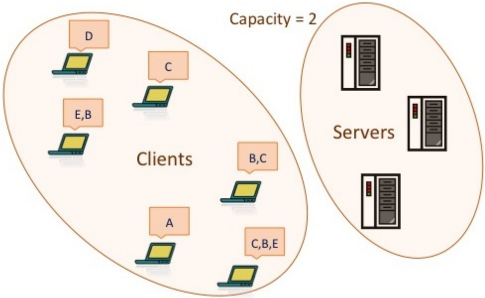
Theme

How do we effectively handle bottleneck resources in networks?

Data

Video content providers, such as Netflix, must replicate data to minimize client latency.

Content Distribution Network



Motivation

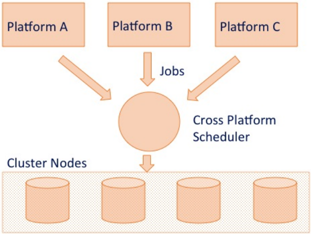
Theme

How do we effectively handle bottleneck resources in networks?

Computational Resources

Cross platform schedulers must fairly allocate cluster resources to various platforms.

Cluster Network



Motivation

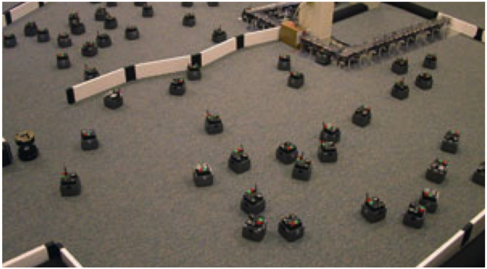
Theme

How do we effectively handle bottleneck resources in networks?

Energy

Wireless ad hoc networks have nodes with limited battery life. Key issues here are routing, target monitoring and interference.

Wireless Adhoc Network



Papers in this Talk

Improved Approximation Algorithms for Resource Replication Problems. APPROX 2012

Khuller, Saha, S.

Container Selection with Applications to Cloud Computing.

Nagarajan, S., Schieber, Shachnai, Wolf

Analyzing the Optimal Neighborhood: Approximation Algorithms for Partial and Budgeted Connected Dominating Set. SODA 2014

Khuller, Purohit, S.

Other Papers

The X-Flex Cross-Platform Scheduler: Who's the Fairest of Them All? Middleware 2014

Wolf, Nabi, Nagarajan, Saccone, Wagle, Hildrum, Pring, S.

Approximation Algorithms for Covering Problems in Energy Constrained Wireless Networks.

Khuller, Purohit, S.

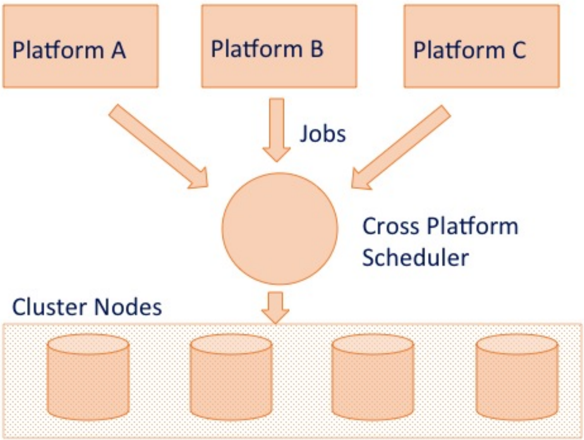
Improved Approximation Algorithms for Steiner tree and Cheapest Tour Oracles.

Bhatia, Gupta, S.

Part I

Computational Resources: Container Selection Problem

Cross platform scheduler



Examples of Cross Platform Schedulers

Dominant Resource Fairness (DRF) - NSDI 2011

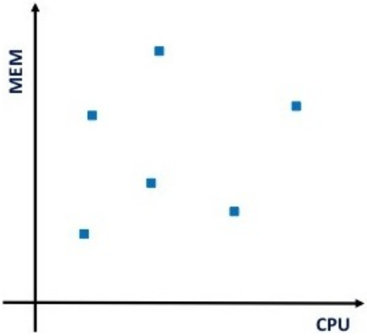
Ghodsi, Zaharia, Hindman, Konwinski, Shenker, Stoica

X-Flex Cross Platform Scheduler- Middleware 2014

Wolf, Nabi, Nagarajan, Saccone, Wagle, Hildrum, Pring, S.

Problem Definition

Example



Input

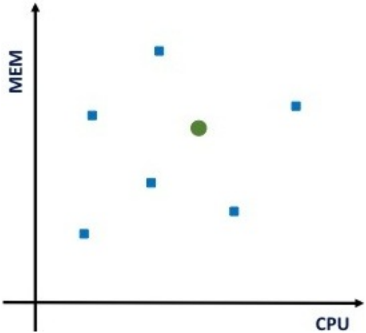
Given N jobs requiring two resources say CPU and memory.
Number of dimensions $d = 2$.

Goal

Find a few representative points for all the input points.

Problem Definition

Example



Container point

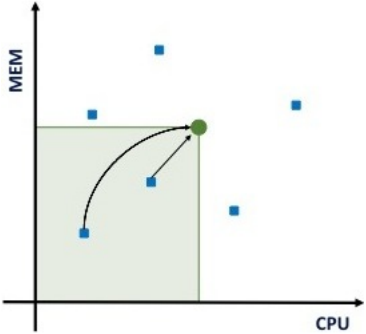
A point (x, y) *dominates* another point (x', y') if

$$x' \leq x \text{ and } y' \leq y$$

We call such a point (x, y) a container point.

Problem Definition

Example



Container point

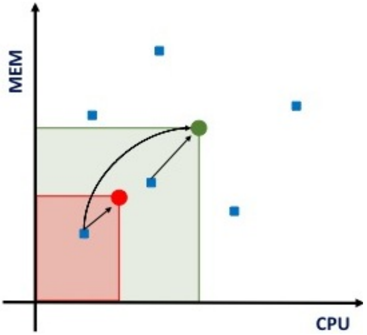
A point (x, y) *dominates* another point (x', y') if

$$x' \leq x \text{ and } y' \leq y$$

We call such a point (x, y) a container point.

Problem Definition

Example

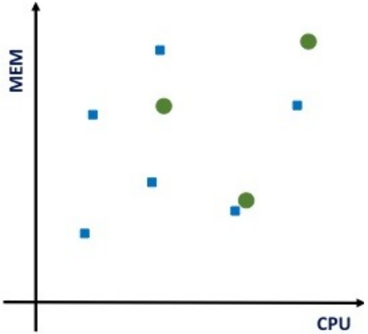


Cost of assignment

Cost of assigning an input point to a container point $(x, y) = x + y$

Problem Definition

Example

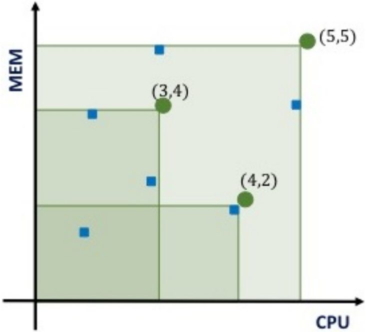


Objective

Find k container points that minimize the total assignment cost of all input points.

Problem Definition

Example

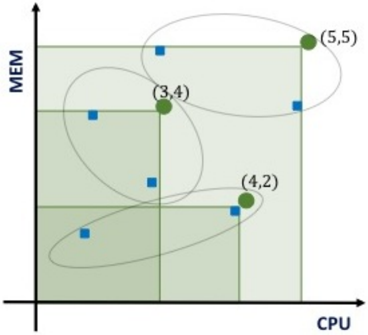


Objective

Find k container points that minimize the total assignment cost of all input points.

