A Method for Determining Optimal Integer Numbers of Circuits in a Telephone Network

L.T.M. Berry
University of Adelaide, Adelaide, Australia

ABSTRACT

For a given telephone network, there exist many different junction allocations which achieve specified overall traffic congestions between each pair of exchanges. This paper considers the problem of finding a Minimum Cost network, that is, a network which satisfies the performance criterion at a minimum total junction cost. Previous models have relaxed the integrality condition on junction numbers.

1. INTRODUCTION

The minimum cost telephone network problem has been formulated as a mathematical program [1,2]

\[ \text{Minimize } C(h) \]

\[ \frac{1}{k} \sum_{j=1}^{j(k)} h^k_j = b^k \cdot t^k ; \quad k = 1, \ldots, K. \]

The variable \( h^k_j \) is the mean carried traffic on the \( j \)th route (or chain) between the \( k \)th origin-destination pair of exchanges. There are \( j(k) \) distinct permissible routes for OD pair \( k \). A feasible point \( h \), with non-negative elements \( h^k_j \), represents a chain flow pattern on the network with the property that for each OD pair \( k \) the sum of the carried traffic is a prescribed fraction \( b^k \) of the total offered traffic \( t^k \). The planned OD traffic congestion \( g^k \) \((=b^k \cdot t^k)\) may be chosen to give different performance priorities to the OD pairs.

For a full availability network, a mathematical model has been developed [1,3] which gives the number of circuits \( n \) required on the links of the network as a function of the chain flow pattern \( h \). \( C(h) \) is the total circuit cost. The nonlinear program (1) has been solved using cost and traffic dispersion data for the Adelaide metropolitan network. Two successful solution techniques have been reported [4,5]. Both approaches assume \( n \) is a continuous function of \( h \) and require some kind of rounding to integer numbers at the completion of the optimisation.

Whereas previous optimisation methods have relaxed the integrality condition on the number of circuits, this paper describes a method for determining optimal integer numbers of circuits.

2. PROBLEMS ASSOCIATED WITH THE DIRECT INTEGER FORMULATION

As an integer program, with variables \( n \) denoting the number of circuits on link 1, the minimum cost problem may be formulated

\[ \text{Minimize } C(n) \]

\[ \frac{1}{j(k)} \sum_{j=1}^{j(k)} h^k_j = n^k \cdot b^k \cdot t^k \]

\[ n = 0 \]

\[ n \text{ integer}. \]

It is usual to consider constant costs per unit circuit, \( n \). Thus the objective function representing total circuit cost, unlike that in (1), is a linear function. On the other hand the performance constraints lose the simple form of (1) and are nonlinear functions of \( n \).

The techniques used to solve (1) were successful because of the echelon-diagonal structure of the linear chain flow constraints. Not only are the corresponding constraints in (2) nonlinear, there is the added difficulty of having a non-convex set of feasible \( n \). This undesirable property is illustrated by the following example. Consider a network with a single OD pair, having two routes to the destination D. Let the components of \( n \) denote the number of circuits on links 1 and 2 respectively. If the point \((n_1,0)\) gives the required OD congestion \( E_n(t^1) \), by symmetry \((0,n_2)\) is also a feasible point. But as

\[ E_n(t^1) \neq 2E_n(t^1/2) \]

the point \( n = (n_1/2, n_2/2) \) is not a feasible point.

Let us next consider the functions \( h^k_\ast(n) \). Assuming random traffic \( t^k \) is offered to \( n \) on OD junctions on the direct route for OD pair \( k \), the functions \( h^k_\ast(n) \) are readily obtained from the Erlang loss formula

\[ h^k_\ast(n) = \frac{1}{1 - E_n(t^k)} t^k. \]

The non-randomly distributed overflow calls offered to second choice routes of the network compete for access to common links and produce the chain flows \( h^k_\ast(n) \). But, although it is possible to estimate the total carried traffic on the common link (for example by application of the equivalent random method) this total cannot be apportioned to the individual streams with sufficient accuracy. Similarly no sufficiently accurate model exists at the present time to allow \( h^k_\ast(n) \) to be determined as a function of the junction vector \( n \).

It is concluded then, that a direct formulation in terms of numbers of junctions leads to problems of considerable difficulty. No general purpose integer programming algorithm exists at the present time for solving large linear programs. We have a large nonlinear program for which the constraint set is non-convex. In addition, there is the further problem that although \( n \) may be determined as a function of \( h \) the inverse function is not known.

3. AN EXTERIOR PERIODIC PENALTY FUNCTION TECHNIQUE

It was observed, when solving problem 1, that rounding the junction numbers to integer values at intervals of 20 successive iterates gave a monotonic decreasing sequence of total cost values for the first 100 iterations. This suggests that the nonlinear function \( C(h) \) is 'well behaved' and that a reasonable approximation to the solution to problem (2) may be found by solving (1) and rounding-off circuit numbers to the nearest integer. It may be thought that a safer approach would be to round-up \( n_1 \) to the next integer.

