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 $S$ with values $v(S)$ find prices $p$, such that

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

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.

In general:
- $O(\log |U| + \log |S|)$-approximation
- inapproximable within $O(\log^\delta |U|)$ 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:
- \(O(\log |U| + \log |S|)\)-approximation
- inapproximable within \(O(\log^\delta |U|)\) 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

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

Design gadgets that capture the discrete nature of this problem.
Weight Gadgets

Maximum profit out of $W_j$ is $2w_j$.
It is obtained iff $p(W_j) = p(e^j_1) + p(e^j_2)$ 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 $\sum_{j \in S} w_j = \sum_{j \notin S} w_j$. □
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

- $\ell$, 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 $\delta(S) = \nu(S)/|S|$ be *price per product* of set $S$.

1. Round all $\delta(S)$ to powers of 2. Let $S = S_0 \cup \ldots \cup S_t$ where $t = \lceil \log \ell^2 B \rceil - 1$. In $S_i$: $\delta(S) > \delta(T) \Rightarrow \delta(S)/\delta(T) \geq \ell^2 B$.

2. In each $S_i$ select non-intersecting sets with maximum $\delta$-value and compute optimal prices.
Analysis:

- \( \text{Opt}(S) \leq \sum_{i=1}^{t} \text{Opt}(S_i) \)
- Let \( S \in S_i, \mathcal{I}(S) \) intersecting sets with smaller \( \delta \)-values:
  \[ v(S) \geq \sum_{T \in \mathcal{I}(S)} v(T) \]
- Let \( S_i^* \) be non-intersecting sets with max. \( \delta \) as in the algo. Then
  \[ \text{Opt}(S_i) \leq 2 \cdot \text{Opt}(S_i^*), \]
  and, since we compute \( \max_i \text{Opt}(S_i^*) \):

Theorem

The above algorithm has approximation ratio \( O(\log \ell + \log B) \).
Upper bounding technique

We relate $\text{Opt}(S_i^*)$ to $\text{Opt}(S_i)$ by using as an upper bound

$$\text{Opt}(S_i) \leq \sum_{S \in S_i} v(S),$$

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

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 $e \in U$ compute the optimal price $p^*(e)$ assuming all other prices were 0.
2. Our upper bound: $Opt \leq \sum_{e \in U} p^*(e)$
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 (⇒ partitioning)
  - $O(\ell^2)$-approximation (⇒ 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 $r_c : \mathcal{U} \rightarrow \{1, \ldots, |\mathcal{U}|\}$.

For prices $p : \mathcal{U} \rightarrow \mathbb{R}_0^+$:

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

$A(p) = \{c \in \mathcal{C} | \exists e \in \mathcal{U} : p(e) \leq b(c, e)\} = \text{consumers affording to buy any product.}$

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

$$\sum_{c \in A(p)} \min\{p(e) | p(e) \leq b(c, e)\} \quad (\text{UDP-MIN-NPL})$$
Unit-Demand Pricing (UDP)

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

For prices $p : \mathcal{U} \to \mathbb{R}_0^+$:

$A(p) = \{ c \in \mathcal{C} | \exists e \in \mathcal{U} : p(e) \leq b(c, e) \} = \text{consumers affording to buy any product.}$

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

- $\sum_{c \in A(p)} \min\{p(e) | p(e) \leq b(c, e)\}$ (UDP-MIN-NPL)
- $\sum_{c \in A(p)} \max\{p(e) | p(e) \leq b(c, e)\}$ (UDP-MAX-NPL)
Unit-Demand Pricing \((UDP)\)

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

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

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

- \[\sum_{c \in A(p)} \min\{p(e) | p(e) \leq b(c, e)\}\] \((\text{UDP-Min-NPL})\)
- \[\sum_{c \in A(p)} \max\{p(e) | p(e) \leq b(c, e)\}\] \((\text{UDP-Max-NPL})\)
- \[\sum_{c \in A(p)} p(\arg\min\{r_c(e) | e : p(e) \leq b(c, e)\})\] \((\text{UDP-Rank-NPL})\)
## 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 $r_c : \mathcal{U} \rightarrow \{1, \ldots, |\mathcal{U}|\}$.