Problem Definition

Example



Total cost computation

$$2(2+4)+2(4+3)+2(5+5)=46$$

Main Result for the Continuous Setting

Definition (continuous container selection)

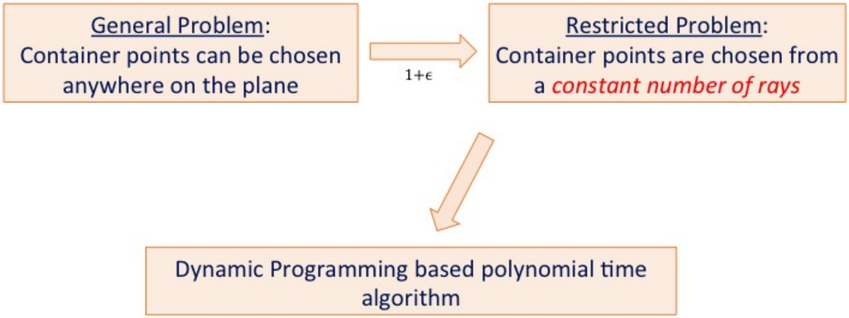
In an instance of the problem, we are given a set of input points $\mathscr{C}$ in $\mathbb{R}^d$ and a budget k. The goal is to find a subset S of k container points in $\mathbb{R}^d$, such that the following cost is minimized:

$$\sum_{p \in \mathscr{C}} \min_{\substack{c \in S \\ p \prec c}} \|c\|$$

Theorem

For any fixed dimension d, there is a polynomial time approximation scheme for the container selection problem.

The Main Idea



Potential Container Points

The Sets X and Y

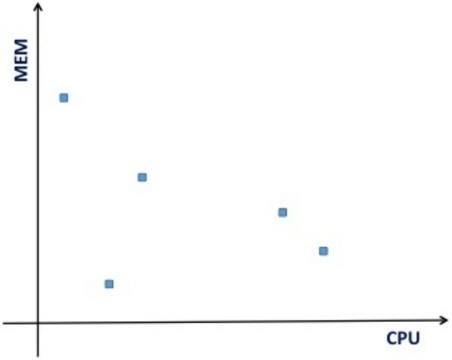
X be the set of all input x-coordinates and Y be the set of all input y-coordinates.

Observation

Any container point chosen by an optimal solution must be in $X \times Y$.

Reducing General Case to Restricted Case

Transformation

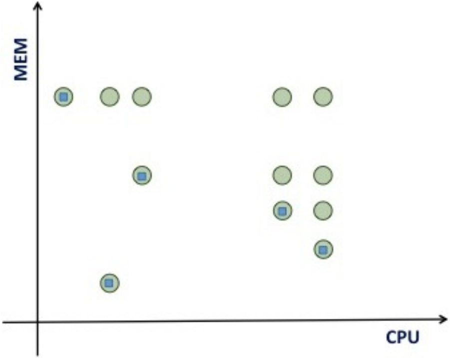


Input:

Set of N points and a budget k on the number of containers.

Reducing General Case to Restricted Case

Transformation

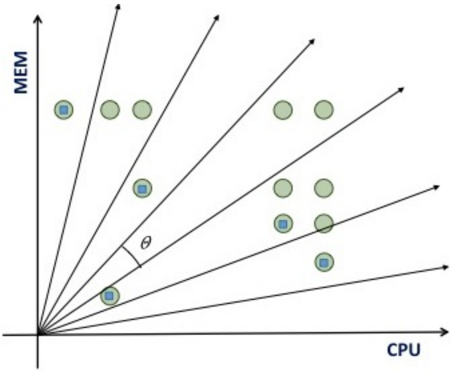


Compute:

The set of potential container points.

Reducing General Case to Restricted Case

Transformation

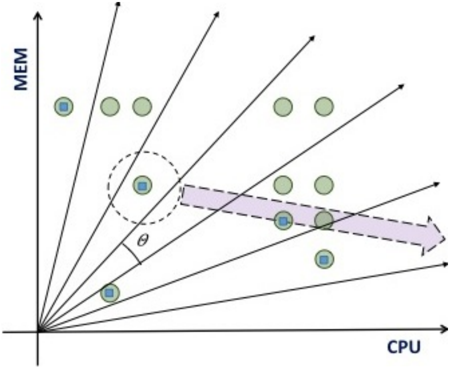


Constant number of lines:

For a given ε, we construct equiangular rays separated by $\theta \approx \varepsilon/2$.

Reducing General Case to Restricted Case

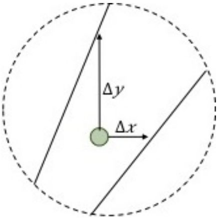
Transformation



Shifting:

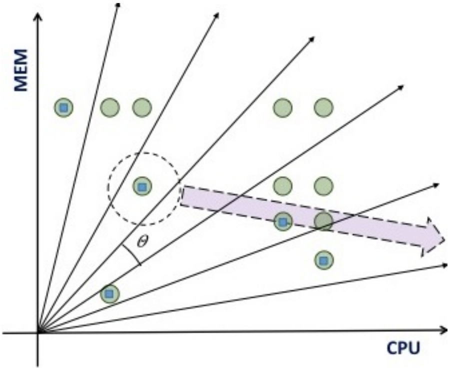
Shift potential container points onto these rays.

Magnified View:



Reducing General Case to Restricted Case

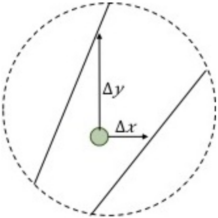
Transformation



Shifting:

Using basic trigonometry, we obtain
$\min(\Delta x, \Delta y) \leq (x+y)2\theta \leq (x+y)\varepsilon$

Magnified View:



Theorem

There is a poly-time algorithm for the restricted container selection problem.

PTAS in Two and Higher Dimensions

Higher Dimensions

The transformation can be extended to higher fixed dimensions!

Restricted Problem

There is a poly-time algorithm for the restricted container selection problem in any fixed dimension d.

Theorem (PTAS)

This implies a PTAS for the continuous container selection problem in any fixed dimension.

Theorem (NP-hard)

We further show that the problem is NP-hard in any fixed dimension $d \geq 3$.

PTAS in Two and Higher Dimensions

Higher Dimensions

The transformation can be extended to higher fixed dimensions!

Restricted Problem

There is a poly-time algorithm for the restricted container selection problem in any fixed dimension d.

Theorem (PTAS)

This implies a PTAS for the continuous container selection problem in any fixed dimension.

Theorem (NP-hard)

We further show that the problem is NP-hard in any fixed dimension $d \geq 3$.

PTAS in Two and Higher Dimensions

Higher Dimensions

The transformation can be extended to higher fixed dimensions!

Restricted Problem

There is a poly-time algorithm for the restricted container selection problem in any fixed dimension d.

Theorem (PTAS)

This implies a PTAS for the continuous container selection problem in any fixed dimension.

Theorem (NP-hard)

We further show that the problem is NP-hard in any fixed dimension $d \geq 3$.

PTAS in Two and Higher Dimensions

Higher Dimensions

The transformation can be extended to higher fixed dimensions!

Restricted Problem

There is a poly-time algorithm for the restricted container selection problem in any fixed dimension d.

Theorem (PTAS)

This implies a PTAS for the continuous container selection problem in any fixed dimension.

Theorem (NP-hard)

We further show that the problem is NP-hard in any fixed dimension $d \geq 3$.

