An Efficient Solution Approach for Combinatorial Bandwidth Packing Problem with Queuing Delays

Sachin Jayaswal, Navneet Vidyarthi, Sagnik Das

W.P. No. 2014-12-05

December 2014

**INDIAN INSTITUTE OF MANAGEMENT
AHMEDABAD-380 015
INDIA**

An Efficient Solution Approach for Combinatorial Bandwidth Packing Problem with Queuing Delays

Sachin Jayaswal

Production & Quantitative Methods, Indian Institute of Management Ahmedabad
sachin@iimahd.ernet.in

Navneet Vidyarthi

Department of Supply Chain and Technology Management
John Molson School of Business, Concordia University
navneetv@jmsb.concordia.ca

Sagnik Das

Production & Quantitative Methods, Indian Institute of Management Ahmedabad
sagnikd@iimahd.ernet.in

Abstract

The combinatorial bandwidth packing problem (CBPP), arising in a telecommunication network with limited bandwidth, is defined as: given a set of requests, each with its potential revenue and consisting of calls with their bandwidth requirements, deciding (i) a subset of the requests to accept/reject, and (ii) a route for each call in an accepted request, so as to maximize total revenue earned. However, telecommunication networks are generally characterized by variability in the call (bits) arrival rates and service times, resulting in delays in the network. In this paper, we present a non-linear integer programming model for CBPP accounting for such delays. By using simple transformation and piecewise outer-approximation, we linearize the model, and present an efficient cutting plane based approach to solve the resulting linear mixed integer program to ϵ-optimality.

Keywords:OR in telecommunications; Bandwidth packing; Mixed integer programming; Queuing; Cutting Plane

An Efficient Solution Approach for Combinatorial Bandwidth Packing Problem with Queuing Delays

1 Introduction

The recent advent of bandwidth intensive telecommunication services like video conferencing, social networking, online interactive gaming, interactive television, real-time simulations, telemedicine, and collaborative software development using global virtual teams makes the efficient management of the available telecommunication network bandwidth critical. In most of such applications, users generally schedule their requests in advance (Parker and Ryan, 1993). In this respect, a critical problem that network administrators commonly face is: given a set of submitted call requests, each with its potential revenue and its associated bandwidth requirement, deciding (i) a subset of the calls to accept/reject, and (ii) a route for each accepted call, so as to maximize total revenue earned. This is the classical version of what is referred to in the literature as the bandwidth packing problem (BPP) (Cox et al., 1991; Laguna and Glover, 1993; Anderson et al., 1993; Parker and Ryan, 1993; Park et al., 1996). Several variants of BPP have been widely studied in the literature. Amiri (2005) presents a version of BPP involving scheduling of selected calls within given time windows. A multi-hour version of BPP accounting for the variation in traffic between peak and off-peak hours of the day is studied by (Amiri and Barkhi, 2000).

The bandwidth requirements of telecommunication calls involving data and video transmission are usually bursty (variable bit rates). In this respect, sole focus on revenue maximization in BPP may significantly affect the quality of service to users due to excessive utilization of existing network resources (bandwidth) (Amiri et al., 1999). This has been addressed in the literature by extending the classical BPP to account for the delays on links caused due to high bandwidth utilization. Amiri et al. (1999), Rolland et al. (1999), and Han et al. (2012) explicitly account for such network delays by incorporating queuing delay terms in their model. All of these papers model the links on the network as independent M/M/1 queues with the implicit underlying assumption that bit (of calls) arrivals are Poisson and their service times on links have exponential distribution. While Amiri et al. (1999) and Vidyarthi et al. (2014) discourage such delays in their model by penalizing them in the objective function, Rolland et al. (1999) and Han et al. (2012) impose an explicit constraint to limit maximum queuing delays. Bose (2009) presents a two-priority version of BPP, wherein the calls belonging to lower priority consume less bandwidth but are longer and generate lower revenue compared to those belonging to higher priority. For this, he models each link as a preemptive priority M/M/1 queue. Amiri (2003) extends the multi-hour BPP, earlier studied by Amiri and Barkhi (2000), with delay guarantees. Gavish and Hantler (1983) studied BPP with delays due to congestion, although the acceptance/rejection of calls is not a decision in their problem.

BPP and its variants studied in the literature mostly permit only a single path for each of the accepted calls, as applicable to video data, which makes the problem NP-hard (Parker and Ryan, 1993). Solution methods for BPP and its variants reported in the literature can be broadly classified into two groups: metaheuristics, and decomposition based approaches. Within metaheuristics, Tabu Search has been applied by Anderson et al. (1993) and Laguna and Glover (1993), while Cox et al. (1991) has reported the use of Genetic Algorithm. Within the decomposition based approaches, Lagrangian relaxation has been a popular choice, reported by Gavish and Hantler (1983), Rolland et al. (1999), Amiri et al. (1999), Amiri and Barkhi (2000), Amiri (2003), Amiri (2005), and Amiri and Barkhi (2012). Other decomposition based approaches reported are Column Generation (Parker and Ryan, 1993; Park et al., 1996; Villa and Hoffman, 2006), and Dantzig-Wolfe Decomposition (Han et al., 2012). Vidyarthi et al. (2014) report the use of Cutting Plane based method to deal with non-linearity in the model arising due to queuing delays.

