Pricing Methods for E-Content Servers

Work in Progress

Katie Batten, David John, and Errin W. Fulp

WAKE FOREST
UNIVERSITY

August 4, 2003

Network E-Content Servers

- QoS-enabled networks are becoming more prevalent
  - Making resource sensitive applications truly plausible
- An example is Video on Demand (VoD) systems
  - A server can offer a variety of videos to customers
  - Videos are encoded and provided at different rates (QoS)
  - Videos require different resource amounts for transmission
- The server has a **fixed** amount of bandwidth available
  - *Which videos should be transmitted to whom?*

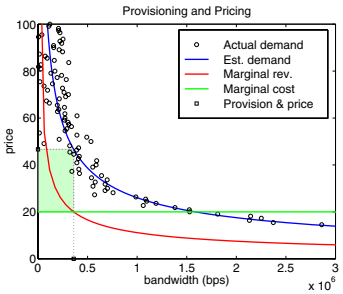
E-Content Pricing

- Accepted that consumers will pay for e-content (video)
  - Video server operator will seek to maximize profit
- Pricing resolves *which videos* and *to whom*, but...
  - *How are prices determined?* focus of this research
- First, must address how to model consumer demand
  - System consists of relatively few consumers
  - Each consumer will make a discrete choice

Demand Models

- Cobb-Douglas demand model

$$x_{i,t} = \beta_{i,t} \cdot \prod_{j \in Q} p_{j,t}^{-\alpha_{ij,t}}$$



  - Model is for large aggregates, does not fit a *smaller* problem
  - Demand curve for e-content is not necessarily continuous
- Logit and probit
  - Model single choice demand, but not very applicable
- Our problem is different, demand is discrete

Economic Model

- Assume an auction is used for selling e-content
  - Users bid for e-content from a batched system
  - Server is interested in determining auction winners **quickly**
- The server (seller) offers N product types, let $i = 1, ..., N$
  - Product i represents a video at a certain quality
  - Each product requires c_i resources (bandwidth)
  - Server has a fixed supply of S bandwidth, a **constraint**
- There are M consumers, where $j = 1, ..., M$
  - Consumer j offers a bid for product i denoted as $b_{i,j}$,
  - $b_{i,j}$ represents the max price the consumer will pay for i

Example

- Assume a server has 70 units of bandwidth available ($S = 70$)
- Server offers three products for sale ($N = 3$)
  - Assume product type 3 requires 30 resource units, product 2 requires 20 units, and product 1 requires 10 units
- Assume 9 consumers ($M = 9$) offer the following bids

Product Types		**Bids**
i	c_i	$b_{i,j}$
3	30	$b_{3,7} = 7$, $b_{3,8} = 9$, $b_{3,9} = 10$
2	20	$b_{2,4} = 4$, $b_{2,5} = 5$, $b_{2,6} = 5$
1	12	$b_{1,1} = 1$, $b_{1,2} = 2$, $b_{1,3} = 3$

  - Only one bid per customer

System Objective

- **Objective** of the seller is to maximize revenue
  - Determine price of product i for consumer j, denoted as $p_{i,j}$
  - Let A_i represent the set of consumers who accept $p_{i,j}$
  - Denote the number of consumers in set A_i as d_i $(d_i := |A_i|)$

$$\max_{p_{i,j}} \left\{ \sum_{i=1}^{N} \sum_{\forall j \in A_i} p_{i,j} \right\}, \quad \sum_{i=1}^{N} d_i \cdot c_i \leq S$$

- Note, the model does **not** price resources
  - Revenue per unit resource $(\frac{p_{i,j}}{d_i \cdot c_i})$ may differ per product
  - As a result the model captures **content** and **quality** value
- *How are prices determined?*... depends on the *type* of pricing

Discriminatory Market

- *Accepted* consumers have distinct prices equal to their bid

$$p_{i,j} = b_{i,j}, \forall j \in A_i$$

  - Consumers may pay different prices for the same product
- Determining the bids to accept = **Knapsack Problem** (KP)
  - Assume a knapsack has a volume of S units
  - Each bid is an object with value $b_{i,j}$ and volume c_i
  - Determining which bids to accept is a $\mathcal{NP}$-complete problem
  - Proven in [Sandholm & Suri, 2002]
- This is a problem since prices must be determined quickly

Example Solutions

Greedy (*by density*)

Prods		Accepted Bids
i	c_i	A_i
3	30	$p_{3,8} = 9$, $p_{3,9} = 10$
2	20	
1	12	
Total revenue:		19

