

2006

Solution of a certain class of network flow problems with cascaded demand aggregation and capacity allocation

F. Safaei

University of Wollongong, farzad@uow.edu.au

I. Ouveysi

University of Melbourne

Publication Details

This is an electronic version of an article originally published as: Safaei, F & Ouveysi, I, Solution of a certain class of network flow problems with cascaded demand aggregation and capacity allocation, *Optimization Methods and Software*, 2006, 21(4), 583–595. The journal *Optimization Methods and Software* can be found [here](#) through Taylor & Francis journals.

Solution of a certain class of network flow problems with cascaded demand aggregation and capacity allocation

Abstract

This article develops analytical models for a class of networking problems that includes two cascaded stages of demand aggregation and capacity allocation. The solutions to these problems are required in real time as the demand fluctuates rapidly. The capacity allocation problem makes a large-scale integer programming problem too complex for practical applications. Using the Lagrangian relaxation technique and a suitably developed heuristic for multiplier adjustment, the computational complexity is reduced to such a degree that a real-time implementation of the algorithm is feasible. This article also develops efficient heuristics to aggregate demand. The proposed algorithm produces a near-optimal solution in pseudo-polynomial time.

Keywords

Optimization, Mathematical programming, Lagrangian relaxation technique, Telecommunications

Disciplines

Physical Sciences and Mathematics

Publication Details

This is an electronic version of an article originally published as: Safaei, F & Ouveysi, I, Solution of a certain class of network flow problems with cascaded demand aggregation and capacity allocation, *Optimization Methods and Software*, 2006, 21(4), 583–595. The journal *Optimization Methods and Software* can be found [here](#) through Taylor & Francis journals.

Solution of a certain class of Network Flow Problems with cascaded demand aggregation and capacity allocation

FARZAD SAFAEI* AND IRADJ OUVEYSI**

*University of Wollongong, Australia, farzad@uow.edu.au

**The University of Melbourne, Australia, iradjouveysi@yahoo.co.uk

Abstract

This paper develops analytical models for a class of networking problems that include two cascaded stages of demand aggregation and capacity allocation. The solutions to these problems are required in real-time as the demand fluctuates rapidly. The capacity allocation problem leads to a large-scale integer programming problem too complex for practical applications. Using the Lagrangian relaxation technique and a suitably developed heuristic for multiplier adjustment, the computational complexity is reduced to such a degree that a real-time implementation of the algorithm is feasible. The paper also develops efficient heuristics to aggregate demand. The proposed algorithm produces a near optimal solution in pseudo-polynomial time.

Keywords: Optimization; Mathematical Programming; Lagrangian Relaxation Technique; Telecommunications

1 Introduction

Optimization of a class of network flow problems has become increasingly important in certain industrial applications. These problems involve two cascaded (but interrelated) stages of demand aggregation and capacity optimization. Typically, the demand is highly dynamic, that is, it fluctuates significantly with time. Consequently, the optimization is required to be performed on short time scales, often in real-time. In addition, the demand aggregation stages are spatially distributed. This geographical separation would necessitate distribution, and to a certain degree independence, of decision making processes for the overall system optimization.

Figure 1 shows a representation of this class of problems. In this Figure, there is a supply node on the right hand side that is the source of some goods for consumption of sink nodes on the left hand side. The sink nodes are grouped in several geographical locations. At a given short time interval there is a certain level of demand from every sink node for the goods supplied. Note that we require the demand from a particular sink node to either be satisfied in full or completely rejected. In other words, we cannot accept partial fulfilment of demand.

Every sink node can access the supply source from a set of demand concentrators (labelled Access Concentrator nodes in the Figure). In a given location, every sink node is connected to every Access Concentrator node, but to avoid cluttering, we have only shown a subset of these links. There is no capacity limit on these links.

The supply node, however, has a capacity limit on the amount of goods that it can ship to the Access Concentrator nodes. This limit is denoted by Nk where N is an integer and k is a fixed value that we call “channel capacity”. The supply node has potential links to all the Access Concentrator nodes. However, at

** Iradj Oveysi is an honorary research fellow with the Department of Electrical and Electronic Engineering of the University of Melbourne.

each instant of time at most N of these links will be allocated capacity of k . There is also a limit of how many channels can be allocated to a given location.

In a typical situation, the demand from several sink nodes in a given location is aggregated into a single Access Concentrator node so that the total demand is less than k . The supply node will then allocate a channel of capacity k to this Access Concentrator node and use this to satisfy the demand. For example, Figure 1 shows that in location 1, three sink nodes are connected to the first Access Concentrator node and a channel has been allocated to this node from the supply for this purpose.

The problem of interest to us, therefore, can be summarised as follows: Given a network of supply node and sink nodes as described before, we intent to maximise the amount of demand satisfied by this network at each time interval by aggregating a subset of demands from multiple sink nodes into Access Concentrator nodes and then allocating channel capacity to these. There is a cost associated with re-allocation of channel capacity from one link to another. Channel capacity can only be allocated in whole and partial fulfilment of demand for a particular sink node is not allowed.

