

# Constant-Time Distributed Dominating Set Approximation

Kuhn & Wattenhofer

Sagar Mittal

November 1, 2007

# Outline

## Dominating Set Problem

Centralized algorithm

## Linear Programming Formulation

ILP approximation through LP rounding

## Approximating the Linear Program with $\Delta$

Analysis

## Approximating the Linear Program without $\Delta$

Analysis

# Outline

## Dominating Set Problem

Centralized algorithm

## Linear Programming Formulation

ILP approximation through LP rounding

## Approximating the Linear Program with $\Delta$

Analysis

## Approximating the Linear Program without $\Delta$

Analysis

# Dominating Set

- ▶ In a graph  $G = (V, E)$ , select a set of nodes  $D \subset V$  such that every node is either in  $D$  or adjacent to a node in  $D$ .
- ▶ Used for routing in a wireless peer-to-peer network; nodes only communicate with the closest member of the dominating set, which is responsible for routing messages.
- ▶ Since finding a set  $D$  with minimum size (MDS) is NP-hard, we are interested in finding reasonable approximations.

# Centralized Greedy Algorithm

- ▶ Repeatedly select the node that covers the most uncovered nodes until all nodes are covered.
- ▶  $O(\log \Delta)$  approximation, where  $\Delta$  is the maximum degree.
- ▶ Proved to be best approximation possible unless  $P$  almost equals  $NP$  [Feige98].
- ▶ Forms basis of most approximation algorithms.

# Outline

Dominating Set Problem  
Centralized algorithm

Linear Programming Formulation  
ILP approximation through LP rounding

Approximating the Linear Program with  $\Delta$   
Analysis

Approximating the Linear Program without  $\Delta$   
Analysis

## MDS as a Linear Program

$$\begin{aligned} \min \text{OPT} &= \sum_{v \in V} x_v \\ \text{s.t.} \quad &x_v + \sum_{u \in \Gamma(v)} x_u \geq 1 \quad \forall v \in V \\ &x_i \in \{0, 1\} \end{aligned}$$

Apply standard *relaxation* procedure to last constraint:

$$\begin{aligned} \min \text{LP} &= \sum_i x_i \\ \text{s.t.} \quad &x_v + \sum_{u \in \Gamma(v)} x_u \geq 1 \quad \forall v \in V \\ &x_i \geq 0 \end{aligned}$$

Since we've only increased the search space,  $\text{LP} \leq \text{OPT}$ .

# Linear Programming Duality

In general, a linear program can be expressed in matrix form:

$$\begin{aligned} \max \text{ LP} &= c^T x \\ \text{s.t.} & Ax \leq b \\ & x \geq 0 \end{aligned}$$

The *dual* has the same optimal solution (by strong duality):

$$\begin{aligned} \min \text{ DLP} &= b^T y \\ \text{s.t.} & A^T y \geq c \\ & y \geq 0 \end{aligned}$$

By weak duality, any feasible solution to the dual is smaller than LP.

# Linear Programming Duality

Write the MDS LP relaxation in matrix form, with adjacency matrix  $N$ :

$$\begin{aligned} \min \text{ LP} &= \sum x_i \\ \text{s.t.} & N \cdot x \geq 1 \\ & x \geq 0 \end{aligned}$$

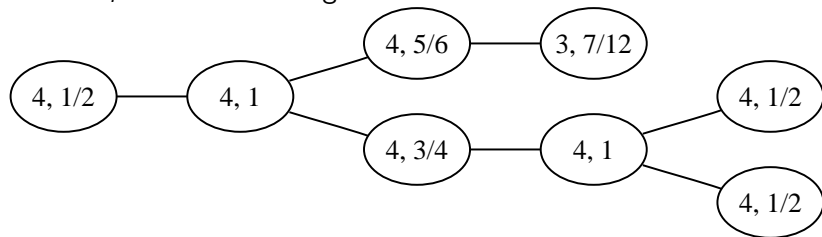
$$\begin{aligned} \max \text{ DLP} &= \sum y_i \\ \text{s.t.} & N \cdot y \leq 1 \\ & y \geq 0 \end{aligned}$$

## A feasible solution to the dual problem

$$y_i = \frac{1}{\delta_i^{(1)} + 1}$$

$$\delta_i^{(1)} = \max_{u \in N_i} d(u)$$

where  $N_i$  is the closed neighborhood of vertex  $i$ .



So,  $\sum_{i=1}^n \frac{1}{\delta_i^{(1)} + 1} \leq \text{LP} \leq \text{OPT}$ . This will be used to show our approximation bound.

