A Primal-Dual Resource Augmentation Analysis of a Constant Approximate Algorithm for Stable Coalitions in a Cluster

Nedialko B. Dimitrov and Indrajit Roy

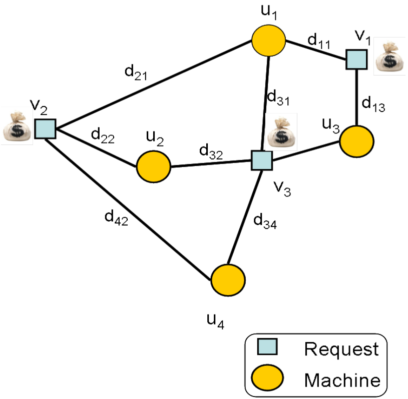
Department of Computer Science
University of Texas at Austin

Overview

- Cluster Profit Problem
- Contributions
- Algorithm
- Clarifying Examples
- Analysis
- Conclusion

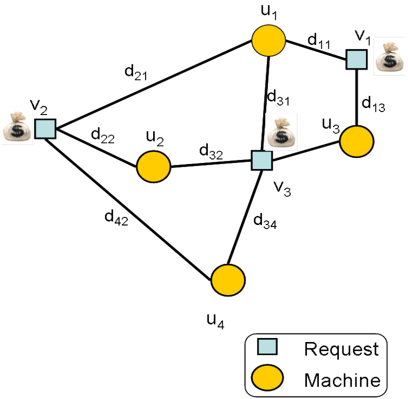
Cluster Profit Problem

- Highly parallelizable requests are present on some network nodes
- Each request is associated with a tuple (g, r).
- The requester is willing to pay g to each machine that executes the request.



Cluster Profit Problem

- Machines must pay the request processing cost, r.
- Machines also pay the connection cost d_v to serve a request v.



Objective

Find a profit maximizing assignment of machines to requests, such that each machine works on at most one request.

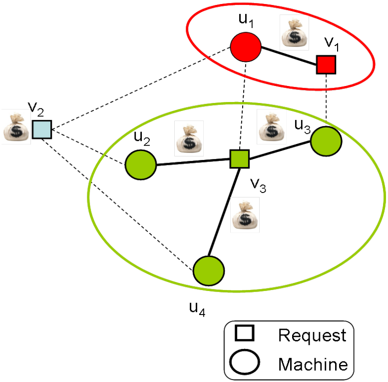


Figure: A possible assignment of machines to requests.

Application: Facility Location Variant

- Cluster Profit Problem (CPP) has more general interpretations
- The problem can be viewed as a variant of the Facility Location Problem
- Each facility is associated with a tuple (g, r)
- The number g signifies the quality of the facility
- Each customer is willing to pay g for being connected to a facility.
- Note that the standard Facility Location Problem has no quality values

Application: Coalition Formation

- CPP can also be viewed as an instance of a coalition formation problem.
- We would like to partition a set of agents and assign each resulting subset to some task
- Ensure that the assignment maximizes some utility function
- Consider tasks as the requests, the agents as the machines, and the utility function as the profit
- We would like the coalition to be resilient to selfish behavior of the system's participants !

Cluster Profit Problem

- Cluster Profit Problem is NP-hard
- CPP can be viewed as a profit maximizing variant of facility location problem
- However facility location algorithms provide no guarantees on the profit
  - There could be assignments with high costs that yield high profits
  - So a facility location algorithm that finds lowest cost assignment provides no guarantees on the maximum profit earned

Cluster Profit Problem

- We analyze this problem under **Resource Augmentation (RA)** [3]
- In resource augmentation a given algorithm competes with constant factor resource advantage over optimal algorithm
  - LRU caching analysis gives constant factor larger cache size [4]
- Our algorithm (ALG) operates on an instance where distances are shorter by a constant factor

CPP: Example Scenario

- Consider a large company with millions of machines which the company does not own.
- The company suggests coalitions to the participating machines.
- If the machines follow the company's suggestion, they use the company's proprietary bandwidth to communicate.
- If they deviate, the machines must use some other method of communication and incur slightly larger communication costs.
- Resource augmentation captures this increase in communication costs

Contributions

Result

We show ALG gives a constant RA approximation.

- ALG gains at least a constant fraction of the profit of an optimal algorithm
- ALG is given the resource advantage to operate on the same graph instance with edge lengths decreased by a constant factor

Contributions

Result

We show ALG gives a constant RA approximation.

- ALG gains at least a constant fraction of the profit of an optimal algorithm
- ALG is given the resource advantage to operate on the same graph instance with edge lengths decreased by a constant factor

Result

We show ALG is constant RA stable to group deviations.