PTAS in Two and Higher Dimensions

Higher Dimensions

The transformation can be extended to higher fixed dimensions!

Restricted Problem

There is a poly-time algorithm for the restricted container selection problem in any fixed dimension d.

Theorem (PTAS)

This implies a PTAS for the continuous container selection problem in any fixed dimension.

Theorem (NP-hard)

We further show that the problem is NP-hard in any fixed dimension $d \geq 3$.

Discrete Variant

Discrete vs Continuous

- Containers must be chosen from a given set of potential container points $\mathscr{F}$.
- The previous transformation fails as we are not allowed to "move" them!
- Much harder than continuous version! In fact, we show that:

Theorem (Hardness of Approximation)

For any dimension $d \geq 3$, the discrete container selection problem is NP-hard to obtain any approximation.

What do we do?

Relax the problem a little

We relax the restriction on the number of container points. What do we know?

Bi-approximation Results

- Special case of non-metric k-median problem.
- There is a $(1+\varepsilon, (1+\frac{1}{\varepsilon})\ln n)$ bi-approximation algorithm (Lin and Vitter, STOC 1992).

The Question

Can we still use the geometric properties of our special problem to do better?

Our Results

2-Dimensions

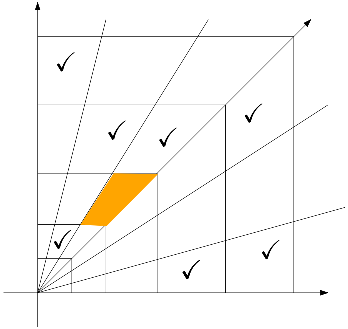
- We obtain a $(1+\varepsilon, 3)$ bi-approximation algorithm.
- This approach does not work for higher dimensions.

Higher Dimensions

- We obtain a $(1+\varepsilon, O(\frac{d}{\varepsilon}\log dk))$ bi-approximation algorithm for dimension d.
- Based on an LP relaxation.

Our Algorithm for 2-Dimensions

Cells

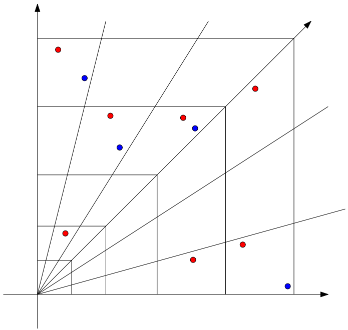


Decompose

Decompose the space in $O(\log n)$ cells. “Guess” which cells are touched by a fixed optimal solution. Call them *good* cells.

Our Algorithm for 2-Dimensions

Cells

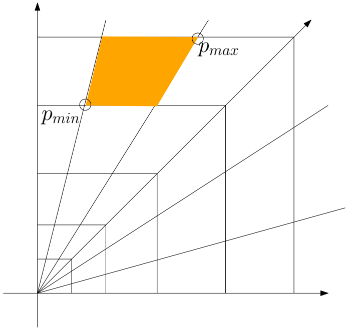


Representative points

From the good cells choose two container points - one with maximum x-coordinate and the other with maximum y-coordinate.

Our Algorithm for 2-Dimensions

Cells

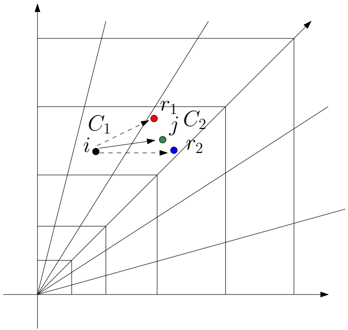


Cells are approximately uniform

Using a trigonometric argument, we can show that the costs of container points in any given cell are approximately the same, i.e., $p_{max}/p_{min} \leq (1+\varepsilon)$.

Our Algorithm for 2-Dimensions

Cells



Decoupling the cells

Given input point $i \in C_1$ and container point $j \in C_2$ such that $i \prec j$, then $i \prec r_1$ or $i \prec r_2$.

Our Algorithm for 2-Dimensions

Single Cell Problem

For a given budget k_1 to a cell, we try and satisfy input points only from that cell. We can use a simple DP to solve this optimally.

Restricted problem

- The only “inter cell” allocations are to the representative container points.
- We use a dynamic program based scheme to solve the problem under this restrictions.

Results Summary

Continuous CSP

PTAS for any $d \geq 2$

NP-hard for any $d \geq 3$

Discrete CSP

- For two dimensions, a $(1+\varepsilon, 3)$-bi-approximation algorithm
- For any fixed dimension d, a $(1+\varepsilon, O(\frac{1}{\varepsilon}\log dk))$ bi-approximation algorithm.

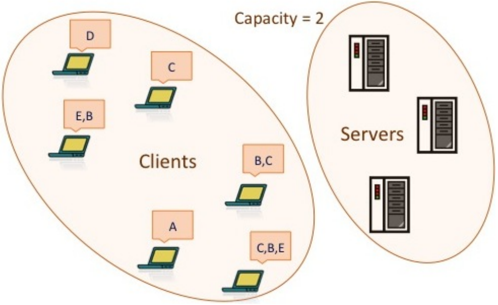
NP-hard to approximate, for any $d \geq 3$

Part II

Data: Resource Replication Problems

Resource Replication Problems

Example

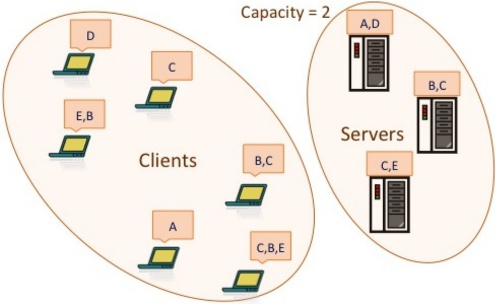


Framework

- Clients and servers are embedded into a **metric space**.
- Clients need a subset of data objects $\{A, B, C, D, E\}$.
- Servers have limited capacities to store data objects.

Resource Replication Problems

Example

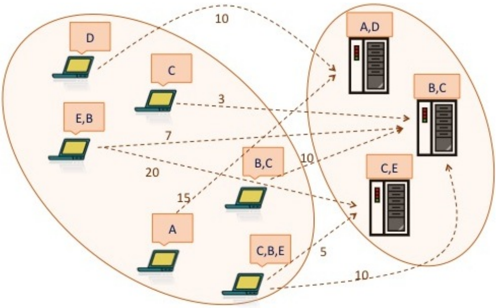


Framework

Goal: Place data items on different servers to meet the demands of all clients. Minimize the distance a client has to travel to go to get a required data item.

Resource Replication Problems

Example

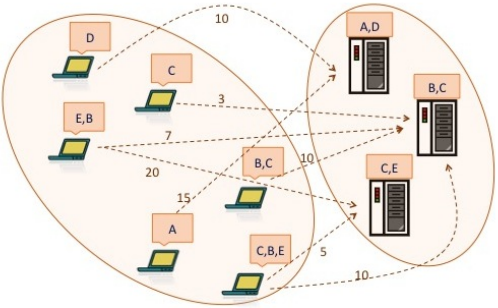


Framework

Goal: Place data items on different servers to meet the demands of all clients. Minimize the distance a client has to travel to go to get a required data item.

Resource Replication Problems

Example



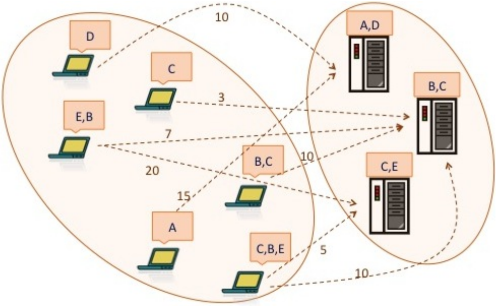
Objectives: Min Sum

Minimize the aggregate distance travelled by all clients to obtain all of their required data objects.

