CONTRAM 5: AN ENHANCED TRAFFIC ASSIGNMENT MODEL

by N B Taylor

The views expressed in this report are not necessarily those of the Department of Transport.
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>1</td>
</tr>
<tr>
<td>1. Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2. Delay, cost and assignment</td>
<td>1</td>
</tr>
<tr>
<td>3. Speed/flow relationships</td>
<td>2</td>
</tr>
<tr>
<td>4. Geometric delay</td>
<td>3</td>
</tr>
<tr>
<td>5. Signal coordination model</td>
<td>3</td>
</tr>
<tr>
<td>6. Queue and delay model</td>
<td>4</td>
</tr>
<tr>
<td>7. Signal queues with coordination</td>
<td>4</td>
</tr>
<tr>
<td>8. Generalised cost</td>
<td>5</td>
</tr>
<tr>
<td>9. Fuel consumption model</td>
<td>5</td>
</tr>
<tr>
<td>10. Packet generation and packet sizes</td>
<td>6</td>
</tr>
<tr>
<td>11. Assignment</td>
<td>7</td>
</tr>
<tr>
<td>11.1 Optimum route assignment</td>
<td>7</td>
</tr>
<tr>
<td>11.2 Stabilisation and freezing of packets’ routes</td>
<td>7</td>
</tr>
<tr>
<td>11.3 Storage of route information</td>
<td>7</td>
</tr>
<tr>
<td>12. Program characteristics</td>
<td>7</td>
</tr>
<tr>
<td>12.1 Program structure</td>
<td>7</td>
</tr>
<tr>
<td>12.2 Files</td>
<td>8</td>
</tr>
<tr>
<td>12.3 Data storage</td>
<td>8</td>
</tr>
<tr>
<td>12.4 Method of calculation</td>
<td>8</td>
</tr>
<tr>
<td>12.5 Code size, memory requirement and run time</td>
<td>8</td>
</tr>
<tr>
<td>13. Summary and conclusions</td>
<td>9</td>
</tr>
<tr>
<td>14. Acknowledgements</td>
<td>9</td>
</tr>
<tr>
<td>15. References</td>
<td>10</td>
</tr>
<tr>
<td>Appendix A Speed/flow relationships</td>
<td>10</td>
</tr>
<tr>
<td>Appendix B Geometric delay</td>
<td>11</td>
</tr>
<tr>
<td>Appendix C Signal coordination</td>
<td>11</td>
</tr>
<tr>
<td>Appendix D Queue and delay models</td>
<td>13</td>
</tr>
<tr>
<td>Appendix E Generalised cost functions</td>
<td>15</td>
</tr>
<tr>
<td>Appendix F Fuel consumption model</td>
<td>16</td>
</tr>
<tr>
<td>Appendix G Packet generator</td>
<td>18</td>
</tr>
<tr>
<td>Appendix H Algorithms</td>
<td>18</td>
</tr>
<tr>
<td>Appendix I Memory requirement and run time</td>
<td>20</td>
</tr>
<tr>
<td>Appendix F Fuel consumption model</td>
<td>16</td>
</tr>
<tr>
<td>Appendix G Packet generator</td>
<td>18</td>
</tr>
<tr>
<td>Appendix H Algorithms</td>
<td>18</td>
</tr>
<tr>
<td>Appendix I Memory requirement and run time</td>
<td>20</td>
</tr>
</tbody>
</table>

© CROWN COPYRIGHT 1990

Extracts from the text may be reproduced, except for commercial purposes, provided the source is acknowledged.
CONTRAM 5: AN ENHANCED TRAFFIC ASSIGNMENT MODEL

ABSTRACT

CONTRAM 5 is the latest version of the Transport and Road Research Laboratory's traffic assignment program CONTRAM which models time varying traffic demands on urban and other road networks subject to capacity restraint and transient overload, and predicts the variation through time of the resulting routes, queues and delays.

CONTRAM 5 contains many improvements to the original modelling and introduces some new facilities. The queue and delay models are compatible with those in the junction analysis programs ARCADY2, PICADY2 and OSCADY. The effects of signal coordination and geometric delay are modelled. Speed/flow relationships, compatible with those in COBA, can be used to represent traffic speeds on high-speed and limited-access roads and to represent the aggregate effect of delays in buffer networks. A revised model is used to estimate fuel consumption and the number of stops at junctions is estimated.

The CONTRAM 5 program incorporates improved algorithms for loading and optimum route assignment of traffic and benefits from a major restructuring of the software compared to previous versions of CONTRAM. The need to re-configure the software for different networks is almost completely eliminated by the use of automatic array dimensioning. Methods of predicting memory use and run time have been established.

The main text of the Report outlines and discusses the new and modified features of CONTRAM 5, while formal definitions of models and procedures are given in Appendices. This Report is complementary to RR 178 (CONTRAM: structure of the model. Leonard, Gower and Taylor 1989) and describes changes and features specific to CONTRAM 5.

1 INTRODUCTION

CONTRAM 5 (Continuous Traffic Assignment Model Version 5) is the latest version of the CONTRAM program originally written in the early 1970s. The purpose of this Research Report is to describe specifically those modelling features which are new or significantly enhanced in CONTRAM 5. The basic structure and operation of CONTRAM are described in RR 178 (Leonard, Gower and Taylor 1989), which this Report complements.

Urban and peri-urban road networks frequently operate close to capacity or under transient overload, where dynamic interactions of traffic and time-dependent queueing and delay processes become dominant. These processes are complex and can only be modelled satisfactorily by a method which takes proper account of time variation. CONTRAM 5 models time-varying traffic on urban and other road networks which are subject to transient overload and capacity restraint (based on junction or speed/flow effects), and predicts the variation through time of routes, queues, delays, the effects of signal timings and coordination, fuel consumption, and numbers of stops.

The representation of queueing and delay in CONTRAM 5 is based upon robust models supported by extensive research. The models of queues are compatible with those in TRRL's Junction Model Programs ARCADY2, PICADY2 and OSCADY (Semmens 1985a, b, Burrow 1987). The formulations of speed/flow relationships and generalised cost are based on those developed for COBA (COBA9 1987).

The models and features covered by this Report are the following:

- Speed/flow relationships
- Geometric delay
- Signal coordination
- Queues and delays, with extension to allow for signal coordination
- Generalised cost
- Fuel consumption
- Packet generation and packet sizes
- Optimum routes and assignment
- Program structure and performance

While this latest version of CONTRAM is capable of representing traffic conditions on a road network to a high degree of precision it is essential that users interpret its results with an appreciation of the quality of input data and the program structure and with a practical understanding of the variability of driver behaviour and network performance. Readers are advised to consult RR178 as essential background to the present Report.

2 DELAY, COST AND ASSIGNMENT

Route assignment in CONTRAM 5 consists of finding the route of minimum cost between a vehicle’s origin and destination, given the state of the network as the driver would find it. The cost of the route is the sum of the costs incurred in travelling along the links making up the route. Possible routes are extended
one link at a time, the current link being included if the links or destination immediately downstream cannot already be reached by a cheaper route (see Section 11).

The cost function can be pure travel time or a generalised cost function depending mainly on travel distance and travel time, with certain additional components described later in Section 8. The function contains an element of fuel cost, but CONTRAM makes a separate, more detailed, estimate of fuel consumption which does not enter into assignment (see Section 9). The only flow-dependent element of cost is travel time and it is this that consumes the most computational effort. The total travel time along a link is calculated as:

\[ t_{\text{depart}} - t_{\text{enter}} = t_{\text{cruise}} + t_{\text{geometric}} + t_{\text{queue}} \quad (2.1) \]

or \[ t_{\text{speed/flow}} + t_{\text{geometric}} + t_{\text{queue}} \]

where \( t_{\text{enter}} \) is the time at which the vehicle enters the upstream end of the link, and \( t_{\text{depart}} \) is the time at which the vehicle leaves the downstream end of the link, calculated by the following steps:

1. \[ t_{\text{moving}} = t_{\text{enter}} + t_{\text{cruise}} \quad \text{or} \quad t_{\text{moving}} = t_{\text{enter}} + t_{\text{speed/flow}} \quad (2.2) \]

   where \( t_{\text{cruise}} \) is a flow-independent cruise or running time and \( t_{\text{speed/flow}} \) is the alternative cruise time calculated by means of a speed/flow relationship, which depends on the flow entering the upstream end of the link (see Section 3).

2. \[ t_{\text{arrive}} = t_{\text{moving}} + t_{\text{geometric}} \quad (2.3) \]

   where \( t_{\text{geometric}} \) is the geometric delay involved in negotiating the upstream junction, which depends on the cruise speed established by (1) above (see Section 4). (Note: It is assumed that any overriding effect of queueing over geometric delay is taken into account in the queueing delay.)

3. \[ t_{\text{depart}} = t_{\text{arrive}} + t_{\text{queue}} \quad (2.4) \]

   where \( t_{\text{queue}} \) is the queueing delay, evaluated at the time \( t_{\text{arrive}} \) when the vehicle joins the queue or arrives at the stop line (vertical queueing is assumed.) This delay depends on the demand and capacity on the link and the effects of opposing flows, signal settings and signal coordination (see Sections 5, 6 and 7).

3 SPEED/FLOW RELATIONSHIPS

Speed/flow relationships are intended to be used in CONTRAM for two main purposes: to represent cruise speeds on high-speed and limited-access roads; and to take account of the aggregate effect of delays in buffer networks where these are considered appropriate (see also CONTRAM Userguides). The effect of a speed/flow relationship is in addition to any delay due to explicit queueing at the downstream end of the link to which it applies (see Section 2).

CONTRAM 5 uses COBA-type speed/flow relationships whose general form consists of two linear sections of different slope (see Figure 1). The exact form of each relationship is determined by entering as data three points through which it passes—the free speed (where flow is zero), the 'break point' (flow and speed) where the slope changes, and the 'capacity point' (flow and speed) which is a point through which the second section passes. This last point need not actually represent 'capacity', which is in any case difficult to determine for a road section carrying moving traffic, but it is convenient to identify it with the COBA 'capacity' which is 'set to correspond roughly to the highest level of traffic flow that has been observed' (COBA9 Section 5.7.1). A minimum speed cut-off can also be entered. The method of data entry follows the COBA convention of specifying the relationship in terms of flow per lane, and uses a factor to convert this to total link flow.