For prices $p : \mathcal{U} \rightarrow \mathbb{R}_0^+$:

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

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

- $\sum_{c \in A(p)} \min\{p(e) \mid p(e) \leq b(c, e)\}$ (UDP-MIN-NPL)
- $\sum_{c \in A(p)} \max\{p(e) \mid p(e) \leq b(c, e)\}$ (UDP-MAX-NPL)
- $\sum_{c \in A(p)} p(\arg\min\{r_c(e) \mid e : p(e) \leq b(c, e)\})$ (UDP-RANK-NPL)

Given a price ladder constraint (PL), $p(e_1) \leq \cdots \leq p(e_{|\mathcal{U}|})$, 

UDP-$\{\text{MIN,MAX,RANK}\}$-PL asks for prices $p$ satisfying PL.
UDP introduced as non-parametric multi-product pricing in [1].


**Udp-Min-\{Pl,Npl\}**: 
- **Udp-Min-Pl** poly-time for uniform budgets consumers [1].
- **Udp-Min-Npl** APX-hard, has $O(\log |C|)$-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].


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


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


**Udp-Min-\{Pl,Npl\}:**

- **Udp-Min-Pl** poly-time for uniform budgets consumers [1].
- **Udp-Min-Npl** APX-hard, has $\mathcal{O}(\log |C|)$-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].


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


**UDP-Min-{Pl,Npl}**:  
- **UDP-Min-Pl** poly-time for uniform budgets consumers [1].
- **UDP-Min-Npl** APX-hard, has $O(\log |C|)$-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: $\text{UDP-MIN-NPL}$ has $O(\log |C|)$-approx [2]. We prove:

**Theorem**

$\text{UDP-MIN-}\{\text{PL, NPL}\}$ is not approximable within $O(\log^\varepsilon |C|)$ for some $\varepsilon > 0$, unless $\text{NP} \subseteq \text{DTIME}(n^{O(\log \log n)})$. 
Recall: $\text{UDP-MIN-NPL}$ has $O(\log |C|)$-approx [2]. We prove:

**Theorem**

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

**Sketch of Proof:**

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

Recall: \textsc{Udp-Min-Npl} has $O(\log |C|)$-approx [2]. We prove:

**Theorem**

\textsc{Udp-Min-\{Pl,Npl\}} is not approximable within $O(\log^\varepsilon |C|)$ for some $\varepsilon > 0$, unless NP $\subseteq$ DTIME($n^{O(\log \log n)}$).

**Sketch of Proof:**

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

**Proposition [Alon, Feige, Wigderson, Zuckerman’95]**

$n \in \mathbb{N}$, $\mathcal{G} = \{G : G = (V, E) \text{ with max degree } O(\log n), |V| = n\}$. There is $\varepsilon > 0$, s.t. $O(\log^\varepsilon n)$-approx to $\alpha(G)$ is NP-hard for $G \in \mathcal{G}$. 
Recall: $\text{UDP-MIN-NPL}$ has $O(\log |C|)$-approx [2]. We prove:

**Theorem**

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

**Sketch of Proof:**

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

**Proposition [Alon, Feige, Wigderson, Zuckerman’95]**

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

Given $G \in \mathcal{G}$, we reduce finding $\alpha(G)$ to $\text{UDP-MIN-PL}$. 
Recall: $\text{UDP-MIN-NPL}$ has $\mathcal{O}(\log |C|)$-approx [2]. We prove:

**Theorem**

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

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

**Proposition [Alon, Feige, Wigderson, Zuckerman’95]**

$n \in \mathbb{N}$, $\mathcal{G} = \{G : G = (V, E) \text{ with max degree } \mathcal{O}(\log n), |V| = n\}$. 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 $\text{UDP-MIN-PL}$.

Assume a.c.: $\text{UDP-MIN-PL}$ has $\mathcal{O}(\log^{\varepsilon-\delta} |C|)$-approx for some $\delta > 0$. 
Recall: $\textsc{UDP-MIN-NPL}$ has $\mathcal{O}(\log |C|)$-approx [2]. We prove:

**Theorem**

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

**Sketch of Proof:**

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

**Proposition [Alon, Feige, Wigderson, Zuckerman’95]**

$n \in \mathbb{N}$, $\mathcal{G} = \{G : G = (V, E) \text{ with max degree } \mathcal{O}(\log n), |V| = n\}$. 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 $\textsc{UDP-MIN-PL}$.