For this example, total cost = $10+3+(20+7)+15+(10+10)+(10+5+5)=95$

Resource Replication Problems

Example



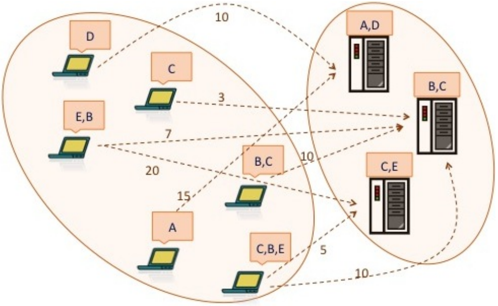
Objectives: Min Sum

Minimize the aggregate distance travelled by all clients to obtain all of their required data objects.

For this example, total cost = $10 + 3 + (20 + 7) + 15 + (10 + 10) + (10 + 5 + 5) = 95$

Resource Replication Problems

Example



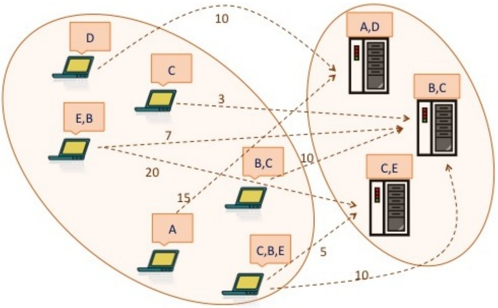
Objectives: Min Max

Minimize the maximum distance travelled by all clients to obtain all of their required data objects.

For this example, cost = 20

Resource Replication Problems

Example



Objectives: Min Max

Minimize the maximum distance travelled by all clients to obtain all of their required data objects.
For this example, cost = 20

Related Work

Min Sum Objective : Baev, Rajaraman and Swamy - SIAM J. Compt. 2008

- LP-based approximation algorithm.
- Has an approximation guarantee of 10.

Min Max Objective : Ko and Rubenstein - ICNP 2003, ICNP 2004

- Heuristic approach.
- 3-approximation algorithm for a basic version - but the algorithm does not necessarily terminate in poly time.
- No approximation guarantee for the general problem.

Related Work

Min Sum Objective : Baev, Rajaraman and Swamy - SIAM J. Compt. 2008

- LP-based approximation algorithm.
- Has an approximation guarantee of 10.

Min Max Objective : Ko and Rubenstein - ICNP 2003, ICNP 2004

- Heuristic approach.
- 3-approximation algorithm for a basic version - but the algorithm does not necessarily terminate in poly time.
- No approximation guarantee for the general problem.

Related Work

Min Sum Objective : Baev, Rajaraman and Swamy - SIAM J. Compt. 2008

- LP-based approximation algorithm.
- Has an approximation guarantee of 10.

Min Max Objective : Ko and Rubenstein - ICNP 2003, ICNP 2004

- Heuristic approach.
- 3-approximation algorithm for a basic version - but the algorithm does not necessarily terminate in poly time.
- No approximation guarantee for the general problem.

Basic Resource Replication

Definition

Given a graph $G = (V, E)$, a metric $d : E \rightarrow R^+ \cup \{0\}$ and data types set $\mathscr{C}$. Find a mapping $\phi : V \rightarrow \mathscr{C}$ to minimize the following quantity:

$$\max_{v \in V, r \in \mathscr{C}} \min_{u \ni \phi(u) = r} d(u, v)$$

- Every node needs all data items.
- Every node has a unit storage capacity.
- We give a simple 3-approximation algorithm.

Basic Resource Replication

Definition

Given a graph $G = (V, E)$, a metric $d : E \to R^{+} \cup \{0\}$ and data types set $\mathscr{C}$.
Find a mapping $\phi : V \to \mathscr{C}$ to minimize the following quantity:

$$\max_{v \in V, r \in \mathscr{C}} \min_{u \ni \phi(u) = r} d(u, v)$$

- Every node needs all data items.
- Every node has a unit storage capacity.
- We give a simple 3-approximation algorithm.

Basic Resource Replication

Definition

Given a graph $G = (V, E)$, a metric $d : E \rightarrow R^+ \cup \{0\}$ and data types set $\mathscr{C}$. Find a mapping $\phi : V \rightarrow \mathscr{C}$ to minimize the following quantity:

$$\max_{v \in V, r \in \mathscr{C}} \min_{u \ni \phi(u) = r} d(u, v)$$

- Every node needs all data items.
- Every node has a unit storage capacity.
- We give a simple 3-approximation algorithm.

Basic Resource Replication

Definition

Given a graph $G = (V, E)$, a metric $d : E \rightarrow R^+ \cup \{0\}$ and data types set $\mathscr{C}$. Find a mapping $\phi : V \rightarrow \mathscr{C}$ to minimize the following quantity:

$$\max_{v \in V, r \in \mathscr{C}} \min_{u \ni \phi(u) = r} d(u, v)$$

- Every node needs all data items.
- Every node has a unit storage capacity.
- We give a simple 3-approximation algorithm.

Basic Resource Replication

Definition

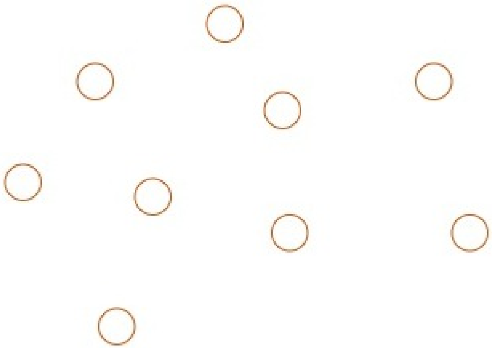
Given a graph $G = (V, E)$, a metric $d : E \rightarrow R^{+} \cup \{0\}$ and data types set $\mathscr{C}$. Find a mapping $\phi : V \rightarrow \mathscr{C}$ to minimize the following quantity:

$$\max_{v \in V, r \in \mathscr{C}} \min_{u \ni \phi(u) = r} d(u, v)$$

- Every node needs all data items.
- Every node has a unit storage capacity.
- We give a simple 3-approximation algorithm.

Algorithm for Basic Resource Replication

Example



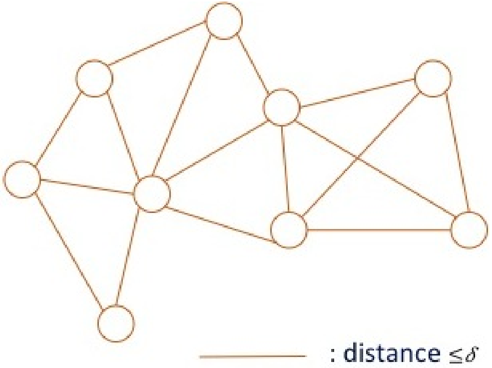
Input

Set of nodes and objects

$\{R, G, B\}$

Algorithm for Basic Resource Replication

Example

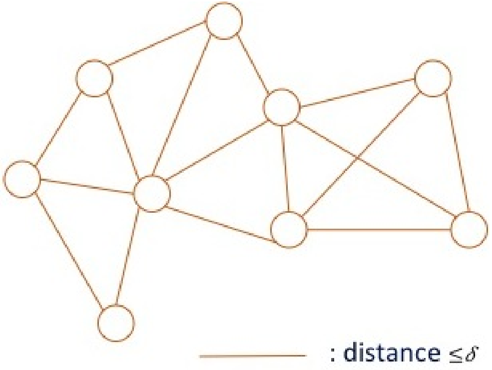


Construct Threshold Graph:

- "Guess" optimal dist. δ.
- Add *uv* if $d(u,v) \leq \delta$.