Most traffic streams contain a mix of vehicle types resulting in speeds intermediate between those of pure light and heavy vehicle streams. On some road types heavy vehicles have their own speed/flow relationships. CONTRAM allows these to be linked to the corresponding relationships for light vehicles. On other road types the speed/flow function of a stream of heavy vehicles is assumed to be equivalent to that of a stream of light vehicles shifted by a fixed amount of km/h. In this case the free speed of heavy vehicles is given together with the flow at which the speeds of light and heavy vehicles become equal due to their interactions. CONTRAM uses PCU factors to characterise individual vehicles and the whole traffic stream, assuming that a typical light vehicle has a PCU value of 1 and a typical heavy vehicle has a PCU value of 2, intermediate between COBA classes OGV1 and OGV2 (other goods vehicle classes 1 and 2—see COBA9 Section 4.5). Speeds for vehicle types with PCU values between these limits are interpolated. The speed/flow model is defined formally in Appendix A.
4 GEOMETRIC DELAY

Geometric delay is defined in CONTRAM 5 to be the component of delay due to deceleration and acceleration at a junction, excluding queueing delay, which contributes to actual journey time and hence may influence assignment (this differs from ARCADY2, PICADY2 and OSCADY in which geometric delay is defined to be the difference between the actual journey time through the junction, excluding queueing delay, and the journey time through an idealised junction).

In previous versions geometric delay was included implicitly in link cruise times, but CONTRAM 5 provides the option to specify it explicitly for each separate turning movement (in which case the link cruise time or speed represents travel at free speed rather than average cruise speed). The geometric delay effect on a turning movement can be entered either as a number or seconds delay or in the form of speed in the junction. Formulae for calculating speeds for various manoeuvres at different types of junction are incorporated in ARCADY2, PICADY2 and OSCADY and are given in Appendices to RR35, RR36 and RR105 (Semmens 1985a,b, Burrow 1987).

Because geometric delay is treated as an integral part of journey time, it is not recorded separately but is included in the category of 'freemoving' travel time, which is defined to be the travel time in the absence of the effects of other traffic such as queueing delay and speed/flow effects. The geometric delay model is given formally in Appendix B.

5 SIGNAL COORDINATION MODEL

The signal coordination model in CONTRAM 5 estimates the effect of coordination but does not explicitly model platoon structure, dispersion or the detailed timing of arrivals with respect to signal phases. Two signals are assumed to be coordinated if:

(1) Their cycle times are identical, or differ by a factor of two exactly.

(2) The offset of the start of stage 1 green (relative to an arbitrary zero) is specified for both signals in the network data.

In addition, in order that the effect of coordination may be calculated, the signals must be adjacent in the sense that some traffic flows between them without passing through any intervening signals. The following assumptions are made in the calculations:

(a) The maximum effective green time seen by the traffic stream arriving at the downstream signal is that obtained by mapping green waves of uniform traffic flow from the upstream signal onto the phasing of the downstream signal, with due allowance for travel time and the offset between the signals. The probability that an arriving vehicle (packet) meets green is then just the proportion of the arriving green wave which coincides with a green phase at the downstream signal. A queue at the downstream signal is assumed to have no direct effect—ie the back of the queue is assumed to move in step with the front, and due to the assumption of vertical queueing, travel (cruise) time between the signals is not affected.

(b) The effect of dispersion is described by the formula (Robertson and Hunt 1982):

\[ k = \frac{1}{1 + t} \]  

where \( t \) is travel (cruise) time in minutes. This has the effect of reducing the efficiency of coordination as travel time increases. For example, a travel time of 60 seconds between signals halves the benefit of coordination.

(c) The effect of ageing of fixed time plans can be accounted for by reducing the efficiency of coordination by a factor entered in the data as a percentage.

The theoretical maximum efficiency of coordination, as used here, is defined by:

\[ e_m = (1 - \lambda) / (1 - \lambda) \]  

and \( g \) and \( c \) are respectively the total green and cycle times on the arriving arm of the downstream signal. \( \lambda_{\text{max}} \) is the maximum effective green fraction—ie the probability that arriving traffic meets green in the absence of dispersion or other effects. If there is no coordination then \( \lambda_{\text{max}} = \lambda \) and \( e_m = 0 \). If \( \lambda_{\text{max}} = 1 \), ie arriving traffic always finds the signal at green, then \( e_m = 1 \). If \( \lambda_{\text{max}} < \lambda \) then the signal coordination actually causes a disbenefit to the arriving traffic and \( e_m \) is negative. This can occur in practice for some minor or cross movements in networks where coordination is designed to favour major or through movements (ibid).

The combined effects of dispersion and plan ageing are then taken into account by defining the effective coordination efficiency \( e \) as follows:

\[ e = f \cdot e_m \]  

where \( f \) is the factor mentioned in (c) above, and the effective green fraction for traffic arriving at the downstream junction is obtained by inverting equation (5.2) to give:

\[ \lambda_b = e + (1 - e) \lambda \]  

Cruise time of individual vehicles in CONTRAM can vary according to vehicle class, the effect of speed/flow relationships and the crossing of time-slice boundaries, so the total flow arriving at the downstream junction may be composed of a mixture of streams with differing coordination which
contribute to the queue to varying degrees. Therefore CONTRAM calculates the coordination efficiency of each individual vehicle and obtains the mean coordination of the traffic stream as a whole by summing over the packet flows \( q_j \):

\[
E = \frac{\sum q_j \rho_j}{\sum q_j} \quad (5.5)
\]

The main effect of signal coordination is that a systematic difference arises between the mean arrival rates of vehicles in green and red signal phases. This effect is the basis of the modified signal queue model discussed in Section 7. The coordination model is defined formally in Appendix C.

6 QUEUE AND DELAY MODEL

CONTRAM 5 calculates the lengths of queues using time-dependent stochastic queueing theory. Random- and oversaturation queues are calculated using the 'divided' queue formulae based on the sheared queue model (Kimber and Hollis 1979, Kimber and Daly 1986), and other formulae are used to calculate phase queues due to signals. 'Vertical' queueing is assumed—that is the queueing process is formally defined as occurring at the stop line. The basic queue formulae are the same as those used in ARCADY2, PICADY2 and OSCADY (Semmens 1985a,b, Burrow 1987). The queue lengths on a link are recalculated for each vehicle (packet) that arrives, as a function of its arrival time.

Interpolation is not used, as was the case in some earlier versions of CONTRAM. Both the random- and oversaturation and the phase queues at signals are affected by signal coordination, which is taken into account by a modification to the queue model (Section 7). The formal definition of the queue and delay models is given in Appendix D.

Delay is calculated per arriving vehicle (not per unit time). As in earlier versions of CONTRAM, the method of calculation must allow for the possibility that a vehicle may be subject to different capacities if it queues in more than one time slice. The model differs from that used in earlier versions in its incorporation of the explicit rate of change of queue length. The rate of vehicle departures from the queue at time \( t \) is the product of the utilisation \( U(t) \) and the capacity \( \mu \), which satisfies:

\[
dL(t)/dt = \rho U(t) \mu = \text{arrival rate} - \text{departure rate} \quad (6.1)
\]

where \( L(t) \) is the queue length and \( \rho \) is the traffic intensity (arrivals divided by capacity). The utilisation \( U(t) \) is the mean proportion of the available capacity currently being used, ie the probability that a queue is present.

After a vehicle joins the queue it is unaffected by changes in the arrival rate but, for as long as it remains in the queue, it is directly affected by changes in capacity, in proportion to their duration. CONTRAM represents this by defining the total delay to the vehicle as:

\[
\begin{align*}
d &= \sum d_i, \\
L &= U \sum \mu_i d_i
\end{align*}
\]

where \( \mu_i \) and \( d_i \) are respectively the capacity and delay in time slice \( i \), and \( L \) and \( U \) are the queue length and utilisation at the moment when the vehicle joins the queue, satisfying equation (6.1).

Each component of delay \( d_i \) is equal to the part or whole of the time slice \( i \) during which the vehicle remains in the queue. The \( d_i \) are calculated by a straightforward iterative method (see Appendix D). In the case where only one time slice is involved (or where all the \( \mu_i \) are equal) the delay formula simplifies to:

\[
d = \frac{L}{\mu} \quad (6.3)
\]

Total delay to all vehicles in each time slice is obtained by summing the components of vehicle delay \( d_i \) for each \( i \), thus ensuring that all delay estimates are internally consistent.

Delay at signals includes a component due to the phase queue, which is taken into account by adding the phase queue to the random queue \( L \). An estimate of the probability of a vehicle having to stop is also output (see Appendix D).

7 SIGNAL QUEUES WITH COORDINATION

In the absence of coordination the arrival rate of traffic at a signal is assumed to be equal to the mean arrival rate \( q \) in both the green and red phases. However, in the presence of coordination, indicated by a non-zero value of \( E \) in equation (5.5), the mean arrival rates in the green and red phases are different and are given by (see equations 5.4-5):

\[
\begin{align*}
q_g &= \lambda q \lambda = (E + (1 - E)\lambda)q/\lambda \\
q_r &= (1 - E)q
\end{align*}
\]

where \( \lambda = g/c \) at the downstream signal. It is easily shown that equations (7.1) satisfy the same identity as in the uncoordinated case, viz:

\[
q = \lambda q_g + (1 - \lambda)q_r \quad (7.2)
\]

To simplify the analysis variables are transformed so that the transformed arrival rate is the same in both green and red phases, and the capacity is adjusted accordingly, thus making the situation formally equivalent to the uncoordinated case:
Let \( q^* = q_i \)

Define \( \mu^* \) by \( q^* - \mu^*/\lambda = q_0 - \mu/\lambda \) \( (7.3) \)

Define \( q^* \) by \( q^* = q_i/\mu^* \)

Substituting for \( q_0 \) and \( q_r \) from equations (7.1) gives the explicit transformations:

\[
\begin{align*}
q^* &= (1 - E) q_i \\
(1 - E \mu)
\end{align*}
\]

The transformed variables \( q^* \) and \( \mu^* \) reduce to \( q \) and \( \mu \) if \( E = 0 \). As \( E \) increases both \( q \) and \( \mu \) decrease, which has the effect of reducing the phase queues in both red and green phases. The random-and-oversaturation queue, which mainly represents the net effect of random variations or oversaturation causing phase queues to persist from one red to the next, is assumed to be given by standard queue formulae with \( q^* \) and \( \mu^* \) substituted for \( q \) and \( \mu \). The possible residual effect of coordination on the randomness parameter (see Kimber and Hollis 1979) is thought to be small, so no allowance is made for this. The transformation (7.4) is such that if \( q = 1 \) then \( q^* = 1 \) also. The singularity of the transformation at \( q = 1/E \) (which is always \( > 1 \)) causes no difficulties with queue calculations because \( q \) always appears either explicitly multiplied by \( \mu \), so cancelling the singular term, or in an expression valid only for \( q < 1 \), and \( \mu \) never appears as a divisor.