## Convert feasible LP solution to dominating set

Intuitively, the components of  $x$  can be randomly rounded to obtain an approximation with expected cost  $O(LP)$ .

**Require:**  $x$  is a feasible LP solution.

**Ensure:**  $x_{DS}$  is a feasible ILP solution (dominating set).

calculate  $\delta_i^{(2)} = \max_{j \in N_i} \delta_j^{(1)}$

$p_i = \min\{1, x_i \log(\delta_i^{(2)} + 1)\}$

$x_{DS,i} = \begin{cases} 1 & \text{with probability } p_i \\ 0 & \text{otherwise} \end{cases}$  {Dominator selection}

send  $x_{DS,i}$  to all neighbors

**if**  $x_{DS,j} = 0 \forall$  neighbors  $j$  **then**

$x_{DS,i} = 1$  {Feasibility guarantee}

**end if**

## Approximation ratio of the LP-derived ILP solution

Let  $x$  be an  $\alpha$ -approximation for LP, and  $X$  be a random variable indicating the number of nodes chosen in the dominator selection step.

$$\begin{aligned}\sum_i x_i &= \alpha \sum_i x_i^* = \alpha(\text{LP}) \\ \mathbb{E}[X] &= \sum_{i=1}^n p_i \leq \sum_{i=1}^n x_i \log(\delta_i^{(2)} + 1) \\ &\leq \log(\Delta + 1) \sum_{i=1}^n x_i \\ &\leq \alpha \log(\Delta + 1)(\text{LP}) \\ &\leq \alpha \log(\Delta + 1)(\text{OPT})\end{aligned}$$

## Approximation ratio of the LP-derived ILP solution (2)

Now examine  $Y$ , the number of nodes chosen in feasibility guarantee step and  $q_i$ :

$$\begin{aligned}q_i &= \prod_{j \in N_i} (1 - p_j) \leq \prod_{j \in N_i} (1 - x_j \log(\delta_i^{(1)} + 1)) \\&\leq \left(1 - \frac{\sum_{j \in N_i} x_j \log(\delta_i^{(1)} + 1)}{\delta_i + 1}\right)^{\delta_i + 1}, \text{ since } \prod x_i \leq \left(\frac{\sum x}{|x|}\right)^{|x|} \\&\leq \left(1 - \frac{\log(\delta_i^{(1)} + 1)}{\delta_i + 1}\right)^{\delta_i + 1}, \text{ since } \sum x_j > 1 \\&\leq e^{-\log(\delta_i^{(1)} + 1)}, \text{ since } 1 - x \leq e^{-x} \\&= \frac{1}{\delta_i^{(1)} + 1} \\E[Y] &= \sum_{i=1}^n q_i \leq (\text{OPT})\end{aligned}$$

## Approximation ratio of the LP-derived ILP solution (3)

So, given  $x$ , an  $\alpha$ -approximation of LP, we can find a  $(1 + \alpha \log(\Delta + 1))$ -approximation of the minimum dominating set OPT.

# Outline

Dominating Set Problem  
Centralized algorithm

Linear Programming Formulation  
ILP approximation through LP rounding

**Approximating the Linear Program with  $\Delta$   
Analysis**

Approximating the Linear Program without  $\Delta$   
Analysis

# Motivation

- ▶ The LP formulation of MDS is not useful in a centralized setting, as using the centralized greedy algorithm is much faster.
- ▶ We find a  $O(k\Delta^{2/k})$  approximation to the LP in  $O(k^2)$  time.
- ▶ This yields a  $O(k\Delta^{2/k} \log(\Delta))$  approximation of MDS.

## LP approximation with $\Delta$ known

```
 $x_i = 0, \text{color}_i = \text{'white'}$   
for  $l = k - 1, k - 2, \dots, 0$  do  
  for  $m = k - 1, k - 2, \dots, 0$  do  
    send  $\text{color}_i$  to all neighbors  
     $\tilde{\delta}(v_i) = |\{j \in N_i \mid \text{color}_j = \text{'white'}\}|$   
    if  $\tilde{\delta}(v_i) \geq (\Delta + 1)^{l/k}$  then  
       $x_i = \max \left\{ x_i, \frac{1}{(\Delta + 1)^{m/k}} \right\}$   
    end if  
    send  $x_i$  to all neighbors  
    if  $\sum_{j \in N_i} x_j \geq 1$  then  
       $\text{color}_i = \text{'gray'}$   
    end if  
  end for  
end for
```

## Algorithm overview

- ▶ Each node gradually increases its  $x_i$ , until the constraint associated with it is tight.
- ▶ Nodes with large degrees increase their  $x_i$ 's first.
- ▶ Each iteration of the outer loop reduces the max dynamic degree  $\tilde{\delta}$  by a factor of  $(\Delta + 1)^{1/k}$ .
- ▶ Each iteration of the inner loop reduces the number of active nodes of max dynamic degree.
- ▶ The amount that the  $x_i$ 's increase is bounded with a cost accounting method.

