Pricing for Revenue Maximization in General Scenarios and in Networks

Patrick Briest Piotr Krysta

Dept. of Computer Science
University of Dortmund
Germany

03-10-2006

Websites gather data about consumer preferences / budgets.

Computation of profit maximizing prices.

Different approaches taken to model markets. Here:

- Single-Minded Unlimited-Supply Pricing:
  - *single-minded* customers, each interested in a single set of products,
  - *unlimited supply*, i.e., no production constraints.
  - Customer buys if the sum of prices is below her budget.
- Unit-Demand Pricing:
  - *unit-demand* customers, each buy a single product in a set of products,
  - *unlimited or limited supply*,
  - Customer buys only products with prices below their budgets.

1. Introduction
2. Single-Minded Unlimited-Supply Pricing
   - Hardness Results
   - Approximation Algorithms
3. Unit-Demand Pricing
   - Hardness Results
   - Approximation Algorithms

Single-Minded Unlimited-Supply Pricing

Single-Minded Unlimited-Supply Pricing *(SUSP)*

Given products $\mathcal{U}$ and sets $\mathcal{S}$ with values $v(S)$ find prices p, such that

$$\sum_{S:\sum_{e\in S} p(e)\leq v(S)} \quad \sum_{e\in S} p(e) \quad \longrightarrow \max.$$

$\rightsquigarrow$ models pricing of direct connections in computer or transportation networks.

Pricing in Graphs*(G-SUSP)*

Given graph $G = (V, E)$ and paths $\mathcal{P}$, assign profit-maximizing prices p to edges.

First investigated by *Guruswami et al. (2005)*. Recent inapproximability result due to *Demaine et al. (2006)*.

In general:

- $O(\log|\mathcal{U}| + \log|\mathcal{S}|)$-approximation
- inapproximable within $O(\log^{\delta}|\mathcal{U}|)$ for some $0 < \delta < 1$

With G being a line *(Highway Problem)*:

- poly-time algo for integral valuations of constant size
- pseudopolynomial time algo for paths of constant length

Q: Is there a poly-time algorithm for the Highway Problem?

First investigated by *Guruswami et al. (2005)*. Recent inapproximability result due to *Demaine et al. (2006)*.

In general:

- $O(\log |\mathcal{U}| + \log |\mathcal{S}|)$-approximation
- inapproximable within $O(\log^{\delta} |\mathcal{U}|)$ for some $0 < \delta < 1$

With G being a line *(Highway Problem)*:

- poly-time algo for integral valuations of constant size
- pseudopolynomial time algo for paths of constant length

Q: Is there a poly-time algorithm for the Highway Problem? No!

Hardness Results

The Highway Problem

Theorem

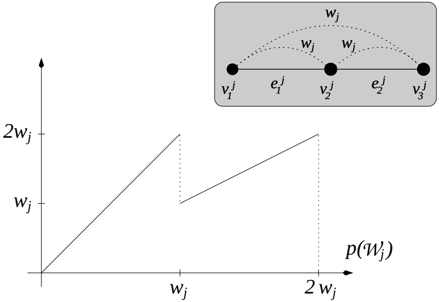
The Highway Problem is NP-hard.

Sketch of Proof: PARTITION problem:

Given positive weights $w_1, \ldots, w_n$, does there exist $S \subset \{1, \ldots, n\}$, such that

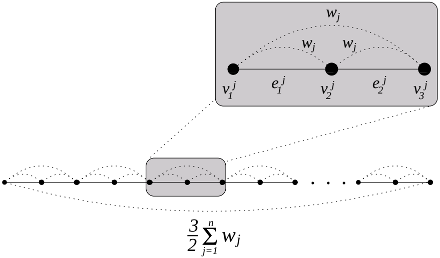
$$\sum_{j \in S} w_j = \sum_{j \notin S} w_j \text{ ?}$$

Design gadgets that capture the discrete nature of this problem.



Weight Gadgets

Maximum profit out of $\mathcal{W}_j$ is $2w_j$.
It is obtained iff $p(\mathcal{W}_j) = p(e_1^j) + p(e_2^j)$ is set to w_j or $2w_j$.



Maximum profit $\frac{7}{2}\sum_{j=1}^{n} w_j$ is obtained iff there exists $S \subset \{1, \ldots, n\}$ with $\sum_{j \in S} w_j = \sum_{j \notin S} w_j$. $\square$

The sets in this instance are *nested*, i.e.,

- $S \subseteq T$, $T \subseteq S$, or
- $S \cap T = \emptyset$.

Every instance of SUSP with nested sets can be viewed as an instance of the Highway Problem.

Dynamic programming / scaling:

Theorem

SUSP with nested sets allows an FPTAS.

G-SUSP

Inapproximability of Sparse Problem Instances

APX-hardness of G-SUSP due to *Guruswami et al. (2005)*.

Applications in realistic network settings often lead to sparse problem instances. Hardness of approximation still holds if:

- G has constant degree d
- paths have constant lengths $\leq \ell$
- at most a constant number B of paths per edge
- only constant height valuations

Theorem

G-SUSP on sparse instances is APX-hard.

Approximation Algorithms

Best ratio in the general case: $\log |\mathcal{U}| + \log |\mathcal{S}|$
Guruswami, Hartline, Karlin, Kempe, Kenyon, McSherry (2005)

Not approximable within $\log^{\delta} |\mathcal{U}|$ for some $0 < \delta < 1$.
Demaine, Feige, Hajiaghayi, Salavatipour (2006)

Can we do better on sparse problem instances, i.e., can we obtain approximation ratios depending on

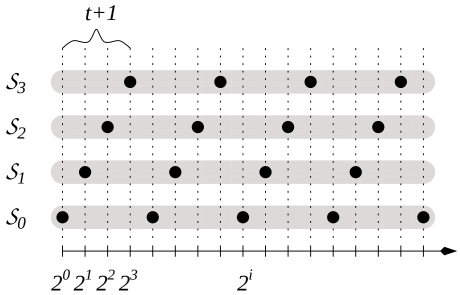
- ℓ, the maximum cardinality of any set $S \in \mathcal{S}$
- B, the maximum number of sets containing some product $e \in \mathcal{U}$

rather than $|\mathcal{U}|$ and $|\mathcal{S}|$?

An $O(\log \ell + \log B)$-Approximation

Let $\delta(S) = v(S)/|S|$ be *price per product* of set S.

1. Round all $\delta(S)$ to powers of 2. Let $\mathcal{S} = \mathcal{S}_0 \cup \ldots \cup \mathcal{S}_t$ where $t = \lceil \log \ell^2 B \rceil - 1$. In $\mathcal{S}_i$: $\delta(S) > \delta(T) \Rightarrow \delta(S)/\delta(T) \geq \ell^2 B$.



2. In each $\mathcal{S}_i$ select non-intersecting sets with maximum δ-value and compute optimal prices.

Analysis:

- $Opt(\mathcal{S}) \leq \sum_{i=1}^{t} Opt(\mathcal{S}_i)$
- Let $S \in \mathcal{S}_i$, $\mathcal{I}(S)$ intersecting sets with smaller δ-values:

$$v(S) \geq \sum_{T \in \mathcal{I}(S)} v(T)$$

- Let $\mathcal{S}_i^*$ be non-intersecting sets with max. δ as in the algo. Then

$$Opt(\mathcal{S}_i) \leq 2 \cdot Opt(\mathcal{S}_i^*),$$

and, since we compute $\max_i Opt(\mathcal{S}_i^*)$:

Theorem

The above algorithm has approximation ratio $O(\log \ell + \log B)$.

Upper bounding technique

We relate $Opt(\mathcal{S}_i^*)$ to $Opt(\mathcal{S}_i)$ by using as an upper bound

$$Opt(\mathcal{S}_i) \leq \sum_{S \in \mathcal{S}_i} v(S),$$

i.e., the sum of all valuations.

Using this upper bounding technique, no approximation ratio $o(\log B)$ can be achieved.

In many applications: $B >> \ell$.

Can we obtain ratios independent of B?

An $O(\ell^2)$-Approximation

Define *(smoothed)* s-SUSP by changing the objective to

$$\sum_{S \in \Lambda(p)} \sum_{e \in S} p(e),$$

where $\Lambda(p) = \{S \in \mathcal{S} \mid p(e) \leq \delta(S) \, \forall \, e \in S\}$.