Optimal

Prods		Accepted Bids
i	c_i	A_i
3	30	$p_{3,9} = 10$
2	20	$b_{2,5} = 5$, $b_{2,6} = 5$
1	12	
Total revenue:		20

- Heuristics exist to solve certain KP problems
  - [Yamada et al. , 2002] for smaller problems
  - [Sandholm & Suri, 2002] *linearize* the problem
- Discriminatory pricing is generally not preferred by consumers
  - Our model will focus on non-discriminatory pricing

Non-Discriminatory Market

- Seller determines a single price for each product type

$$p_{i,j} = p_{i,k}, \forall j,k \in A_i$$

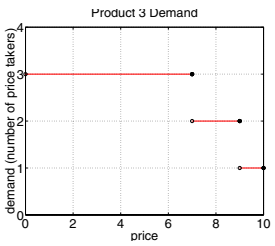
  - The optimization problem becomes

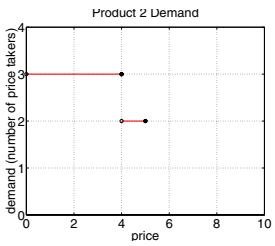
$$\max_{p_i} \left\{ \sum_{i=1}^{N} p_i \cdot d_i \right\}, \quad \sum_{i=1}^{N} d_i \cdot c_i \leq S$$

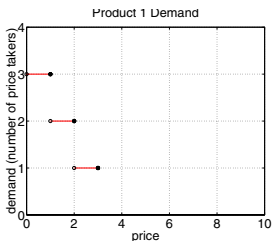
- Want to determine p_i^*, know supply *what is demand?*
  - Aggregate product demand curve is a step function

$$d_3(p_3) = \begin{cases} 3, & 0 \leq p_3 \leq 7 \\ 2, & 7 < p_3 \leq 9 \\ 1, & 9 < p_3 \leq 10 \\ 0, & 10 < p_3 \end{cases} \qquad d_2(p_2) = \begin{cases} 3, & 0 \leq p_2 \leq 4 \\ 2, & 4 < p_2 \leq 5 \\ 0, & 5 < p_2 \end{cases}$$

$$d_1(p_1) = \begin{cases} 3, & 0 \leq p_1 \leq 1 \\ 2, & 1 < p_1 \leq 2 \\ 1, & 2 < p_1 \leq 3 \\ 0, & 3 < p_1 \end{cases}$$







- Maximize revenue only at the end-points of each level
  - Only interested in the following price-demand pairs

Prouduct i	Price-Demand Pairs
3	$\{(7,3),(9,2),(10,1),(10+\epsilon,0)\}$
2	$\{(4,3),(5,2),(5+\epsilon,0)\}$
1	$\{(1,3),(2,2),(3,1),(3+\epsilon,0)\}$

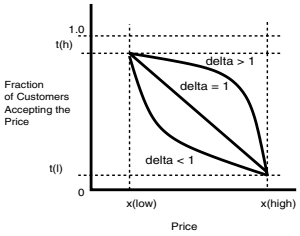
- *How difficult is it to determine the optimal prices?*

Disjunctively Constrained Knapsack Problem

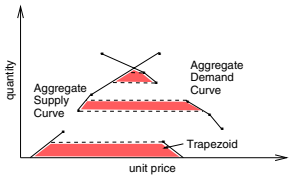
- A variation of the knapsack problem [Martello & Toth, 1990]
  - Certain objects are **incompatible**
  - Incompatible objects **cannot** be selected together
  - DCKP is $\mathcal{NP}$-complete [Martello & Toth, 1990]
- Determining the optimal non-discriminatory prices = DCKP
  - Let each demand-price pair represent an object
  - The value is the price and the volume is the demand
  - Only one object per product type is permitted in the knapsack
- Determining optimal non-discriminatory prices is $\mathcal{NP}$-complete
  - *Want to determine the prices quickly*

Other Approaches

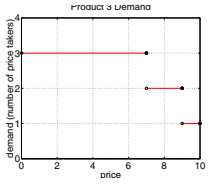
- Methods exist for **discriminatory** and non-discriminatory pricing
  - Continuous curves and inf. supply [Jagannathan & Almeroth, 2002]
  - Linear demand curves [Sandholm & Suri, 2002]