## Outer loop invariant: maximum dynamic degree

At the beginning of the outer loops, the following invariant holds:

$$\tilde{\delta}(v_i) \leq (\Delta + 1)^{(l+1)/k}$$

For the first iteration ( $l = k - 1$ ), this is trivial. For subsequent iterations, in the last step of the inner loop

**if**  $\tilde{\delta}(v_i) \geq (\Delta + 1)^{l/k}$  **then**  
     $x_i = \max \left\{ x_i, \frac{1}{(\Delta+1)^{m/k}} \right\} = 1$   
**end if**

since  $m = 0$ . So nodes with dynamic degree at least  $(\Delta + 1)^{l/k}$  must be covered at the end of each outer loop iteration.

## Inner loop invariant: number of active nodes

- ▶ In each outer loop iteration, participating nodes must have

$$\tilde{\delta}(v_i) > (\Delta + 1)^{l/k}.$$

Such nodes are called *active*.

- ▶ Define  $a(v_i) = |\{u \in N_i \mid u \text{ is active}\}|$ .
- ▶ In the inner loop we have the invariant

$$a(v_i) \leq (\Delta + 1)^{(m+1)/k}.$$

## Inner loop invariant: number of active nodes

$$a(v_i) \leq (\Delta + 1)^{(m+1)/k}$$

For  $m = k - 1$ , this is trivial. Recall that in the inner loop,

**if**  $v_i$  is active **then**

$$x_i = \max \left\{ x_i, \frac{1}{(\Delta+1)^{m/k}} \right\} \geq \frac{1}{(\Delta+1)^{m/k}}$$

**end if**

If  $a(v_i) \geq (\Delta + 1)^{m/k}$ , then

$$\begin{aligned} \sum_{j \in N_i} x_j &\geq (\Delta + 1)^{m/k} \frac{1}{(\Delta + 1)^{m/k}} \\ &= 1 \end{aligned}$$

So any such  $v_i$  would become covered.

# Cost accounting

- ▶ Define a variable  $z_i$  for each node
  - ▶ set to 0 at the beginning of each outer loop.
  - ▶ Whenever a  $x_i$  is increased, distribute the increase equally among all  $z_j$ 's in  $N_i$ .
- ▶ We will show that  $z_i \leq \frac{1}{(\Delta+1)^{(l-1)/k}}$  for all  $l, i$ .
- ▶ With our other invariants, this will allow us to bound LP.

## Phase one: $v_i$ is white

- ▶ We break the inner loop into two phases: the iterations where  $v_i$  is white, and where it turns gray.
- ▶ When  $v_i$  is white,  $\sum_{j \in N_i} x_j < 1$ . So all increases in  $x_i$ 's must be distributed among  $(\Delta + 1)^{1/k}$  nodes.

$$z_i < \frac{\sum_{j \in N_i} x_j}{(\Delta + 1)^{1/k}} \leq \frac{1}{(\Delta + 1)^{1/k}}$$

## Phase two: $v_i$ turns gray

- ▶ Some  $v_j \in N_i$  have increased their  $x_j$  values from  $\frac{1}{(\Delta+1)^{(m+1)/k}}$  to  $\frac{1}{(\Delta+1)^{m/k}}$ .
- ▶ This change must be distributed over  $(\Delta + 1)^{l/k}$  vertices.
- ▶ At most  $a(v_i)$  such increases can occur.

$$z_i \leq \frac{\frac{1}{(\Delta+1)^{m/k}} - \frac{1}{(\Delta+1)^{(m+1)/k}}}{(\Delta + 1)^{l/k}} a(v_i)$$

where  $a(v_i) \leq (\Delta + 1)^{(m+1)/k}$ .

$$z_i \leq \frac{(\Delta + 1)^{1/k} - 1}{(\Delta + 1)^{l/k}}$$

## Total per-node, per-iteration cost

Adding the costs from phase one and phase two, we find

$$\begin{aligned} z_i &\leq \frac{1}{(\Delta + 1)^{l/k}} + \frac{(\Delta + 1)^{1/k} - 1}{(\Delta + 1)^{l/k}} \\ &= \frac{1}{(\Delta + 1)^{\frac{l-1}{k}}} \end{aligned}$$

# Total cost

Consider the total  $z_i$  seen by any node during an iteration.