Algorithm for Basic Resource Replication

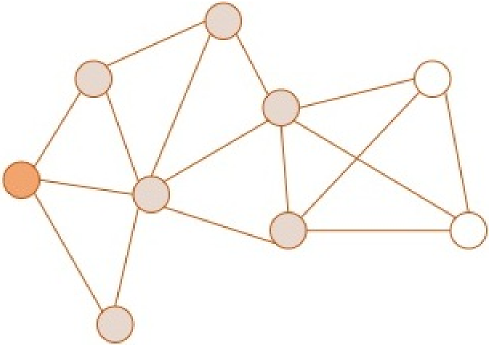
Example



Compute 2-hop MIS:

Keep picking nodes and delete nodes within two hops.

Example

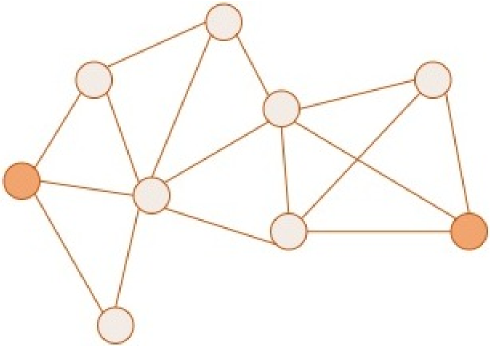


Compute 2-hop MIS:

Keep picking nodes and delete nodes within two hops.

Algorithm for Basic Resource Replication

Example

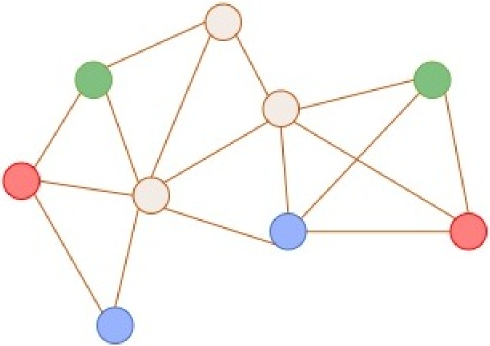


Compute 2-hop MIS:

Keep picking nodes and delete nodes within two hops.

Algorithm for Basic Resource Replication

Example



Assign colors:

For each vertex in *MIS*, we place $k = 3$ resources in its neighborhood in G_δ.

Algorithm for the Basic Resource Replication

Analysis

- Every vertex has a degree at least $k-1$ in G_δ. Therefore, our coloring is valid.
- By definition, every vertex is within a 2-hop distance of some vertex in *MIS*.
- Hence, every vertex has all the colors within a 3-hop distance.

Theorem

There is a 3-approximation algorithm for the basic resource replication problem.

Subset Resource Replication

Given:

- a graph $G = (V, E)$ embedded into a metric $d : E \rightarrow R^{+} \cup \{0\}$ and a set of data items $\mathscr{C}$
- each vertex has a storage capacity of s_v
- each vertex needs a subset of data items $\mathscr{C}_v$

Goal:

find a mapping $\phi : V \rightarrow 2^{\mathscr{C}}$ that assigns at most s_v data items to v and minimizes the following quantity:

$$\max_{v \in V} \min_{\substack{r \in \mathscr{C}_v \\ u \ni \phi(u) = r}} d(u, v)$$

Subset Resource Replication

Given:

- a graph $G = (V, E)$ embedded into a metric $d : E \rightarrow R^{+} \cup \{0\}$ and a set of data items $\mathscr{C}$
- each vertex has a storage capacity of s_v
- each vertex needs a subset of data items $\mathscr{C}_v$

Goal:

find a mapping $\phi : V \rightarrow 2^{\mathscr{C}}$ that assigns at most s_v data items to v and minimizes the following quantity:

$$\max_{v \in V} \min_{\substack{r \in \mathscr{C}_v \\ u \ni \phi(u) = r}} d(u, v)$$

Subset Resource Replication

Given:

- a graph $G = (V, E)$ embedded into a metric $d : E \rightarrow R^+ \cup \{0\}$ and a set of data items $\mathscr{C}$
- each vertex has a storage capacity of s_v
- each vertex needs a subset of data items $\mathscr{C}_v$

Goal:

find a mapping $\phi : V \rightarrow 2^{\mathscr{C}}$ that assigns at most s_v data items to v and minimizes the following quantity:

$$\max_{v \in V} \min_{\substack{r \in \mathscr{C}_v \\ u \ni \phi(u) = r}} d(u, v)$$

Subset Resource Replication

Given:

- a graph $G = (V, E)$ embedded into a metric $d : E \rightarrow R^{+} \cup \{0\}$ and a set of data items $\mathscr{C}$
- each vertex has a storage capacity of s_v
- each vertex needs a subset of data items $\mathscr{C}_v$

Goal:

find a mapping $\phi : V \rightarrow 2^{\mathscr{C}}$ that assigns at most s_v data items to v and minimizes the following quantity:

$$\max_{v \in V} \min_{\substack{r \in \mathscr{C}_v \\ u \ni \phi(u) = r}} d(u, v)$$

Subset Resource Replication

Given:

- a graph $G = (V, E)$ embedded into a metric $d : E \rightarrow R^+ \cup \{0\}$ and a set of data items $\mathscr{C}$
- each vertex has a storage capacity of s_v
- each vertex needs a subset of data items $\mathscr{C}_v$

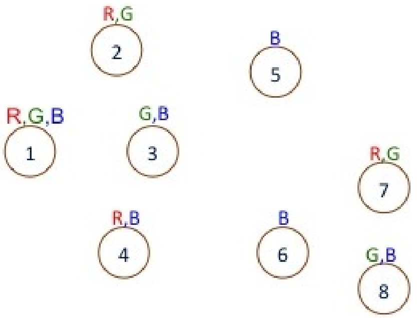
Goal:

find a mapping $\phi : V \rightarrow 2^{\mathscr{C}}$ that assigns at most s_v data items to v and minimizes the following quantity:

$$\max_{v \in V} \min_{\substack{r \in \mathscr{C}_v \\ u \ni \phi(u) = r}} d(u, v)$$

Algorithm for Subset Resource Replication

Example

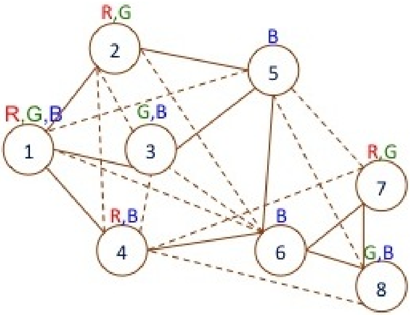


Input:

- Objects set $\{R, G, B\}$
- Subsets of data items needed by vertices are next to them
- Storage capacities (for this example) are unit

Algorithm for Subset Resource Replication

Example



Threshold Graph:

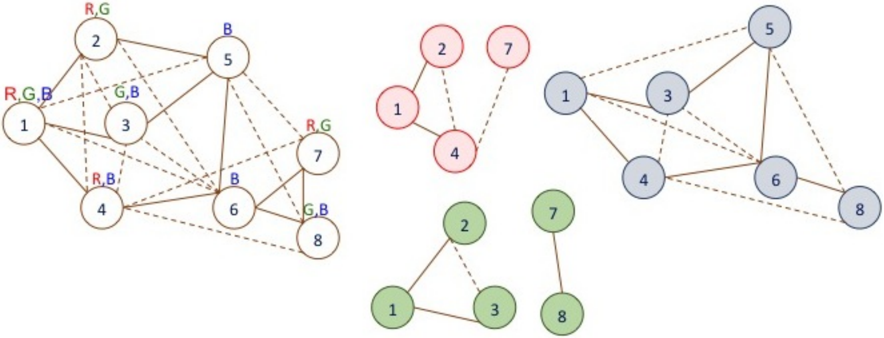
- Guess optimal δ
- Construct threshold G_δ
- Mark 2-hop edges, represented by dashed lines

Algorithm for Subset Resource Replication

Decompose:

For each color *r*, construct a subgraph on nodes needing resource *r*.