To examine the sensitivity of OD congestions to changes of 1 circuit on the last links of final choice routes a number of simulation tests were made with a 12 OD pair alternative routing network. It was found that increases of 1 circuit on selected links could result in an in...
A necessary condition for a stationary point is that the derivative with respect to $x$ of the objective function be zero. That is,

$$3 - 2x = p \sin 2x.$$  

(12)

It can be seen from fig.1 that the solution to (12) is not unique, its value depending on both the starting point for the minimisation and the sequence of values for $p$.

Problem (2) is equivalent to

$$\text{Minimize } C(h)$$

$$\sum_{j=1}^{n} h_j = b^x + h_j; \quad k = 1, \ldots, K.$$  

(4)

$$h_k \geq 0$$

$$r_k = (p(h))_k \text{ integer.}$$

To maintain feasibility we must consider only those chain flow patterns $h$ for which $r_k$ are integers. The author recalls the difficulty of finding, by tedious adjustments, a set of flows $h$ during a simulation experiment in which the calculated numbers of circuits were required to be integers (to within 2 decimal places).

Instead, the integrality constraints are incorporated implicitly in a penalty function $\phi(q)$ which adds a positive value to the objective function whenever some $r_k$ is not an integer. That is, we replace (4) by

$$\text{Minimize } C(h) + \sum_{i=1}^{l} r_i \phi(r_i(h)).$$  

(5)

$$\sum_{j=1}^{n} h_j = b^x + h_j; \quad k = 1, \ldots, K.$$  

(6)

$$h_k^* \geq 0,$$  

(7)

I being the total number of links in the network. The parameters $r_i$ are simply non-negative weights.

Clearly, the function $\phi(q)$ must be periodic, with period 1, and it is desirable for $\phi$ to achieve a maximum value whenever the fractional part of $r_k$ is 0.5. In this paper results are reported for the case

$$\phi(r_i) = \sin^2 r_i.$$  

(8)

Let $S_i$ be the set of points satisfying (6) and (7), and define

$$F(p) = \min_{S_i} [C(h) + \sum_{i=1}^{l} r_i \phi(r_i(h))].$$  

(9)

Suppose, for some sequence of $p$ values with each component $p_i$ tending to $\infty$, that $F(p)$ approaches a finite value $L$ and that this value is achieved by the point $h^*$. It is clear from (9) that each $r_i$ must be integral otherwise $\phi(r_i(h^*))$ is positive and $F(p)$ could be made arbitrarily large. It should be noted that in general $L$ is not unique, that is, two different sequences of $p$ values may be chosen giving different limiting values.

Before outlining the proposed solution procedure in full detail the following simple example is given to illustrate certain difficulties.

$$\text{Minimize } x(x-1.5)^2$$

$$x \text{ integer.}$$  

(10)

Equation (9) gives for each $p$ the unconstrained minimisation problem

$$F(p) = \min_x [x(x-1.5)^2 + p \sin^2 2x].$$  

(11)

A necessary condition for a stationary point is that the derivative with respect to $x$ of the objective function for the unconstrained problem be zero. That is,

$$3 - 2x = p \sin 2x.$$  

(12)

It can be seen from fig.1 that the solution to (12) is not unique, its value depending on both the starting point for the minimisation and the sequence of values for $p$.

For the case $p=1$ there are two points satisfying (11). If we choose our starting point for the minimisation less than 1.5 we are likely to obtain the solution $x^* = 1.422$ (4 sig. figs.). Solving (11) for increasing values of $p$ we approach the value $x^*=1$, a solution of (10). Table 1 shows the convergence with a sequence of increasing values of $p$.

<table>
<thead>
<tr>
<th>$p$</th>
<th>10</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x^*$</td>
<td>1.422</td>
<td>1.002</td>
</tr>
</tbody>
</table>

Table 1 A sequence of values of $p$ giving convergence to a solution of (10).

It is clear that as $p$ increases the number of solutions to (12) increases; many of these solutions correspond to local minima of (11). For example, with starting point $x = 15.4$ and $p = 10,000$ Powell's algorithm finds the solution $x^* = 7$, a local minima of (12). To solve the above problem it is necessary to start either with a small value of $p$ or an initial value of $x$ near the point $x = 1.5$ (the solution to the continuous problem, i.e. $p=0$).

We now consider problem (2). The proposed approach is to find a near-optimal solution to the program given by (5), (6) and (7) with $p_i=0$ for all $i$. This point is then taken as our initial point. The parameters $r_i$ are increased to some small positive value and a new point determined by applying the minimization algorithm (in our case the Gradient Projection algorithm) for a selected number of iterations. $r_i$ are subsequently further increased and the procedure repeated until the solution vector $h$ leads to values of $r_i$ sufficiently close to integers.