- Every subset of the machines gains at least a constant fraction of the profit they would gain under an optimal deviation
- Assume deviating increases communication costs by a constant factor

Overview

Algorithm

- Greedily find a subset of machines and a request such that the corresponding assignment gives the maximum profit density.

Algorithm

- Greedily find a subset of machines and a request such that the corresponding assignment gives the maximum profit density.

Definition

- Let $S = (v_S, \mathcal{A}_S)$, denote a tuple of a request, v_S, and a set of machines, $\mathcal{A}_S$
- The profit of S is $p_S = (|S| \cdot g_{v_S} - \sum_{u \in S} d_{u,v_S} - r_{v_S})$.
- The profit density of S is $p_S/|S|$.

Algorithm

- Greedily find a subset of machines and a request such that the corresponding assignment gives the maximum profit density.

Definition

- Let $S = (v_S, \mathcal{A}_S)$, denote a tuple of a request, v_S, and a set of machines, $\mathcal{A}_S$
- The profit of S is $p_S = (|S| \cdot g_{v_S} - \sum_{u \in S} d_{u,v_S} - r_{v_S})$.
- The profit density of S is $p_S/|S|$.

- Find $S^* = (v_{S^*}, \mathcal{A}_{S^*})$ with the highest profit density and assigns the machines in $\mathcal{A}_{S^*}$ to work on the request v_{S^*}.

Algorithm

Primal

$$\max \quad \sum_{S \in \mathcal{T}} p_S \cdot x_S \qquad \text{(P)}$$

s.t.

$$\sum_{S \in \mathcal{T} : u \in S} x_S \leq 1 \qquad \forall u \in \mathcal{U} \qquad (1)$$

$$x_S \in \{0, 1\} \qquad \forall S \in \mathcal{T} \qquad (2)$$

- Let x_S be a binary variable denoting whether S has been picked

Algorithm

Dual

$$\min \quad \sum_{u \in \mathcal{U}} \alpha_u \tag{D}$$

s.t.

$$\sum_{u \in \mathcal{U}} \max(0, g_v - \alpha_u - d_{u,v}) \leq r_v \qquad \forall v \in \mathcal{V}. \tag{3}$$

$$\alpha_u \geq 0 \qquad \forall u \in \mathcal{U}$$

- Interpret α_u as the profit made by the machine u after paying its distance costs and its share of the resource cost

Algorithm

Initialization Set each α_u to $g_{\max}$.

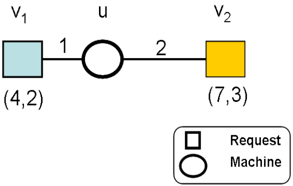
Loop Decrease α_u for each machine in $u \in \mathcal{U}$ at a uniform rate until one of the following happens:

- If α_u becomes zero or $\mathcal{U}$ is empty, then we stop the algorithm.
- If some inequalities of type become tight, pick one arbitrarily. Say we picked the inequality corresponding to v. Assign all machines in $\mathcal{A} = \{u \mid g_v - \alpha_u - d_{u,v} \geq 0, u \in \mathcal{U}\}$ to request v. Set $r_v = 0$ and $\mathcal{U} = \mathcal{U} - \mathcal{A}$ and proceed with the uniform decrease once again.

Example

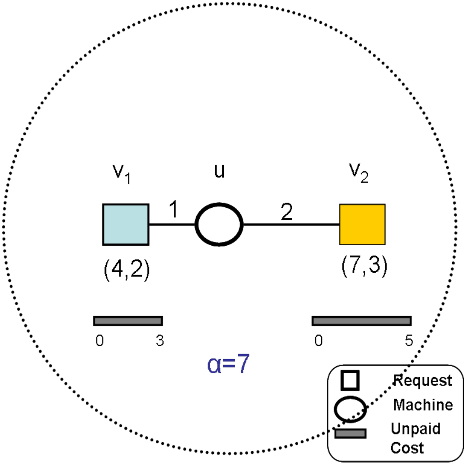
- A single machine u and two requests v_1, v_2

Request	Payment	Cost	Distance from u
v_1	4	2	1
v_2	7	3	2

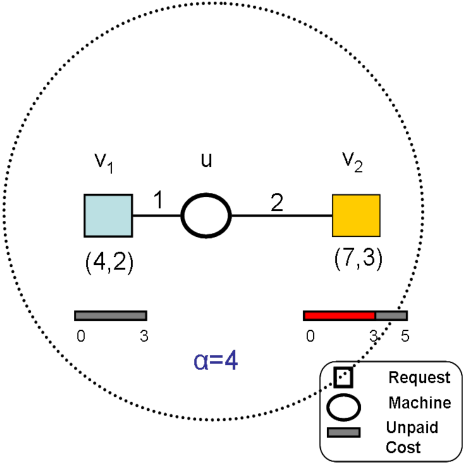


Algorithm: Example

- Initialize a ball of radius 7 around u (i.e. $\alpha = g_{\max} = 7$)