Example

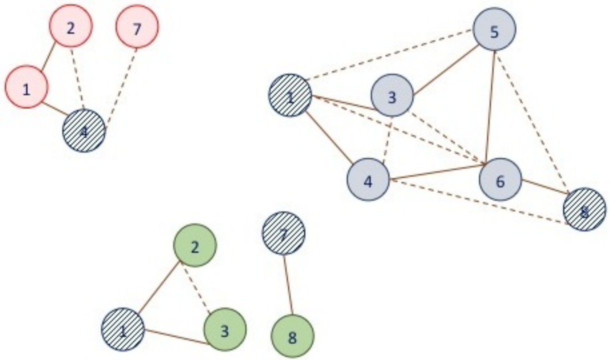


Algorithm for Subset Resource Replication

Decompose:

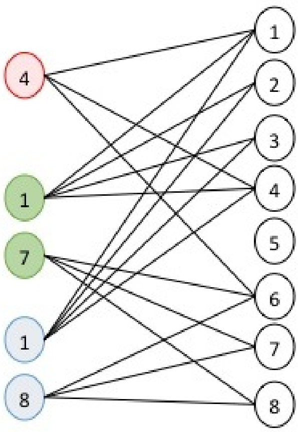
Compute 2-hop maximal independent set in each of these subgraphs.

Example



Algorithm for Subset Resource Replication

Example



Side A:

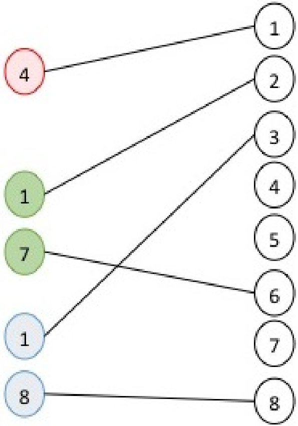
Union of 2-hop maximal independent sets in each subgraph

Side B:

Vertices of the graph.

Algorithm for Subset Resource Replication

Example

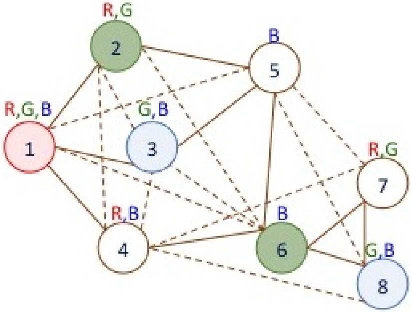


Compute:

A b-matching with bounds

- s_v on the vertex v on the right side
- 1 on the vertices on the left.

Example



Color:

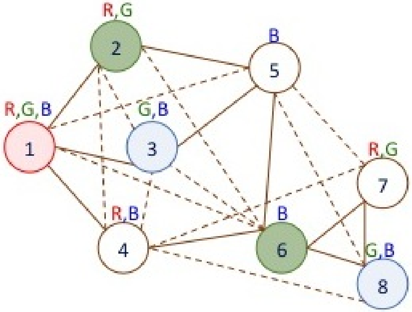
Use the determined matching to color the nodes

Theorem

There is a 3-approximation algorithm for the subset resource replication problem.

Algorithm for Subset Resource Replication

Example



Color:

Use the determined matching to color the nodes

Theorem

There is a 3-approximation algorithm for the subset resource replication problem.

Results Summary

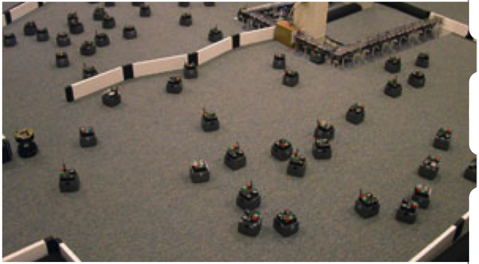
Problem	Approx. Guar.	Hardness
Basic Resource Replication (BRR)	3	$2-\varepsilon$
Subset Resource Replication (SRR)	3	$3-\varepsilon$
Robust BRR	3	$2-\varepsilon$
Robust K-BRR	5	$2-\varepsilon$
Robust SRR	-	NP-hard to approx.
Capacitated BRR	$(4,2)$-bi-approx.	-

Part III

Energy: Partial and Budgeted Connected Dominating Set

Energy Issues in Wireless Adhoc Network

Wireless Adhoc Network



Routing

Communication backbones, i.e., virtual backbone

Monitoring

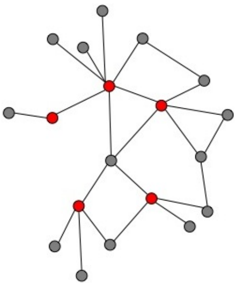
Target monitoring in sensor networks

Interference

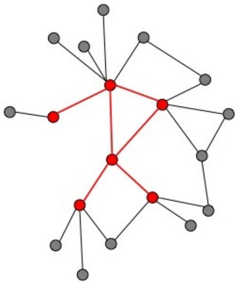
Message propagation in radio networks

Problem Definitions

Dominating Set (DS)



Connected Dominating Set (CDS)



Definition (CDS)

Given an undirected graph $G = (V, E)$, find a connected subgraph with fewest number of nodes that dominates all the nodes.

Previous Work on CDS

Guha and Khuller, ESA 1996

$\ln \Delta + 3$ approximation algorithm in general graphs. Set cover hard.

Dubhashi, Mei, Panconesi, Radhakrishnan, and Srinivasan, SODA 2003

Distributed $O(\ln \Delta)$ approximation algorithm in general graphs.

Demaine and Hajiaghayi, SODA 2005

PTAS in planar graphs.

Cheng, Huang, Li, Wu, and Du, Networks 2003

PTAS in unit disk graphs.

Previous Work on CDS

Guha and Khuller, ESA 1996

$\ln \Delta + 3$ approximation algorithm in general graphs. Set cover hard.

Dubhashi, Mei, Panconesi, Radhakrishnan, and Srinivasan, SODA 2003

Distributed $O(\ln \Delta)$ approximation algorithm in general graphs.

Demaine and Hajiaghayi, SODA 2005

PTAS in planar graphs.

Cheng, Huang, Li, Wu, and Du, Networks 2003

PTAS in unit disk graphs.

Previous Work on CDS

Guha and Khuller, ESA 1996

$\ln \Delta + 3$ approximation algorithm in general graphs. Set cover hard.

Dubhashi, Mei, Panconesi, Radhakrishnan, and Srinivasan, SODA 2003

Distributed $O(\ln \Delta)$ approximation algorithm in general graphs.

Demaine and Hajiaghayi, SODA 2005

PTAS in planar graphs.

Cheng, Huang, Li, Wu, and Du, Networks 2003

PTAS in unit disk graphs.

Previous Work on CDS

Guha and Khuller, ESA 1996

$\ln \Delta + 3$ approximation algorithm in general graphs. Set cover hard.