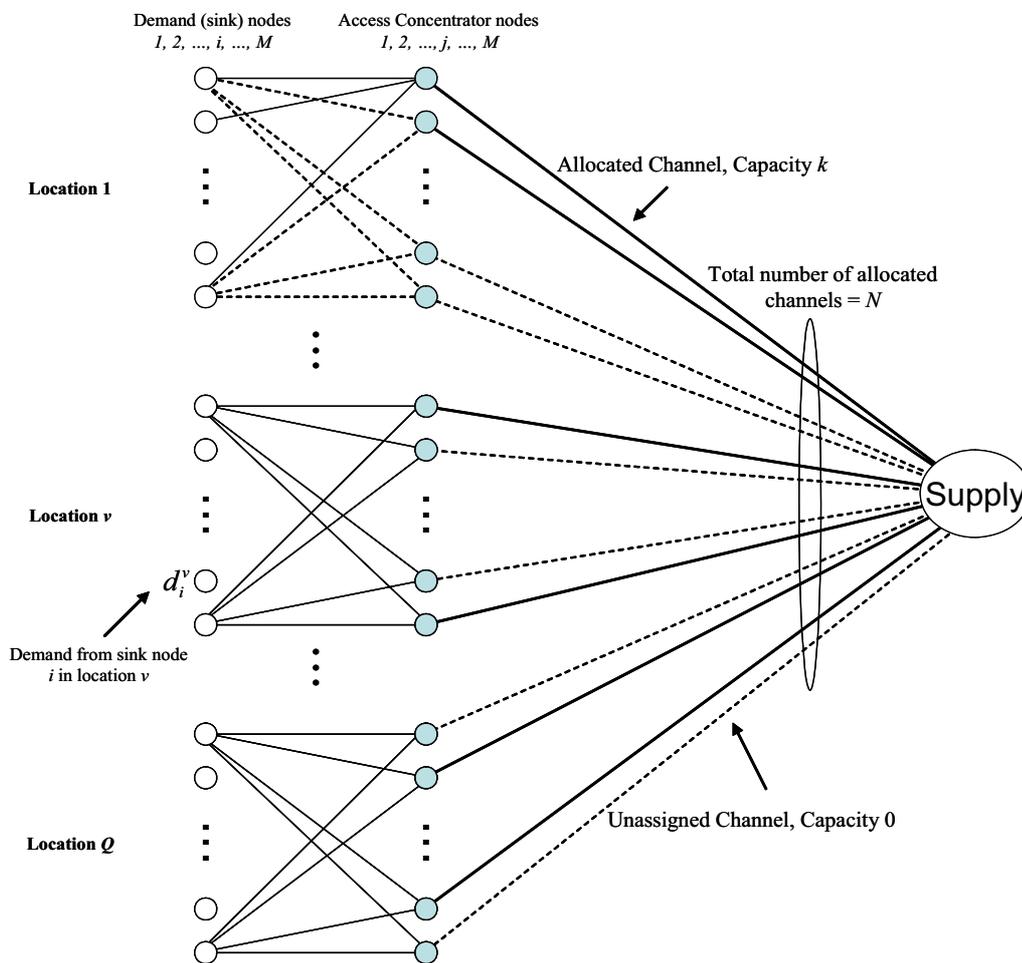


Figure 1: The structure of the network flow problem

1.1 Example application

The above problem is frequently encountered in planning and operation of telecommunication networks to supply broadband Internet access to residential customers. In this case, an “Access Concentrator” is situated in vicinity of a group of customers, say a given street. This device is usually called a “multiplexer” in telecommunication jargon. Several of these Access Concentrators in a given geographical region are then connected to the first point of entry to the Internet which is typically a router installed in the Internet Service Providers premises. This router performs the role of the supply node in our model and will deliver the information requested by the customers to them. The router, however, has limited capacity and it would be very expensive to connect it to all nodes statically. The typical interconnection, therefore, is based on an “optical fibre ring” network where a number of channels of given capacity can be assigned to the Access Concentrators based on demand. These channels are implemented using different optical wavelengths on the optical fibre ring. Given that more than 50% of a telecommunications carrier’s total investment is in the access network [1], the cost savings of the above optimization could be substantial.

1.2 The aim of this paper

This paper develops mathematical models and algorithms for optimal supply of demand for the above problem. There are two phases of channel capacity allocation and demand concentration.

- The supply node must dynamically allocate channel capacity to Access Concentrator nodes as the demand from these fluctuates in time.
- The Access Concentration phase requires algorithms to pack as much demand as possible onto the available number of allocated channels.

In practice, these two stages are geographically distributed and it would not be useful to model the problem as a single monolithic problem. Instead, one needs to develop optimization algorithms for both phases in such a way that they can operate more or less independently on short time intervals.