An interesting generalization of the classical BPP presented byAmiri and Barkhi (2012) considers the case where each request consists of a set of calls, requiring one-to-many or many-to-many connections. The resulting problem, referred to as combinatorial bandwidth packing problem (CBPP), is defined as: given a set of requests, each with its potential revenue and consisting of calls with their bandwidth requirements, deciding (i) a subset of the requests to accept/reject, and (ii) a route for each call in an accepted request, so as to maximize total revenue earned in a telecommunication network with limited bandwidth. The combinatorial nature of the problem makes BPP, which is NP-hard, even harder to solve.

In this paper, we extend CBPP, as presented by Amiri and Barkhi (2012), to account for queuing delays on the links. For this, links in the network are modelled as independent single server queues with Poission arrivals and exponential service times (M/M/1 queues). This results in a non-linear integer programming (IP) model. We, therefore, reformulate the model, using simple transformation and piecewise linear outer-approximation, as a linear mixed integer program (MIP) with a large number of constraints. We further present an efficient cutting plane based solution approach, which performs very well in terms of optimality gap and computational time.

The remainder of the paper is organized as follows. In Section 2, we formally describe the problem and present its non-linear IP formulation. Section 3 describes an approach to linearize the model, followed by an exact cutting plane based method to solve the resulting linear MIP. Computational results are reported in Section 4. Section 5 concludes with possible directions for future research.

2 Problem Formulation

CBPP assumes a telecommunication network comprising of nodes $\{i : i \in N\}$ or $\{j : j \in N\}$ with a limited bandwidth Q_{ij} (bits per unit time) on every link $\{(i,j) : (i,j) \in E\}$. Given a set of requests $\{b : b \in B\}$, each with its potential revenue r^b and consisting of calls $\{c : c \in C_b\}$ between origin-destination pairs $O(b,c) - D(b,c)$ with bandwidth requirements d^{bc} (bits per unit time), the objective of CBPP is to select: (i) a subset of these requests; and (ii) a single path (sequence of links) to route each call of the selected requests, such that the total revenue generated from the accepted requests is maximized without violating the bandwidth capacities on the links. For this, let $V^b = 1$ if request b is accepted, 0 otherwise, and $X_{ij}^{bc} = 1$ if call c in request b is routed through a path that uses a directed link (i,j), 0 otherwise. The notations used in the problem are summarized below:

Parameters:

N	:	Set of nodes in the network indexed by i and j, where $i, j \in N$
E	:	Set of links (i, j) in the network, where $i, j \in N$
B	:	Set of requests indexed by b, where $b \in B$
C_b	:	Set of calls of request b indexed by c, where $c \in C$
$O(b, c)$	:	Origin node of call c of request b; $O(b, c) \in N$
$D(b, c)$	:	Destination node of call c of request b; $D(b, c) \in N$
d^{bc}	:	Demand (bits per unit time) of call c of request b
r^b	:	Potential revenue from request b
Q_{ij}	:	Bandwidth capacity (bits per unit time) of link (i, j)

Decision Variables:

$$V^b = 1 \text{ if request } b \text{ is accepted; 0 otherwise.}$$

$$X_{ij}^{bc} = \begin{array}{l} 1 \text{ if call } c \text{ of request } b \text{ is routed through a path that uses link } (i,j) \\ \text{ in the direction from } i \text{ to } j\text{; 0 otherwise.} \end{array}$$

Using the above notations, an adaptation of the IP formulation for CBPP introduced by Amiri and Barkhi (2012) can be stated as follows:

$$\max \sum_{b \in B} r^b V^b \tag{1}$$

$$\text{s.t.} \sum_{j:(i,j) \in E} \left(X_{ij}^{bc} - X_{ji}^{bc} \right) = \begin{cases} V^b & \text{if } i = O(b,c); \\ -V^b & \text{if } i = D(b,c); \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in N, b \in B, c \in C_b \tag{2}$$

$$\sum_{b \in B} \sum_{c \in C_b} d^{bc} (X_{ij}^{bc} + X_{ji}^{bc}) \leq Q_{ij} \qquad \forall (i,j) \in E : i < j \tag{3}$$

$$V^b \in \{0, 1\} \qquad \forall b \in B \tag{4}$$

$$X_{ij}^{bc} \in \{0, 1\} \qquad \forall (i,j) \in E, b \in B, c \in C_b \tag{5}$$

The objective function (1) is the total revenue from accepted requests. Constraint set (2) are the flow conservation equations on each node for each call. Constraint set (3) ensures that the total demand on each link is less than its available bandwidth. Constraint sets (4) and (5) are binary restrictions on the variables. As discussed in section 1, the above model may result in poor response times due to excessive utilization of link capacities (bandwidth) in presence of variable call bit rates d^{bc}. To overcome this drawback, instead of maximizing the total revenue, we determine the optimal trade-off between the revenue and the response time delay due to excessive bandwidth usage.