We derive an $O(\ell)$-approximation for s-SUSP.

1. For every $e \in \mathcal{U}$ compute the optimal price $p^*(e)$ assuming all other prices were 0.
2. Resolve existing *conflicts*.

Set S is *conflicting*, if

$$\exists \, e, f \in S \; : \; p^*(e) \leq \delta(S) < p^*(f).$$

Upper bounding technique

1. For every $e \in \mathcal{U}$ compute the optimal price $p^*(e)$ assuming all other prices were 0.
2. Our upper bound: $Opt \leq \sum_{e \in \mathcal{U}} p^*(e)$

Summary (SUSP):

- Hardness results
  - NP-hardness of the Highway Problem
  - APX-hardness of G-SUSP for sparse instances
- Approximation Algorithms
  - $O(\log \ell + \log B)$-approximation ($\rightsquigarrow$ partitioning)
  - $O(\ell^2)$-approximation ($\rightsquigarrow$ conflict graph)

Unit-Demand Pricing

Unit-Demand Pricing *(UDP)*

Given products $\mathcal{U}$ and consumer samples $\mathcal{C}$ consisting of budgets $b(c,e) \in \mathbb{R}_0^+ \ \forall c \in \mathcal{C}, e \in \mathcal{U}$, and rankings $r_c : \mathcal{U} \rightarrow \{1, \ldots, |\mathcal{U}|\}$.

Unit-Demand Pricing *(UDP)*

Given products $\mathcal{U}$ and consumer samples $\mathcal{C}$ consisting of budgets $b(c,e) \in \mathbb{R}_0^+ \ \forall c \in \mathcal{C}, e \in \mathcal{U}$, and rankings $r_c : \mathcal{U} \to \{1, \ldots, |\mathcal{U}|\}$.

For prices $p : \mathcal{U} \to \mathbb{R}_0^+$:
$\mathcal{A}(p) = \{c \in \mathcal{C} \mid \exists e \in \mathcal{U} : p(e) \leq b(c,e)\}$ = consumers affording to buy any product.

Unit-Demand Pricing *(UDP)*

Given products $\mathcal{U}$ and consumer samples $\mathcal{C}$ consisting of budgets $b(c,e) \in \mathbb{R}_0^+ \ \forall c \in \mathcal{C}, e \in \mathcal{U}$, and rankings $r_c : \mathcal{U} \rightarrow \{1, \ldots, |\mathcal{U}|\}$.

For prices $p : \mathcal{U} \rightarrow \mathbb{R}_0^+$:
$\mathcal{A}(p) = \{c \in \mathcal{C} \mid \exists e \in \mathcal{U} : p(e) \leq b(c,e)\}$ = consumers affording to buy any product.

In no price ladder scenario (NPL) we find prices p that maximize:

- $\sum_{c \in \mathcal{A}(p)} \min\{p(e) \mid p(e) \leq b(c,e)\}$ (Udp-Min-Npl)

Unit-Demand Pricing *(UDP)*

Given products $\mathcal{U}$ and consumer samples $\mathcal{C}$ consisting of budgets $b(c,e) \in \mathbb{R}_0^+ \ \forall c \in \mathcal{C}, e \in \mathcal{U}$, and rankings $r_c : \mathcal{U} \to \{1, \ldots, |\mathcal{U}|\}$.

For prices $p : \mathcal{U} \to \mathbb{R}_0^+$:
$\mathcal{A}(p) = \{c \in \mathcal{C} \mid \exists e \in \mathcal{U} : p(e) \leq b(c,e)\}$ = consumers affording to buy any product.

In no price ladder scenario (NPL) we find prices p that maximize:

- $\sum_{c \in \mathcal{A}(p)} \min\{p(e) \mid p(e) \leq b(c,e)\}$ (Udp-Min-Npl)
- $\sum_{c \in \mathcal{A}(p)} \max\{p(e) \mid p(e) \leq b(c,e)\}$ (Udp-Max-Npl)

Unit-Demand Pricing *(UDP)*

Given products $\mathcal{U}$ and consumer samples $\mathcal{C}$ consisting of budgets $b(c,e) \in \mathbb{R}_0^+ \; \forall c \in \mathcal{C}, e \in \mathcal{U}$, and rankings $r_c : \mathcal{U} \to \{1, \ldots, |\mathcal{U}|\}$.

For prices $p : \mathcal{U} \to \mathbb{R}_0^+$:
$\mathcal{A}(p) = \{c \in \mathcal{C} \,|\, \exists e \in \mathcal{U} : p(e) \leq b(c,e)\}$ = consumers affording to buy any product.

In no price ladder scenario (NPL) we find prices p that maximize:

- $\sum_{c \in \mathcal{A}(p)} \min\{p(e) \,|\, p(e) \leq b(c,e)\}$ (UDP-MIN-NPL)
- $\sum_{c \in \mathcal{A}(p)} \max\{p(e) \,|\, p(e) \leq b(c,e)\}$ (UDP-MAX-NPL)
- $\sum_{c \in \mathcal{A}(p)} p\big(\text{argmin}\{r_c(e) \,|\, e : p(e) \leq b(c,e)\}\big)$ (UDP-RANK-NPL)

Unit-Demand Pricing *(UDP)*

Given products $\mathcal{U}$ and consumer samples $\mathcal{C}$ consisting of budgets $b(c, e) \in \mathbb{R}_0^+ \; \forall c \in \mathcal{C}, e \in \mathcal{U}$, and rankings $r_c : \mathcal{U} \to \{1, \ldots, |\mathcal{U}|\}$.

For prices $p : \mathcal{U} \to \mathbb{R}_0^+$:
$\mathcal{A}(p) = \{c \in \mathcal{C} \,|\, \exists e \in \mathcal{U} : p(e) \leq b(c, e)\}$ = consumers affording to buy any product.

In no price ladder scenario (NPL) we find prices p that maximize:

- $\sum_{c \in \mathcal{A}(p)} \min\{p(e) \,|\, p(e) \leq b(c, e)\}$ (Udp-Min-Npl)
- $\sum_{c \in \mathcal{A}(p)} \max\{p(e) \,|\, p(e) \leq b(c, e)\}$ (Udp-Max-Npl)
- $\sum_{c \in \mathcal{A}(p)} p\big(\text{argmin}\{r_c(e) \,|\, e : p(e) \leq b(c, e)\}\big)$ (Udp-Rank-Npl)

Given a price ladder constraint (PL), $p(e_1) \leq \cdots \leq p(e_{|\mathcal{U}|})$, Udp-{Min,Max,Rank}-Pl asks for prices p satisfying PL.

UDP introduced as non-parametric multi-product pricing in [1].

[1] *Glynn, Rusmevichientong and Van Roy (2003)*

[2] *Aggarwal, Feder, Motwani and Zhu (2004)*

UDP-MIN-{PL,NPL}:

- UDP-MIN-PL poly-time for uniform budgets consumers [1].
- UDP-MIN-NPL APX-hard, has $\mathcal{O}(\log|\mathcal{C}|)$-approx [2].

UDP-MAX-{PL,NPL}:

- UDP-MAX-PL has a PTAS [2].
- UDP-MAX-PL, limited supply: 4-approx [2].
- UDP-MAX-NPL 16/15-hard, has 1.59-approx [2].

Q: UDP-MIN-NPL: Is there a const approx ?

UDP introduced as non-parametric multi-product pricing in [1].

[1] *Glynn, Rusmevichientong and Van Roy (2003)*

[2] *Aggarwal, Feder, Motwani and Zhu (2004)*

UDP-MIN-{PL,NPL}:

- UDP-MIN-PL poly-time for uniform budgets consumers [1].
- UDP-MIN-NPL APX-hard, has $\mathcal{O}(\log |\mathcal{C}|)$-approx [2].

UDP-MAX-{PL,NPL}:

- UDP-MAX-PL has a PTAS [2].
- UDP-MAX-PL, limited supply: 4-approx [2].
- UDP-MAX-NPL 16/15-hard, has 1.59-approx [2].

Q: UDP-MIN-NPL: Is there a const approx ? No!

UDP introduced as non-parametric multi-product pricing in [1].

[1] *Glynn, Rusmevichientong and Van Roy (2003)*

[2] *Aggarwal, Feder, Motwani and Zhu (2004)*

UDP-MIN-{PL,NPL}:

- UDP-MIN-PL poly-time for uniform budgets consumers [1].
- UDP-MIN-NPL APX-hard, has $\mathcal{O}(\log |\mathcal{C}|)$-approx [2].