2 Problem Formulation – Access Concentrator

Figure 2 depicts the Access Concentrator model as used in this paper. There are M inputs to the concentrator indexed by $i = 1, 2, \dots, M$. Each input i has a demand requirement of d_i . It is assumed that these values are sensible. That is, $d_i \leq k \forall i$, where k is the channel capacity described before. It is also assumed that d_i values are integers (which can be enforced by selecting a suitable demand unit).

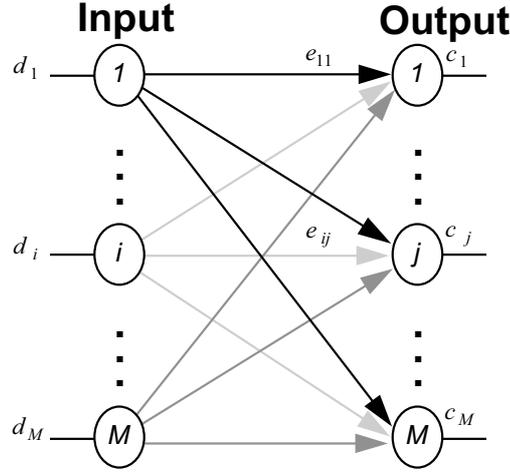


Figure 2: Access Concentrator Model

For conceptual simplicity, the number of output ports is also equal to M . The output ports are indexed by $j = 1, 2, \dots, M$. The main task of the concentrator is to aggregate as many inputs onto the output ports without exceeding the channel capacity k . Naturally, some of the outputs may be assigned with no demand. These outputs are presented to the supply node for allocation of channels which attempts to maximise the total supply.

Although not essential, it would be easier to understand the interaction between the two optimization entities if the concentrator output is ordered. To this end, the weighting factor c_j can be properly assigned to enforce the desired ordering. For example, by setting $c_j = j$, one can force the Access Concentrator optimization routine to provide an ordered set of outputs from maximum to minimum. Another possibility is to set the values of c_j according to the following:

$$c_j = \begin{cases} 1 & \text{for } j = 1, 2, \dots, m \\ F & \text{otherwise} \end{cases} \quad (1)$$

Where F is a large number. This forces the concentrator to aggregate as much as possible onto the first m output ports. (Commonly, there is a limit for maximum number of channels which can be assigned to a given location due to hardware constraints. This maximum value is denoted here by m .)

Therefore, the Integer Linear Programming model for the Access concentrator can be formulated as follows:

$$\text{Min } z = \sum_{i=1}^M \sum_{j=1}^M e_{ij} d_i c_j \quad \mathbf{P1}$$

$$\text{Subject to: } \sum_{j=1}^M e_{ij} = 1 \quad \forall i = 1, 2, \dots, M \quad (2)$$

$$\sum_{i=1}^M e_{ij} d_i \leq k \quad \forall j = 1, 2, \dots, M \quad (3)$$

$$e_{ij} \in \{0, 1\} \quad \forall i, j \quad (4)$$

In this formulation, e_{ij} is the decision variable representing the input-output connection matrix for the concentrator. The constraint set (2) ensures that each input is connected to precisely one output port and the constraint set (3) enforces the channel capacity limit.

2.1 Solution strategy

The Integer Linear Programming P1 problem belongs to the class of NP-Complete problems. It is similar to the star-star concentrator location problem with the difference that any terminal i here may have a different demand d_i .

Different heuristics have been developed to solve this type of problems, among which 'ADD' and 'DROP' algorithms are often used in the literature [2]. However, in ADD and DROP algorithms a sequence of optimal assignment problems are solved and hence these approaches quickly become impractical as the problem size grows.

2.2 Greedy Algorithm

Here, a Greedy Algorithm for solving P1 is provided. This algorithm attempts to solve the problem iteratively. In each step the remaining output port with the least weighting value c_j will be maximally filled using the remaining input demands.

Denoting the set of remaining input and output ports by S and Q respectively, and the output port index with minimum weighting value in Q by l , at each iteration of the algorithm the following ILP sub-problem will be solved.

$$\text{Max } z' = \sum_{i \in S} e_{il} d_i \quad \mathbf{P2}$$

$$\text{Subject to: } \sum_{i \in S} e_{il} d_i \leq k \quad (5)$$

$$e_{il} \in \{0,1\} \quad \forall i \in S \quad (6)$$

Problem P2 may be recognized as a special case of a 0-1 *knapsack* problem (cost and constraint vectors coincide).

Formally, the proposed Greedy Algorithm can be outlined as follows.

2.2.1 Main Algorithm

Step 1: Given the number of ports M , input demands $d_i \leq k$ for $i=1,2,\dots,M$, where $k \in \mathbb{Z}^+$ is the output port capacity, initialize $S = \{1,2,\dots,M\}$ and $Q = \{1,2,\dots,M\}$ as the set of all unassigned input and output ports respectively.

Step 2: Consider an output port $l: c_l \leq c_j \quad \forall j \in Q$. Solve the sub-problem P2 according to procedure outlined below. Let the solution be $S_l \subseteq S$.