The results of the calculations given in Appendix D consist of \( L_q \) and \( L_r \), the mean phase queue lengths in the green and red phases respectively. However, each individual vehicle has its own probability of arriving in a red or green phase as determined by its coordination efficiency \( e \). Therefore the expected phase queue encountered by a vehicle depends on the vehicle's effective green proportion (equation (5.4)), as follows:

\[
L_q = \lambda_q L_q + (1 - \lambda_q) L_r \quad (7.5)
\]

The random-and-oversaturation queue is phase-independent and depends only on \( E \).

### 8 GENERALISED COST

CONTRAM 5 calculates two kinds of generalised cost which, though similar in functional form, are quite distinct in interpretation. **Perceived cost** is the notional cost that is perceived by drivers and which they seek to minimise by their choice of route (behavioural cost is almost the same thing—drivers' perceived cost inferred from their actual behaviour). **Resource cost** is assumed to represent the 'real' cost of travel and in CONTRAM is purely an output quantity which has no effect on route choice. For example, the perceived cost of a litre of petrol is the cost at the pump, including taxes, while its resource cost is the value of the petrol net of tax but possibly including the cost of pollution resulting from its production and consumption.

Assignment by perceived cost other than pure journey time in UK Scheme Assessment is subject to the advice given in TAM, Section 9.6.1.

In CONTRAM the functional form of both perceived and resource cost is based on the form of resource cost function defined in COBA9 Section 2, viz:

\[
C = aD + bT + cV^2D \quad (8.1)
\]

which expresses cost in terms of distance \( D \), time \( T \) and average speed \( V \). CONTRAM has extensions which allow the following to be taken into account:

- tolls on links
- notional extra cost of queueing delay
- notional cost of stopping
- notional cost of making a deterrent turning movement
- notional cost of negotiating a junction

A 'deterrent' movement is a link-to-link movement indicated as deterrent in the data. This could represent a right turn in countries which 'drive on the left' or a left turn in countries which 'drive on the right', which might be ascribed a cost over that associated with delay, for example because of an increased risk of accidents. The generalised cost functions are defined formally in Appendix E.

### 9 FUEL CONSUMPTION MODEL

This model is used in the final iteration of a CONTRAM run to calculate fuel consumption. It is derived from several sources. At its heart is the model of fuel consumption at steady speed developed by Everall 1968, to which the form of the COBA generalised cost formulae (8.1) is closely related, viz:

\[
F = AD + BT + CV^2D \quad (9.1)
\]

Extensions to take account of kinematic effects are based on an elemental model developed by the Australian Road Research Board (ARRB) (Bowyer, Akçelik and Biggs 1985, Biggs and Akçelik 1986). The effects of queueing are derived from a regression model of fuel consumption at roundabouts by TRRL (Gardiner, Baker and Lucas 1986). Finally, parameters for goods vehicles are obtained from work by TRRL (Renouf 1981).

In addition to the three coefficients of Everall's formula there are three coefficients relating to kinematic effects, namely fluctuations in cruise speed, and deceleration and acceleration at junctions.
The first coefficient is vehicle mass \( M \). The remaining two, \( e_1 \) energy efficiency and \( e_2 \) energy/acceleration efficiency, are linked to engine performance (see Akçelik et al).

Fuel consumption while queueing depends only on queue length, queueing time and idling fuel consumption (equivalent to coefficient \( B \) in equation (9.1)) and covers both waiting and moving up at crawl pace.

CONTRAM 5 has three built-in sets of coefficients, one for light vehicles (class C), one for buses (class \( B \)) and one for heavy vehicles (class \( L \)). The coefficients \( A \), \( B \) and \( C \) for vehicle class \( C \) are derived from unpublished research by TRRL up to 1985 and are representative of British and European cars of the early 1980s. The coefficients for class \( L \) are broadly representative of the class OGV1 (other goods vehicle class 1—see COBA9 Manual), and those for class \( B \) are identical except for vehicle mass. User-defined sets of coefficients can be input if required (see CONTRAM Userguides). Values of \( e_2 \) is known only for cars but results are not particularly sensitive to it so it has been assumed to apply to all vehicles.

Appendix F contains a formal definition of the model, and Tables which give the values of coefficients that have so far been determined for cars and goods vehicles.

The predictions of the CONTRAM 5 fuel model differ quite significantly from those of earlier versions of CONTRAM (Leonard and Gower 1982) in regard to excess fuel consumed during delay, as shown by Table 1.

**TABLE 1**

Average fuel consumption on a simple signal-controlled link, using parameters for Wolseley 2.2L car. Ranges in RH column are due to time variation. Level of demand is defined as (initial queue + arrivals)/capacity.

<table>
<thead>
<tr>
<th>Cruise speed (km/h)</th>
<th>Level of demand</th>
<th>Fuel consumption at Cruise speed (mL/m)</th>
<th>Excess fuel consumption per unit delay (mL/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>CONTRAM 4F</td>
<td>CONTRAM 5</td>
</tr>
<tr>
<td>36</td>
<td>.45</td>
<td>.104</td>
<td>.101</td>
</tr>
<tr>
<td></td>
<td>.60</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>.90</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.20</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>.45</td>
<td>.092</td>
<td>.091</td>
</tr>
<tr>
<td></td>
<td>.60</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>.90</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.20</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.50</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

10 PACKET GENERATION AND PACKET SIZES

CONTRAM loads traffic onto the network in increments called ‘packets’, each consisting of a whole number (typically in the range 1–20) of vehicles of the same type assigned at the same time between the same origin and destination. For a full explanation of packets see Leonard, Gower and Taylor 1989.

The origin-destination (o-d) demand data determine the flow rates entering the network for each o-d pair and vehicle class in each time slice. These are translated into numbers of vehicles which in turn translated into numbers and sizes of packets. Packets waiting to be loaded onto the network (from all o-d pairs and vehicle classes) are placed together in a ‘heap’ (Knuth 1973), which is in fact a list of the o-d movements partially sorted in order of increasing journey start time of the next packet. The method is designed to mingle the different o-d movements in a repeatable but pseudo-random fashion, producing an even distribution of traffic over the network.

The default mode of packet generation in CONTRAM 5 is ‘variable packet size’. This means that packet size can be adjusted up to a certain maximum value, which is also the target value, so as to match to within 1 vehicle the demand specified in the o-d data. The mechanism of packet generation is defined formally in Appendix G.

The maximum packet size for each classified o-d movement can be specified in the data or calculated automatically (subject to an optional scaling factor and an optional upper limit). The optimum choice of packet size is necessarily a compromise: too small a
packet size results in a large number of packets and increased run time without any significant benefit to accuracy; too large a packet size produces a 'grainy' loading and possibly an unrealistic assignment. A useful target value for the packet size for a particular o-d movement is given by the following formula:

\[ P = \frac{\sum Q_i}{\sum (2Q_i)^{1/2}} \]  

(10.1)

where \( Q_i \) is the number of vehicles to be loaded in the ith time slice. For example, if all \( Q_i \) were 100 then the target packet size would be 14. In practice this would mean that 8 packets would be generated (because 7 packets of 14 vehicles make a total of only 98 vehicles) in an alternating sequence of packets of size 12 and 13 vehicles.

11 ASSIGNMENT

This Section outlines certain algorithmic techniques which contribute significantly to CONTRAM 5 performance by reducing memory requirements or run time. They are described in more detail in Appendix H. The techniques described in Sections 11.1–2 are employed by default, but can optionally be suppressed by settings in the Control File (see Section 12.2 and CONTRAM Userguides).

11.1 OPTIMUM ROUTE ALGORITHM

The method of assignment in CONTRAM 5 is a modified form of Dijkstra's algorithm (Dijkstra 1959, Whiting and Hillier 1960) which at any point on a route seeks to minimise the sum of the actual cost from the origin to that point and an estimate of the minimum cost from that point to the destination, which is generated using D'Esopo's algorithm (Van Vliet 1977). The minimum cost to destination acts heuristically to direct the search towards the destination without altering the final result. The method is capable of reducing substantially the number of queue evaluations, leading to a reduction in overall run time of over 50 per cent compared with a conventional algorithm (Taylor 1989).

11.2 STABILISATION AND FREEZING OF PACKETS' ROUTES

CONTRAM 5 takes advantage of the fact that the routes and link arrival times of packets entering the network near the beginning or the end of the modelled period tend to settle down first. As a first step the routes of these packets can be stabilised, that is their sequence of links fixed but arrival times allowed to vary. As a second step certain packets whose times are unlikely to change can be frozen and effectively removed from all subsequent iterations, only the flow they generate on the links of their routes being retained. The saving in computation time which results from these methods is network dependent, but in the case of the User Guide Test Network (see CONTRAM Userguides), the time taken to perform an iteration is reduced by a maximum of 43 per cent and overall run time is reduced by 15 per cent.

11.3 STORAGE OF ROUTE INFORMATION

CONTRAM 5 uses a compression technique which allows routes to be stored in internal memory, thereby avoiding the use of the large temporary disc files required by earlier versions. The amount of memory taken by routes is typically less than 10% of the total memory devoted to all data and results (see Table 2, Section 12). In the event of internal memory being insufficient, temporary disc files are used to 'back up' the internal route store in transparent 'virtual memory' fashion. These files are much smaller than the temporary files used by earlier versions and are accessed much less frequently.

The method offers the greatest benefit on computers with relatively long disc access times, such as PCs, and can also increase efficiency on time-shared mainframes by reducing the number of times that the program is 'swapped' in and out of the computer's memory during execution. In technical terms, CONTRAM 5 is strongly 'core-bound'.

12 PROGRAM CHARACTERISTICS

12.1 PROGRAM STRUCTURE

Whilst the logical principles and structure remain, the 'physical' structure of the CONTRAM 5 code has been substantially revised compared with Version 4, and much of the code rewritten. The main code of CONTRAM 5 is written in ANSI-standard FORTRAN 66 to ensure that it can be compiled on a wide range of users' machines. The program also contains a package of machine-dependent subroutines which act as an interface with the host machine and the user. The main code is divided into 41 subprograms which are listed in an Appendix to the CONTRAM 5 User Guide (CONTRAM Userguides). These are grouped logically into five segments, which are executed in sequence:

Root—containing the main program, memory management, process control and machine-dependent interface and the packet generator and route storage routines. COMMON areas are designed to be included at this level.

Pre-read—which reads the data files to establish the required dimensions of arrays.