Assume a.c.: $\textsc{UDP-MIN-PL}$ has $\mathcal{O}(\log^{\varepsilon-\delta} |C|)$-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 $\Delta$, def instance of $\text{UDP-MIN-PL}$:
Sketch of Proof:

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

\[ V = V_0 \cup \ldots \cup V_\Delta: (\Delta + 1)-\text{vertex-coloring of } G. \]
Sketch of Proof:

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

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

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

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

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

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

$\mathcal{V}(v_{ij}) = \{v_{k\ell} | \{v_{ij}, v_{k\ell}\} \in E \text{ and } k < i\} = \text{vertices adjacent to } v_{ij}$ in color class with index $< i$. 
Sketch of Proof:

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

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

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

$\mathcal{V}(v_{ij}) = \{v_{k\ell} | \{v_{ij}, v_{k\ell}\} \in E \text{ and } k < i\}$ = 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:

Let $G = (V, E) \in \mathcal{G}$ of max degree $\Delta$, define instance of $\text{UDP-MIN-PL}$:

$$V = V_0 \cup \cdots \cup V_\Delta: (\Delta + 1)\text{-vertex-coloring of } G.$$ 

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

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

**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:

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

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

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

$\mathcal{V}(v_{ij}) = \{v_{k\ell} | \{v_{ij}, v_{k\ell}\} \in E \text{ and } k < i\} =$ 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 $\gamma = \mu^{-\Delta - 1}/n$. 
Sketch of Proof:

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

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

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

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

**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

$$C_{ij} = \{ c_{ij}^t | t = 0, \ldots, \mu^{\Delta-i} - 1 \}$$

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

$$C_{ij} = \{c_{ij}^t \mid t = 0, \ldots, \mu^{\Delta-i} - 1\}$$

of identical consumers with budgets

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

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

$$C = C_0 \cup \ldots \cup C_\Delta,$$

where $C_i = \bigcup_j C_{ij}$. 
Sketch of Proof:
Illustration of the construction:
Sketch of Proof: **Soundness:** $\text{opt}_{\text{UDP}} \geq \alpha(G)$, that is:

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

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:** $\text{opt}_{\text{UDP}} \geq \alpha(G)$, that is:

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

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 $C_{ij}$.

($\forall v_{k\ell} \in V(v_{ij}) : v_{k\ell} \notin V'$)
Sketch of Proof: **Soundness:** $\text{opt}_{UDP} \geq \alpha(G)$, that is:

\[
\text{(large IS in } G \text{) } \Rightarrow \text{(high revenue in } UDP)\]

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 $C_{ij}$.

- $(\forall v_{k\ell} \in V(v_{ij}) : v_{k\ell} \notin V')$

$\Rightarrow$ each consumer $c_{ij} \in C_{ij}$ can afford product $e_{ij}$ at price $p_{ij}$
Sketch of Proof: **Soundness:** \( \text{opt}_{\text{UDP}} \geq \alpha(G) \), that is:

\[
\text{(large IS in } G \text{)} \implies \text{(high revenue in } \text{UDP)}
\]

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

Consider \( v_{ij} \in V' \) and corresponding consumers \( C_{ij} \).
\( (\forall v_{k\ell} \in V(v_{ij}) : v_{k\ell} \notin V') \)
\( \implies \text{each consumer } c_{ij}^{t} \in C_{ij} \text{ can afford product } e_{ij} \text{ at price } p_{ij} \)
\( \implies \text{prices of products } e_{k\ell} \text{ exceed thresholds } p_{k\ell} \text{ of each } c_{ij}^{t} \)
Sketch of Proof: **Soundness:** $\text{opt}_{UDP} \geq \alpha(G)$, that is:

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

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 $C_{ij}$.

($\forall v_{k\ell} \in V(v_{ij}) : v_{k\ell} \notin V'$)

$\Rightarrow$ each consumer $c_{ij}^t \in C_{ij}$ 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:** \( \text{opt}_{\text{UDP}} \geq \alpha(G) \), that is:

\[
(\text{large IS in } G) \Rightarrow (\text{high revenue in } \text{UDP})
\]

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 \( C_{ij} \).

(\( \forall v_{k\ell} \in V(V_{ij}) : v_{k\ell} \notin V' \))

\( \Rightarrow \) each consumer \( c_{ij}^t \in C_{ij} \) 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 \( C_{ij} \geq |C_{ij}| \cdot p_{ij} = \mu^{\Delta-i} (\mu^{i-\Delta} + j\gamma) \geq 1 \)
Sketch of Proof: **Soundness:** \( \text{opt}_{\text{UDP}} \geq \alpha(G) \), that is:

\[
\text{(large IS in } G \text{) } \Rightarrow \text{(high revenue in } \text{UDP)}
\]

For an IS \( V' \) of \( G \), define prices \( p \):

for \( v_{ij} \in V' \) set \( p(e_{ij}) = p_{ij} \), else set \( p(e_{ij}) = p_{ij} + \gamma \).