To model response time delays in the network, we assume that the arrivals of bits associated with calls on the network occur according to a Poisson process. Further, the nodes are assumed to have unlimited buffers to store calls waiting for transmission (due to finite capacities on links). We also assume that the call lengths (in bits) follow an exponential distribution with a mean $1/\mu$. The service rate (in bits per second) of the link (i, j) is proportional to the capacity of the link Q_{ij}. Then, the service time per call on link (i, j) also follows an exponential distribution with a mean $1/\mu Q_{ij}$. Under these assumptions, each link can be modelled as a single server M/M/1 queue. These assumptions are in line with the literature (Gavish and Hantler, 1983; Amiri et al., 1999; Rolland et al., 1999; Han et al., 2012). Thus, the arrival of bits on link (i, j), due to superposition of several Poisson processes, follows a Poisson process with a rate $\sum_{b\in B}\sum_{c\in C_b} d^{bc}(X_{ij}^{bc} + X_{ji}^{bc})$, and the arrival rate of calls on link (i, j) is $\lambda_{ij} = \mu\sum_{b\in B}\sum_{c\in C_b} d^{bc}(X_{ij}^{bc} + X_{ji}^{bc})$. The average utilization of link (i, j) is given by:

$$\rho_{ij} = \frac{\lambda_{ij}}{\mu Q_{ij}} = \frac{\sum_{b\in B}\sum_{c\in C_b} d^{bc}(X_{ij}^{bc} + X_{ji}^{bc})}{Q_{ij}} \tag{6}$$

Under steady state conditions ($\rho_{ij} < 1$) and first-come first-serve (FCFS) queuing discipline, the *mean sojourn time* (waiting time in queue + service time) of a call on link (i, j), which is modelled as an M/M/1 queue, is given by: $E[w_{ij}] = \frac{\rho_{ij}}{1-\rho_{ij}}$. The total network delay (W) is, therefore, given by:

$$W = \sum_{(i,j)\in E} \frac{\rho_{ij}}{1-\rho_{ij}} = \sum_{(i,j)\in E} \frac{\sum_{b\in B}\sum_{c\in C_b} d^{bc}(X_{ij}^{bc} + X_{ji}^{bc})}{Q_{ij} - \sum_{b\in B}\sum_{c\in C_b} d^{bc}(X_{ij}^{bc} + X_{ji}^{bc})} \tag{7}$$

Using the above notations, the mathematical model for CBPP with queuing delays can be stated as:

$$[P] : \max \ Z(\mathbf{X}, \mathbf{V}) = \sum_{b \in B} r^b V^b - K \sum_{(i,j) \in E} \frac{\sum_{b \in B} \sum_{c \in C_b} d^{bc} \left(X_{ij}^{bc} + X_{ji}^{bc} \right)}{Q_{ij} - \sum_{b \in B} \sum_{c \in C_b} d^{bc} \left(X_{ij}^{bc} + X_{ji}^{bc} \right)} \tag{8}$$

$$\text{s.t. } (2) - (5)$$

The objective function (8) is the net revenue (gross revenue - total queuing delay cost) from the accepted requests. The second term in (8) captures the average queuing delay cost due to all accepted calls, where K is the queuing delay cost per unit time. The above formulation $[P]$ is a non-linear integer program due to the queueing delay term in the objective function. In the following section, we present an alternate method to solve it, which is efficient and exact.

3 Solution Approach

3.1 Linear Reformulation

To linearize the queueing delay term in (8), we define non-negative auxiliary variables R_{ij}, such that:

$$R_{ij} = \frac{\sum_{b \in B} \sum_{c \in C_b} d^{bc} (X_{ij}^{bc} + X_{ji}^{bc})}{Q_{ij} - \sum_{b \in B} \sum_{c \in C_b} d^{bc} (X_{ij}^{bc} + X_{ji}^{bc})} \qquad \forall (i,j) \in E \tag{9}$$

This implies,

$$\sum_{b \in B} \sum_{c \in C_b} d^{bc} (X_{ij}^{bc} + X_{ji}^{bc}) = \frac{R_{ij}}{1 + R_{ij}} Q_{ij} \qquad \forall (i,j) \in E \tag{10}$$

For a given set of points $h \in H$, the function $f(R_{ij})$ can be approximated, using Taylor series approximation, by a set of piecewise linear functions that are tangent to $f(R_{ij})$ at points $(R_{ij}^h)_{h \in H}$ as follows: $f(R_{ij}) \approx f(R_{ij}^h) + f'(R_{ij}^h)(R_{ij} - R_{ij}^h) = \frac{1}{(1+R_{ij}^h)^2} R_{ij} + \frac{(R_{ij}^h)^2}{(1+R_{ij}^h)^2}$. Since $f(R_{ij})$ is concave in $R_{ij} \in [0, \infty)$, it can be expressed as follows:

$$\frac{R_{ij}}{1 + R_{ij}} = \min_{h \in H} \left\{ \frac{1}{(1 + R_{ij}^h)^2} R_{ij} + \frac{(R_{ij}^h)^2}{(1 + R_{ij}^h)^2} \right\}$$