It is reasonable to expect, in many applications, that the objective function is roughly quadratic near its minimum. The technique described provides a means of moving to an integer point near the point which solves the continuous problem. In other words, for our problem, the technique provides an automatic means of adjusting flows on links of the network to obtain integral numbers of circuits. We are able to use the same minimization algorithm which solves (1), only minor changes in the objective function and gradient subroutines being required.
It is interesting to contrast the technique with the sequential unconstrained minimization techniques proposed by Fiacco and McCormick [6]. The approach is an exterior penalty function method, that is, feasibility (n_t integer) is not maintained. A sequence of points is found converging to a feasible point. The auxiliary function, however, is nonmonotone. We have a periodic auxiliary function. Also, whilst it is usual to obtain a sequence of minimizing points, it has been found that it is not necessary to solve (9) to find minima for each p.

4. RESULTS

The telephone network considered has 1141 OD pairs with 755 direct junctions and 212 overflow junctions. The mathematical program has 3,037 variables H, 1141 performance constraints and 967 integrality constraints.

With very large non-linear programs it is not usually possible to achieve the optimal solution. Some convenient stopping criterion must be determined. Let

\[ q_i = \min(f_i, 1, 0) \]  

where \( f_i \) is the fractional part of \( q_i \). The algorithm is terminated when the following inequality is satisfied

\[ S(d) = \sum_{i=1}^{n} q_i < K. \]  

The value of S(d) gives a convenient measure of the 'distance' from an integer point p.

If the number of circuits in the network is large we may expect, for a continuous variable optimisation, that the values \( q_i \) are approximately uniformly distributed between 0 and 0.5 with mean 0.25. Hence S(d) would have an expected value 0.251. In our case this value is 241.75. The constant K=7 was arbitrarily chosen in the optimisation.

The objective function has a maximum value greater than $4,000,000 (corresponding approximately to all flow on overflow routes). It is known from previous computation, for the continuous problem, that the minimum value is approximately $2,800,000. The starting point for application of the solution technique outlined above gave a total circuit cost of $3,325,499. The parameters \( p_1 \) were set to the same value, 

\[ PZ = 10, \]  

where \( H(z) \) is not known, but \( z = H(y) \) is known.

The problem was reformulated as

\[ \min f(x) \] 

\[ g(H(x)) \] 

\[ x_i \text{ integer}, \]  

where \( f(x) \) is not known, but \( x = H(y) \) is known.

The problem was reformulated as

\[ \min f(H(y)) + \sum_{i=1}^{n} p_i \psi(H(y_i)) \]  

\[ g(H) \] 

where \( \psi(t) \) is an appropriate periodic penalty function, and the parameters \( p_i \) are increased during the minimization. Although the choice of both a sequence of \( p_i \) and the number of iterates between changes in their values is somewhat of an art, it has been demonstrated using real data that the method converges satisfactorily.

An extension of the method to restrict the number of circuits for each link to multiples \( 0,1,2,\ldots \) may be accomplished by choosing a suitable function \( \phi \) with period \( L \), for example \( \sin^m \pi L \).

6. BIBLIOGRAPHY


Table 2a Values for overflow link numbers \( n_i (i=1, \ldots, 100) \) at iteration 300.

<table>
<thead>
<tr>
<th>( n_i )</th>
<th>( f_{10} )</th>
<th>( f_{20} )</th>
<th>( f_{30} )</th>
<th>( f_{40} )</th>
<th>( f_{50} )</th>
<th>( f_{60} )</th>
<th>( f_{70} )</th>
<th>( f_{80} )</th>
<th>( f_{90} )</th>
<th>( f_{100} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0</td>
<td>20.0</td>
<td>15.0</td>
<td>10.0</td>
<td>5.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>4.0</td>
<td>21.0</td>
<td>16.0</td>
<td>11.0</td>
<td>6.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>3.0</td>
<td>22.0</td>
<td>17.0</td>
<td>12.0</td>
<td>7.0</td>
<td>2.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>2.0</td>
<td>23.0</td>
<td>18.0</td>
<td>13.0</td>
<td>8.0</td>
<td>3.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.0</td>
<td>24.0</td>
<td>19.0</td>
<td>14.0</td>
<td>9.0</td>
<td>4.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Table 2b Values for overflow link numbers \( n_i (i=1, \ldots, 100) \) at iteration 359.

<table>
<thead>
<tr>
<th>( n_i )</th>
<th>( f_{10} )</th>
<th>( f_{20} )</th>
<th>( f_{30} )</th>
<th>( f_{40} )</th>
<th>( f_{50} )</th>
<th>( f_{60} )</th>
<th>( f_{70} )</th>
<th>( f_{80} )</th>
<th>( f_{90} )</th>
<th>( f_{100} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0</td>
<td>20.0</td>
<td>15.0</td>
<td>10.0</td>
<td>5.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>4.0</td>
<td>21.0</td>
<td>16.0</td>
<td>11.0</td>
<td>6.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>3.0</td>
<td>22.0</td>
<td>17.0</td>
<td>12.0</td>
<td>7.0</td>
<td>2.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>2.0</td>
<td>23.0</td>
<td>18.0</td>
<td>13.0</td>
<td>8.0</td>
<td>3.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.0</td>
<td>24.0</td>
<td>19.0</td>
<td>14.0</td>
<td>9.0</td>
<td>4.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>