UDP-MAX-{PL,NPL}:

- UDP-MAX-PL has a PTAS [2].
- UDP-MAX-PL, limited supply: 4-approx [2].
- UDP-MAX-NPL 16/15-hard, has 1.59-approx [2].

Q: UDP-MIN-NPL: Is there a const approx ? No!
UDP-MAX-PL: Is PTAS best possible approx ?

UDP introduced as non-parametric multi-product pricing in [1].

[1] *Glynn, Rusmevichientong and Van Roy (2003)*

[2] *Aggarwal, Feder, Motwani and Zhu (2004)*

UDP-MIN-{PL,NPL}:

- UDP-MIN-PL poly-time for uniform budgets consumers [1].
- UDP-MIN-NPL APX-hard, has $\mathcal{O}(\log |\mathcal{C}|)$-approx [2].

UDP-MAX-{PL,NPL}:

- UDP-MAX-PL has a PTAS [2].
- UDP-MAX-PL, limited supply: 4-approx [2].
- UDP-MAX-NPL 16/15-hard, has 1.59-approx [2].

Q: UDP-MIN-NPL: Is there a const approx ? No!
UDP-MAX-PL: Is PTAS best possible approx ? Yes!

UDP introduced as non-parametric multi-product pricing in [1].

[1] *Glynn, Rusmevichientong and Van Roy (2003)*

[2] *Aggarwal, Feder, Motwani and Zhu (2004)*

UDP-MIN-{PL,NPL}:

- UDP-MIN-PL poly-time for uniform budgets consumers [1].
- UDP-MIN-NPL APX-hard, has $\mathcal{O}(\log |\mathcal{C}|)$-approx [2].

UDP-MAX-{PL,NPL}:

- UDP-MAX-PL has a PTAS [2].
- UDP-MAX-PL, limited supply: 4-approx [2].
- UDP-MAX-NPL 16/15-hard, has 1.59-approx [2].

Q: UDP-MIN-NPL: Is there a const approx ? No!
UDP-MAX-PL: Is PTAS best possible approx ? Yes!
UDP-MAX-NPL, limit'd supply: const-approx, APX-hard ?

UDP introduced as non-parametric multi-product pricing in [1].

[1] *Glynn, Rusmevichientong and Van Roy (2003)*

[2] *Aggarwal, Feder, Motwani and Zhu (2004)*

UDP-MIN-{PL,NPL}:

- UDP-MIN-PL poly-time for uniform budgets consumers [1].
- UDP-MIN-NPL APX-hard, has $\mathcal{O}(\log|\mathcal{C}|)$-approx [2].

UDP-MAX-{PL,NPL}:

- UDP-MAX-PL has a PTAS [2].
- UDP-MAX-PL, limited supply: 4-approx [2].
- UDP-MAX-NPL 16/15-hard, has 1.59-approx [2].

Q: UDP-MIN-NPL: Is there a const approx ? No!
UDP-MAX-PL: Is PTAS best possible approx ? Yes!
UDP-MAX-NPL, limit'd supply: const-approx, APX-hard ? Yes!

Hardness Results

UDP-MIN-NPL: Is there a const approx ? No!

Recall: UDP-MIN-NPL has $\mathcal{O}(\log|\mathcal{C}|)$-approx [2]. We prove:

Theorem

UDP-MIN-{PL,NPL} is not approximable within $\mathcal{O}(\log^{\varepsilon}|\mathcal{C}|)$ for some $\varepsilon > 0$, unless NP $\subseteq$ DTIME$(n^{\mathcal{O}(\log\log n)})$.

Recall: UDP-MIN-NPL has $\mathcal{O}(\log|\mathcal{C}|)$-approx [2]. We prove:

Theorem

UDP-MIN-{PL,NPL} is not approximable within $\mathcal{O}(\log^{\varepsilon}|\mathcal{C}|)$ for some $\varepsilon > 0$, unless NP $\subseteq$ DTIME$(n^{\mathcal{O}(\log\log n)})$.

Sketch of Proof:

Let $\alpha(G)$ = size of the maximum independent set in graph G.

Recall: UDP-MIN-NPL has $\mathcal{O}(\log |\mathcal{C}|)$-approx [2]. We prove:

Theorem

UDP-MIN-{PL,NPL} is not approximable within $\mathcal{O}(\log^{\varepsilon} |\mathcal{C}|)$ for some $\varepsilon > 0$, unless NP $\subseteq$ DTIME$(n^{\mathcal{O}(\log \log n)})$.

Sketch of Proof:

Let $\alpha(G)$ = size of the maximum independent set in graph G.

Proposition [Alon, Feige, Wigderson, Zuckerman'95]

$n \in \mathbb{N}$, $\mathcal{G} = \{G : G = (V, E)$ with max degree $\mathcal{O}(\log n), |V| = n\}$. There is $\varepsilon > 0$, s.t. $\mathcal{O}(\log^{\varepsilon} n)$-approx to $\alpha(G)$ is NP-hard for $G \in \mathcal{G}$.

Recall: UDP-MIN-NPL has $\mathcal{O}(\log |\mathcal{C}|)$-approx [2]. We prove:

Theorem

UDP-MIN-{PL,NPL} is not approximable within $\mathcal{O}(\log^{\varepsilon} |\mathcal{C}|)$ for some $\varepsilon > 0$, unless NP $\subseteq$ DTIME$(n^{\mathcal{O}(\log \log n)})$.

Sketch of Proof:

Let $\alpha(G)$ = size of the maximum independent set in graph G.

Proposition [Alon, Feige, Wigderson, Zuckerman'95]

$n \in \mathbb{N}$, $\mathcal{G} = \{G : G = (V, E)$ with max degree $\mathcal{O}(\log n), |V| = n\}$. There is $\varepsilon > 0$, s.t. $\mathcal{O}(\log^{\varepsilon} n)$-approx to $\alpha(G)$ is NP-hard for $G \in \mathcal{G}$.

Given $G \in \mathcal{G}$, we reduce finding $\alpha(G)$ to UDP-MIN-PL.

Recall: UDP-MIN-NPL has $\mathcal{O}(\log |\mathcal{C}|)$-approx [2]. We prove:

Theorem

UDP-MIN-{PL,NPL} is not approximable within $\mathcal{O}(\log^{\varepsilon} |\mathcal{C}|)$ for some $\varepsilon > 0$, unless NP $\subseteq$ DTIME$(n^{\mathcal{O}(\log \log n)})$.

Sketch of Proof:

Let $\alpha(G)$ = size of the maximum independent set in graph G.

Proposition [Alon, Feige, Wigderson, Zuckerman'95]

$n \in \mathbb{N}$, $\mathcal{G} = \{G : G = (V, E)$ with max degree $\mathcal{O}(\log n), |V| = n\}$.
There is $\varepsilon > 0$, s.t. $\mathcal{O}(\log^{\varepsilon} n)$-approx to $\alpha(G)$ is NP-hard for $G \in \mathcal{G}$.

Given $G \in \mathcal{G}$, we reduce finding $\alpha(G)$ to UDP-MIN-PL.

Assume *a.c.*: UDP-MIN-PL has $\mathcal{O}(\log^{\varepsilon - \delta} |\mathcal{C}|)$-approx for some $\delta > 0$.

Recall: UDP-MIN-NPL has $\mathcal{O}(\log|\mathcal{C}|)$-approx [2]. We prove:

Theorem

UDP-MIN-{PL,NPL} is not approximable within $\mathcal{O}(\log^{\varepsilon}|\mathcal{C}|)$ for some $\varepsilon > 0$, unless NP $\subseteq$ DTIME$(n^{\mathcal{O}(\log\log n)})$.

Sketch of Proof:

Let $\alpha(G)$ = size of the maximum independent set in graph G.

Proposition [Alon, Feige, Wigderson, Zuckerman'95]

$n \in \mathbb{N}$, $\mathcal{G} = \{G : G = (V, E)$ with max degree $\mathcal{O}(\log n), |V| = n\}$.
There is $\varepsilon > 0$, s.t. $\mathcal{O}(\log^{\varepsilon} n)$-approx to $\alpha(G)$ is NP-hard for $G \in \mathcal{G}$.

Given $G \in \mathcal{G}$, we reduce finding $\alpha(G)$ to UDP-MIN-PL.