Step 3: Update $Q \leftarrow Q \setminus \{l\}$ and $S \leftarrow S \setminus S_l$.

Step 4: If $Q = \emptyset$ or $S = \emptyset$ stop. Otherwise go to Step 2.

2.2.2 Procedure for solving P2

The P2 sub-problem can be solved by the dynamic programming algorithm described in [5], which is capable of reaching the optimal solution in pseudo-polynomial time. Without loss of generality, assume that $d_1 \leq d_2 \leq \dots \leq d_M$. The above-mentioned algorithm behaves as follows.

For $t=1,2,\dots,M$, define $N_t = \{1,2,\dots,t\}$, and

$$z_t(p) = \text{Max} \left\{ \sum_{i \in N_t} e_{il} d_i : \sum_{i \in N_t} e_{il} d_i \leq p, e_{il} \in \{0,1\} \forall i \right\} \text{ for } p = 0,1,\dots,k$$

The recursion is initialized with

$$z_1(p) = \begin{cases} d_1 & \text{if } d_1 \leq p \\ 0 & \text{otherwise} \end{cases}$$

As shown in [5], for $t=2,3,\dots,M$ and $p=0,1,\dots,k$

$$z_t(p) = \begin{cases} z_{t-1}(p) & \text{if } d_t > p \\ \text{Max} \{ z_{t-1}(p), d_t + z_{t-1}(p - d_t) \} & \text{if } d_t \leq p \end{cases}$$

Upon termination $z_M(k) = z'$.

In the case of multiple solutions to P2 (i.e., more than one possibility for forming S_t), based on Lemma 1 below if these sets are disjoint, a random selection will suffice. In the case that these sets are not disjoint, the solution set with minimum cardinality will be selected. It will be shown in the next sub-section that this is more likely to lead to the optimal solution as it provides further opportunity of multiplexing in the subsequent iterations.

2.3 Effectiveness of Greedy Algorithm

The complexity of Greedy Algorithm can be estimated as follows. As indicated in [5], the dynamic programming algorithm will solve P2 in $O(MK)$ operations for each iteration of main algorithm. As there are M iterations, the total complexity is $O(M^2K)$.

In general, the output of the Greedy Algorithm as defined in the previous sub-section may not be optimal. As an example, consider the case of Figure 3 where an Access Concentrator with $M=6$ and $k=20$ is depicted.

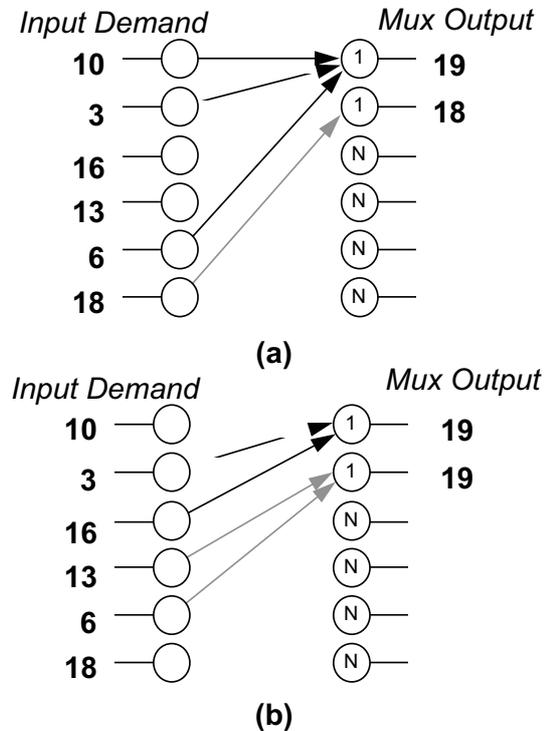


Figure 3: Non optimal nature of the Greedy Algorithm

The values of c_j have been selected according to Equation 1 with $m=2$, that is

$$c_j = \begin{cases} 1 & j = 1,2 \\ F & \text{otherwise} \end{cases} \quad (7)$$

Where F is a large number. This forces the concentrator to aggregate as much as possible onto the first two output ports.

A comparison between Figure 3-(a) and Figure 3-(b) will reveal that the Greedy Algorithm did not pick the optimal solution in Figure 3-(a) (the output values for $j > 2$ are not shown as these are of no concern).

It is instructive to note that in the example of Figure 3, there is more than one solution to the P2 problem when applied to the output node 1 (i.e., there is more than one way to obtain the value of 19 from the input demands). The following lemma, together with the above observations, will be useful in designing an appropriate heuristic to deal with this case (this heuristic was explained in the previous sub-section). For simplicity, the lemma is proved for $m=2$. Extension to general case is also possible.