Dubhashi, Mei, Panconesi, Radhakrishnan, and Srinivasan, SODA 2003

Distributed $O(\ln \Delta)$ approximation algorithm in general graphs.

Demaine and Hajiaghayi, SODA 2005

PTAS in planar graphs.

Cheng, Huang, Li, Wu, and Du, Networks 2003

PTAS in unit disk graphs.

Previous Work on CDS

Guha and Khuller, ESA 1996

$\ln \Delta + 3$ approximation algorithm in general graphs. Set cover hard.

Dubhashi, Mei, Panconesi, Radhakrishnan, and Srinivasan, SODA 2003

Distributed $O(\ln \Delta)$ approximation algorithm in general graphs.

Demaine and Hajiaghayi, SODA 2005

PTAS in planar graphs.

Cheng, Huang, Li, Wu, and Du, Networks 2003

PTAS in unit disk graphs.

Motivation

CDS as Virtual Backbone

A "small" CDS is a good model for a virtual backbone (Bhargavan and Das, ICC 1997)

Outliers

A few "far off" nodes might necessitate a large CDS - making it a bad model for a backbone.

More Robust Model?

Can we find a good backbone if we were to "serve" say only 90% of all nodes?

Motivation

CDS as Virtual Backbone

A "small" CDS is a good model for a virtual backbone (Bhargavan and Das, ICC 1997)

Outliers

A few "far off" nodes might necessitate a large CDS - making it a bad model for a backbone.

More Robust Model?

Can we find a good backbone if we were to "serve" say only 90% of all nodes?

Motivation

CDS as Virtual Backbone

A “small” CDS is a good model for a virtual backbone (Bhargavan and Das, ICC 1997)

Outliers

A few “far off” nodes might necessitate a large CDS - making it a bad model for a backbone.

More Robust Model?

Can we find a good backbone if we were to “serve” say only 90% of all nodes?

Motivation

CDS as Virtual Backbone

A “small” CDS is a good model for a virtual backbone (Bhargavan and Das, ICC 1997)

Outliers

A few “far off” nodes might necessitate a large CDS - making it a bad model for a backbone.

More Robust Model?

Can we find a good backbone if we were to “serve” say only 90% of all nodes?

Partial Connected Dominating Set (PCDS)

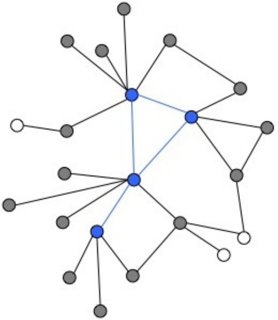


Figure: PCDS on with quota $Q = 19$

Definition (PCDS)

Given:

- undirected graph $G = (V, E)$
- a quota Q

Find a connected subgraph with fewest number of nodes that dominates at least Q nodes.

Problem Definitions

Partial Connected Dominating Set (PCDS)

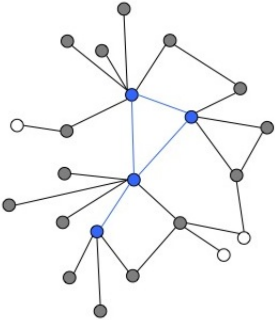


Figure: PCDS on with quota $Q = 19$

Definition (PCDS)

Given:

- undirected graph $G = (V, E)$
- a quota Q

Find a connected subgraph with fewest number of nodes that dominates at least Q nodes.

Problem Definitions

Partial Connected Dominating Set (PCDS)

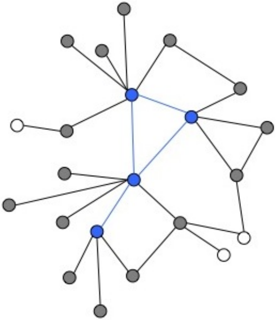


Figure: PCDS on with quota $Q = 19$

Definition (PCDS)

Given:

- undirected graph $G = (V, E)$
- a quota Q

Find a connected subgraph with fewest number of nodes that dominates at least Q nodes.

Problem Definitions

Partial Connected Dominating Set (PCDS)

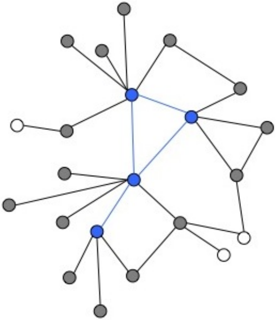


Figure: PCDS on with quota $Q = 19$

Definition (PCDS)

Given:

- undirected graph $G = (V, E)$
- a quota Q

Find a connected subgraph with fewest number of nodes that dominates at least Q nodes.

Problem Definitions

Budgeted Connected Dominating Set (BCDS)

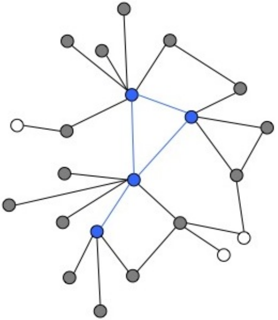


Figure: BCDS on with budget $k = 4$

Definition (BCDS)

Given:

- undirected graph $G = (V, E)$
- a budget k

Find a connected subgraph on at most k nodes that dominates as many nodes as possible.

Our Results

Theorem

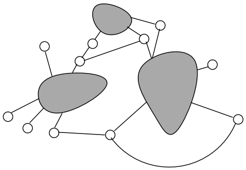
A polynomial time algorithm with $4ln\Delta + 2$ approximation guarantee for the PCDS problem.

Theorem

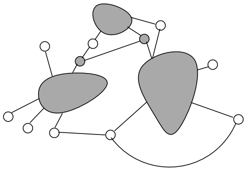
A polynomial time algorithm with $\frac{1}{13}(1 - \frac{1}{e})$ approximation guarantee for the BCDS problem.

Challenges in Solving the PCDS problem.

Dominating Set (DS)



Connected Dominating Set (CDS)



Converting DS to CDS

It can be shown that any dominating set of size D can be connected using at most $2D$ extra vertices.
This yields a simple $O(\log \Delta)$ approximation.

Challenges in Solving the PCDS problem.

How about PCDS?

Unfortunately, such an approach does not work for the PCDS problem.

Greedy Approach?

- Greedily picking vertices until Q vertices are satisfied and then connecting them - bad idea.
- The components could be far away from each other.

Conservative Greedy?

- Greedily picking vertices while maintaining connectivity.
- Fails! Favors “locally” productive areas over “globally” rich areas.

Challenges in Solving the PCDS problem.

How about PCDS?

Unfortunately, such an approach does not work for the PCDS problem.

Greedy Approach?

- Greedily picking vertices until Q vertices are satisfied and then connecting them - bad idea.
- The components could be far away from each other.

Conservative Greedy?

- Greedily picking vertices while maintaining connectivity.
- Fails! Favors "locally" productive areas over "globally" rich areas.

Challenges in Solving the PCDS problem.

How about PCDS?

Unfortunately, such an approach does not work for the PCDS problem.

Greedy Approach?

- Greedily picking vertices until Q vertices are satisfied and then connecting them - bad idea.
- The components could be far away from each other.

Conservative Greedy?

- Greedily picking vertices while maintaining connectivity.
- Fails! Favors “locally” productive areas over “globally” rich areas.

An Idea

An "easier" problem.

- Profit function in PCDS is a non-linear "coverage" function.
- What happens if the profit function is "linear"?
- We obtain (a simpler variant of) the well known quota Steiner tree.