Assume *a.c.*: UDP-MIN-PL has $\mathcal{O}(\log^{\varepsilon-\delta}|\mathcal{C}|)$-approx for some $\delta > 0$.

We will show that this gives $\mathcal{O}(\log^{\varepsilon} n)$-approx for $\alpha(G)$ in time $n^{\mathcal{O}(\log\log n)}$.

Sketch of Proof:

$G = (V, E) \in \mathcal{G}$ of max degree Δ, def instance of UDP-MIN-PL:

Sketch of Proof:

$G = (V, E) \in \mathcal{G}$ of max degree Δ, def instance of UDP-MIN-PL:

$V = V_0 \cup \ldots \cup V_\Delta$: $(\Delta + 1)$-vertex-coloring of G.

Sketch of Proof:

$G = (V, E) \in \mathcal{G}$ of max degree Δ, def instance of UDP-MIN-PL:

$V = V_0 \cup \ldots \cup V_\Delta$: $(\Delta + 1)$-vertex-coloring of G.

Let $V_i = \{v_{ij} \mid j = 0, \ldots, |V_i| - 1\}$.

Sketch of Proof:

$G = (V, E) \in \mathcal{G}$ of max degree Δ, def instance of UDP-MIN-PL:

$V = V_0 \cup \ldots \cup V_\Delta$: $(\Delta + 1)$-vertex-coloring of G.

Let $V_i = \{v_{ij} \mid j = 0, \ldots, |V_i| - 1\}$.

$\mathcal{V}(v_{ij}) = \{v_{k\ell} \mid \{v_{ij}, v_{k\ell}\} \in E \text{ and } k < i\}$ = vertices adjacent to v_{ij} in color class with index $< i$

Sketch of Proof:

$G = (V, E) \in \mathcal{G}$ of max degree Δ, def instance of UDP-MIN-PL:

$V = V_0 \cup \ldots \cup V_\Delta$: $(\Delta + 1)$-vertex-coloring of G.

Let $V_i = \{v_{ij} \mid j = 0, \ldots, |V_i| - 1\}$.

$\mathcal{V}(v_{ij}) = \{v_{k\ell} \mid \{v_{ij}, v_{k\ell}\} \in E \text{ and } k < i\}$ = vertices adjacent to v_{ij} in color class with index $< i$

Products / PL: For every $v_{ij} \in V$ we have a product e_{ij}.

Sketch of Proof:

$G = (V, E) \in \mathcal{G}$ of max degree Δ, def instance of $\textsc{Udp-Min-Pl}$:

$V = V_0 \cup \ldots \cup V_\Delta$: $(\Delta + 1)$-vertex-coloring of G.

Let $V_i = \{v_{ij} \mid j = 0, \ldots, |V_i| - 1\}$.

$\mathcal{V}(v_{ij}) = \{v_{k\ell} \mid \{v_{ij}, v_{k\ell}\} \in E \text{ and } k < i\}$ = vertices adjacent to v_{ij} in color class with index $< i$

Products / PL: For every $v_{ij} \in V$ we have a product e_{ij}.

Define PL: $p(e_{00}) \leq p(e_{01}) \leq \cdots \leq p(e_{0,|V_0|-1}) \leq p(e_{10}) \leq \cdots$.

Sketch of Proof:

$G = (V, E) \in \mathcal{G}$ of max degree Δ, def instance of UDP-MIN-PL:

$V = V_0 \cup \ldots \cup V_\Delta$: $(\Delta + 1)$-vertex-coloring of G.

Let $V_i = \{v_{ij} \mid j = 0, \ldots, |V_i| - 1\}$.

$\mathcal{V}(v_{ij}) = \{v_{k\ell} \mid \{v_{ij}, v_{k\ell}\} \in E \text{ and } k < i\}$ = vertices adjacent to v_{ij} in color class with index $< i$

Products / PL: For every $v_{ij} \in V$ we have a product e_{ij}.

Define PL: $p(e_{00}) \le p(e_{01}) \le \cdots \le p(e_{0,|V_0|-1}) \le p(e_{10}) \le \cdots$.

Let $\mu = 4(\Delta + 1)$ and $\gamma = \mu^{-\Delta-1}/n$.

Sketch of Proof:

$G = (V, E) \in \mathcal{G}$ of max degree Δ, def instance of UDP-MIN-PL:

$V = V_0 \cup \ldots \cup V_\Delta$: $(\Delta + 1)$-vertex-coloring of G.

Let $V_i = \{v_{ij} \mid j = 0, \ldots, |V_i| - 1\}$.

$\mathcal{V}(v_{ij}) = \{v_{k\ell} \mid \{v_{ij}, v_{k\ell}\} \in E \text{ and } k < i\}$ = vertices adjacent to v_{ij} in color class with index $< i$

Products / PL: For every $v_{ij} \in V$ we have a product e_{ij}.

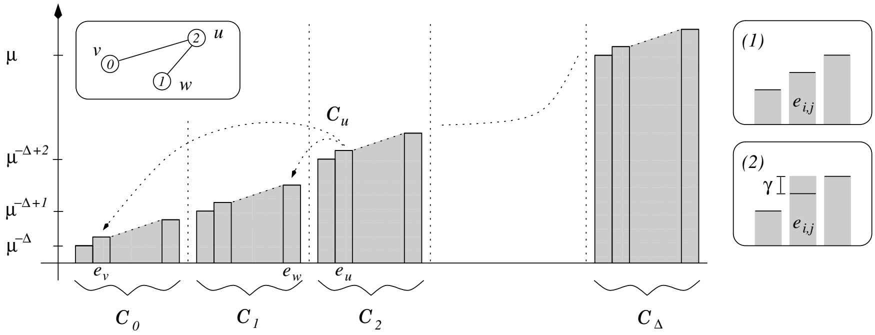
Define PL: $p(e_{00}) \leq p(e_{01}) \leq \cdots \leq p(e_{0,|V_0|-1}) \leq p(e_{10}) \leq \cdots$.

Let $\mu = 4(\Delta + 1)$ and $\gamma = \mu^{-\Delta-1}/n$.

For every product e_{ij} define $p_{ij} = \mu^{i-\Delta} + j\gamma$.

Sketch of Proof:

Illustration of the construction:



Sketch of Proof:

Consumers: For $v_{ij} \in V$ define a set

$$\mathcal{C}_{ij} = \{c_{ij}^t \mid t = 0, \ldots, \mu^{\Delta - i} - 1\} \text{ of identical consumers}$$

with budgets

$$b(c_{ij}^t, e_{ij}) = p_{ij} \text{ and}$$

$$b(c_{ij}^t, e_{k\ell}) = p_{k\ell} \text{ for all } k, \ell \text{ with } v_{k\ell} \in \mathcal{V}(v_{ij}).$$

Sketch of Proof:

Consumers: For $v_{ij} \in V$ define a set

$$\mathcal{C}_{ij} = \{c_{ij}^t \mid t = 0, \ldots, \mu^{\Delta - i} - 1\}$$ of identical consumers

with budgets

$b(c_{ij}^t, e_{ij}) = p_{ij}$ and

$b(c_{ij}^t, e_{k\ell}) = p_{k\ell}$ for all k,ℓ with $v_{k\ell} \in \mathcal{V}(v_{ij})$.

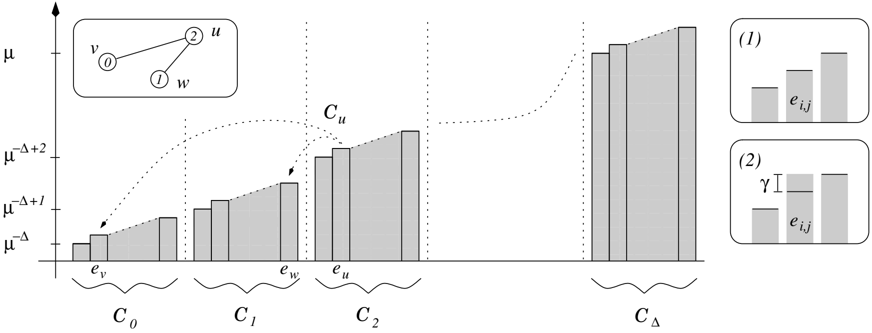
In analogy to coloring $V = V_0 \cup \ldots \cup V_\Delta$ denote consumers

$$\mathcal{C} = \mathcal{C}_0 \cup \ldots \cup \mathcal{C}_\Delta,$$

where $\mathcal{C}_i = \bigcup_j \mathcal{C}_{ij}$.