Lemma 1: Let S_j denote the set of inputs assigned to output j by the concentrator. That is,

$$S_j = \{i : e_{ij} = 1\}.$$

Let c_j be defined as in Equation 7 and α_j denote the output of P2 in the j^{th} iteration of the main algorithm. Also assume that there are $\bar{S}_j, j=1,2$ with solution values $\bar{\alpha}_j, j=1,2$ such that $\bar{\alpha}_1 = \alpha_1$. Then the claim is that if $S_1 \cap \bar{S}_1 = \emptyset$, the Greedy Algorithm will result in an optimal solution.

Proof: Assume to the contrary that $S_1 \cap \bar{S}_1 = \emptyset$, but the output of Greedy Algorithm is not optimal, that is, $\bar{\alpha}_1 + \bar{\alpha}_2 > \alpha_1 + \alpha_2$.

Now, since $\bar{\alpha}_1 = \alpha_1$, it follows that $\bar{\alpha}_2 > \alpha_2$. However, $\bar{S}_1 \subseteq S \setminus S_1$ and $S_2 \subseteq S \setminus S_1$ which imply that $\alpha_2 = \alpha_1$ (due to optimality of P'2). This, in turn, implies that $\bar{\alpha}_2 > \alpha_1$, which is a contradiction and the proof is complete. \square

3 Problem Formulation – Supply Node Channel Allocation

The supply node channel allocation algorithm will examine the offered demand on regular time intervals. At each time interval, some existing channels are allowed to continue and some new channels are optimally rearranged. This assignment procedure is performed based on the output of the following mathematical model for each time interval.

Q : Number of locations.

v : Index for the location number, $v = 1, 2, \dots, Q$

M : Number of inputs in any location.

i : Index for node inputs, $i = 1, 2, \dots, M$.

N : Total number of channels available to the supply node.

j : Index for channel numbers, $j = 1, 2, \dots, N$.

d_i^v : Demand at input number i location v at the given time interval. d_i^v is assumed to be less than or equal to the channel capacity.

$e_{ij}^v = 1$ if input i in location v is connected to channel j at the current time interval, and is equal to zero otherwise.

c_j^v : Cost of establishing a connection between channel j and location v .

$w_j^v \in \{0,1\}$: Connection status of channel j to location v before the current time interval (an input to the optimization based on the status of the system, not a decision variable).

n_v : Maximum number of channels for location v .

r_j^v : The revenue from carrying a unit demand on channel j from location v .

h_j^v : The connection cost of channel j to location v defined as

$$h_j^v = \begin{cases} 0 & \text{if } w_j^v = 1 \\ c_v & \text{if } w_j^v = 0 \end{cases}$$

This indicates that the cost of connecting a channel to a given location depends on the previous state of the system before the current time interval (no cost arises if the channel was already connected to the specified location). A dummy node $N + 1$ is considered and the following defined.

r_{N+1}^v : Lost revenue from rejection of a unit demand from location v .

$e_{i,N+1}^v$: Status of rejection of demand i on location v .

The intended problem can then be modelled as the following integer linear programming problem:

$$\text{Min } \mathbf{Z}(\mathbf{PI}) = \sum_{v=1}^Q \sum_{i=1}^M d_i^v e_{i,N+1}^v r_{N+1}^v + \sum_{v=1}^Q \sum_{i=1}^M \sum_{j=1}^N h_j^v e_{ij}^v - \sum_{v=1}^Q \sum_{i=1}^M \sum_{j=1}^N d_i^v e_{ij}^v r_j^v \quad (\mathbf{P3})$$

Subject to:

$$\sum_{j=1}^{N+1} e_{ij}^v = 1, \quad i = 1, 2, \dots, M; v = 1, 2, \dots, Q \quad (8)$$

$$\sum_{v=1}^Q \sum_{i=1}^M e_{ij}^v = 1, \quad j = 1, 2, \dots, N \quad (9)$$

$$\sum_{j=1}^N \sum_{i=1}^M e_{ij}^v \leq n_v, \quad \forall v \quad (10)$$

$$e_{ij}^v \in \{0, 1\} \quad \forall i, j, v \quad (11)$$

Constraint set (8) indicates that a given input i of location v can either be connected to at most one output channel or to the dummy node (demand rejection). Constraint set (9) ensures that channel j is connected to only one input and finally constraint set (10) guarantees that not more than n_v channels are assigned to location v .

3.1 Lagrangian relaxation

The integer linear programming (ILP) problem (P3) has $\alpha = NMQ$ binary variables and the total number of constraints is also in the order of $\alpha = NMQ$. (A typical problem will have its α in the order of 6000.) General relaxation algorithms, such as Fractional Cutting Plane or Branch and Bound algorithms may be used to solve ILP problems [5]. However for our binary ILP problem, considering the number of binary variables, these methods would appear to be impractical. A closer examination of the structure of (P3) shows that removing constraint set (10) will yield a Linear Min-Sum Assignment problem.