Jagannathan 2002



Sandholm 2002



discrete problem

- We are interested in addressing the discrete problem

Genetic Algorithms

- Genetic Algorithm (GA) is a search algorithm
  - First developed by John Holland in the 1970's [Holland, 1975]
  - Finds *better* solutions to an optimization problem
  - Uses random selection and survival of the fittest
- The general GA idea
  1. Start with a set (**population**) of solutions (**chromosomes**)
     - A chromosome has a fitness value (how good of a solution)
  2. Create a new population
     - Fitness to select parents and create children (new solutions)
     - Children are created via **crossover** and **mutation**
  3. New population becomes the old, repeat the process

Encoding a Chromosome

- Chromosome represents a solution, for non-discriminatory pricing
  - Solution consists of a set of prices $[p_N, ..., p_1]$
    $[p_3 = 11, p_2 = 5, p_1 = 2]$, $[10, 6, 4]$, $[10, 5, 4]$, ...
  - Prices can be any bid submitted for that product
    $p_3 = \{7, 9, 10, 11\}$, $p_2 = \{4, 5, 6\}$, $p_1 = \{1, 2, 3, 4\}$
  - Referred to as *value encoding*
- Fitness, f, of the chromosome is the revenue

$$f = \sum_{i=1}^{N} p_i \cdot d_i$$

- All chromosomes must be feasible

$$\sum_{i=1}^{N} c_i \cdot d_i \leq S$$

Creating a New Population

- Parents are randomly selected according to fitness
  - Higher fitness, higher probability of selection
  - Parents will create two children via **crossover** and **mutation**
- **Crossover** occurs with a certain probability
  - Randomly select a *location* (product index) in the chromosome
  - Part of parents are copied over to created child

Parents		Crossover Children
$[\overline{11},\ \overline{5, 2}]$ f:14 $\otimes$ $[\overline{10},\ \overline{6, 3}]$ f:13	=	$[11, 6, 3]$ f:3 and ~~$[10, 5, 2]$ f:22~~
$[\overline{11, 5},\ \overline{2}]$ f:14 $\otimes$ $[\overline{10, 6},\ \overline{3}]$ f:13	=	$[11, 5, 3]$ f:13 and $[10, 6, 2]$ f:12
$[11, 5, 2]$ f:14 $\otimes$ $[10, 6, 3]$ f:13	=	$[11, 5, 2]$ f:14 and $[10, 6, 3]$ f:13

  - Strong parents should create strong children

- For each child **mutation** occurs with a certain probability
  - Randomly select product and randomly change price
  - Price can change to any submitted bid for the product

Child	Select Products	After Mutation
$[11, 5, 2]$ f:14	$[11, 5, 2]$ f:14	$[10, 4, 2]$ f:24
$[10, 6, 3]$ f:13	$[10, 6, 3]$ f:13	$[11, 6, 3]$ f:3
$[11, 6, 3]$ f:3	$[11, 6, 3]$ f:3	$[10, 5, 4]$ f:20

  - Seeks to prevent converging at a local optimization point
- **Elitism** can be used, copy strongest parents to new population
  - Ensures the best previous solutions are not lost

Example Generations

Using the first example, there are 48 possible price combinations

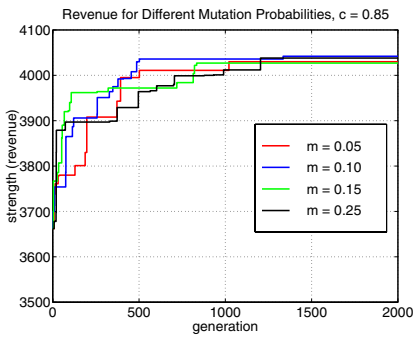
Generation 1	Generation 2	Generation 3	Generation 4
[10 6 2] f:14	[10 5 4] f:20	[10 5 4] f:20	[10 5 4] f:20
[10 6 2] f:14	[10 6 2] f:14	[10 6 2] f:14	[10 6 2] f:14
[10 6 2] f:14 →	[10 6 2] f:14 →	[10 6 2] f:14 →	[10 6 2] f:14 …
[11 5 4] f:10	[10 6 2] f:14	[10 6 2] f:14	[10 6 2] f:14
[10 6 4] f:10	[10 6 2] f:14	[10 6 2] f:14	[10 6 2] f:14
[11 6 3] f:3	[11 6 2] f:4	[11 6 2] f:4	[11 6 2] f:4