This is equivalent to the following set of constraints:

$$\frac{R_{ij}}{1+R_{ij}} \leq \frac{1}{(1+R_{ij}^h)^2}R_{ij} + \frac{(R_{ij}^h)^2}{(1+R_{ij}^h)^2}, \qquad \forall (i,j) \in E, h \in H$$

Using (10), the above set of constraints can be rewritten as:

$$\sum_{b \in B}\sum_{c \in C_b} d^{bc}(X_{ij}^{bc} + X_{ji}^{bc}) - \frac{Q_{ij}}{(1+R_{ij}^h)^2}R_{ij} \leq \frac{Q_{ij}(R_{ij}^h)^2}{(1+R_{ij}^h)^2} \qquad \forall (i,j) \in E, h \in H \tag{11}$$

provided $\exists h \in H$ such that (11) holds with equality.

The above substitutions result in the following linear MIP model:

$$[PL(H)]: \ \max \sum_{b \in B} r^b V^b - K \sum_{(i,j) \in E} R_{ij} \tag{12}$$

$$\text{s.t. } (2)-(5), (11)$$

$$R_{ij} \geq 0 \qquad \forall (i,j) \in E \tag{13}$$

For equivalence between $[P]$ and $[PL(H)]$, there should exist at least one $h \in H$ such that (11) holds with equality. Proposition 1 confirms that there indeed exists one such $h \in H$ at optimality.

Proposition 1: *There exists at least one of the constraints (11) in $[PL(H)]$ that will be binding at optimality.*

After rearranging the terms, (11) can be rewritten as:

$$R_{ij} \geq (1+R_{ij}^h)^2 \frac{\sum_{b \in B}\sum_{c \in C_b} d^{bc}(X_{ij}^{bc} + X_{ji}^{bc})}{Q_{ij}} - (R_{ij}^h)^2 \tag{14}$$

Since R_{ij} appears in the objective function with a negative coefficient, $[PL(H)]$ attains its optimum value only when R_{ij} is minimized. This implies that $\forall (i,j) \in E$, $\exists h \in H$ such that (14) holds with equality if $(1+R_{ij}^h)^2 \frac{\sum_{m \in M} d^{bc}(X_{ij}^{bc}+X_{ji}^{bc})}{Q_{ij}} - (R_{ij}^h)^2 \geq 0$, else $R_{ij} = 0$.

Further,

$$\begin{aligned}
0 &\leq (1+R_{ij}^h)^2 \frac{\sum_{b\in B}\sum_{c\in C_b} d^{bc}(X_{ij}^{bc}+X_{ji}^{bc})}{Q_{ij}} - (R_{ij}^h)^2 \\
&= (1+R_{ij}^h)^2\rho_{ij} - (R_{ij}^h)^2 \quad \text{(using (6))} \\
&= (\rho_{ij}-1)(R_{ij}^h)^2 + 2\rho_{ij}R_{ij}^h + \rho_{ij} \\
&\Leftrightarrow R_{ij}^h \in \left[0, \frac{\rho_{ij}+\sqrt{\rho_{ij}}}{1-\rho_{ij}}\right] \ \forall h \in H \quad \text{(since } \rho_{ij} \leq 1 \text{ and } R_{ij} \geq 0 \text{ using (9))}
\end{aligned}$$

Thus, to prove that $\exists h \in H$ such that (11) holds with equality, we need to show that $R_{ij}^h \in \left[0, \frac{\rho_{ij}+\sqrt{\rho_{ij}}}{1-\rho_{ij}}\right]$. Since R_{ij}^h is an approximation to R_{ij}, we obtain:

$$\begin{aligned}
0 \leq R_{ij}^h \approx R_{ij} &= \frac{\lambda_{ij}}{\mu Q_{ij} - \lambda_{ij}} \qquad \text{(using (9))} \\
&= \frac{\rho_{ij}}{1-\rho_{ij}} \\
&\leq \frac{\rho_{ij}+\sqrt{\rho_{ij}}}{1-\rho_{ij}}
\end{aligned}$$

This proves that $\forall (i,j) \in E$, $\exists h \in H$ such that, at optimality, (11) always holds with equality.