Sketch of Proof:

Illustration of the construction:



Sketch of Proof: **Soundness:** $opt_{UDP} \geq \alpha(G)$, that is:

(large IS in G) $\Rightarrow$ (high revenue in UDP)

For an IS V' of G, define prices p:
for $v_{ij} \in V'$ set $p(e_{ij}) = p_{ij}$, else set $p(e_{ij}) = p_{ij} + \gamma$.
(p_{ij}'s differ by $\geq \gamma$) $\Rightarrow$ prices $p(\cdot)$ fulfill PL

Sketch of Proof: **Soundness:** $opt_{UDP} \geq \alpha(G)$, that is:

(large IS in G) $\Rightarrow$ (high revenue in UDP)

For an IS V' of G, define prices p:
for $v_{ij} \in V'$ set $p(e_{ij}) = p_{ij}$, else set $p(e_{ij}) = p_{ij} + \gamma$.
(p_{ij}'s differ by $\geq \gamma$) $\Rightarrow$ prices $p(\cdot)$ fulfill PL

Consider $v_{ij} \in V'$ and corresponding consumers $\mathcal{C}_{ij}$.
$(\forall v_{k\ell} \in \mathcal{V}(v_{ij}) : v_{k\ell} \notin V')$

Sketch of Proof: **Soundness:** $opt_{UDP} \geq \alpha(G)$, that is:

(large IS in G) $\Rightarrow$ (high revenue in UDP)

For an IS V' of G, define prices p:
for $v_{ij} \in V'$ set $p(e_{ij}) = p_{ij}$, else set $p(e_{ij}) = p_{ij} + \gamma$.
(p_{ij}'s differ by $\geq \gamma$) $\Rightarrow$ prices $p(\cdot)$ fulfill PL

Consider $v_{ij} \in V'$ and corresponding consumers $\mathcal{C}_{ij}$.
$(\forall v_{k\ell} \in \mathcal{V}(v_{ij}) : v_{k\ell} \notin V')$

$\Rightarrow$ each consumer $c_{ij}^t \in \mathcal{C}_{ij}$ can afford product e_{ij} at price p_{ij}

Sketch of Proof: **Soundness:** $opt_{UDP} \geq \alpha(G)$, that is:

(large IS in G) $\Rightarrow$ (high revenue in UDP)

For an IS V' of G, define prices p:
for $v_{ij} \in V'$ set $p(e_{ij}) = p_{ij}$, else set $p(e_{ij}) = p_{ij} + \gamma$.
(p_{ij}'s differ by $\geq \gamma$) $\Rightarrow$ prices $p(\cdot)$ fulfill PL

Consider $v_{ij} \in V'$ and corresponding consumers $\mathcal{C}_{ij}$.
$(\forall v_{k\ell} \in \mathcal{V}(v_{ij}) : v_{k\ell} \notin V')$

$\Rightarrow$ each consumer $c_{ij}^t \in \mathcal{C}_{ij}$ can afford product e_{ij} at price p_{ij}
$\Rightarrow$ prices of products $e_{k\ell}$ exceed thresholds $p_{k\ell}$ of each c_{ij}^t

Sketch of Proof: **Soundness:** $opt_{UDP} \geq \alpha(G)$, that is:

(large IS in G) $\Rightarrow$ (high revenue in UDP)

For an IS V' of G, define prices p:
for $v_{ij} \in V'$ set $p(e_{ij}) = p_{ij}$, else set $p(e_{ij}) = p_{ij} + \gamma$.
(p_{ij}'s differ by $\geq \gamma$) $\Rightarrow$ prices $p(\cdot)$ fulfill PL

Consider $v_{ij} \in V'$ and corresponding consumers $\mathcal{C}_{ij}$.
$(\forall v_{k\ell} \in \mathcal{V}(v_{ij}) : v_{k\ell} \notin V')$

$\Rightarrow$ each consumer $c_{ij}^t \in \mathcal{C}_{ij}$ can afford product e_{ij} at price p_{ij}
$\Rightarrow$ prices of products $e_{k\ell}$ exceed thresholds $p_{k\ell}$ of each c_{ij}^t
$\Rightarrow$ e_{ij} is the product with smallest price that any c_{ij}^t can afford

Sketch of Proof: **Soundness:** $opt_{UDP} \geq \alpha(G)$, that is:

(large IS in G) $\Rightarrow$ (high revenue in UDP)

For an IS V' of G, define prices p:
for $v_{ij} \in V'$ set $p(e_{ij}) = p_{ij}$, else set $p(e_{ij}) = p_{ij} + \gamma$.
(p_{ij}'s differ by $\geq \gamma$) $\Rightarrow$ prices $p(\cdot)$ fulfill PL

Consider $v_{ij} \in V'$ and corresponding consumers $\mathcal{C}_{ij}$.
$(\forall v_{k\ell} \in \mathcal{V}(v_{ij}) : v_{k\ell} \notin V')$

$\Rightarrow$ each consumer $c^t_{ij} \in \mathcal{C}_{ij}$ can afford product e_{ij} at price p_{ij}
$\Rightarrow$ prices of products $e_{k\ell}$ exceed thresholds $p_{k\ell}$ of each c^t_{ij}
$\Rightarrow$ e_{ij} is the product with smallest price that any c^t_{ij} can afford

$\Rightarrow$ revenue of consumers $\mathcal{C}_{ij} \geq |\mathcal{C}_{ij}| \cdot p_{ij} = \mu^{\Delta - i}\left(\mu^{i-\Delta} + j\gamma\right) \geq 1$

Sketch of Proof: **Soundness:** $opt_{UDP} \geq \alpha(G)$, that is:

(large IS in G) $\Rightarrow$ (high revenue in UDP)

For an IS V' of G, define prices p:
for $v_{ij} \in V'$ set $p(e_{ij}) = p_{ij}$, else set $p(e_{ij}) = p_{ij} + \gamma$.
(p_{ij}'s differ by $\geq \gamma$) $\Rightarrow$ prices $p(\cdot)$ fulfill PL

Consider $v_{ij} \in V'$ and corresponding consumers $\mathcal{C}_{ij}$.
$(\forall v_{k\ell} \in \mathcal{V}(v_{ij}) : v_{k\ell} \notin V')$

$\Rightarrow$ each consumer $c_{ij}^t \in \mathcal{C}_{ij}$ can afford product e_{ij} at price p_{ij}
$\Rightarrow$ prices of products $e_{k\ell}$ exceed thresholds $p_{k\ell}$ of each c_{ij}^t
$\Rightarrow$ e_{ij} is the product with smallest price that any c_{ij}^t can afford

$\Rightarrow$ revenue of consumers $\mathcal{C}_{ij} \geq |\mathcal{C}_{ij}| \cdot p_{ij} = \mu^{\Delta-i}\left(\mu^{i-\Delta} + j\gamma\right) \geq 1$
$\Rightarrow$ prices p result in revenue $\geq |V'|$, so $opt_{UDP} \geq \alpha(G)$

Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} revenue(\mathcal{C})$, V'-IS:

(high revenue in UDP) $\Rightarrow$ (large IS in G)

Let $p()$–prices found by approx algo, $r(\mathcal{C})$, $r(\mathcal{C}_{ij})$, $r(c_{ij}^t)$-revenues.

W.l.o.g.: price of each product $e_{ij} \in \{p_{ij}, p_{ij} + \gamma\}$

Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} revenue(\mathcal{C})$, V'-IS:

(high revenue in UDP) $\Rightarrow$ (large IS in G)

Let $p()$–prices found by approx algo, $r(\mathcal{C})$, $r(\mathcal{C}_{ij})$, $r(c_{ij}^t)$-revenues.
W.l.o.g.: price of each product $e_{ij} \in \{p_{ij}, p_{ij} + \gamma\}$

$\mathcal{C}^+ \stackrel{df}{=} \{c_{ij}^t \,|\, r(c_{ij}^t) = p_{ij}\}$, $\mathcal{C}^- = \mathcal{C} \backslash \mathcal{C}^+$.

Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} revenue(\mathcal{C})$, V'-IS:

(high revenue in UDP) $\Rightarrow$ (large IS in G)

Let $p()$–prices found by approx algo, $r(\mathcal{C})$, $r(\mathcal{C}_{ij})$, $r(c_{ij}^t)$-revenues.
W.l.o.g.: price of each product $e_{ij} \in \{p_{ij}, p_{ij} + \gamma\}$

$\mathcal{C}^+ \stackrel{df}{=} \{c_{ij}^t \,|\, r(c_{ij}^t) = p_{ij}\}$, $\mathcal{C}^- = \mathcal{C} \setminus \mathcal{C}^+$. Obs: $\forall i, j$: $\mathcal{C}_{ij} \subseteq \mathcal{C}^+$ or $\mathcal{C}_{ij} \subseteq \mathcal{C}^-$.

Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} revenue(\mathcal{C})$, V'-IS:

(high revenue in UDP) $\Rightarrow$ (large IS in G)

Let $p()$–prices found by approx algo, $r(\mathcal{C})$, $r(\mathcal{C}_{ij})$, $r(c_{ij}^t)$-revenues.
W.l.o.g.: price of each product $e_{ij} \in \{p_{ij}, p_{ij} + \gamma\}$

$\mathcal{C}^+ \stackrel{df}{=} \{c_{ij}^t \mid r(c_{ij}^t) = p_{ij}\}$, $\mathcal{C}^- = \mathcal{C} \setminus \mathcal{C}^+$. Obs: $\forall i, j$: $\mathcal{C}_{ij} \subseteq \mathcal{C}^+$ or $\mathcal{C}_{ij} \subseteq \mathcal{C}^-$.

Goal: define $V' = \{v_{ij} \mid \mathcal{C}_{ij} \subseteq \mathcal{C}^+\}$, show V'–large IS.

Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} revenue(\mathcal{C})$, V'-IS:

(high revenue in UDP) $\Rightarrow$ (large IS in G)

Let $p()$–prices found by approx algo, $r(\mathcal{C})$, $r(\mathcal{C}_{ij})$, $r(c_{ij}^t)$-revenues.
W.l.o.g.: price of each product $e_{ij} \in \{p_{ij}, p_{ij} + \gamma\}$

$\mathcal{C}^+ \stackrel{df}{=} \{c_{ij}^t \,|\, r(c_{ij}^t) = p_{ij}\}$, $\mathcal{C}^- = \mathcal{C} \setminus \mathcal{C}^+$. Obs: $\forall i, j$: $\mathcal{C}_{ij} \subseteq \mathcal{C}^+$ or $\mathcal{C}_{ij} \subseteq \mathcal{C}^-$.

Goal: define $V' = \{v_{ij} \,|\, \mathcal{C}_{ij} \subseteq \mathcal{C}^+\}$, show V'–large IS.

Obs: $r(\mathcal{C}^-) = \sum_{\mathcal{C}_{ij} \subseteq \mathcal{C}^-} r(\mathcal{C}_{ij}) \leq \frac{n}{2(\Delta+1)}$

Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} revenue(\mathcal{C})$, V'-IS:

(high revenue in UDP) $\Rightarrow$ (large IS in G)

Let $p()$–prices found by approx algo, $r(\mathcal{C})$, $r(\mathcal{C}_{ij})$, $r(c_{ij}^t)$-revenues.
W.l.o.g.: price of each product $e_{ij} \in \{p_{ij}, p_{ij} + \gamma\}$

$\mathcal{C}^+ \stackrel{df}{=} \{c_{ij}^t \,|\, r(c_{ij}^t) = p_{ij}\}$, $\mathcal{C}^- = \mathcal{C} \backslash \mathcal{C}^+$. Obs: $\forall i, j$: $\mathcal{C}_{ij} \subseteq \mathcal{C}^+$ or $\mathcal{C}_{ij} \subseteq \mathcal{C}^-$.

Goal: define $V' = \{v_{ij} \,|\, \mathcal{C}_{ij} \subseteq \mathcal{C}^+\}$, show V'–large IS.

Obs: $r(\mathcal{C}^-) = \sum_{\mathcal{C}_{ij} \subseteq \mathcal{C}^-} r(\mathcal{C}_{ij}) \leq \frac{n}{2(\Delta+1)}$

Obs: $\alpha(G) \geq n/(\Delta + 1)$; easy to find prices resulting in revenue $n/(\Delta + 1)$

Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} revenue(\mathcal{C})$, V'-IS:

(high revenue in UDP) $\Rightarrow$ (large IS in G)

Let $p()$–prices found by approx algo, $r(\mathcal{C})$, $r(\mathcal{C}_{ij})$, $r(c_{ij}^t)$-revenues.
W.l.o.g.: price of each product $e_{ij} \in \{p_{ij}, p_{ij} + \gamma\}$

$\mathcal{C}^+ \stackrel{df}{=} \{c_{ij}^t \mid r(c_{ij}^t) = p_{ij}\}$, $\mathcal{C}^- = \mathcal{C} \setminus \mathcal{C}^+$. Obs: $\forall i, j$: $\mathcal{C}_{ij} \subseteq \mathcal{C}^+$ or $\mathcal{C}_{ij} \subseteq \mathcal{C}^-$.

Goal: define $V' = \{v_{ij} \mid \mathcal{C}_{ij} \subseteq \mathcal{C}^+\}$, show V'–large IS.

Obs: $r(\mathcal{C}^-) = \sum_{\mathcal{C}_{ij} \subseteq \mathcal{C}^-} r(\mathcal{C}_{ij}) \leq \frac{n}{2(\Delta+1)}$

Obs: $\alpha(G) \geq n/(\Delta + 1)$; easy to find prices resulting in revenue $n/(\Delta + 1)$

$\Rightarrow$ wlog: $r(\mathcal{C}) \geq \frac{n}{\Delta+1}$

Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} revenue(\mathcal{C})$, V'-IS:

(high revenue in UDP) $\Rightarrow$ (large IS in G)

Let $p()$–prices found by approx algo, $r(\mathcal{C})$, $r(\mathcal{C}_{ij})$, $r(c^t_{ij})$-revenues.
W.l.o.g.: price of each product $e_{ij} \in \{p_{ij}, p_{ij} + \gamma\}$

$\mathcal{C}^+ \stackrel{df}{=} \{c^t_{ij} \mid r(c^t_{ij}) = p_{ij}\}$, $\mathcal{C}^- = \mathcal{C} \setminus \mathcal{C}^+$. Obs: $\forall i, j$: $\mathcal{C}_{ij} \subseteq \mathcal{C}^+$ or $\mathcal{C}_{ij} \subseteq \mathcal{C}^-$.

Goal: define $V' = \{v_{ij} \mid \mathcal{C}_{ij} \subseteq \mathcal{C}^+\}$, show V'–large IS.

Obs: $r(\mathcal{C}^-) = \sum_{\mathcal{C}_{ij} \subseteq \mathcal{C}^-} r(\mathcal{C}_{ij}) \leq \frac{n}{2(\Delta+1)}$

Obs: $\alpha(G) \geq n/(\Delta + 1)$; easy to find prices resulting in revenue $n/(\Delta + 1)$

$\Rightarrow$ wlog: $r(\mathcal{C}) \geq \frac{n}{\Delta+1} \Rightarrow r(\mathcal{C}^+) = r(\mathcal{C}) - r(\mathcal{C}^-) \geq (1/2) r(\mathcal{C})$

Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} revenue(\mathcal{C})$, V'-IS:

(high revenue in UDP) $\Rightarrow$ (large IS in G)

Let $p()$–prices found by approx algo, $r(\mathcal{C})$, $r(\mathcal{C}_{ij})$, $r(c_{ij}^t)$-revenues.
W.l.o.g.: price of each product $e_{ij} \in \{p_{ij}, p_{ij} + \gamma\}$

$\mathcal{C}^+ \stackrel{df}{=} \{c_{ij}^t \,|\, r(c_{ij}^t) = p_{ij}\}$, $\mathcal{C}^- = \mathcal{C} \backslash \mathcal{C}^+$. Obs: $\forall i, j$: $\mathcal{C}_{ij} \subseteq \mathcal{C}^+$ or $\mathcal{C}_{ij} \subseteq \mathcal{C}^-$.

Goal: define $V' = \{v_{ij} \,|\, \mathcal{C}_{ij} \subseteq \mathcal{C}^+\}$, show V'–large IS.