- ▶  $\tilde{\delta}(v_i) \leq (\Delta + 1)^{(l+1)/k}$
- ▶ Only white nodes increase their  $z_i$ .
- ▶ So the total  $z_i$  increase seen by each node is at most

$$\sum_{j \in N_i} z_j \leq (\Delta + 1)^{(l+1)/k} \cdot (\Delta + 1)^{-(l-1)/k} = (\Delta + 1)^{2/k}$$

- ▶ Due to this,  $y_i = z_i / (\Delta + 1)^{2/k}$  is a feasible solution for the DLP!
- ▶ So, the  $\sum_i z_i \leq (\Delta + 1)^{2/k}(\text{OPT})$
- ▶ Since there are  $k$  iterations

$$\sum x = k \sum z_i \leq k(\Delta + 1)^{2/k}(\text{OPT})$$

# Analysis summary

Before rounding:

- ▶ Time complexity:  $O(k^2)$ .
- ▶ Approximation ratio:  $O(k\Delta^{2/k})$

After rounding:

- ▶ Time complexity:  $O(k^2)$  (rounding step is constant time).
- ▶ Approximation ratio:  $O(k\Delta^{2/k} \log \Delta)$

However, this requires knowledge of  $\Delta$ .

# Outline

Dominating Set Problem  
Centralized algorithm

Linear Programming Formulation  
ILP approximation through LP rounding

Approximating the Linear Program with  $\Delta$   
Analysis

Approximating the Linear Program without  $\Delta$   
Analysis

## Algorithm, outer loop

$x_i = 0$

calculate  $\delta_i^{(2)}$

$\gamma^{(2)}(v_i) = \delta_i^{(2)} + 1; \tilde{\delta}(v_i) = \delta_i + 1$

**for**  $l = k - 1, k - 2, \dots, 0$  **do**

$\{\tilde{\delta}(v_i) \leq (\Delta + 1)^{(l+1)/k}, z_i = 0\}$

**for**  $m = k - 1, k - 2, \dots, 0$  **do**

inner loop steps: reduce number of active nodes

**end for**

$\{z_i \leq (1 + (\Delta + 1)^{1/k})/\gamma^{(1)}(v_i)^{l/(l+1)}\}$

send  $\tilde{\delta}(v_i)$  to all neighbors

calculate and send to all neighbors  $\gamma^{(1)}(v_i) = \max_{j \in N_i} \tilde{d}(v_j)$

$\gamma^{(2)}(v_i) = \max_{j \in N_i} \gamma^{(1)}(v_j)$

**end for**

## Algorithm, inner loop

**if**  $\tilde{\delta}(v_i) \geq \gamma^{(2)}(v_i)^{\frac{1}{l+1}}$  **then**  
    mark self as active and notify neighbors  
**end if**  
 $a(v_i) = \begin{cases} 0 & \text{color}_i = \text{'gray' } \\ |\{j \in N_i \mid v_j \text{ is active}\}| & \text{otherwise} \end{cases}$   
send  $a(v_i)$  to all neighbors  
 $a^{(1)}(v_i) = \max_{j \in N_i} a(v_j)$   
**if**  $\tilde{\delta}(v_i) \geq \gamma^{(2)}(v_j)^{\frac{1}{l+1}}$  **then**  
     $x_i = \max \left\{ x_i, a^{(1)}(v_i)^{-\frac{m}{m+1}} \right\}$   
**end if**  
send  $x_i$  to all neighbors  
**if**  $\sum_{j \in N_i} x_j \geq 1$  **then**  
     $\text{color}_i = \text{'gray'}$   
**end if**  
send  $\text{color}_i$  to all neighbors  
 $\tilde{\delta}(v_i) = |\{j \in N_i \mid \text{color}_j = \text{'white'}\}|$

## Outer loop invariant: dynamic degree

At the beginning of the outer loop, the following invariant holds:

$$\tilde{\delta}(v_i) \leq (\Delta + 1)^{(l+1)/k}$$

- ▶ Holds trivially for  $l = k - 1$ .
- ▶ Use induction; assume true for  $l - 1$ .
- ▶ Notice that in the last step of the inner loop,