3.2 Bounds and Exact Approach

For any given subset of points $\{R_{ij}^h\}_{h\in H^q \subseteq H}$, $[PL(H^q)]$ is the relaxation of the full problem $[PL(H)]$. Hence, $v(PL(H^q)) \geq v(PL(H))$, where $v(\bullet)$ represents the optimal objective function value of the maximization problem $(\bullet)$. Thus, $v(PL(H^q))$ provides an upper bound to $[PL(H)]$, given by:

$$UB = v(PL(H^q)) = \sum_{b\in B} r^b V^{bq} - K \sum_{(i,j)\in E} R_{ij}^q \tag{15}$$

For any given subset of points $\{R_{ij}^h\}_{h\in H^q \subseteq H}$, the part of the solution $(\mathbf{X}^q, \mathbf{Y}^q)$ to $[PL(H^q)]$ is also a feasible solution to $[P]$, and hence the objective function (8) evaluated at the solution

$(\mathbf{X}^q, \mathbf{V}^q)$, gives a lower bound to $[PN]$. Hence, a lower bound to $[PN]$ is given by:

$$LB = Z(\mathbf{X^q}, \mathbf{V^q}) = \sum_{b \in B} r^b V^{bq} - K \sum_{(i,j) \in E} \frac{\sum_{b \in B} \sum_{c \in C_b} d^{bc}(X_{ij}^{bcq} + X_{ji}^{mq})}{Q_{ij} - \sum_{b \in B} \sum_{c \in C_b} d^{bc}(X_{ij}^{bcq} + X_{ji}^{bcq})} \tag{16}$$

The model $[PL(H)]$ consists of a large number of constraints (11). However, not all of them need to be generated a priori. The algorithm starts with an initial subset of carefully selected points, $H^q \subset H$. The resulting model $[PL(H^q)]$ is solved and the upper bound (UB^q) and the lower bound (LB^q) are computed using equations (17) and (18) respectively. If the upper bound (UB^q) equals the best known lower bound (LB^q) within accepted tolerance (ϵ) at any given iteration q, then $(\mathbf{X}^q, \mathbf{V}^q)$ is an optimal solution to $[P]$ and the algorithm is terminated. Otherwise, a new set of candidate points $R_{ij}^{h_{new}}$ is generated using the current solution ($\mathbf{X}^q$) as follows:

$$R_{ij}^{h_{new}} = \frac{\sum_{b \in B} \sum_{c \in C_b} d^{bc}(X_{ij}^{bcq} + X_{ji}^{bcq})}{Q_{ij} - \sum_{b \in B} \sum_{c \in C_b} d^{bc}(X_{ij}^{bcq} + X_{ji}^{bcq})}$$

This generated set of "cuts /constraints" eliminates the best solution found so far and improve the upper bound on the remaining solutions. This new set of points is appended to $(R_{ij}^h)_{H^q \subset H}$ and the procedure is repeated until the gap between the current upper bound and the best lower bound is within the tolerance limits. The algorithm is outlined below:

Algorithm 1 Solution Algorithm for $[PL(H)]$

```
1: $q \leftarrow 1; UB^{q-1} \leftarrow +\infty; LB^{q-1} \leftarrow -\infty;$
2: Choose an initial set of points $\{R_{ij}^h\}_{h \in H^q}$ to approximate $R_{ij}/(1 + R_{ij})\ \ \forall (i,j) \in E$ .
3: while $(UB^{q-1} - LB^{q-1})/UB^{q-1} > \epsilon$ do
4:     Solve $[PL(H^q)]$ to obtain $(\mathbf{X}^q, \mathbf{Y}^q)$.
5:     Update the upper bound: $UB^q \leftarrow v(PL(H^q))$.
6:     Update the lower bound: $LB^q \leftarrow \max\{LB^{q-1}, Z(\mathbf{X}^q, \mathbf{Y}^q)\}$.
7:     Compute new points: $R_{ij}^{h_{new}} = \frac{\sum_{b \in B} \sum_{c \in C_b} d^{bc}(X_{ij}^{bcq} + X_{ji}^{bcq})}{Q_{ij} - \sum_{b \in B} \sum_{c \in C_b} d^{bc}(X_{ij}^{bcq} + X_{ji}^{bcq})}\ \ \forall (i,j) \in E$
8:     $H^{q+1} \leftarrow H^q \cup \{h_{new}\}$
9:     $q \leftarrow q + 1$
10: end while
```

The initial set H^1 in the Algorithm 1 can be empty. However, our preliminary computational experiments show that starting with a set of carefully selected cuts H^1 helps in faster convergence of the algorithm.

4 Computational Results

We conduct computational experiments to evaluate the performance of the proposed exact solution procedure. The solution procedure is coded in Visual C++, while $[PL(H^q)]$ at every iteration q is solved using IBM ILOG CPLEX 12.4. The experiments are conducted on a machine with the following specifications: Intel Core i5-3230M, 2.60 GHz CPU; 4.00 GB RAM; Windows 64-bit Operating System. The computational performance of proposed solution approach on problems with varying sizes are presented below.

In the implementation of the algorithm, we start with an initial set of cuts, generated at points $h \in H^1$, for the function $f(R_{ij}) = \frac{R_{ij}}{1+R_{ij}}$. These cuts are generated based on the piecewise linear approximation by $\hat{f}(R_{ij})$ of the function $f(R_{ij})$ such that the approximation error measured by $\hat{f}(R_{ij}) - f(R_{ij})$ is at most 0.001 (see Elhedhli (2005)). This is in part motivated by our initial computational results, which show that the option of starting with a carefully chosen initial set of cuts improves the performance of the solution approach substantially. Hence, in all the test problems, we use 32 cuts which corresponds to a maximum approximation error $(\hat{f}(R_{ij}) - f(R_{ij}))$ of 0.001.