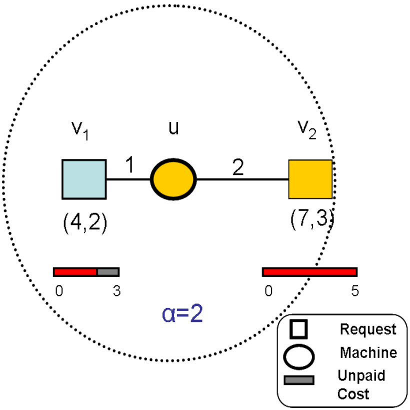


- At $\alpha = 4$, 60% of the costs involved with servicing v_2 has been paid



Algorithm: Example

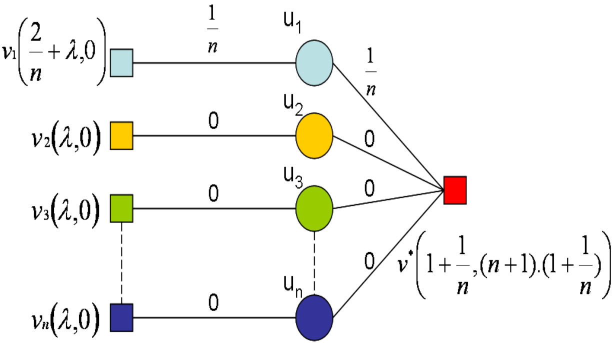
- At $\alpha = 2$ the u has paid all the costs of v_2 and hence gets assigned.
- Total profit is 2
- Assigning u to v_1 would give a profit of 1.



Overview

- Cluster Profit Problem
- Contributions
- Algorithm
- **Clarifying Examples**
- Analysis
- Conclusion

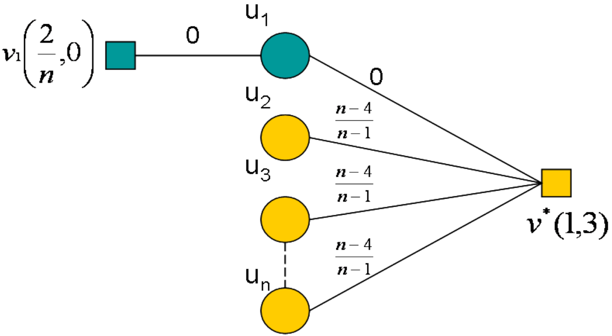
Clarifying Example I



- ALG assigns machine u_i to request v_i. Total profit is $\frac{1}{n} + n \cdot \lambda$ $(\lambda \to 0)$.
- If all machines are assigned to request v^*, then the total profit is 1
- In this case ALG has an approximation ratio of $\Theta(n)$

Constraint I

- Observation: v^* has high resource cost and would require almost all machines to work on it
- Need to constrain the values of resource cost
- Assume requester pays at least the resource cost if ω machines work on the request ($r_v \leq \omega g_v$)



- ALG assigns u_1 to request v_1 and remaining machines to v^*. Total profit is $\frac{2}{n}$.
- If all machines are assigned to v^* then total profit is 1
- Again ALG has an approximation ratio of $\Theta(n)$

Constraint II

- Observation: If distances are θ-factor shorter then ALG can recover $\frac{(\theta-1)(n-4)}{\theta} + \frac{2}{n}$ profit in this instance
- Previous examples motivate the need for another constraint
- If ALG operates on a graph G with distance metric d, we compare with OPT operating on a graph G' with distance metric $\theta \cdot d$
- These constraints help us prove meaningful bounds

Overview

- Cluster Profit Problem
- Contributions
- Algorithm
- Clarifying Examples
- **Analysis**
- Conclusion

Analysis

- Our analysis is based on the technique of *factor revealing* LP as used by Jain et. al. [1]

Definition

- Let P be the problem, π the corresponding variables and $\mathrm{OPTVAL}_{\mathrm{P}}$ its optimal values
- Let D be the dual problem, and α the variables
- Let P^{L} and D^{L} denote the primal and relaxed dual pair for the resource hampered instance

Analysis

General approach of primal-dual algorithms:

- Find real-valued dual variables α' feasible in D
- Find corresponding integer-valued primal variables π' feasible in P s.t. $\mathrm{D}(\alpha') \leq \gamma \cdot \mathrm{P}(\pi')$
- This gives γ-approximate algorithm since $\mathrm{P}(\pi') \leq \mathrm{OPTVAL}_{\mathrm{P}} \leq \mathrm{D}(\alpha') \leq \gamma \cdot \mathrm{P}(\pi')$

Analysis

Using factor-revealing LP technique:

- We find an π' feasible in P and α' which is infeasible in D s.t. $\mathrm{D}(\alpha') = \mathrm{P}(\pi')$.
- We attempt to find a γ, such that $\gamma\alpha'$ is feasible in D.
- We then have $\mathrm{D}(\gamma\alpha') = \gamma\mathrm{D}(\alpha') = \gamma\mathrm{P}(\pi')$,

Analysis

Steps for the resource augmented version:

- Find γ such that $\gamma\alpha'$ is feasible in the optimal algorithm's $\mathrm{D^L}$.
- We can then show
  $\mathrm{OPTVAL_{PL}} \leq \mathrm{D^L}(\gamma\alpha') = \gamma\mathrm{D^L}(\alpha') = \gamma\mathrm{D}(\alpha') = \gamma\mathrm{P}(\pi')$

Analysis

Theorem

Let ALG finish with variables α. Then, there exists a constant γ, only dependent on the constants ω and θ, such that $\gamma\alpha$ is feasible in D^{L}.

Conclusion

- We formalize the Cluster Profit Problem that has applications as profit maximizing facility location and coalition formation problem.
- We give an algorithm that gives a constant RA approximation.
- We show our algorithm is constant RA stable to group deviations.

References

K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. V. Vazirani.
Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp.
J. ACM, 50(6):795–824, 2003.

M. Pál and E. Tardos.
Group strategy proof mechanisms via primal-dual algorithms.
In *Proceedings of the 44th annual IEEE Symposium on Foundations of Computer Science*, pages 584–593, October 2003.

Cynthia A. Phillips, Cliff Stein, Eric Torng, and Joel Wein.
Optimal time-critical scheduling via resource augmentation (extended abstract).
In *STOC '97: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 140–149, 1997.

Daniel D. Sleator and Robert E. Tarjan.
Amortized efficiency of list update and paging rules.
Commun. ACM, 28(2):202–208, 1985.

Thank You !

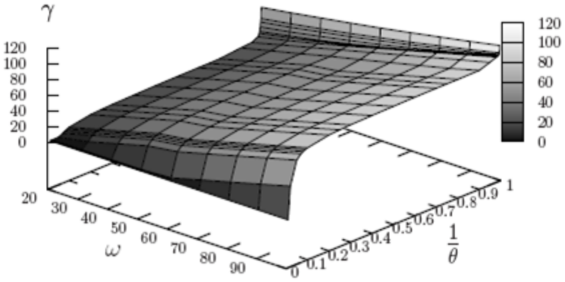


Figure: The exact competitiveness factor, γ, as a function of the parameters ω and θ. In the experiments, ω was varied between 20 and 100, while θ was varied between 0 and 1.

- As θ goes to zero, γ approaches one.
- As θ goes to one, γ approaches infinity.
- For the mid-ranges of θ, γ varies almost linearly with ω.

Extra Slides: Analysis

$z_k = \min_\mu \max_{g,r,\alpha,d} \mu$
s.t.

$$kg - r - \theta \sum_{i=1}^{k} d_i \leq \mu \sum_{i=1}^{k} \alpha_i$$

$$\alpha_i \leq \alpha_{i-1} \qquad \forall i \in \{2, ..., k\}$$

$$\alpha_i \leq \alpha_j + d_i + d_j \qquad \forall i, j \in \{1, ..., k\}$$

$$\theta \mathbf{d_i} \leq \mathbf{g} \qquad \forall i \in \{1, ..., k\}$$

$$\mathbf{r} \leq \omega \cdot \mathbf{g}$$

$$\sum_{j=i}^{k} \max(0, g - \alpha_i - d_j) \leq r \qquad \forall i \in \{1, ..., k\}$$

$$\alpha_i, d_i, r, g \geq 0 \qquad \forall i \in \{1, ..., k\}$$

- z_k represents the worst approximation factor given that at most k machines contribute to the resource cost of any request

Extra Slides: RA Stability

Definition

A payoff vector π is γ-*RA stable* under a transformation L if it satisfies the following inequalities:

Stability inequalities: $\sum_{a\in\mathcal{A}} \pi_a \geq \frac{1}{\gamma} V^{\mathrm{L}(x)}(\mathcal{A})$ for all $\mathcal{A} \subseteq \mathcal{P}$

Conservation inequality: $\sum_{a\in\mathcal{P}} \pi_a \leq V^x(\mathcal{P})$.

- $V(\mathcal{A})$ is the maximum amount of profit the machines in $\mathcal{A}$ can achieve working alone.

Lemma (Main Lemma)

If a vector α is feasible in the non-integer-constrained relaxed dual for $V^{\mathrm{L}}(\mathcal{P})$, it is also feasible in the non-integer-constrained relaxed dual for $V^{\mathrm{L}}(\mathcal{A})$ for any $\mathcal{A} \subseteq \mathcal{P}$.