\( (p_{ij}'s \text{ differ by } \geq \gamma) \Rightarrow \text{prices } p(\cdot) \text{ fulfill PL} \)

Consider \( v_{ij} \in V' \) and corresponding consumers \( C_{ij} \).

\( (\forall v_{kl} \in V(v_{ij}) : v_{kl} \notin V') \)

\( \Rightarrow \text{each consumer } c_{ij}^t \in C_{ij} \text{ can afford product } e_{ij} \text{ at price } p_{ij} \)

\( \Rightarrow \text{prices of products } e_{kl} \text{ exceed thresholds } p_{kl} \text{ of each } c_{ij}^t \)

\( \Rightarrow e_{ij} \text{ is the product with smallest price that any } c_{ij}^t \text{ can afford} \)

\( \Rightarrow \text{revenue of consumers } C_{ij} \geq |C_{ij}| \cdot p_{ij} = \mu^{\Delta - i} (\mu^{i - \Delta} + j \gamma) \geq 1 \)

\( \Rightarrow \text{prices } p \text{ result in revenue } \geq |V'|, \text{ so } \text{opt}_{\text{UDP}} \geq \alpha(G) \)
Sketch of Proof: Completeness: \( |V'| \geq \frac{1}{4} \text{revenue}(C) \), \( V'\)-IS:

\[
(\text{high revenue in } \text{UDP}) \Rightarrow (\text{large IS in } G)
\]

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

W.l.o.g.: price of each product \( e_{ij} \in \{p_{ij}, p_{ij} + \gamma\} \)
Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} revenue(C)$, **$V'$-IS:**

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

Let $p()$-prices found by approx algo, $r(C)$, $r(C_{ij})$, $r(c^t_{ij})$-revenues.

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

$C^+ \overset{df}{=} \{c^t_{ij} | r(c^t_{ij}) = p_{ij}\}$, $C^- = C \setminus C^+$. 
Sketch of Proof: **Completeness:** \( |V'| \geq \frac{1}{4} \text{revenue}(C) \), \( V'-\text{IS} \):

\[
\text{(high revenue in } \text{UDP} \text{) } \Rightarrow \text{(large IS in } G \text{)}
\]

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

\( C^+ \overset{df}{=} \{c_{ij}^t | r(c_{ij}^t) = p_{ij}\}, C^- = C \setminus C^+ \). Obs: \( \forall i, j: C_{ij} \subseteq C^+ \text{ or } C_{ij} \subseteq C^- \).
Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} revenue(C)$, $V'$-IS:

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

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

$C^+ \overset{df}{=} \{c_{ij}^t | r(c_{ij}^t) = p_{ij}\}$, $C^- = C \setminus C^+$. Obs: $\forall i,j$: $C_{ij} \subseteq C^+$ or $C_{ij} \subseteq C^-$. **Goal:** define $V' = \{v_{ij} | C_{ij} \subseteq C^+\}$, show $V'$–large IS.
Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4}\text{revenue}(C)$, $V'$-IS:

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

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

$C^+ \overset{df}{=} \{c_{ij}^t \mid r(c_{ij}^t) = p_{ij}\}$, $C^- = C \setminus C^+$. Obs: $\forall i, j: C_{ij} \subseteq C^+$ or $C_{ij} \subseteq C^-$. **Goal:** define $V' = \{v_{ij} \mid C_{ij} \subseteq C^+\}$, show $V'$—large IS.

Obs: $r(C^-) = \sum_{C_{ij} \subseteq C^-} r(C_{ij}) \leq \frac{n}{2(\Delta+1)}$
Sketch of Proof: **Completeness:** \( |V'| \geq \frac{1}{4} \text{revenue}(C) \), \( V' \)-IS:

\[
(\text{high revenue in } \text{UDP}) \Rightarrow (\text{large IS in } G)
\]

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

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

\[
C^+ \overset{df}{=} \{c_{ij}^t | r(c_{ij}^t) = p_{ij}\}, \quad C^- = C \setminus C^+.
\]

Obs: \( \forall i, j: C_{ij} \subseteq C^+ \) or \( C_{ij} \subseteq C^- \).

**Goal:** define \( V' = \{v_{ij} | C_{ij} \subseteq C^+\} \), show \( V' \)-large IS.

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

Obs: \( \alpha(G) \geq \frac{n}{(\Delta + 1)} \); easy to find prices resulting in revenue \( \frac{n}{(\Delta + 1)} \)
Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4}\text{revenue}(C)$, $V'$-IS:

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

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

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

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

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

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

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

$\Rightarrow$ wlog: $r(C) \geq \frac{n}{\Delta + 1}$
Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} \text{revenue}(C)$, $V'$-IS:

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

Let $p()$—prices found by approx algo, $r(C)$, $r(C_{ij})$, $r(c_{ij}^t)$—revenues.