Input—which reads the data files again, organises the data and checks for errors.
Assignment—which contains all the traffic modelling routines, including calculation of routes, flows, queues and delays, setting of optimised signals, calculation of fuel consumption etc., and within which all the iterations of the assignment take place.

Output—which collects and tabulates the results from the final iteration of the assignment, and Convergence Monitor information collected from all iterations.

Note: Segmentation is necessary in order to accommodate the program on TRRL’s Cyber 730 and 815 mainframes and on IBM PC compatibles, but many modern computers have virtual memory which does not require segmentation of programs.

12.2 FILES
CONTRAM 5 uses up to six permanent files, including three data files: Network, Demand and Control; and three output files: Results, Packet Routes and Post-Analysis Output. These are all fully portable ASCII line files. In some cases two small binary temporary files may also be used (see Section 11.3).

The contents and formats of the data and output files are described in detail in the CONTRAM Userguides. To summarise, the Network File contains all data for the network, including time slice definitions, zones, links, signal timings, fixed routes and generalised cost functions. The Demand File contains the time-sliced traffic demands for each origin, destination and vehicle class. The Control File specifies the number of iterations to be performed and the outputs to be produced. The Results File contains an annotated listing of the data and tables of results in a form suitable for printing. The Post-Analysis Output File contains the same information in a structured form suitable for use by another computer program. The Routes File contains the final routes and link departure times of all packets in the order in which they were assigned.

12.3 DATA STORAGE
In order to be able to accommodate many different network configurations within a given available amount of memory CONTRAM 5 uses soft arrays defined by partitioning a single large main array. Only this main array needs to be dimensioned before compilation. The dimensions and locations of the soft arrays are determined from the data at the pre-read stage. This is an enhanced version of the method originally developed for the PC version of CONTRAM 4F by the firm of consultants Mott, Hay and Anderson.

A machine-dependent memory management facility allows memory to be allocated or extended at run time, and accessed by over-indexing the main array name. These facilities enable CONTRAM 5 to reduce resource usage by acquiring memory only when it is needed (as on TRRL’s Cybers), or to fit into varying amounts of available memory on PCs (which may have resident software). Details are given in the CONTRAM Userguides.

The first index of each soft array is used to select a sub-array within it. The sub-arrays of a soft array share a common dimensional structure but can contain otherwise unrelated data. This system increases code efficiency by substantially reducing the lengths of the argument lists needed for transferring arrays to subroutines, since only a relatively small number of soft array names need to be transferred, sub-arrays being indexed using a COMMON table of PARAMETERS or constants included in each subroutine. A complete list of all soft arrays and their sub-arrays is included in an Appendix to the CONTRAM 5 User Guide (CONTRAM Userguides).

12.4 METHOD OF CALCULATION
CONTRAM 5 relies heavily on integer calculations both for the sake of efficiency and to ensure reproducibility of results on different types of computer. Packet timing, cost and route calculations are particularly sensitive to small changes in values which, though they might not be significant in practical terms, could lead to a different assignment being generated. Queue length and similar sub-model calculations are mostly performed using real arithmetic since their outcomes are not significantly affected by arithmetical precision.

The use of integer arithmetic where appropriate ensures that the outcomes of sensitive calculations are precisely defined in a machine-independent way. However, it can carry an attendant risk of arithmetic overflow. This is prevented in CONTRAM either by providing REAL back-up code to handle out-of-range values where necessary, or by constraining the permitted range of the data and designing calculations so that their intermediate and final results are always within range. These measures inevitably add considerably to the program’s complexity.

12.5 CODE SIZE, MEMORY REQUIREMENT AND RUN TIME
The CONTRAM 5 source code contains around 17800 lines of FORTRAN, including comments, after execution of all INCLUDE statements. The root, COMMON areas and largest segment of object code together occupy around 22,000 words of memory on TRRL’s 60-bit Cyber mainframes and around 190,000 bytes on an IBM-compatible PC, when compiled using the Prospero™ Fortran 77 iid compiler.

The remaining memory requirement is for the main array used to store data and results. The maximum size of this array on an IBM-compatible PC with
'640K' bytes of RAM is typically in the range 80,000–100,000 depending on the amount of other software already resident. As the PC version is able to make use of INTEGER*2 this memory is in practice worth about 20% more than when only INTEGER*4 arrays are used. The data memory required for a given network depends on several network parameters but, on the basis of empirical data and some general arguments, the following simplified formulae predict memory requirement $W$ in terms of number of links $L$ only:

If all integer arrays are declared INTEGER*4:

$$W(I=4) = 250 + 295L + L^{634} (L + 7.5 L^{634} + 21) \text{ words}$$

(12.1)

If INTEGER*2 arrays are used wherever permissible:

$$W(I=2) = 235 + 270L + L^{634} (L + 4 L^{634} + 10.5) \text{ words}$$

(12.2)

Run time of CONTRAM 5 depends on the type of computer being used, the number of links in the network, the number of packets assigned and the number of iterations performed. On the same basis as above the following simplified formula gives an estimate of run time on TRRL's Cyber mainframes or a Compaq 386/25 in terms of number of links $L$ and the number of iterations $N$:

$$T_{CPU} = 0.14 N L^{1.384} \text{ seconds}$$

(12.3)

The derivation of these formulae is described in more detail in Appendix 1. Some actual examples taken from runs on TRRL's Cyber mainframes are given in the Table 2 below.

### 13 SUMMARY AND CONCLUSIONS

CONTRAM 5 contains a number of improved features compared with earlier versions. These have been designed, in particular:

- To make the program's queue length and delay calculations compatible with those of TRRL's Junction Analysis Programs ARCADY2, PICADY2 and OSCADY;
- To support the modelling of buffer networks and high-speed or limited-access roads through the use of speed/flow relationships; and
- To improve, and from the user's point of view simplify, the way in which the packet method is used to load traffic onto the network.

Together they amount to a considerable increase in power and reliability obtained, thanks to improved program structure and algorithms, at little if any extra cost in computing resources. Methods for predicting the performance of the program as a function of the data set have been given, enabling the computational cost of CONTRAM studies and the resources needed to be estimated in advance.

### 14 ACKNOWLEDGEMENTS

The work described in this Report was carried out in the Traffic Safety Division (Division Head: Dr R M Kimber) of the Traffic Group of TRRL. The author is grateful to CONTRAM users for providing feedback and examples of networks.

### TABLE 2

Examples of CONTRAM 5 runs on some actual networks

<table>
<thead>
<tr>
<th>Links</th>
<th>Zones</th>
<th>Signals</th>
<th>O-ds</th>
<th>T/slices</th>
<th>Iterations</th>
<th>Memory (words)</th>
<th>Routes (words)</th>
<th>Run time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>8</td>
<td>0</td>
<td>12</td>
<td>4</td>
<td>8</td>
<td>5 824</td>
<td>355</td>
<td>109</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>4</td>
<td>8</td>
<td>9</td>
<td>9</td>
<td>15 586</td>
<td>284</td>
<td>278</td>
</tr>
<tr>
<td>58</td>
<td>13</td>
<td>4</td>
<td>99</td>
<td>9</td>
<td>5</td>
<td>23 325</td>
<td>1 476</td>
<td>392</td>
</tr>
<tr>
<td>120</td>
<td>24</td>
<td>6</td>
<td>473</td>
<td>6</td>
<td>16</td>
<td>41 847</td>
<td>18 886</td>
<td>3 293</td>
</tr>
<tr>
<td>150</td>
<td>28</td>
<td>15</td>
<td>360</td>
<td>6</td>
<td>8</td>
<td>48 594</td>
<td>4 569</td>
<td>1 449</td>
</tr>
<tr>
<td>152</td>
<td>22</td>
<td>3</td>
<td>254</td>
<td>7</td>
<td>5</td>
<td>50 097</td>
<td>3 470</td>
<td>1 449</td>
</tr>
<tr>
<td>163</td>
<td>37</td>
<td>13</td>
<td>494</td>
<td>6</td>
<td>8</td>
<td>55 652</td>
<td>1 500</td>
<td>991</td>
</tr>
<tr>
<td>166</td>
<td>23</td>
<td>3</td>
<td>254</td>
<td>7</td>
<td>8</td>
<td>54 303</td>
<td>3 298</td>
<td>1 955</td>
</tr>
<tr>
<td>173</td>
<td>11</td>
<td>4</td>
<td>121</td>
<td>12</td>
<td>8</td>
<td>68 453</td>
<td>4 720</td>
<td>4 498</td>
</tr>
<tr>
<td>184</td>
<td>37</td>
<td>14</td>
<td>380</td>
<td>11</td>
<td>8</td>
<td>83 860</td>
<td>3 219</td>
<td>5 019</td>
</tr>
<tr>
<td>190</td>
<td>29</td>
<td>7</td>
<td>328</td>
<td>10</td>
<td>8</td>
<td>75 028</td>
<td>3 562</td>
<td>5 061</td>
</tr>
<tr>
<td>232</td>
<td>50</td>
<td>17</td>
<td>813</td>
<td>6</td>
<td>8</td>
<td>82 136</td>
<td>2 644</td>
<td>2 814</td>
</tr>
<tr>
<td>928</td>
<td>46</td>
<td>15</td>
<td>800</td>
<td>9</td>
<td>4</td>
<td>100 316</td>
<td>9 365</td>
<td>9 863</td>
</tr>
<tr>
<td>411</td>
<td>38</td>
<td>23</td>
<td>643</td>
<td>7</td>
<td>5</td>
<td>131 557</td>
<td>—</td>
<td>10 313</td>
</tr>
</tbody>
</table>

Notes: a: run on Compaq 386/16. b: run on Microvax II
15 REFERENCES


EVERALL P F (1968). The effect of road and traffic conditions on fuel consumption. TRRL Laboratory Report LR 226. Transport and Road Research Laboratory, Crowthorne.


APPENDIX A

SPEED/FLOW RELATIONSHIPS

A speed/flow relationship is defined by the following data (see Figure 1):

- \( N \) = effective number of lanes making up link
- \( V_0 \) = free speed (km/h) (at zero flow)
- \( V_b \) = break speed (km/h) \( \{ \) point at which \( Q_b \) = break flow (veh/h/lane) \( \} \) slope changes
- \( V_c \) = capacity speed (km/h) \( \{ \) notional \( Q_c \) = capacity flow (veh/h/lane) \( \} \) 'capacity' of lane
- \( V_m \) = minimum speed (km/h) (see below)
If no value for $V_M$ is given then a 'safety-net' value is calculated which is the speed at which the speed/flow relationship predicts an inter-vehicle headway equal to the minimum headway $H_{min}$ supplied elsewhere in the CONTRAM data or defaulted to the value 5.75m:

$$V_M = \frac{V_B Q_c - V_c Q_b}{(Q_c - Q_b) + 1000 (V_B - V_c)/H_{min}}$$

Note: The minimum speed $V_M$ applies only to the cruise portion of travel and does not represent a lower limit on the average speed of travel including the effect of queueing.