Obs: $r(\mathcal{C}^-) = \sum_{\mathcal{C}_{ij} \subseteq \mathcal{C}^-} r(\mathcal{C}_{ij}) \leq \frac{n}{2(\Delta+1)}$

Obs: $\alpha(G) \geq n/(\Delta + 1)$; easy to find prices resulting in revenue $n/(\Delta + 1)$

$\Rightarrow$ wlog: $r(\mathcal{C}) \geq \frac{n}{\Delta+1} \Rightarrow r(\mathcal{C}^+) = r(\mathcal{C}) - r(\mathcal{C}^-) \geq (1/2) r(\mathcal{C})$

$\forall v_{ij} \in V'$: $\mathcal{C}_{ij} \subseteq \mathcal{C}^+$, $r(\mathcal{C}_{ij}) = |\mathcal{C}_{ij}| \cdot p_{ij} = \mu^{\Delta - i} \left(\mu^{i-\Delta} + j\gamma\right) \leq 2$

Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} revenue(\mathcal{C})$, V'-IS:

(high revenue in UDP) $\Rightarrow$ (large IS in G)

Let $p()$–prices found by approx algo, $r(\mathcal{C})$, $r(\mathcal{C}_{ij})$, $r(c_{ij}^t)$-revenues.
W.l.o.g.: price of each product $e_{ij} \in \{p_{ij}, p_{ij} + \gamma\}$

$\mathcal{C}^+ \stackrel{df}{=} \{c_{ij}^t \,|\, r(c_{ij}^t) = p_{ij}\}$, $\mathcal{C}^- = \mathcal{C} \backslash \mathcal{C}^+$. Obs: $\forall i, j$: $\mathcal{C}_{ij} \subseteq \mathcal{C}^+$ or $\mathcal{C}_{ij} \subseteq \mathcal{C}^-$.

Goal: define $V' = \{v_{ij} \,|\, \mathcal{C}_{ij} \subseteq \mathcal{C}^+\}$, show V'–large IS.

Obs: $r(\mathcal{C}^-) = \sum_{\mathcal{C}_{ij} \subseteq \mathcal{C}^-} r(\mathcal{C}_{ij}) \leq \frac{n}{2(\Delta+1)}$

Obs: $\alpha(G) \geq n/(\Delta + 1)$; easy to find prices resulting in revenue $n/(\Delta + 1)$

$\Rightarrow$ wlog: $r(\mathcal{C}) \geq \frac{n}{\Delta+1} \Rightarrow r(\mathcal{C}^+) = r(\mathcal{C}) - r(\mathcal{C}^-) \geq (1/2) r(\mathcal{C})$

$\forall v_{ij} \in V'$: $\mathcal{C}_{ij} \subseteq \mathcal{C}^+$, $r(\mathcal{C}_{ij}) = |\mathcal{C}_{ij}| \cdot p_{ij} = \mu^{\Delta - i} \left(\mu^{i-\Delta} + j\gamma\right) \leq 2$

$\Rightarrow |V'| = |\{v_{ij} \,|\, \mathcal{C}_{ij} \subseteq \mathcal{C}^+\}| \geq (1/2) r(\mathcal{C}^+) \geq (1/4) r(\mathcal{C})$

Sketch of Proof: **Finish:**

Recall: $|\mathcal{C}| = \sharp$ consumers, and $\log |\mathcal{C}| \leq \log n\mu^{\Delta} = \mathcal{O}(\log^{1+\varepsilon'} n)$ for any $\varepsilon' > 0$.

Sketch of Proof: **Finish:**

Recall: $|\mathcal{C}| = \sharp$ consumers, and $\log |\mathcal{C}| \leq \log n\mu^{\Delta} = \mathcal{O}(\log^{1+\varepsilon'} n)$ for any $\varepsilon' > 0$.

$r(\mathcal{C})$ is $\mathcal{O}(\log^{\varepsilon-\delta} |\mathcal{C}|)$-approx to opt_{UDP}

Sketch of Proof: **Finish:**

Recall: $|\mathcal{C}| = \sharp$ consumers, and $\log |\mathcal{C}| \leq \log n\mu^{\Delta} = \mathcal{O}(\log^{1+\varepsilon'} n)$ for any $\varepsilon' > 0$.

$r(\mathcal{C})$ is $\mathcal{O}(\log^{\varepsilon-\delta} |\mathcal{C}|)$-approx to opt_{UDP}

$$\Rightarrow |V'| \geq \frac{1}{4} r(\mathcal{C}) \geq \frac{1}{\mathcal{O}(\log^{\varepsilon-\delta} |\mathcal{C}|)} opt_{UDP} \geq \frac{1}{\mathcal{O}(\log^{\varepsilon} n)} \alpha(G)$$

Sketch of Proof: **Finish:**

Recall: $|\mathcal{C}| = \sharp$ consumers, and $\log |\mathcal{C}| \leq \log n\mu^{\Delta} = \mathcal{O}(\log^{1+\varepsilon'} n)$ for any $\varepsilon' > 0$.

$r(\mathcal{C})$ is $\mathcal{O}(\log^{\varepsilon-\delta} |\mathcal{C}|)$-approx to opt_{UDP}

$\Rightarrow |V'| \geq \frac{1}{4} r(\mathcal{C}) \geq \frac{1}{\mathcal{O}(\log^{\varepsilon-\delta} |\mathcal{C}|)} opt_{UDP} \geq \frac{1}{\mathcal{O}(\log^{\varepsilon} n)} \alpha(G)$

By Proposition finding such an IS set is NP-hard.

The size of our UDP-MIN-PL instance is roughly

$$n \cdot (\log n)^{\log n} = n^{\mathcal{O}(\log \log n)}$$

and the running time of our approx algo is polynomial in this expression.

UDP-MAX-PL: Is PTAS best possible approx ?
Yes!

Theorem

UDP-MAX-PL with unlimited supply is strongly NP-hard, even if each consumer has at most 2 non-zero budgets.

Approximation Algorithms

UDP-MAX-NPL, limit'd supply: const-approx, APX-hard ? Yes!

Theorem

UDP-MAX-{PL,NPL} with unit-supply can be solved in polynomial time.

Theorem

UDP-MAX-NPL with limited supply 2 or larger is APX-hard.

Theorem

There is a 2-approximation algorithm for UDP-MAX-NPL with limited supply.

Theorem

There is a 2-approximation algorithm for UDP-MAX-NPL with limited supply.

Sketch of Proof:

Let $r(p, a)$ be the revenue of price assignment p and corresponding (optimal) allocation a (*Maximum Weighted Bipartite b-Matching*).

Given prices p let $[p \mid p(e) = p']$ be prices obtained by changing price of e to p'. We prove that the following is a 2-approx algo:

> LOCALSEARCH: Initialize p arbitrarily and compute the optimal allocation a. While there exists product e and price $p' \neq p(e)$ such that
>
> $$r(p, a) < r([p \mid p(e) = p'], a'),$$
>
> where a' is the optimal allocation given prices $[p \mid p(e) = p']$, set $p(e) = p'$.

Summary (UDP):

- UDP-MIN-{PL,NPL} is intractable (no const approx), even with PL
- UDP-MAX-{PL,NPL} is tractable (const approx), even with NPL and limited supply

APX-hardness of G-SUSP due to *Guruswami et al. (2005)*.

Applications in realistic network settings often lead to sparse problem instances. Hardness of approximation still holds if:

- G has constant degree d
- paths have constant lengths $\leq \ell$
- at most a constant number B of paths per edge
- only constant height valuations

Theorem

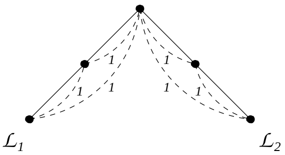
G-SUSP on sparse instances is APX-hard.

We reduce a variant of MaxSat.

Max2Sat(3): clauses of length 2, every literal appears in at most 3 clauses

We need to design gadgets that imitate clauses in the SAT instance.

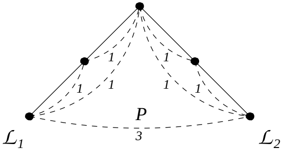
We start out from the *weight gadgets* which will model literals.



$p(\mathcal{L}_1)$	$p(\mathcal{L}_2)$	prof		
1	1	4		
1	2	4		
2	2	4		

Clause Gadgets

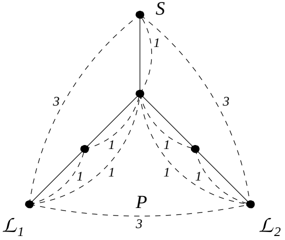
Literal gadget $\mathcal{L}_j$ for every occurrence of literal l_j,
gives maximum profit 2 if $p(\mathcal{L}_j) \in \{1, 2\}$.
Connect $\mathcal{L}_1$ and $\mathcal{L}_2$, if l_1, l_2 form a clause.