**if**  $\tilde{\delta}(v_i) \geq \gamma^{(2)}(v_i)^{\frac{l}{l+1}}$  **then**

$x_i = 1$

**end if**

- ▶ By inductive hypothesis,  $\gamma^{(2)}(v_i) \leq (\Delta + 1)^{(l+1)/k}$ .  
Then  $\gamma^{(2)}(v_i)^{\frac{l}{l+1}} \leq (\Delta + 1)^{l/k}$ .
- ▶ So, all nodes with  $\tilde{\delta}(v_i) \geq (\Delta + 1)^{l/k}$  become covered.

## Inner loop invariant: active nodes

$$a(v_i) \leq (\Delta + 1)^{(m+1)/k}$$

- ▶ Holds trivially for  $m = k - 1$ .
- ▶ Again, use induction
- ▶ Each active neighbor assigns

$$\begin{aligned}x_j &\geq \frac{1}{a^{(1)}(v_j)^{\frac{m}{m+1}}} \\ &\geq \frac{1}{(\Delta + 1)^{\frac{m+1}{k} \cdot \frac{m}{m+1}}} \\ &= \frac{1}{(\Delta + 1)^{m/k}}\end{aligned}$$

- ▶ So, nodes with  $a(v_i) > (\Delta + 1)^{m/k}$  become covered.

## Cost accounting: phase one

- ▶ In analogy to the first analysis, we consider the cost in two phases: when the node  $i$  is white and when it turns gray.
- ▶ Again, during phase one  $\sum_{j \in N_i} x_j < 1$ .

$$\begin{aligned} z_i^1 &\leq \sum_{j \in N_i} \frac{x_j}{\gamma^{(2)}(v_j)^{\frac{1}{l+1}}} \\ &\leq \frac{1}{\gamma^{(1)}(v_i)^{\frac{1}{l+1}}} \end{aligned}$$

## Cost accounting: phase two

There are up to  $a(v_i)$  active members contributing increases in  $x_i$  values, distributed across  $\gamma^{(1)}(v_i)^{l/(l+1)}$  vertices so

$$\begin{aligned} z_i^2 &\leq a(v_i) \cdot \frac{1}{a^{(1)}(v_i)^{\frac{m}{m+1}}} \frac{1}{\gamma^{(1)}(v_i)^{\frac{l}{l+1}}} \\ &= \frac{a(v_i)^{\frac{1}{m+1}}}{\gamma^{(1)}(v_i)^{\frac{l}{l+1}}} \\ &\leq \frac{(\Delta + 1)^{1/k}}{\gamma^{(1)}(v_i)^{\frac{l}{l+1}}} \end{aligned}$$

So, in total

$$z_i \leq \frac{(\Delta + 1)^{1/k} + 1}{\gamma^{(1)}(v_i)^{\frac{l}{l+1}}}$$

## Total cost

Again, in analogy to the first analysis we consider the total cost of  $z_j$ 's in the closed neighborhood of  $i$ :

$$\begin{aligned}\sum_{j \in N_i} z_j &\leq \frac{1 + (\Delta + 1)^{1/k}}{\gamma^{(1)}(v_i)^{\frac{l}{l+1}}} \tilde{\delta}(v_i) \\ &\leq \frac{1 + (\Delta + 1)^{1/k}}{\gamma^{(1)}(v_i)^{\frac{l}{l+1}}} \gamma^{(1)}(v_i) \\ &\leq \left(1 + (\Delta + 1)^{1/k}\right) \gamma^{(1)}(v_i)^{1/(l+1)} \\ &\leq \left(1 + (\Delta + 1)^{1/k}\right) \left((\Delta + 1)^{(l+1)/k}\right)^{1/(l+1)} \\ &= (\Delta + 1)^{1/k} + (\Delta + 1)^{2/k}\end{aligned}$$

## Approximation ratio for LP

Now set

$$y_i = \frac{z_i}{(\Delta + 1)^{1/k} + (\Delta + 1)^{2/k}}$$

to find a feasible solution to the dual linear program. Then

$$\sum_i x_i = k \sum z_i \leq k \left( (\Delta + 1)^{1/k} + (\Delta + 1)^{2/k} \right) \text{ (LP)}$$

After rounding, this is an overall approximation of  $O(k(\Delta + 1)^{2/k} \log(\Delta))$ . The algorithm clearly runs in  $O(k^2)$  rounds.

Questions?