If no corresponding relationship for heavy vehicles is given:

$$V_w = \text{free speed for heavy vehicles (km/h)} \text{--if this is not given then it is assumed to be equal to } V_0$$

$$Q_w = \text{flow at which all vehicle speeds become equal (veh/h/lane)} \text{--if this is not given then it is assumed to be equal to } Q_c$$

The magnitudes of the slopes of the two limbs of the relationship are given by:

$$S_1 = \frac{(V_0 - V_b)}{N Q_b}$$

$$S_2 = \frac{(V_b - V_c)}{N (Q_c - Q_b)}$$

The speed function is given by:

$$V(Q_E) = \max(\min[V_L - S_1 Q_E, V_B - S_2(Q_E - N Q_b)], V_M)$$

where $Q_E$ is the flow entering the upstream end of the link in the current time slice.

The effective PCU excess factors for the packet currently being assigned (packet flow $q$) and the traffic stream (total flow $Q$—which includes the current packet) are given in terms of the packet or accumulated* PCU and vehicle flows by:

$$\rho = \max(\min( [q_{PCU} - q_{veh}], 1), 0)$$

$$P = \max(\min( [Q_{PCU} - Q_{veh}], 1), 0)$$

The speed of a light vehicle $V_L$ is obtained by evaluating the speed function. If a separate relationship exists for heavy vehicles the function is re-evaluated to give the speed of a heavy vehicle $V_H$. Then the speed of the current vehicle is given by:

$$V = V_L - \rho(V_L - V_H)$$

If the free speed of heavy vehicles $V_w$ is given then:

If $Q_w > Q_c$ then $\rho' = P$

If $Q_w < Q_c$ then define a hybrid of the vehicle and traffic stream factors:

$$\rho' = \frac{P Q_w + p(Q_w-Q_c)}{Q_w}$$

$$V = \min[V_L - \rho'(V_0 - V_w), V_0 - \rho'(V_0 - V_w)]$$

where the second term in the minimisation ensures that $V$ cannot exceed the interpolated free speed of the current vehicle.

**APPENDIX B**

**GEOMETRIC DELAY**

If the junction crossing time $t_j$ only is given in the data then the geometric delay is given by:

$$t_g = t_j$$

If the junction crossing speed $V_j$ is given then let:

$$V_u = \text{cruise speed on upstream link}$$

$$V_c = \text{cruise speed on current link}$$

Then if the speeds are given in km/h the respective deceleration and acceleration rates in m/s$^2$ are given by (Semmens 1985a,b, Burrow 1987):

$$\delta = 1.06 (V_u - V_j)/V_u + .23$$

$$\alpha = 1.11 (V_c - V_j)/V_c + .02$$

and the total geometric delay is obtained by calculating the time taken while respectively decelerating and accelerating, and subtracting the time which would have been taken to cover the same distances at the respective cruise speeds, giving:

$$t_g = \frac{1}{7.2} \left[ \frac{(V_u - V_j)^2}{\delta V_u} + \frac{(V_c - V_j)^2}{\alpha V_c} \right]$$

Note: Geometric delay can only be evaluated for the upstream junction of a link, because it must be calculated at the same time as other delays affecting a vehicle during assignment, at a point where the vehicle’s path through the downstream junction is still undecided.

**APPENDIX C**

**SIGNAL COORDINATION**

Coordination takes place between two signals if:

(1) the cycle times of the upstream and downstream signal are equal or differ by a factor of 2 exactly;

(2) the offsets of the start times of the stage 1 greens of the signals relative to an arbitrary common zero are defined.
The model assumes that, in the absence of dispersion or other degrading effects, the effective green fraction met by traffic arriving at the downstream signal (D) is obtained by mapping the green waves (assumed to be uniform) leaving the upstream signal (U) onto the phasing of D taking into account the travel time between the signals, and their offsets (see Figure 2).

For each combination of upstream and downstream green stages let:

\[ c^* = \max(c_u, c_d) \] be the 'standardised' cycle time, where

\[ c_u = \text{cycle time of upstream signal U} \]
\[ c_d = \text{cycle time of downstream signal D} \]
\[ o_u = \text{offset of U} \]
\[ o_d = \text{offset of D} \]
\[ t = \text{travel time between signals excluding queuing time} \]
\[ s_u = \text{time displacement of green at U relative to start of standardised cycle*} \]
\[ s_d = \text{time displacement of green at D relative to start of standardised cycle*} \]

("*if a signal is double-cycled then each plan green generates two greens within the standardised cycle, separated by the cycle time")

Then the effective displacement between the start of the sending green at U and the start of the receiving green at D is given by:

\[ s_e = \mod(t + o_u + s_u - o_d - s_d, c^*) \]

\[ \text{Note. Green times in CONTRAM are always effective green plus amber times. The effective green times specified in the Signal Plan Data may include or exclude the lost time due to the build up and decay of saturated flow at the stop line, depending on whether a general value for the lost time has been specified (see CONTRAM Userguides).} \]

Further, let:

\[ g_u = \text{length of green phase at U} \]
\[ g_d = \text{length of green phase at D} \]

and define:

\[ r_u^* = c^* - g_u = \text{standardised upstream red} \]
\[ r_d^* = c^* - g_d = \text{standardised downstream red} \]

if \( s_u < g_d \) then let \( x = g_d - s_u \)
else if \( s_u > r_u^* \) then let \( x = s_u - r_u^* \)
else let \( x = 0 \)

Then the overlap \( v \) of the greens is calculated as follows (see Figure 3):

\[ v = \max(\min(x, g_u, g_d), g_u - r_u^*) \]

If one signal is double-cycled relative to the other then it is treated as having two identical stage sequences within the standardised cycle. Within each plan cycle each signal arm may have up to two independent green stages. Hence each arm may give up to four green phases to the movement under consideration, making a total of up to 16 combinations to be evaluated. Once this has been done the theoretical maximum effective green fraction over all combinations \( j \) is given by:

\[ \lambda_{(\text{max})} = \frac{\sum N_i j}{\sum (g_{\text{ui}})} \]

Increasing the number of green stages at signal U only affects the distribution of arriving traffic, not its quantity, so that the effect of the green stages at U is averaged. However, multiple green stages at D can increase the chance that an arriving vehicle meets green, so their effects add.

Finally, as described in the main text, the coordination efficiency of the arriving vehicle, taking into account dispersion and plan ageing, is given by:

\[ e = \frac{f k \cdot \lambda_{(\text{max})} - \lambda}{1 - \lambda} \]

where

\[ k = 1/(1 + t) = \text{dispersion factor (t in minutes)} \]
\[ f = \text{a network-wide factor \( < 1 \) to allow for the ageing of fixed-time plans.} \]
APPENDIX D

QUEUE AND DELAY MODELS

The queue models in CONTRAM 5 are based on time-dependent stochastic queueing theory as developed by Kimber and Hollis 1979, Kimber and Daly 1986, and are compatible with those used by ARCADY2, PICADY2 and OSCADY (Semmens 1985a, b, Burrow 1987). However, CONTRAM incorporates special modifications to allow for the effect of signal coordination and to enable delay to be calculated vehicle by vehicle.

Mean queue

The mean queue length $L(t)$ and utilisation $U(t)$ at time $t$ are given by:

1. for an Uncontrolled link
   
   $L(t) = L_0(L_0, q, \mu, t)$
   
   $U(t) = 1$

2. for an Give-way (priority-controlled) link
   
   $L(t) = L_R(L_0, q, \mu, t, 0, 1, 1)$
   
   $U(t) = e^{-L_R/\mu}$

3. for a Signal-controlled link
   
   $L(t) = L_R(L_0, q, \mu, t, E, 0, 1, 1) + L_P(e, \mu, t, E, g, c)$
   
   $U(t) = e^{-L_R/\mu}$

where:

- $L_0(),$ is the deterministic queue function
- $L_R(\ldots, J, K)$ is the random queue function with vehicle-in-service parameter $J$ and randomness parameter $K$ (see below)
- $L_R()$ is the mean phase queue function for signals
- $L'_R$ is the derivative of $L_R$ with respect to time: ie $dL_R/dt$

and the independent variables are defined as follows:

- $t =$ time after beginning of current time slice
- $L_0 =$ mean random queue at beginning of time slice
- $\mu =$ mean capacity in current time slice
- $q =$ traffic intensity (arrivals/capacity) in current time slice
- $E =$ mean coordination efficiency of arriving traffic stream (zero for uncoordinated or non-signal junction—see Appendix C)
- $e =$ coordination efficiency of the current arriving vehicle
- $g =$ signal effective green time
- $c =$ signal cycle time
- $J =$ vehicle-in-service parameter, which represents the effective contribution to queue length associated with any inherent delay involved in negotiating the stop line

- $K =$ randomness parameter, which reflects the statistical properties of the arrival stream and service mechanism

For a full explanation of the significance of the parameters $J, E$ and $K$ see Kimber and Hollis 1979, Kimber and Daly 1986, Kimber, Summersgill and Burrow 1986. The queue length functions are defined as follows:

**Deterministic Queue**

$L_0 = \max(L_0 + (q - 1)\mu, t, 0)$

**Random Queue**

Define transformed traffic intensity and capacity by the formulae (see main text):

$\rho^* = \frac{(1 - E)q}{(1 - E_0)}$

$\mu^* = \frac{(1 - E_0)\mu}{1 - \rho^*}$

The steady-state equilibrium queue (for $\rho<1$) is given by:

$L = J\rho^* + \frac{K\rho^{*2}}{1 - \rho^*}$

Now define the basic sheared queue function $F$ and its gradient function $G$ by:

$F(x) = \begin{cases} 
-\frac{B + (B^2 - 4A)C^2}{2A} & \text{if } A \neq 0 \\
-\frac{C}{B} & \text{if } A = 0 
\end{cases}$

$G(x) = \frac{-\mu^*(F^2 + B'F + C')}{2A + B}$

There are a number of cases depending on the values of $q$ and $L_0$:

(a) Where $q<1$ and the queue is already at the steady-state value:

- If $L_0 = \ell$ then $L_R = \ell, L'_R = 0$

(b) Where the queue is growing the origin of the queue function is shifted so that $t=0$ corresponds to the point where the queue length is $L_0$. Where the queue is decaying and its initial value is no greater than twice the steady-state

**Note:** 'J' is used here instead of the more usual '1' to avoid confusion with the numeral 1
value a similar origin shift is applied to a mirror image form of the basic queue function F(t).