Therefore, to solve problem (P3), the constraint set (10) is relaxed using the Lagrangian relaxation technique [5] and incorporated into the objective function as a penalty. Let β_v , $v = 1, 2, \dots, Q$, be the set of non-negative Lagrangian multipliers corresponding to the constraints in (10). Then the relaxation of problem (P3), denoted here by $LR(\beta)$, will be as follows:

$$\text{Min } Z(LR(\beta)) = \sum_{v=1}^Q \left\{ \sum_{i=1}^M \sum_{j=1}^{N+1} e_{ij}^v c_{ij}^v - \beta_v n_v \right\} \quad LR(\beta)$$

$$\text{Subject to: } \sum_{j=1}^{N+1} e_{ij}^v = 1, \quad i = 1, 2, \dots, M; v = 1, 2, \dots, Q \quad (12)$$

$$\sum_{v=1}^Q \sum_{i=1}^M e_{ij}^v = 1, \quad j = 1, 2, \dots, N \quad (13)$$

$$e_{ij}^v \in \{0, 1\} \quad \forall i, j, v \quad (14)$$

$$\text{Where: } c_{ij}^v = \begin{cases} \beta_v + h_j^v - d_i^v r_j^v & \text{for } j = 1, 2, \dots, N \\ d_i^v r_{N+1}^v & \text{for } j = N + 1 \end{cases} \quad (15)$$

It is possible to convert this problem into an assignment problem provided the cost of rejecting a demand can be considered sufficiently high. The transformation of Lagrangian relaxation problem, denoted by $LR^T(\beta)$, can then be written as

$$\text{Min } Z(LR^T(\beta)) = \sum_l^{QM} \sum_{l'}^{QM} f_{ll'} e_{ll'} \quad LR^T(\beta)$$

$$\text{Subject to: } \sum_{l=1}^{QM} e_{ll'} = 1, \quad l' = 1, 2, \dots, QM \quad (16)$$

$$\sum_{l'=1}^{QM} e_{ll'} = 1, \quad l = 1, 2, \dots, QM \quad (17)$$

$$e_{ll'} \in \{0, 1\} \quad l, l' = 1, 2, \dots, QM \quad (18)$$

$$\text{Where: } f_{ll'} = \begin{cases} c_{ij}^v & \text{if } l' = j \leq N \text{ for } l = M \times (v-1) + i \\ \varphi & \text{if } l' > N \text{ for } l = 1, 2, \dots, QM \end{cases} \quad (19)$$

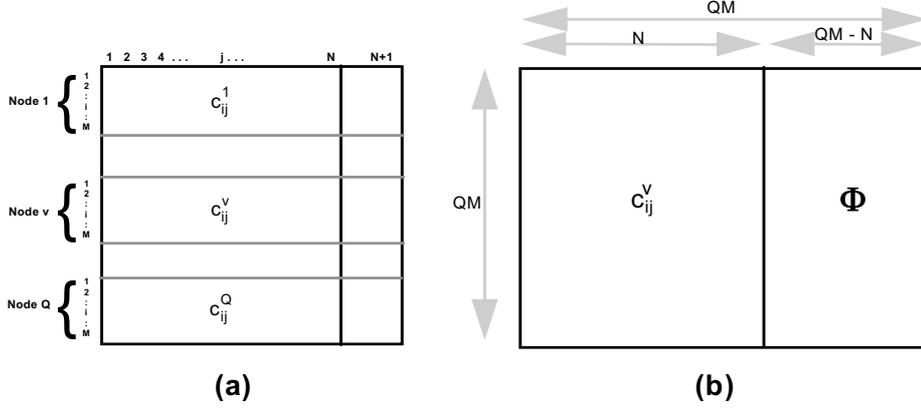


Figure 4: (a) Structure of the Lagrangian relaxation problem. (b) Transformation of the Lagrangian relaxation problem to an assignment problem

Here, φ is a very large number in comparison to the values of c_{ij}^v . Figure 4 provides a pictorial representation of transforming the Lagrangian relaxation problem into an assignment problem as described above. In this Figure, Φ represents a matrix in $R^{QM \times (QM - N)}$ with all the entries equal to φ .

The procedure used to solve the assignment problem is based on transformation of the assignment problem to a Min-Cost Flow problem [3]. This in turn can be solved, through the well known algorithm by Ford and Fulkerson [4], which performs α shortest path computations on the network. The complexity of the overall algorithm is in the order of α^4 .

3.1.1 Multiplier adjustment and termination

A proper heuristic procedure for updating Lagrangian multipliers can help to reduce the computational time. Let us define g_v as the sub-gradient for the relaxed constraints evaluated at the current solution by

$$g_v = \sum_{j=1}^N \sum_{i=1}^M e_{ij}^v - n_v \quad (20)$$

The following rules can be used to change multiplier values:

- If $g_v < 0$, the multiplier β_v will be reduced.
- If $g_v = 0$, the multiplier β_v will not be changed.
- If $g_v > 0$, the multiplier β_v will be increased.