$p(\mathcal{L}_1)$	$p(\mathcal{L}_2)$	prof	P	
1	1	4	2	
1	2	4	3	
2	2	4	0	

Clause Gadgets

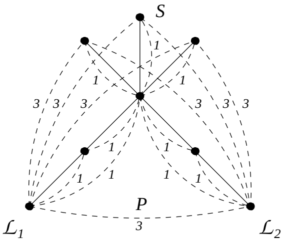
Literal gadget $\mathcal{L}_j$ for every occurrence of literal l_j,
gives maximum profit 2 if $p(\mathcal{L}_j) \in \{1, 2\}$.
Connect $\mathcal{L}_1$ and $\mathcal{L}_2$, if l_1, l_2 form a clause.



$p(\mathcal{L}_1)$	$p(\mathcal{L}_2)$	prof	P	S
1	1	4	2	6
1	2	4	3	6
2	2	4	0	7

Clause Gadgets

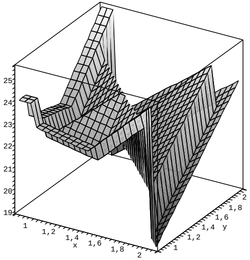
Literal gadget $\mathcal{L}_j$ for every occurrence of literal l_j,
gives maximum profit 2 if $p(\mathcal{L}_j) \in \{1, 2\}$.
Connect $\mathcal{L}_1$ and $\mathcal{L}_2$, if l_1, l_2 form a clause.



$p(\mathcal{L}_1)$	$p(\mathcal{L}_2)$	prof	P	S
1	1	4	2	6
1	2	4	3	6
2	2	4	0	7

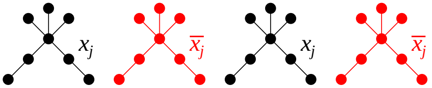
Clause Gadgets

Literal gadget $\mathcal{L}_j$ for every occurrence of literal l_j,
gives maximum profit 2 if $p(\mathcal{L}_j) \in \{1, 2\}$.
Connect $\mathcal{L}_1$ and $\mathcal{L}_2$, if l_1, l_2 form a clause.

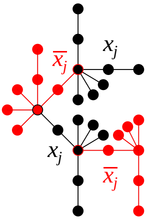


Lemma

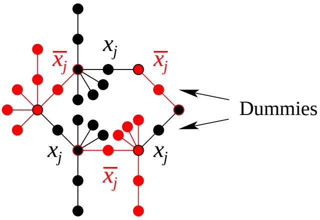
Let $\mathcal{C}$ be a clause gadget with literal gadgets $\mathcal{L}_1$ and $\mathcal{L}_2$. Maximum profit obtainable from $\mathcal{C}$ is 25 and is reached if and only if $\{p(\mathcal{L}_1), p(\mathcal{L}_2)\} = \{1, 2\}$ or $p(\mathcal{L}_1) = p(\mathcal{L}_2) = 2$. $\mathcal{C}$ gives profit 24 if $p(\mathcal{L}_1) = p(\mathcal{L}_2) = 1$.



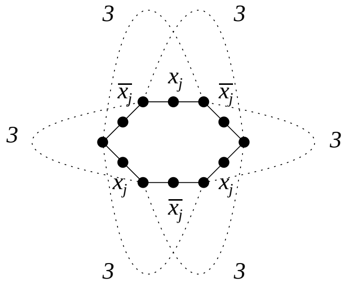
Literal gadgets $\mathcal{L}(x_j)$, $\mathcal{L}(\overline{x}_j)$ belonging to literals x_j or $\overline{x}_j$ are joined together. Adding a sufficient number of *dummies* gives a cyclic structure of exactly 6 literal gadgets.



Literal gadgets $\mathcal{L}(x_j)$, $\mathcal{L}(\overline{x}_j)$ belonging to literals x_j or $\overline{x}_j$ are joined together. Adding a sufficient number of *dummies* gives a cyclic structure of exactly 6 literal gadgets.



Literal gadgets $\mathcal{L}(x_j)$, $\mathcal{L}(\overline{x}_j)$ belonging to literals x_j or $\overline{x}_j$ are joined together. Adding a sufficient number of *dummies* gives a cyclic structure of exactly 6 literal gadgets.



Literal gadgets $\mathcal{L}(x_j)$, $\mathcal{L}(\overline{x}_j)$ belonging to literals x_j or $\overline{x}_j$ are joined together. Adding a sufficient number of *dummies* gives a cyclic structure of exactly 6 literal gadgets.

Prices p on this instances are *SAT-feasible*, if

- $p(\mathcal{L}) \in \{1, 2\}$ for all literal gadgets $\mathcal{L}$
- $p(\mathcal{L}_1(x_j)) = p(\mathcal{L}_2(x_j)) = p(\mathcal{L}_3(x_j))$ and
- $p(\mathcal{L}_1(\overline{x}_j)) = p(\mathcal{L}_2(\overline{x}_j)) = p(\mathcal{L}_3(\overline{x}_j))$ for all x_j.

Lemma

Any price assignment p can be transformed in polynomial time into a SAT-feasible price assignment p^* of no smaller profit.

An $O(\ell^2)$-Approximation (SUSP)

Define *(smoothed)* s-SUSP by changing the objective to

$$\sum_{S \in \Lambda(p)} \sum_{e \in S} p(e),$$

where $\Lambda(p) = \{S \in \mathcal{S} \mid p(e) \leq \delta(S) \, \forall \, e \in S\}$.

We derive an $O(\ell)$-approximation for s-SUSP.

1. For every $e \in \mathcal{U}$ compute the optimal price $p^*(e)$ assuming all other prices were 0.
2. Resolve existing *conflicts*.

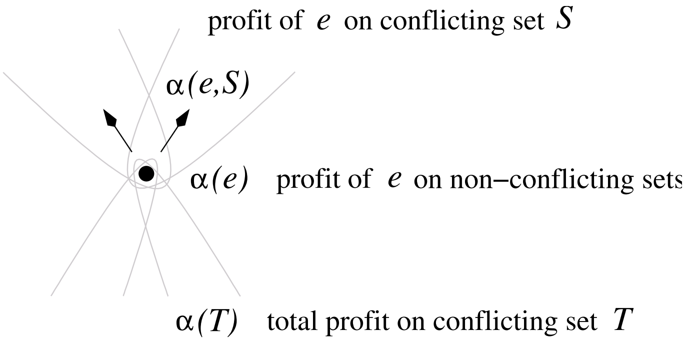
Set S is *conflicting*, if

$$\exists \, e, f \in S \; : \; p^*(e) \leq \delta(S) < p^*(f).$$

Upper bounding technique

1. For every $e \in \mathcal{U}$ compute the optimal price $p^*(e)$ assuming all other prices were 0.
2. Our upper bound: $Opt \leq \sum_{e \in \mathcal{U}} p^*(e)$

Conflict graph for price assignment p^*:



Vertices represent products, directed hyperedges represent conflicting sets. Profit out of non-conflicting sets is assigned to vertices, conflicting profit to hyperedges.

Conflicts are resolved by transforming the conflict graph:

Step 1: In order of increasing prices check for each product e if

$$\sum_{T \in In(e)} \alpha(T) > \frac{1}{2} \sum_{S \in Out(e)} \alpha(e, S),$$

and remove e from all outgoing edges in this case.

Step 2: Let $R = \{e \mid Out(e) = \emptyset\}$, $\mathcal{E} = \{S \widehat{=} (V, W) \mid W \subseteq R\}$. Edges in $\mathcal{E}$ carry half the profit of all edges in the graph. If $\alpha(R) > \alpha(\mathcal{E})$ set $p(e) = 0$ for all $e \in R$.

Step 3: Remove the remaining edges.

We obtain a conflict graph for some non-conflicting price assignment p.

Lemma

In the transformation the overall α-value decreases by at most a factor $O(\ell)$.

Opt of SUSP is upper bounded by ℓ times the α-value of p^*'s conflict graph, thus:

Theorem

The above algorithm computes an $O(\ell^2)$-approximation for SUSP.