(i) If \( p > 1 \), or \( p < 1 \) and \( 0 < p < 1 \) then let \( L^* = L_0 \)

(ii) If \( p < 1 \) and \( 0 < L_0 < 2 \) then let \( L^* = 2 \ell - L_0 \)

In either sub-case (i) or (ii) let

\[
L^* = \frac{L_0 - 2 \ell}{w}
\]

Then in sub-case (i) \( L_R = F(t + t_0) \)
\( L'_R = G(t + t_0) \)

in sub-case (ii) \( L_R = 2 \ell - F(t + t_0) \)
\( L'_R = G(t + t_0) \)

(c) Where the queue is decaying \( (p < 1 \) invariably) and its initial value is greater than twice the steady-state length the queue is assumed to fall linearly to that value. Once the queue length falls below \( 2 \ell \) it assumed to be described by a mirror image form of the basic queue function F(t).

If \( L_0 > 2 \ell \) then

(i) If \( K < J \) then let \( w = \frac{\mu^*(1 - \lambda)}{2 \ell} \)

(ii) If \( K = J \) then let \( w = \frac{\mu^* L_0}{L_0 + J - \ell^*} \)

And let

\[
t_c = \frac{L_0 - 2 \ell}{w}
\]

Then if \( t < t_c \) \( L_R = L_0 - wt \)
\( L'_R = -w \)

if \( t > t_c \) \( L_R = 2 \ell - F(t - t_c) \)
\( L'_R = -G(t - t_c) \)

Phase Queue

The average phase queue is given by:

\[
L_p = \lambda^* L_0 + (1 - \lambda^*) L_R
\]

where

\[
\lambda^* = e + (1 - e) \ell
\]

\[
\lambda = g/c
\]

\[
L_0 = \text{mean queue in green phase \( L_0 \)}
\]

\[
L_R = \text{mean queue in red phase \( L_R \)}
\]

Let \( q_{ep} \) be the effective value of \( q \) in the previous time slice, which may equal the actual value of \( q \) in the previous time slice or may be forced to 1 in accordance with the procedure defined below (see Burrow 1987).

There are a number of cases depending on the value of \( q \) in the current time slice and also, in some cases, on the value of \( q_{ep} \):

(i) If \( q > 1 \) then

\[
L_r = L_q = \mu^* (1 - \lambda) c/2
\]

(ii) If \( q < 1 \) and \( q_{ep} < 1 \) then

\[
L_r = \frac{\mu^* (1 - \lambda) c}{2(1 - \lambda^*)}
\]

(iii) If \( q < 1 \) but \( q_{ep} > 1 \) then the queue lengths are linear combinations of those obtained under (i) and (ii) above, the proportions depending on the approximate length of time for which a random queue in excess of the steady-state value maintains an effective saturated demand at the stop line, viz:

\[
L_r = L_q = \mu^* (1 - \lambda) c/2
\]

where \( q_{ep} = 1 \) for the next time slice

If \( t > t_d \) then

\[
L_r = \frac{\mu^* (1 - \lambda) c}{2 (1 - \lambda^*)}
\]

Delay

The delay to an individual vehicle arriving at time \( t \), when the queue length is \( L \) and the utilisation is \( U \), is the value \( d \) obtained by solving the equations:

\[
d = \sum d_i
\]

\[
L = U \sum d_i
\]

where \( d_i \) is equal to the whole or part of the duration of the ith time slice during which the vehicle remains in the queue, and \( \mu_i \) is the capacity in the ith time slice. The calculation assumes that the same value of \( U \) applies throughout, since \( U \) is the probability of a queue being present on the arrival of the current vehicle.

The equations are solved by the following iterative method: if \( t_i \) represents the start of the ith time slice, and the vehicle arrives at the queue at time \( t_i \) in the ith time slice then:

\[
d_i = \min (L/(\mu_i), t_{i+1} - t_i)
\]

\[
L_{i+1} = L - U_i d_i
\]

Then for \( n = 1 \) to last_time_slice: while \( L_{i+1} > 0 \)

\[
d_{i+n} = \min (L_{i+n}/(\mu_{i+n}), t_{i+n+1} - t_{i+n})
\]

\[
L_{i+n+1} = L_{i+n} - U_i d_{i+n}
\]

The nature of this procedure is illustrated by Figure 4. Total delay in each time slice is obtained by
summing the \( i \)th component of vehicle delay over all vehicles \( j \):

\[ D_i = \sum d_{ij} \]

**Continuity of queue length at time slice boundary**

The random queue is automatically continuous at time slice boundaries, but the phase queue is defined for a whole time slice and there will in general be a step in its calculated value at the boundary between time slices. To ensure conservation of traffic stored in the queue the change in phase queue at a time slice boundary is smoothed by calculating adjusted values of phase queue and initial random queue by the following ad hoc method:

Let
- \( L_p \) = phase queue calculated at time \( t \) in the current time slice
- \( L_0 \) = initial random queue in current time slice
- \( T \) = length of the current time slice
- \( c \) = signal cycle time
- \( L_{p0} \) = phase queue at the end of the previous time slice
- \( L_p^* \) = adjusted value of \( L_p \)
- \( L_0^* \) = adjusted value of \( L_0 \)

There are two cases depending on the relationship between \( L_p \) and \( L_{p0} \):

(a) If \( L_p > L_{p0} \) then let

\[ t_r = \max(\min(\frac{L_p - L_{p0}}{c}, T), c) \]

(i) If \( t < t_r \), then

\[ L_p^* = L_{p0} + \frac{(L_p - L_{p0})t}{t_r} \]

(ii) If \( t \geq t_r \), then

\[ L_p^* = L_p \]

in both cases (i) and (ii)

\[ L_0^* = L_0 \]

(b) If \( L_p \leq L_{p0} \) then

\[ L_p^* = L_p \]

\[ L_0^* = L_0 + L_{p0} - L_p \]

**Estimated number of stops**

This is an estimate of the probability that a vehicle suffers delay at a junction, which can be used as a qualitative indicator of the junction’s performance. It can also enter into generalised cost if an additional (eg subjective) cost is ascribed to stopping (see main text and Appendix E). Because there is some ambiguity in relation to priority junctions it is assumed that the vehicle’s probability of stopping is equal to the formal probability that a queue is present. However, if an absolute stop is specified in the Extra Link Data (geometric delay data) for the vehicle’s movement through the junction then the vehicle’s probability of stopping is set to 1.

The mean proportion of stopped vehicles in a traffic stream is given by summing individual stopping probabilities of individual packets over packet flows \( q_i \):

\[ S = \frac{\sum q_i s_i q_i}{\sum q_i} \]

where the probability of an individual vehicle (packet) being stopped is given as follows:

At an uncontrolled junction:

if \( L = 0 \) then \( s = 0 \)
if \( L > 0 \) then \( s = 1 \)

At a give-way junction:

\[ s = U \]

At a signal junction:

\[ s = 1 - \lambda U (1 - U) \]

In any case, if it is specified in the run data that the particular movement made by the vehicle through the junction involves a complete stop:

\[ s = 1 \]

**APPENDIX E**

**GENERALISED COST FUNCTION**

The generalised cost of travel along a link is given by:

\[ C = aD + bT + cV^2D + b^*T_u + uP_s + k_u H + k_d J_d + (k_p J_p + k_w J_w)l_s \]
where the coefficients are:

- \( a \) = cost per unit distance (entered as cost/100 km)
- \( b \) = cost per unit time (entered as cost/hour)
- \( c \) = cost per unit distance-speed\(^2\) (entered as cost/((100 km)/h\(^2\))
- \( b^* \) = additional cost per unit queueing time (entered as cost/hour)
- \( u \) = cost of one unit of toll
- \( k_H \) = cost per stop
- \( k_d \) = cost per deterrent movement
- \( k_g \) = cost per give-way junction
- \( k_s \) = cost per signal junction

and the variables of parameters are:

- \( D \) = link length
- \( T \) = total link travel time
- \( V \) = average link travel speed
- \( T_q \) = queueing delay on link
- \( P_a \) = number of units of toll attached to link
- \( H \) = probability that vehicle is stopped at junction
- \( J_d \) = 1 if deterrent movement at upstream junction, 0 otherwise
- \( J_g \) = 1 if junction is a give-way (priority-controlled), 0 otherwise
- \( J_s \) = 1 if junction is signal-controlled, 0 otherwise
- \( I_a \) = ‘incidence factor’ attached to link, which modulates the cost of negotiating the junction.

Each set of cost function coefficients is input in a single data card, and a given function can be defined to apply to all links, vehicle classes and time slices or to a subset of these as required. The parameters \( P_a \) and \( I_a \) are entered in Extra Link Data Cards. The units of cost are usually a currency, but for assignment purposes the units of perceived cost can be arbitrary, provided the same units are used throughout the network for each vehicle class, since route choice depends only on relative cost. For resource cost calculations it is necessary to use the same units throughout the network and for all vehicle classes.

Note: The actual method of evaluation of generalised cost in CONTRAM is complicated by the need to be able to reproduce costs precisely on different computers in order to ensure that identical sequences of packet routes and timings are generated. This precludes the use of floating-point arithmetic, requiring instead a system of integer multipliers and divisors carefully chosen to avoid arithmetic overflow over the expected range of the data. The effective precision of the calculations is about one part in 10 000.

APPENDIX F

FUEL CONSUMPTION MODEL

The data for the fuel consumption model consist of six coefficients for each vehicle class:

- \( A \) = distance-dependent consumption (mL/m)
- \( B \) = idling consumption (mL/s)
- \( C \) = distance-and-speed-dependent consumption (mL/s\(^2\)/m\(^3\))
- \( M \) = vehicle mass (tonnes)
- \( e_1 \) = energy efficiency (mL/KJ)
- \( e_2 \) = energy/acceleration efficiency (mL/s\(^2\)/KJ/m)

and a number of variables related to the movement of the vehicle:

- \( D \) = distance travelled (m)
- \( T \) = total time spent (s)
- \( V \) = mean cruise speed (m/s)
- \( T_q \) = queuing time (s)
- \( L_q \) = number of vehicles in queue when joined
- \( G \) = gradient of road section (fraction)
- \( V_u \) = upstream junction crossing speed (m/s)
- \( V_d \) = downstream junction crossing speed (m/s)
- \( S_u \) = proportion of vehicles stopped at upstream junction
- \( S_d \) = proportion of vehicles stopped at downstream junction

Note 1: The gradient of a road \( G \) is not defined in the CONTRAM data so its value is assumed to be zero in the fuel calculation.