Definition

Given an undirected graph $G(V, E)$ and profit function $p : V \rightarrow \mathbb{Z}^{+} \cup \{0\}$ and a quota Q. Find the tree T with least number of vertices with total profit $\geq Q$

Theorem (Johnson et al. [SODA 2000], Garg [STOC 2005])

Quota Steiner tree has a 2-approximation algorithm.

An Idea

An "easier" problem.

- Profit function in PCDS is a non-linear "coverage" function.
- What happens if the profit function is "linear"?
- We obtain (a simpler variant of) the well known quota Steiner tree.

Definition

Given an undirected graph $G(V,E)$ and profit function $p: V \to \mathbb{Z}^+ \cup \{0\}$ and a quota Q. Find the tree T with least number of vertices with total profit $\geq Q$

Theorem (Johnson et al. [SODA 2000], Garg [STOC 2005])

Quota Steiner tree has a 2-approximation algorithm.

An Idea

An “easier” problem.

- Profit function in PCDS is a non-linear “coverage” function.
- What happens if the profit function is “linear”?
- We obtain (a simpler variant of) the well known quota Steiner tree.

Definition

Given an undirected graph $G(V, E)$ and profit function $p : V \rightarrow \mathbb{Z}^+ \cup \{0\}$ and a quota Q. Find the tree T with least number of vertices with total profit $\geq Q$

Theorem (Johnson et al. [SODA 2000], Garg [STOC 2005])

Quota Steiner tree has a 2-approximation algorithm.

Our Approach

Use QST

PCDS $\Longrightarrow$ **QST**

Ideas

- Approximate the coverage function by a linear function? Can we do it with a $\log n$ loss?
- What are the candidates? How about degree function? bad idea!
- Somewhat surprisingly, a natural linear function defined by a greedy scheme to find the complete dominating set works!

Our Approach

Use QST

PCDS $\Longrightarrow$ **QST**

Ideas

- Approximate the coverage function by a linear function? Can we do it with a $\log n$ loss?
- What are the candidates? How about degree function? bad idea!
- Somewhat surprisingly, a natural linear function defined by a greedy scheme to find the complete dominating set works!

Our Approach

Use QST

PCDS $\Longrightarrow$ QST

Ideas

- Approximate the coverage function by a linear function? Can we do it with a $\log n$ loss?
- What are the candidates? How about degree function? bad idea!
- Somewhat surprisingly, a natural linear function defined by a greedy scheme to find the complete dominating set works!

Our Approach

Use QST

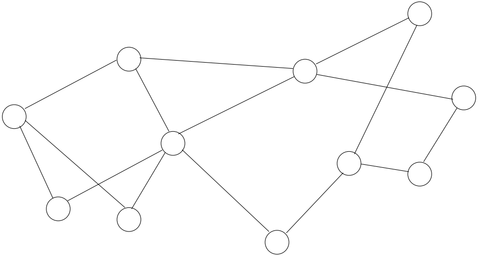
$$\boxed{\textbf{PCDS}} \Longrightarrow \boxed{\textbf{QST}}$$

Ideas

- Approximate the coverage function by a linear function? Can we do it with a $\log n$ loss?
- What are the candidates? How about degree function? bad idea!
- Somewhat surprisingly, a natural linear function defined by a greedy scheme to find the complete dominating set works!

Our Approach

Greedy Linear Function

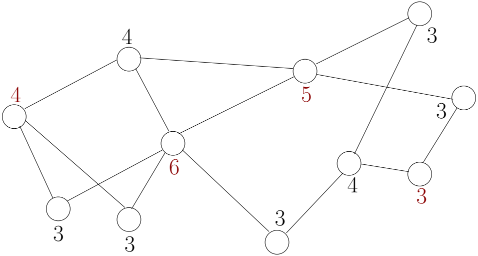


Description

Use the natural greedy algorithm to define a linear function.

Our Approach

Greedy Linear Function

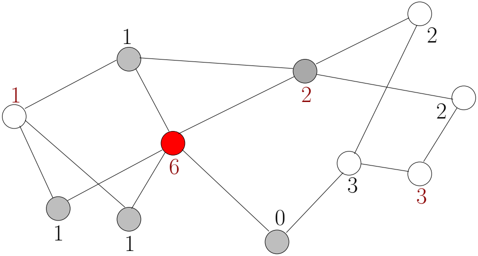


Description

For each vertex, compute the number of uncovered neighbors.

Our Approach

Greedy Linear Function

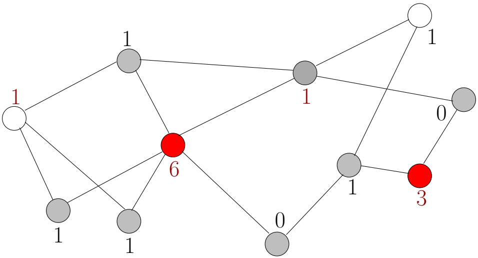


Description

Choose most profitable vertex and recompute for rest.

Our Approach

Greedy Linear Function

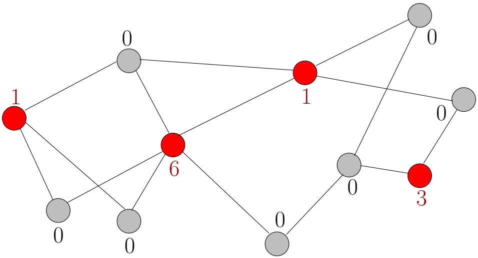


Description

Tie breaking is arbitrary.

Our Approach

Greedy Linear Function

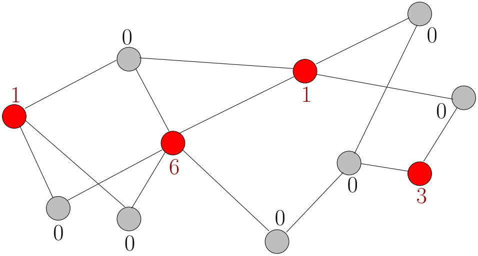


Description

We may choose covered vertices if they qualify.

Our Approach

Greedy Linear Function



The Profit Function

$p(v) =$ # of newly covered neighbors by v.

The Algorithm for PCDS

Input

Given an undirected graph $G = (V, E)$ and a quota Q.

STEP 1

Run the greedy dominating set algorithm and compute the linear profit function $p : V \to \mathbb{Z}^+ \cup \{0\}$.

STEP 2

Solve the quota Steiner tree on the instance (G, Q, p) and return it.

Theorem

Let OPT denote the optimal PCDS solution and T denote the optimal quota Steiner tree. Then T is a feasible solution for the PCDS instance and $|T| \leq (2\log n + 1)|OPT|$.

Part IV

Conclusion

Open Problems

Container Selection Problem

- Resolving the hardness of 2-dimensions problem
- What happens in the case of non-fixed dimensions - esp. the continuous variant
- Improving bounds for the discrete case.

Resource Replication Problem

- Most results are almost tight, except the capacitated variant. Can we tighten the bounds further?

Partial/Budgeted Connected Dominating Set

- Distributed setting? Planar graphs? Unit disk graphs?
- Tighten the bounds
- Capacitated PCDS/BCDS?

Co-authors

- Samir Khuller
- Manish Purohit
- Barna Saha
- Viswanath Nagarajan
- Baruch Schieber
- Hadas Shachnai
- MohammadTaghi Hajiaghayi
- Joel Wolf
- Meghana Mande
- Prabhanjan Ananth
- Guy Kortsarz
- Randeep Bhatia
- Bhawna Gupta
- Robert Saccone
- Rohit Wagle
- Kirsten Hildrum
- Edward Pring
- Zubair Nabi

Thank You! Questions?