The computational results are reported for a set of test instances generated using the scheme proposed by Amiri and Barkhi (2012), and references therein. The links are generated such that each node has a degree equal to 2, 3, or 4 with probability of 0.6, 0.3, and 0.1, respectively and the network is connected. This results in the number of links varying between 30 and 160. Each link in the network is randomly assigned a capacity (Q_{ij}) equal to 960, 1920, 5000, or 10,800 with equal probabilities. The number of calls per request b is generated randomly from the uniform distribution between 2 and 10; as such the minimum and maximum numbers of calls per request are 2 and 20, respectively. The bandwidth requirement for each call c of a request b is generated randomly from the uniform distribution between 20 and 40.

Table 1 shows the computational results for various network sizes with the number of nodes $|N|$ varying from 10 to 100 in steps of 10 while the number of available requests, $|B|$ is set to 500. Table 2 presents the results for the number of available requests set to 1000 and 1500. The queuing delay cost per unit time (K) is selected from the set $\{1, 10, 50, 100, \text{to } 500\}$.

A total of 90 instances are solved. The algorithm is terminated once the Gap (%), computed as $\frac{\text{Upper Bound - Lower Bound}}{\text{Lower Bound}} \times 100\%$, falls below 0.1 or CPU time exceeds 3600 seconds. The table reports the gross revenue (GR), delay costs (DC) expressed as percentage of gross revenue, average and maximum utilization (Util.) of the links, Gap (%), number of iterations ($Iter.$), and computational time (CPU(s)). The upper and lower bounds at every iteration are computed using (15) and (16) respectively. The observations from our experimental results are as follows:

- Results in Table 1 indicate that 24 out of 50 instances are solved with an optimality gap ranging from 0 to 0.09% in less than one hour of CPU time, whereas the optimality gap for the 21 instances that are terminated after one hour of CPU time is in the range 0.031% - 4.328%. The remaining 5 instances failed due to insufficient memory. Similarly, results in Table 2 indicate that 15 out of 40 instances are solved within an optimality gap ranging from 0.001% to 0.048% in less than one hour of CPU time, whereas 17 instances were terminated after one hour of CPU time, resulting in solutions within a gap ranging from 0.035% to 1.166%. The remaining 8 instances failed due to insufficient memory.

- The computation times highlight the efficiency of our proposed solution approach for different values of parameters, whereas the number of iterations imply that only a very few of the constraints in (11) are required. As the number of nodes ($|N|$), the number of calls ($|C|$), and the number of requests ($|B|$) increase, the problem becomes difficult to solve, requiring more CPU time.

- Results in Tables 1 and 2 also show the effects of changes in the number of submitted requests (B). For a given problem size ($|N|$), we vary the number of submitted requests from 500 to 1,000 to 1,500. As the delay cost per unit time K increases, we observe fewer requests are accepted and fewer calls are routed through the network accompanied by reduction in the average link utilization. However, this observation cannot be generalized since with increase in K, reduction in average link utilization is also possible by accepting more requests/calls with lower bandwidth requirements.