Note 2: If junction crossing speed is not given in the data for the run then it is assumed to be the mean of the upstream and downstream cruise speeds (ignoring queues).

The following coefficient, when multiplied by vehicle mass, has been found to represent to good approximation the effect of terms in the ARRB model included to account for speed fluctuations at cruise speeds of 10 km/h or greater (Bowyer et al 1985, pp 40–41):

\[
f = 0.64 e_1 + 1.08 e_2 \text{ (mL/s/tonne)}
\]

Total fuel consumed is made up of a component due to travelling and an excess component due to queueing:

\[
fi = f_1 + f_2
\]

Fuel consumed while travelling

Total fuel consumed while travelling is given by the sum of the consumption while at cruise speed (which is proportional to link length) and the excess ‘geometric’ fuel consumption due to speed changes at the upstream and downstream junctions:

\[
f_1 = \phi_c D + f_1 + f_d
\]

The fuel consumed per metre during cruise is given by:
\[
\phi_c = \max(A + (B + fM)/V + CV^2 + 9.81 e_1 GM, 0)
\]

The excess geometric fuel consumption at the upstream and downstream junctions respectively is:

\[
\begin{align*}
\hat{\phi}_u &= S_u X(G, \phi_c, 0, V_u, V_d) + (1 - S_u) X(G, \phi_c, V_u, V_d) \\
\hat{\phi}_d &= S_d X(G, \phi_c, V_c, 0) + (1 - S_d) X(G, \phi_c, V_c, V_d)
\end{align*}
\]

where the function \(X(G, \phi_c, V_1, V_2)\) is defined as follows:

If \(V_1 = V_2\) then

\[X = 0\]

If \(V_1 < V_2\) then

\[
X = k_1(V_2 - V_1) + (k_2 + k_3)(V_2^2 - V_1^2) + k_4(V_2^4 - V_1^4) - \phi_c D_a
\]

where

\[
\begin{align*}
k_1 &= B/\alpha \\
k_2 &= \frac{1}{2}(A/\alpha + 9.81 e_1 GM/\alpha) \\
k_3 &= \frac{1}{2}(e_1 M + e_2 Ma) \\
k_4 &= \frac{1}{4} C/\alpha
\end{align*}
\]

The function \(X\) above combines the definite integral of Everall's formula, equation (9.1), between lower and upper speed values \(V_1\) and \(V_2\), representing purely speed-dependent consumption (coefficients \(k_1\), \(k_2\) and \(k_3\)) with terms representing additional consumption due specifically to acceleration-dependent effects (coefficient \(k_4\)).

If \(V_1 > V_2\) then

\[X = k_1(V_1 - V_2) - \phi_c D_\delta\]

where

\[
\begin{align*}
\delta &= 1.06(V_1 - V_2)/V_1 + .23 \\
D_\delta &= \frac{1}{2}(V_1 - V_2)^2/\delta \\
k_1 &= B/\delta
\end{align*}
\]

The expressions \(\alpha\) and \(\delta\) are the acceleration and deceleration rates and \(D_a\) and \(D_\delta\) the distances travelled under acceleration or deceleration respectively (Semmens 1985a, b). To calculate the excess fuel consumption the fuel that would have been consumed while covering these distances at cruise speed is subtracted from the fuel consumed during the manoeuvre. It is possible for the excess fuel consumed to be positive or negative—i.e., there can be a saving in fuel compared with travelling the same distance at cruise speed.

**Fuel consumed while queueing**

Fuel consumed in a queue includes a component due to idling and a component due to crawling forward (derived from Gardiner, Baker and Lucas 1986):

\[
\hat{\phi}_q = \max(BT_a + [(0.24 + .5G)B - \phi_c] D_a, 0)
\]

where the physical length of the queue through which the vehicle must move up at crawl speed is given by:

\[
D_a = \max(L_a - L_{dis}, 0)V_{min}/Q
\]

\(L_{dis}\) being the number of vehicles which can discharge in a single movement, given by:

at a give-way or uncontrolled junction:

\[L_{dis} = 1\]

at a signal-controlled junction:

\[L_{dis} = \mu c\]

where \(\mu\) is capacity, \(c\) is signal cycle time, \(V_{min}\) is the crawl speed assumed to be 3.2 m/s (11.5 km/h) based on a nominal lane saturation flow of 2000 veh/h and a headway of 5.75 m, and \(Q\) is saturation flow (in veh/s) which acts partly as a proxy for number of lanes.

**Typical coefficients of fuel consumption model**

The following coefficients are derived from unpublished work at TRRL on cars and measurements on goods vehicles by Renouf 1981, except for the energy efficiency parameters for cars which are taken from Bowyer, Akçelik and Biggs 1985.

<table>
<thead>
<tr>
<th>Vehicle Type</th>
<th>(M) (tonnes)</th>
<th>(A) (mL/m)</th>
<th>(B) (mL/s)</th>
<th>(C) (mL/s²/m³)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ford 2L car</td>
<td>1.080</td>
<td>.0240</td>
<td>.361</td>
<td>.000057</td>
</tr>
<tr>
<td>Wolseley 2.2L</td>
<td>1.204</td>
<td>.0440</td>
<td>.417</td>
<td>.000056</td>
</tr>
<tr>
<td>Petrol Van</td>
<td>&lt;2</td>
<td>.0284</td>
<td>.621</td>
<td>.000136</td>
</tr>
<tr>
<td>Diesel Van</td>
<td>&lt;2</td>
<td>.0079</td>
<td>.743</td>
<td>.000152</td>
</tr>
<tr>
<td>OGV 1</td>
<td>2 - 24</td>
<td>-.0404</td>
<td>2.272</td>
<td>.000334</td>
</tr>
<tr>
<td>OGV 2</td>
<td>&gt;24</td>
<td>-.0933</td>
<td>4.809</td>
<td>.000546</td>
</tr>
</tbody>
</table>
Table F.2
Fuel/energy efficiency of different vehicle types (*ARRB)

<table>
<thead>
<tr>
<th>Vehicle Type</th>
<th>M  (tonnes)</th>
<th>e1 (mL/KJ)</th>
<th>e2 (mL.s²/KJ.m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average car</td>
<td>1.2</td>
<td>.087</td>
<td>.0245</td>
</tr>
<tr>
<td>Petrol Goods</td>
<td>&lt;3.5</td>
<td>.095</td>
<td></td>
</tr>
<tr>
<td>Diesel Goods</td>
<td>&lt;3.5</td>
<td>.082</td>
<td></td>
</tr>
<tr>
<td>Diesel Goods</td>
<td>3.5–22</td>
<td>.074</td>
<td></td>
</tr>
<tr>
<td>Diesel Goods</td>
<td>22–36</td>
<td>.071</td>
<td></td>
</tr>
<tr>
<td>Diesel Goods</td>
<td>36–44</td>
<td>.069</td>
<td></td>
</tr>
</tbody>
</table>

APPENDIX G

PACKET GENERATOR

The target packet size for a given classified origin-destination movement acts both as a datum value and as an upper bound on the packet sizes actually generated. The target packet size is the same in each time slice (though individual packet sizes need not be) and may be entered as data or calculated automatically by the following formula:

\[ P = \frac{\sum_{i=1}^{N} 2Q_i}{\sum_{i=1}^{N} (2Q_i)} \]

where \( Q_i \) is the number of vehicles to be loaded in the ith time slice and \( F \) is a 'change-of-mind' factor entered in the o—d data as a percentage, which allows packet sizes to be generally reduced or increased according to the user's requirements (see main text).

For each classified o—d movement and each time slice packet sizes and start times are calculated by the following algorithm.

1. Let:
   \[ T = T_{i+1} - T_i = \text{length of the ith time slice} \]
   \[ q_i = \text{rate of demand in the ith time slice} \] (as entered in data)
   \[ P = \text{target packet size} \]
   \[ K = \text{int} \left( q_i T + .5 \right) = \text{number of vehicles to be loaded in ith time slice} \]

2. Procedure:

   Note: The following procedure uses integer calculations so that all results are rounded to the nearest whole number below. Rounding is an integral part of the calculation:
   \[ N = \frac{(K+P-1)P}{P} \] ! number of packets to generate

\[ Q_0 = \frac{(200K+N)(100)}{2N} \] ! mean packet size x 100
while \( N>0 \) do
\[ Q_0 = \frac{(QN+50)(100)}{100} \]
\[ Q_0 = T_{i+1} - (Q_0 + Q_1)T_i/(2K) \] ! start time of next packet
\[ N = N - 1 \]
end

APPENDIX H

ALGORITHMS

Optimum route algorithm

The route algorithm in CONTRAM 5 uses a modified form of the algorithm described by Dijkstra 1959, in the linked-based rather than node-based form developed independently at the Road Research Laboratory by Whiting and Hillier 1960. In the conventional algorithm the network links are explored in order of increasing cost (or time) from the origin. In CONTRAM 5 links are explored in order of increasing sum of the actual cost from the origin and the estimated minimum cost to the destination, the latter being calculated by using D'Esopo's algorithm (Van Vliet 1977) on the reversed network, starting at the destination (see below). The minimum costs to destination act heuristically to guide exploration towards the destination without altering the final choice of route. Reductions of 66% in algorithm computation time and 52% in overall program run time have been reported (Taylor 1989). The algorithms given below are simplified versions embodying their essential features.

Dijkstra's algorithm

This algorithm is characterised by its use of a conventional 'bubble' sort method for maintaining a list of next links to be explored in order of increasing cost. Although more sophisticated sorting methods exist (for example the 'heap' sort used by the packet generator, Section 10) their complexity is considered to outweigh any efficiency benefit for the typical list lengths used in CONTRAM. Furthermore, Dijkstra's algorithm can be terminated as soon as the destination has been reached, whereas more inherently efficient algorithms, such as D'Esopo's, do not guarantee that the minimum cost route has been found until all links have been explored, and so do not offer any benefit when searching for the route to just one destination.

1. Set the cost of the origin to zero. Set the costs of all other points in the network to a high value. Undefine backward pointers belonging to all links. Initialise next-link list to contain the origin only.
2. Remove the top link from the list. If this is the destination terminate. Otherwise make this the current link.

3. For each link branching from the current link, do:
   a. Calculate the cost of the upstream end of the branch link. Compare this with the existing cost for that link.
   b. If the new cost of the branch link is greater than the existing cost do nothing.
   c. If the new cost is smaller than the existing cost of the branch link then
      i. Replace the existing cost by the new cost and set the backward pointer belonging to the branch link to point to the current link.
      ii. Place or replace the branch link in the list in the position corresponding to its cost, maintaining the list sorted in order of increasing cost.