Based on experience it was found that incremental changes in β_v of the order of 10% at each step would provide good convergence. Stopping criterion is based on the complementary slackness condition, that is,

$$\beta_v g_v = 0 \quad \forall v \quad (21)$$

In our implementation, the algorithm was terminated when the absolute value of this product was less than a predefined small positive number ε and $g_v \leq 0$ (β_v is non-negative for all v). Note that $g_v > 0$ represents an infeasible solution irrespective of the size of $\beta_v g_v$ and, therefore, cannot be an acceptable condition for

termination. Another criterion for termination was that the total number of iterations be less than a prescribed upper value (500 in this case), although, practically, this condition was never encountered in our simulations. The procedure for multiplier adjustment is presented below:

3.1.2 Procedure for multiplier adjustment:

Step 1: Given δ (incremental factor for (β) adjustment) and S (set of location indexes for which the complementary slackness or feasibility conditions are not satisfied), evaluate $g_v, \forall v \in S$ from Equation (20).

Step 2: If $g_v < 0$ then decrease β_v by δ percent. That is, $\beta_v \leftarrow \beta_v(1 - \delta)$. If $g_v > 0$ then update β_v according to the following:

$$\beta_v \leftarrow \begin{cases} \beta_v(1 + \delta) & \text{if } \beta_v > 0 \\ \text{Min}_{i,j} |c_{ij}^v| & \text{if } \beta_v = 0 \end{cases}$$

The algorithm for solving the original problem is presented below:

Step 1: Given ε (small positive number), and U (upper bound on the number of iterations), set $I = 0, (\beta) = 0, S = \emptyset$.

Step 2: Evaluate c_{ij}^v from Equation (15); set $\varphi = 100 \times \text{Max}_{i,j,v} |c_{ij}^v|$.

Step 3: Evaluate (f) from Equation (19) and solve the assignment problem $LR^T(\beta)$. Update $I \leftarrow I + 1$.

Step 4: If $I > U$ (maximum number of iterations), go to step 7.

Step 5: If $|\beta_v g_v| < \varepsilon$ and $g_v \leq 0 \forall v$ (the complementary slackness condition is almost satisfied, hence, the current solution is near optimal), go to step 7.

Step 6: Set $S = \{v : |\beta_v g_v| > \varepsilon \text{ or } g_v > 0\}$, update (β) according to *procedure (a)* above. Go to step 2.

Step 7: Stop.

4 Numerical Results

This algorithm was used to solve a series of about 16000 randomly generated problems (with $\alpha \approx 6000$). The average running time for each problem was found to be less than one CPU second. The problems were selected to exemplify possible growth scenarios in terms of the market uptake of broadband Internet services. The aim was to demonstrate the impact of using above optimization methods on the ability of the Internet Service Provider to cope with growth in both customer base and demand generated by existing customers.

To obtain the required capacity to be deployed by the telecommunication carrier, we have developed a network planning tool that is comprised of three basic modules: (i) demand generation module, (ii) optimization module, and (iii) network dimensioning module. Each of these is now briefly explained.

The *demand generation module* was developed in an Excel spreadsheet. The aim was to be able to provide the network traffic requirements of customers according to a variety of different growth scenarios. In each scenario, the market penetration of broadband Internet services (that is, percentage of households subscribed to the service) is modelled over an eight-year period based on the characteristics of the scenario. In addition, the usage pattern of each subscriber (number of hours per week using the service, the amount of capacity required during the active periods depending on the type of services used) is also modelled over the same period. Consequently, growth in demand in each scenario is attributed to both increased usage of Internet by existing subscribers, and also by the take up of these by new customers as the prevalence and cost advantages of these services change the nature of work, entertainment and lifestyle. We have developed a range of scenarios from optimistic to pessimistic and tested the algorithm based on these. The capacity curves in Figure 5 are obtained based on a moderately optimistic scenario.

The *optimization module* was written in C and incorporated the algorithms described in this paper for both the Access Concentrator and the Optical Ring. The demand values produced by the demand generation module form the input to this module. The Access Concentrator will attempt to aggregate as much of this demand as possible onto its output channels (assumed to be of fixed capacity). The optical ring optimizer will attempt to carry as many of these channels as possible on the available capacity of the ring.