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

$C^+ \overset{df}{=} \{c_{ij}^t \mid r(c_{ij}^t) = p_{ij}\}$, $C^- = C \setminus C^+$. Obs: $\forall i, j$: $C_{ij} \subseteq C^+$ or $C_{ij} \subseteq C^-$. **Goal:** define $V' = \{v_{ij} \mid C_{ij} \subseteq C^+\}$, show $V'$—large IS.

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

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

$\Rightarrow$ wlog: $r(C) \geq \frac{n}{\Delta + 1} \Rightarrow r(C^+) = r(C) - r(C^-) \geq (1/2)r(C)$
Sketch of Proof: Completeness: $|V'| \geq \frac{1}{4}\text{revenue}(C)$, $V'$-IS:

(high revenue in $\text{Udp}$) $\Rightarrow$ (large IS in $G$)

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

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

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

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

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

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

$\forall v_{ij} \in V'$: $C_{ij} \subseteq C^+$, $r(C_{ij}) = |C_{ij}| \cdot p_{ij} = \mu^{\Delta-i} (\mu^{i-\Delta} + j\gamma) \leq 2$
Sketch of Proof: **Completeness:** $|V'| \geq \frac{1}{4} \text{revenue}(C)$, $V'$-IS:

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

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

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

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

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

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

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

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

$\Rightarrow |V'| = |\{v_{ij} | C_{ij} \subseteq C^+\}| \geq (1/2)r(C^+) \geq (1/4)r(C)$
Sketch of Proof: **Finish:**

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

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

$r(C)$ is $O(\log^{\varepsilon-\delta} |C|)$-approx to $opt_{UDP}$
Sketch of Proof: Finish:

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

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

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

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

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

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

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**

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

\texttt{LOCALSEARCH}: Initialize $p$ arbitrarily and compute the optimal allocation $a$. While there exists product $e$ and price $p' \neq p(e)$ such that

$$r(p, a) < r([p | p(e) = p'], a'),$$

where $a'$ is the optimal allocation given prices $[p | p(e) = p']$, set $p(e) = p'$. 

Patrick Briest, Piotr Krysta

Pricing for Revenue Maximization in General Scenarios and in Network Flows
Summary (UDP):

- \textbf{UDP-MIN}\{-PL,NPL\} is \textit{intractable} (no const approx), even with PL
- \textbf{UDP-MAX}\{-PL,NPL\} is \textit{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.
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.

<table>
<thead>
<tr>
<th>$p(\mathcal{L}_1)$</th>
<th>$p(\mathcal{L}_2)$</th>
<th>prof</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>
**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.
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.
Clause Gadgets

Literal gadget $L_j$ for every occurrence of literal $l_j$, gives maximum profit 2 if $p(L_j) \in \{1, 2\}$. Connect $L_1$ and $L_2$, if $l_1, l_2$ form a clause.
Lemma

Let $C$ be a clause gadget with literal gadgets $L_1$ and $L_2$. Maximum profit obtainable from $C$ is 25 and is reached if and only if 
\{p(L_1), p(L_2)\} = \{1, 2\} or $p(L_1) = p(L_2) = 2$. $C$ gives profit 24 if $p(L_1) = p(L_2) = 1$.
Literal gadgets $\mathcal{L}(x_j), \mathcal{L}(\bar{x}_j)$ belonging to literals $x_j$ or $\bar{x}_j$ 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 $\mathcal{L}(x_j), \mathcal{L}(\bar{x}_j)$ belonging to literals $x_j$ or $\bar{x}_j$ 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 \( \Lambda(p) = \{ S \in S \mid p(e) \leq \delta(S) \ \forall \ e \in S \} \).

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

1. For every \( e \in U \) 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 $p^*$:

- **$\alpha(e, S)$**: profit of $e$ on conflicting set $S$
- **$\alpha(e)$**: profit of $e$ on non-conflicting sets
- **$\alpha(T)$**: total profit on conflicting set $T$

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

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

and remove \( e \) from all outgoing edges in this case.

**Step 2:** Let \( R = \{ e \mid \text{Out}(e) = \emptyset \} \), \( \mathcal{E} = \{ S \supseteq (V, W) \mid W \subseteq R \} \). 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 $\alpha$-value decreases by at most a factor $O(\ell)$.

$Opt$ of SUSP is upper bounded by $\ell$ times the $\alpha$-value of $p^*$'s conflict graph, thus:

**Theorem**

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