4. Go back to Step 2.

Note: 'Cost of a link' in the conventional Dijkstra algorithm would be the cost of travelling from the origin to the upstream end of the link. In CONTRAM it is that cost plus the estimated minimum cost from that point to the destination.

D'Esopo's Algorithm
This algorithm is highly efficient for calculating tables of minimum cost to the destination since costs from all points of the network are required and these costs are derived from minimum link costs which are independent of time and traffic volumes, viz:

$$C_a = \min_k \{a_k D_a + b_k T_{a k} + u_k P_{a k}\}$$

where

- $C_a$ = estimated minimum cost of travel on link $a$
- $D_a$ = length of link $a$
- $T_{a k}$ = cruise or free-speed time on link $a$ for vehicle class $k$ in time slice $i$
- $P_{a k}$ = units of toll allocated to vehicle class $k$ in time slice $i$ (corresponding to $a,b,u$ in Appendix E)

(Note: it is not necessary to know the exact minimum costs, but only to ensure that the estimates cannot exceed them.) Tables of costs are calculated at the start of a run for as many destinations as there is room for in memory, starting with those which attract the largest number of packets. However, the algorithm is so efficient that a saving in run time can be achieved even if a new table has to be created for every packet that is assigned.

1. Set the cost of the origin to zero. Set the costs of all other points in the network to a high value. Undefined backward pointers belonging to all links. Initialise next-link list to contain the origin only. Initialise the status of all links 'unvisited'.

2. If the next-link list is empty terminate. Otherwise remove the top link from the list, mark its status 'visited' and make it the current link.

3. For each link branching from the current link, do:
   a. Calculate the cost of the upstream end of the branch link. Compare this with the existing cost for that link.
   b. If the new cost of the branch link is greater than the existing cost do nothing.
   c. If the new cost is smaller than the existing cost of the branch link then
      i. Replace the existing cost by the new cost and set the backward pointer belonging to the branch link to point to the current link.
      ii. Check the status of the branch link.
         I. If it is not already in the list add it at the bottom.
         II. If it is already in the list but 'unvisited' do nothing.
         III. If its status is 'visited' place it at the top of the list.

4. Go back to Step 2.

Note: In CONTRAM the 'origin' for this algorithm is a network destination and the route tree is built on a reversed form of the network. The principle of the algorithm is to immediately go back and recalculate the part of the route tree branching from any link which has been 'visited'—ie been at some time the current link—and whose cost has subsequently been found to be less than was originally assumed, thus ensuring that any resulting changes downstream are effected as soon as possible.

Stabilisation of packets' routes
The packet generator always repeats the same sequence of packets with non-strictly increasing start times. Therefore packets can be identified by sequence number, running from 1 to $N$, say. Portals $P_e$ and $P_l$ ('early' and 'late' respectively) are initially set to 1 and $N$ respectively. These define the sequence of packets whose routes are allowed to change. If packet $P_e$ has not changed its route significantly since the previous iteration (the first iteration is not considered) then $P_e$ is incremented by one. Similarly if packet $P_l$ has not changed its route significantly since the previous iteration then $P_l$ is decremented by one. Packets before $P_e$ and after $P_l$ are stabilised, meaning that their routes are treated as fixed routes in subsequent iterations, although their times are recalculated in the normal way, taking into account queueing delays etc.

The criterion for a route being 'not significantly changed' is that its 'stations', or links used and time slices arrived or departed in, are the same as in the
previous iteration. This means that subsequent changes in the packet's detailed arrival times are unlikely to affect the packet's contribution to network flows, which are resolved only on time slices.

If packet \( P_e \) enters the network in time slice \( T \), where \( T \geq 1 \), then link flows in time slices 1 to \( T-1 \) are contributed to only by stabilised packets. Therefore, by definition, these flows are unlikely to change. A packet whose journey takes place entirely before the beginning of time slice \( T \) can expect to encounter the same link flows in every subsequent iteration, so both its route and its detailed times are likely to remain the same. Therefore its route and timings can be frozen. The 'melt' portal \( P_m \), initially 1 and never greater than \( P_e \), indicates the first packet which is not frozen. It is advanced one packet at a time in a similar manner to \( P_e \). When a packet is frozen all results which depend on its detailed timings are accumulated in special arrays which are preserved between iterations, and the packets route and link departure times are written to the Route File. The need to write the Route File sequentially precludes late stabilised packets from being frozen. In subsequent iterations the calculation of queues, delays and timings is by-passed, although blocking back effects (see Leonard, Gower and Taylor 1989), which are recalculated from scratch in every iteration, have to be evaluated. Since these effects are likely to be the same in all future iterations, only frozen packets which experienced blocking back in their final assignment are flagged for re-evaluation.

It is not guaranteed that time slices of stabilised packets will not change, nor that detailed timings of frozen packets would not have changed if left free, but, in view of the inherent safeguards in the methods, it is considered that the effect of such deviations on the overall assignment is outweighed by the benefit of reduced computation (see Section 11.2).

**Storage of packets’ routes**

The description of a packet's route, which is stored between iterations of the assignment, consists only of the 'stations' defined above—ie links and time slices. These contain relatively little information, in the technical sense. Each link (or origin) can branch out to at most five downstream links (or destinations), and as the origin is already determined from the packet generator each link on the route can be defined by the corresponding branch index from its predecessor, which is a number in the range from 1 to 5. The origin time slice is also already determined and each increment of time slice number can be represented by a special code inserted before the code for the link to which it applies. In fact two codes are required: 6, meaning increment time slice on arrival (at queue or stop line); and 7, meaning increment time slice on departure from link.

The codes 1 to 7, together with the stop code 0, can be stored in 'codons' of 3 bits, and a single 32-bit computer word can store 10 such codons, representing a route up to 9 links long. In most CONTRAM networks average route length falls in the range of around 4 to 20 links, so typically between half and two computer words are needed for each route. Routes may be split between adjacent words, so there are no gaps.

The route memory is organised as an integer array containing an even number of words, divided physically into two equal blocks. Logically it is treated as an infinite sequence of such blocks, forming an infinite 'tape' along which two 'heads' move, one writing each packet's route as it is assigned, the other reading the next packet's route assigned in the previous iteration. A route remains active from the time when it is written to the time when it is read. Any block containing an active route is itself active and must be represented by a physical block of data. Only two blocks can exist in memory at any one time, so if there are more than two active blocks the remainder must be placed in a temporary file.

Temporary files are opened and managed automatically by a 'virtual memory' system which is transparent to the rest of the program. At any time one file is being written and, after the first iteration, one file is being read. When all the blocks held on the current read file have been processed the file is rewound and becomes the new write file, while the old write file is rewound and becomes the new read file. The sequential nature of the logical 'tape' means that blocks are always read in the same order as they were written, so that only sequential access files are required.

**APPENDIX I**

**MEMORY REQUIREMENT AND RUN TIME**

**Memory requirement**

The following formulae estimate the total number of memory words \( W \) needed to store the data and results for a given network and demand configuration:

\[
W = W_o + W_i
\]

where \( W_o \) is the number of words used for arrays (Section 12.3 in main text) and \( W_i \) is the number of words used for storing routes (Section 11.3 and Appendix H) a guide value for which is given by:

\[
W_i = 200 \frac{L}{C}
\]

(1.1)

If all integer arrays are declared INTEGER*4:

\[
W_o = L(28.5T + H + 53) + S(13T + 40) + X(T + 11) + 21T + 16Z + 5H + 88
\]

(1.2)
If INTEGER*2 arrays are used where permissible:

\[ W_a = L(27T + H + 40) + S(6.5T + 20) + X(0.5T + 6.5) + 21T + 8Z + 2.5H + 72 \]  \hspace{1cm} (I.3)

where

- \( L \) = Number of links in network
- \( Z \) = Number of zones (origins or destinations—whichever is greater)
- \( H \) = Number of destinations with pre-calculated minimum cost trees. This can take any value from 1 to the actual number of destinations, and is set automatically by the program according to the amount of memory available.
- \( S \) = Number of signal junctions
- \( X \) = Number of classified o—d movements (origin, destination and vehicle class)
- \( T \) = Number of time slices modelled
- \( C \) = Number of codons per word = \((\text{bits per word} - 1)/3\). This is the number of 3-bit elements that can be fitted into a positive integer word. A typical value is 10 in a 32-bit word (Section 11.3).

Note: CONTRAM initially assumes that the size of the route memory is the guide value (I.1) and that \( H \) is equal to the actual number of destinations. It calculates the total memory requirement based on these assumptions and the actual data. If the total available memory is exceeded then both the assumed route memory and the assumed value of \( H \) are divided by 2, subject to minima of 128 words and 1 destination respectively, and the memory requirement is recalculated. This is repeated until either all data are accommodated or the program terminates with an 'insufficient memory' error.

**Run time**

Runs of 12 networks ranging in size from 29 to 232 links on TRRL's Cyber computers have yielded the following regression formula for reported CPU time:

\[ T_{CPU} = kPNR \text{ seconds} \]  \hspace{1cm} (12.1)

where

- \( k = 0.005 \pm 0.002 \)
- \( P \) = number of packets assigned (per iteration)
- \( N \) = number of iterations of the assignment
- \( R \) = average length of route (links)

Actual run times can differ from the above estimate by up to ±50%. Run time on a Compaq 386/25 PC is very similar. The number of packets \( P \) can be quite difficult to estimate in advance. However, some typical expressions for a number of network parameters in terms of number of links \( L \), which may be useful in a wider context, have been derived empirically as follows:

- Average number of links per route \( R = L^{0.5} \)
- Number of signal junctions \( S = 0.057L \)
- Number of zones \( Z = L^{0.634} \)
- Number of classified o—d's \( X = 0.42Z^2 \) for large networks
- \( X = 0.5Z^2 \) for networks tested

and, assuming that the total flow on any network of any size is equivalent to 2000 veh/h on each link for a period of 2 hours, and that packet sizes are generated automatically:

- Total traffic loaded \( V = 4000L/R = 4000L^{0.5} \)
- Average demand on each o—d \( \rho = 10,000L^{-0.75} \)
- Number of packets assigned \( P = 28L^{0.634} \)

These together yield the simplified expressions given in the main text and repeated here:

\[ W_{(4)} = 250 + 295L + L^{0.634} (L + 7.5L^{0.634} + 21) \text{ words} \]
\[ W_{(2)} = 235 + 270L + L^{0.634} (L + 4L^{0.634} + 10.5) \text{ words} \]
\[ T_{CPU} = 0.14NL^{1.394} \text{ seconds} \]