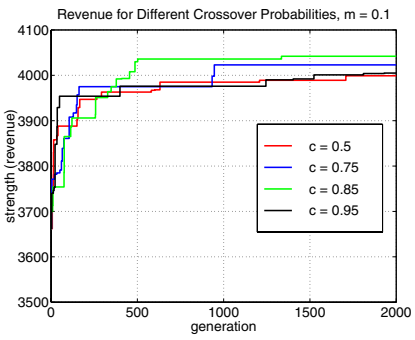
Generation 5	Generation 6	Generation 7	Generation 8
[10 5 4] f:20	[10 5 4] f:20	[10 5 4] f:20	[10 5 4] f:20
[10 6 2] f:14	[10 5 4] f:20	[10 5 4] f:20	[10 5 4] f:20
[10 6 2] f:14 →	[9 6 4] f:18 →	[10 5 4] f:20 →	[10 5 4] f:20 →
[10 6 2] f:14	[10 6 2] f:14	[10 5 4] f:20	[10 5 4] f:20
[10 6 4] f:10	[10 6 2] f:14	[9 6 4] f:18	[10 5 4] f:20
[10 6 4] f:10	[10 6 4] f:10	[10 6 2] f:14	[10 6 4] f:10

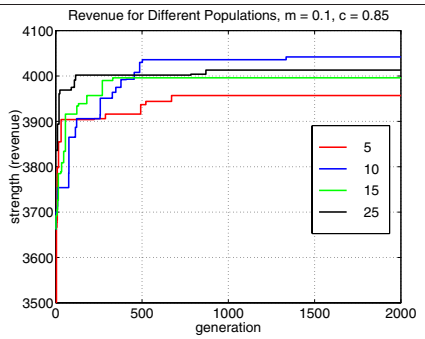
Generation 9
[10 5 4] f:20
[10 5 4] f:20
[10 5 4] f:20
[10 5 4] f:20
[10 5 4] f:20
[10 5 4] f:20

Larger Example

- Consider a server with 10 videos ($N = 10$)
  - Bandwidth supply is 500 Mbps ($S = 500$)
  - Approximately 20 bids per product ($M = 200$)
  - Number of price combinations is 7.18×10^{10}
- GA experiments consisted of the following
  - A population of 5, 10, or 20 chromosomes
  - Probability of mutation was 0.05, 0.1, 0.15, or 0.25
  - Probability of crossover was 0.5, 0.75, 0.85, or 0.95
- Interested in the highest revenue per generation
  - More iterations allow better solutions to be found
  - *More iterations require more time*







- Greedy revenue was 2620
  - GAs were twice as high
- Higher crossover prob. is better
- Lower mutation prob. is better
  - As $m \rightarrow 1$, GA→random search
- 16 msec/generation
  - Population = 10
- All simulations converged

Problem Review

- Pricing e-content given a server with bandwidth constraint
  - An auction format where users bid for a product type
- Determining clearing prices is $\mathcal{NP}$-complete (Knapsack Problem)
  - Both for discriminatory and non-discriminatory pricing
  - As a result, cannot use continuous optimization methods
- Genetic algorithms provide a means for determining prices
  - Search method based on *survival of the fittest*
  - Mutation probability, crossover probability, and population size
  - Scalable to different problem sizes (polynomial time)
- GA's have shown good results for non-discriminatory pricing
- *But this is a work in progress...*

Future Work

- Users may submit **multiple bids** for different products
  - Currently a user can only submit a single bid
  - Alter model to allow multiple choices
  - Demand curves are **still** discrete $\Rightarrow$ use GA
- Change model to allow **fixed pricing**
  - Prices are known *a priori*, based on past demands
  - Demand is again discrete, will continue to use GA
- **Tune GA** for performance
  - Currently using a simple GA
  - Use knowledge of the system to increase performance

References

Holland, J. H. 1975. Adaption in Natural and Artificial Systems. University of Michigan Press.

Jagannathan, Srinivasan, & Almeroth, Kevin C. 2002. Pricing and Resource Provisioning for Delivering E-Content On-Demand with Multiple Levels-of-Service. Pages 325 – 336 of: Proceedings of the ICQT.

Martello, S, & Toth, P. 1990. Knapsack Problems: Algorithms and Computer Implementations. John Wiley & Sons.

Sandholm, Toumas, & Suri, Subhash. 2002. Optimal Clearing of Supply/Demand Curves. In: Proceedings of the 13^{th} Annual International Symposium on Algorithms and Computation.

Yamada, Takeo, Kataoka, Seija, & Watanabe, Kohtaro. 2002. Heuristic and Exact Algorithms for Disjunctively Constrained Knapsack Problem. IPSJ, **43**(9).