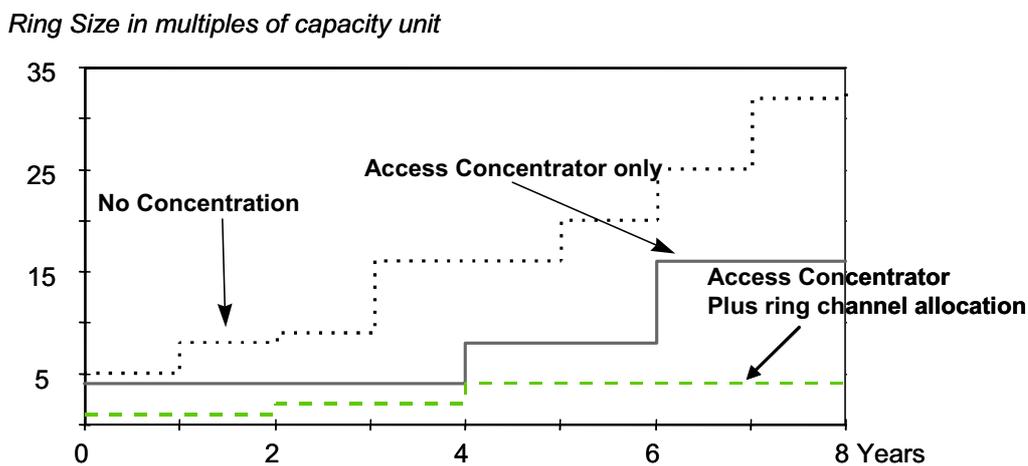


Figure 5: The required ring capacity in response to a typical growth scenario

The *network dimensioning module* was developed in an Excel spreadsheet and its aim is to find the most suitable and cost effective network capacity to carry the optimised demand produced by the previous module. Note that the ring capacity is limited to discrete values as the underlying optical technology can only support certain multiples of optical wavelengths on each fibre. Also, when growing a ring to higher capacity, there are two options available. The first option is to increase the capacity of the whole ring to the next level (for example upgrading a ring that can carry 16 channels at 2.5 Giga bits per second (Gbps) each to one that can carry 64). The second option is to stack another ring on top of the existing ring if optical fibre is available within the cable sheet. The dimensioning module, therefore, will find the most cost effective option to provide sufficient capacity for the required number of channels. As the demand grows, more capacity will be required and if needed the ring has to be upgraded or stacked with another ring. That is why the capacity upgrades in Figure 5 are in steps.

Figure 5 shows an example result where the capacity requirement is compared for three possible situations: 1- “No Concentration” where neither concentration algorithm is employed; 2- “Access Concentrator only” where the channel capacity on the ring is static; and 3- “Access Concentrator plus ring channel allocation” where both levels of optimizations are operating.

As evident in the Figure, by introducing the two levels of traffic concentration, the required capacity is significantly reduced in comparison with a “brute force” capacity assignment to customers.

5 Conclusions

The category of network flow problems identified in this paper has many industrial applications, especially in telecommunications industry where the cost savings of optimal architecture could be very substantial. In this paper, analytical models and efficient algorithms for the aggregation of demand using an Access Concentrator have been developed. The paper also presents an analytical model for allocation of channel capacity to Access Concentrator. The developed algorithms provide near optimal solutions in pseudo-polynomial time.

These algorithms could be useful for real-time operation and management of certain types of telecommunication networks, in particular, for connection of residential customers to Internet.

6 References

- [1] Balakrishnan, A., et al., "A decomposition algorithm for local access telecommunications network expansion planning", *Operations Research*, Vol 43 No 1, January-February 1995.
- [2] Tang, D. T., et al., "Optimization of Teleprocessing Networks with Concentrators and Multiconnected Terminals", *IEEE Transactions on Computers*, Vol C-27, No 7, July 1978.
- [3] Carpento, G., et al., "Algorithms and Codes for the Assignment Problem", *Annals of Operations Research* 13(1988) 193-223.
- [4] Ford Jr., L.R. and Fulkerson, D.R., "Flows in Networks", Princeton University Press, Princeton 1962.
- [5] Nemhauser, G.L., and Wolsey, L.A., "Integer and Combinatorial Optimization", Wiley, 1988.
- [6] Ouveysi, I., "A Lagrangian Relaxation Technique for Multi-connected Terminals and Concentrator Location Problem", *Proceedings of Australian Telecommunication Networks and Applications Conference*, 5-7 December 1994.

Biographies



Farzad Safaei graduated from the University of Western Australia with the degree of Bachelor of Engineering and obtained his PhD in Telecommunications Engineering from Monash University, Melbourne, Australia. He has more than 15 years of experience in conducting and managing advanced research in the field of data communications and networks. Currently, he is the Professor of Telecommunications Engineering and Director of Centre for Emerging Networks and Applications at the University of Wollongong. He is also the Program Manager of the Cooperative Research Centre for Smart Internet Technology. Before joining the University of Wollongong, he was the Manager of Internetworking Architecture and Services Section in Telstra Research Laboratories (TRL). His primary research interest is to design large-scale

telecommunication networks that can adapt autonomously to dynamic characteristics of applications, cost, customer demand, or any other critical influence from outside.



Iradj Ouveysi received the B.Sc. degree in engineering from Middle East Technical University, Ankara, Turkey, in 1987, the M.Sc. degree in operations research from Bilkent University, Ankara, in 1990, and the Ph.D. degree on telecommunication network design from the University of Melbourne, Melbourne, Australia, in 1996. Dr. Ouveysi has been working in the telecommunication industry for the last 12 years. He is also an honorary fellow of the Electrical and Electronic Engineering Department, the University of Melbourne. His areas of interest are network reliability and survivability, optimal location theory, polyhedral theory, combinatorial optimization, MPLS traffic engineering, PON access

networks, optimization and traffic management, dynamic alternative routing, and cost modeling in telecommunication networks.