Table 1: Computational Results for Instances with 500 Requests

$\|N\|$	$\|B\|$	$\|C\|$	K	$\|B_s\|$	$\|C_s\|$	Gross	Delay Cost (DC)	Avg.	Max.	Gap	Iter.	CPU(s)
						Revenue (GR)	(% of GR)	Util.(%)	Util.(%)	(%)		
10	500	2980	1	72	356	14706	1.7	55.8	98.8	0.009	2	444
			10	68	336	14199	5.1	52.2	95.5	0.002	3	143
			50	64	314	13335	12.6	47.8	88.9	0.009	2	106
			100	59	293	12511	17.4	43.7	83.2	0.002	2	77
			500	43	219	9248	42.4	29.9	62.4	0.002	3	78
20	500	2967	1	52	262	10958	1.6	35.8	98.0	0.006	2	3250
			10	48	250	10572	6.3	33.7	94.9	0.002	2	198
			50	45	231	9743	15.1	30.0	89.2	0.002	3	406
			100	42	214	9118	22.4	27.4	84.0	0.002	2	158
			500	30	137	5745	50.2	15.5	52.8	0.004	2	126
30	500	2985	1	52	263	11283	1.7	30.9	98.3	0.014	2	2435
			10	51	254	10804	5.8	28.3	94.4	0.019	2	2210
			50	46	232	10070	16.2	27.8	86.7	0.001	3	2376
			100	43	220	9367	23.9	25.2	79.7	0.001	2	464
			500	27	129	5651	58.1	13.8	45.6	0.000	4	631
40	500	2962	1	60	289	11477	1.7	32.4	98.5	0.028	1	3590
			10	57	276	11043	6.7	30.7	95.5	0.033	2	2708
			50	51	254	10187	19.0	27.1	87.0	0.002	2	2924
			100	48	228	9253	27.6	23.6	77.2	0.007	2	2253
			500	26	105	4517	61.1	9.3	35.2	0.009	3	2926
50	500	3050	1	41	197	7918*	2.1	22.8	97.3	0.204	1	3600*
			10	42	193	7568*	9.1	21.5	93.9	0.031	2	3600*
			50	35	167	6792	23.6	18.7	84.8	0.060	1	2691
			100					Memory				
			500					Memory				
60	500	3000	1	50	232	9281*	2.7	23.3	97.7	0.361	1	3600*
			10					Memory				
			50	43	195	7730*	22.5	18.8	81.3	0.148	1	3600*
			100	38	175	6936*	34.2	16.7	69.5	0.203	1	3600*
			500	17	61	2205*	74.2	5.3	22.1	1.282	1	3600*
70	500	3026	1	49	223	8862*	2.9	33.2	98.5	0.228	1	3600*
			10	47	209	8331	11.1	31.1	92.7	0.050	1	2858
			50	40	181	7169	30.6	25.9	78.7	0.090	1	3591
			100	33	147	5886*	43.6	20.0	62.2	0.153	1	3600*
			500	10	24	959	72.2	2.4	13.4	1.397	1	2755
80	500	3054	1	52	234	8727*	1.6	22.6	97.1	0.324	1	3600*
			10	50	223	8444*	9.2	21.3	95.0	0.100	1	3600*
			50	44	194	7542*	24.5	17.9	87.6	0.175	1	3600*
			100	39	171	6634*	36.3	14.8	79.4	0.246	1	3600*
			500	12	40	1594*	74.7	3.5	19.6	1.714	1	3600*
90	500	3057	1	50	257	10211*	2.1	21.4	98.8	0.375	1	3600*
			10	49	242	9641*	7.3	20.2	91.0	0.822	1	3600*
			50	46	216	8708*	22.5	16.8	81.0	0.195	1	3600*
			100	42	189	7777*	34.6	14.3	71.7	0.290	1	3600*
			500	13	60	2600*	76.5	4.2	34.2	1.605	1	3600*
100	500	3023	1	52	263	10864*	2.2	26.0	98.7	0.291	1	3600*
			10	49	254	10695*	15.1	24.7	98.2	4.328	1	3600*
			50	46	227	9400*	27.2	20.8	84.1	0.147	1	3600*
			100					Memory				
			500					Memory				

* The instance was terminated after 3600 sec of CPU time; "Memory": This refers to instances that run out of memory.
$|N|$: Number of nodes; B: Number of submitted requests; M: Number of Calls; ; M: Unit Delay Costs; $|B_s|$: Number of accepted requests; $|M_s|$: Number of accepted calls;

Table 2: Computational Results for Instances with 1000 and 1500 Requests

$\|N\|$	$\|B\|$	$\|C\|$	K	$\|B_s\|$	$\|C_s\|$	Gross Revenue (GR)	Delay Cost (DC) (% of GR)	Avg. Util.(%)	Max. Util.(%)	Gap (%)	Iter.	CPU(s)
10	1000	6132	1	133	645	26598	1	73	99	0.006	2	247
			10	130	627	25856	4	70	96	0.003	3	271
			50	122	584	24448	10	65	92	0.004	3	249
			100	116	557	23446	14	61	87	0.004	3	268
			500	95	438	18742	33	45	70	0.006	3	179
20	1000	5993	1	84	366	15222	2	48	98	0.031	1	3512
			10	82	352	14669	6	45	95	0.008	2	1728
			50	76	320	13474	15	40	88	0.002	3	2069
			100	72	303	12702	22	37	84	0.001	3	804
			500	49	188	8006	50	20	57	0.009	3	729
30	1000	5980	1	119	610	24774*	1	47	99	0.062	1	3600*
			10	117	592	24205*	5	45	97	0.086	1	3600*
			50	110	553	22944	12	41	93	0.003	2	3122
			100	101	513	21820*	19	38	89	0.082	1	3600*
			500	76	357	15523	48	24	65	0.045	2	2355
40	1000	6056	1					Memory				
			10					Memory				
			50					Memory				
			100	75	370	15270*	26	30	81	0.128	1	3600*
			500					Memory				
50	1000	6036	1	96	467	19563*	2	40	99	0.656	1	3600*
			10	90	441	18870*	7	38	94	0.122	1	3600*
			50	80	401	17302*	17	33	87	0.171	1	3600*
			100	76	367	16039*	26	29	81	0.226	1	3600*
			500	45	194	8589*	59	14	49	0.515	1	3600*
60	1000	6058	1					Memory				
			10					Memory				
			50					Memory				
			100	41	169	7229*	32	18	76	0.284	1	3600*
			500					Memory				
20	1500	9099	1	77	304	11419*	1.3	33	99	0.035	2	3600*
			10	77	298	11152	4.6	31	96	0.037	1	3487
			50	71	276	10579*	11.9	28	89	0.057	1	3600*
			100	65	258	10047	17.4	26	84	0.001	2	2407
			500	50	182	7462	41.5	16	61	0.081	2	2578
40	1500	8944	1	60	255	9728*	1.2	26	98	0.401	1	3600*
			10	58	243	9230*	4.1	24	90	1.166	1	3600*
			50	54	227	8694*	14.0	23	84	0.416	1	3600*
			100	50	213	8264*	23.2	21	79	0.161	1	3600*
			500	31	120	4997*	55.7	10	48	0.467	1	3600*

* The instance was terminated after 3600 sec of CPU time; "Memory": This refers to instances that run out of memory. $|N|$: Number of nodes; B: Number of submitted requests; C: Number of Calls; ; C: Unit Delay Costs; $|B_s|$: Number of accepted requests; $|C_s|$: Number of accepted calls.

5 Conclusion

Telecommunication bandwidth is a scarce resource making its judicious use critical in presence of competing requests by bandwidth intensive video and data services. So, ideally, a network administrator would like to accept as many requests, each possibly consisting of several one-to-many or many-to-many calls, as allowed by the limited bandwidth to maximize the revenue earned. This is referred to as a combinatorial bandwidth packing problem. However, this revenue focus approach is likely to degrade the response time due to excessive bandwidth usage. So, in this paper, we extended this problem to finding the optimal trade-off between revenue maximization and response time delays. To this end, we presented a non-linear IP formulation for the problem. By using simple transformation and piecewise outer-approximation, we linearized the model, and presented an efficient cutting plane based approach to solve the resulting linear MIP without adding a large number of linearization constraints. Our computational experiments over a wide range of problem instances obtained by varying the size of the networks, number of requests and calls, and call bandwidth requirements indicate that the proposed solution method can produce near optimal solutions in reasonable computational time.

Acknowledgements

This research was supported by the Research & Publication Grant, Indian Institute of Management Ahmedabad to the first author, and National Science and Engineering Research Council of Canada (NSERC) grant to the second author. The authors would also like to acknowledge Vikranth B.T. Chetty for his assistance in computational experiments.

References

Amiri, A., 2003. The multi-hour bandwidth packing problem with response time guarantees. Information Technology and Management 4, 113–127.

Amiri, A., 2005. The selection and scheduling of telecommunication calls with time windows. European Journal of Operational Research 167 (1), 243–256.

Amiri, A., Barkhi, R., 2000. The multi-hour bandwidth packing problem. Computer and Operations Research 27, 1–14.

Amiri, A., Barkhi, R., 2012. The combinatorial bandwidth packing problem. European Journal of Operations Research 208, 37–45.

Amiri, A., Rolland, E., Barkhi, R., 1999. Bandwidth packing with queuing delay costs: Bounding and heuristic procedures. European Journal of Operational Research 112, 635–645.

Anderson, C., Fraughnaugh, K., Parkner, M., Ryan, J., 1993. Path assignment for call routing: An application of tabu search. Annnals of Operations Research 41, 301–312.

Barnhart, C., Christopher, A., Hane, Vance, P. H., 2000. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. Operations Research 48 (2), 318–326.

Bose, I., 2009. Bandwidth packing with priority classes. European Journal of Operational Research 192, 313–325.

Cox, L., Davis, L., Qui, Y., 1991. Dynamic anticipatory routing in circuit-switched telecommunications networks. In: Davis, L. (Ed.), Handbook of Genetic Algorithms. Vol. 11. Van Norstrand/Reinhold, New York, pp. 229–340.

Dantzer, J., Haddani, M., Robert, B., 2000. On the stability of a bandwidth packing algorithm. Probability in the Engineering and Information Sciences 14, 57–79.

Elhedhli, S., 2005. Exact solution of a class of nonlinear knapsack problems. Operations Research Letters 33 (6), 615–624.

Gavish, B., Hantler, S., 1983. An algorithm for optimal route selection in sna networks. IEEE Transactions on Communications 31 (10), 1154–1161.

Han, J., Lee, K., Lee, C., Park, S., 2012. Exact algorithms for a bandwidth packing problem with queueing delay guarantees. INFORMS Journal on Computing.

Laguna, M., Glover, F., 1993. Bandwidth packing: A tabu search approach. Management Science 39, 492–500.

Park, K., Kang, S., Park, S., 1996. An integer programming approach to the bandwidth packing problem. Management Science 42, 1277–1291.

Parker, M., Ryan, J., 1993. A column generation algorithm for bandwidth packing. Telecommunication Systems 2 (1), 185–195.

Rolland, E., Amiri, A., Barkhi, R., 1999. Queueing delay guarantees in bandwidth packing. Computers and Operations Research 26, 921–935.

Vidyarthi, N., Jayaswal, S., Chetty, V., ???? Exact solution to bandwidth packing problem with queuing delays. Indian Institute of Management Ahmedabad Working Paper No. 2013-11-04.

Vidyarthi, N., Jayaswal, S., Chetty, V. B., 2014. Bandwidth packing problem with queueing delays: Modelling and exact solution approach. Working Paper, John Molson School of Business, Concordia University, Montreal.

Villa, C., Hoffman, K., 2006. A column-generation and branch-and-cut approach to the bandwidth-packing problem. Journal of Research of the National Institute of Standards and Technology 111, 161–185.