199 - 210

A MULTIPATH TRAFFIC ASSIGNMENT MODEL

Robert B. Dial, Alan M. Voorhees and Associates, Inc., McLean, Virginia

This paper describes the mechanics of a novel traffic assignment model that is able to assign coterminous trips to alternative routes without resorting to reiteration or path enumeration. Under a particular definition of "reasonable path," the model satisfies certain common-sense requirements. The model is a 2-pass Markov model. It calculates node and link transition probabilities in 1 examination of the network and assigns trips in a second examination, when it diverts trips entering a node to all reasonable links ending at the node. The model never explicitly examines a path, and it is not in any sense an "optimization model." It assigns trips to all reasonable paths simultaneously in such a way that the resulting effect is identical to what would have been obtained had each path been assigned trips separately under certain choice probability assumptions. Thus, compared to other multipath assignment techniques, the model is theoretically attractive and computationally very efficient. Presented also are 2 algorithms that differ in their definition of a reasonable path and in the number of times each is executed to assign all trips from a given origin node.

•IN MOST widely used traffic assignment models, all trips between a fixed origin and destination are assigned to the links constituting a single shortest connecting path. (In this article, link "time" and "length" are used interchangeably to mean neither's literal definition. Here they are names for the link disutility measure representing travel cost or impedance. We assume a link's disutility is always a positive number. A path's length is the sum of the disutility of the links that constitute it. The shortest path is one whose links sum to the smallest total disutility—whatever it may be.) This latter technique has been designated the "all-or-nothing" assignment. Because of the effects of trip volumes on travel time and the trip-maker's nondeterministic choice function on route selection, all-or-nothing assignment is known to contradict actual trip behavior; and the link-volume output of these traffic assignment models is sometimes inaccurate to the point of compromising the transportation planner's design decisions.

Many transportation planners feel that a traffic assignment model would be much more useful if it could efficiently reflect, to some degree, the nonoptimal behavior of trip-makers. The quality of the planner's decisions could be improved, and the cost of arriving at them could be decreased. A highway system, particularly when operating at near-capacity volumes, provides many alternate paths that vary slightly with respect to length between the same origin and destination. A realistic model would be a "multipath" assignment model, which would apportion trips to all of these paths in a probabilistic manner reflecting each path's relative likelihood of use.

An easy 3-step solution to this problem would be a model that would (a) relate path choice to path characteristics; (b) find all paths between a given origin and destination; and (c) using relationships found in step a, apportion trips to the paths on the basis of their characteristics. Taken literally, the preceding method has little utility. Even though such a model is fairly easy to design and implement, the large size (4,000 to 15,000 nodes) of the networks precludes this obvious solution. There are too many paths. Computers are not yet fast enough to perform the implied computation in a

Sponsored by Committee on Transportation Forecasting.

reasonable amount of time and, thus, render such a model uneconomical to use. The economics of model utilization demand that the differential value of a sophisticated multipath traffic assignment model be greater than the differential value of the output minus the differential cost of obtaining it. The cost of the output includes data preparation and computation cost. As yet, no "pure" multipath assignment model has managed to achieve a positive differential value. The computation time required to search out and evaluate alternative paths costs more than the information is worth. The proof of the statement is implied by the technique's nonuse.

The probablistic assignment model presented in the following is an attempt to circumvent the path enumeration problem. It assigns trips to all "reasonable" paths simultaneously in such a way that the resulting effect is identical to what would have been obtained had each path been assigned trips separately under certain choice probability assumptions. Compared to other multipath techniques, the model is theoretically attractive and computationally very efficient. On the theoretical side it displays some highly desirable characteristics seldom present in other techniques.

Under a particular definition of "reasonable path," theoretical appeal comes from the model satisfying the 3 following functional specifications:

1. The model gives all reasonable paths between a given origin and destination a nonzero probability of use whereas all unreasonable paths have a probability of zero;
2. All reasonable paths of equal length have an equal probability of use; and
3. When there are 2 or more reasonable paths of unequal length, the shorter has the higher probability of use.

Computational efficiency and flexibility result from the model satisfying 2 additional functional specifications:

4. The model does not explicitly enumerate the paths it loads, but all reasonable paths between a given origin and destination are loaded simultaneously; and
5. The user is able to control the path diversion probability by assigning a value to a parameter θ that affects the slope of the "diversion curve."

Computationally, the model can be called a 2-pass Markov model. It calculates node and link transition probabilities during 1 look at the network and assigns trips during a second look when it diverts trips entering a node to all reasonable (efficient) links ending at the node. In this way, it assigns trips simultaneously to an entire set of reasonable paths. The model never explicitly examines a path, and it is not in any sense an "optimization model."

In the next 2 chapters, the mechanics of 2 models are presented. Both of them satisfy the 5 preceding specifications. Both are Markov models that probabilistically divert trips from nodes to competing converging links. The 2 models differ in their definition of an efficient path and in the number of times their implementing algorithm is executed to assign all trips from a given origin node. This article is quite informal. For a lengthier and more rigorous discussion, the reader is referred to another paper by Dial (70), which presents algorithms in more detail, describes their computer implementation, and provides complete formal justification for some of the unsupported claims made in the following sections.

MODEL 1—PROBABILISTIC MULTIPATH ASSIGNMENT

This first model requires that a reasonable path between nodes o and d be an efficient path, composed only of links possessing the 2 following properties:

1. The initial node of the link is closer to the origin node o than is its final node, and
2. The final node of the link is closer to the destination node d than is its initial node.

These dual constraints restrict the set of efficient paths to those relating symmetrically to the origin and destination nodes. This duality requires that the assignment algorithm be executed once for each pair of nodes o and d. Although this is an understandable requirement, it is time consuming in execution. First, the shortest path length from each node to d must be known, and, second, there are many o-d pairs. The algorithm is described in the following section.

Algorithm 1

Preliminaries—To assign y trips between origin node o and a destination node d requires that the 4 following items be known for each node i: p(i) = the shortest path distance from o to i; q(i) = the shortest path distance from i to d; I_i = the set of all links whose initial node is node i; and F_i = the set of all links whose final node is node i. Letting link e = (i, j) have length t (i, j), we can calculate for each link e its likelihood:

$$a(e) = \begin{cases} \exp \theta\, [p(j) - p(i) - t(i,j)] & \text{if } p(i) < p(j) \text{ and } q(j) < q(i) \\ 0 & \text{if otherwise} \end{cases}$$

Having thus defined a(e), we can describe the algorithm as a 2-pass process, which need concern itself only with those links whose a(e) is not zero.

Forward Pass—By examining all nodes i in ascending sequence with respect to p(i), their distance from the origin, we can calculate for each link e in I_i its link weight:

$$w(e) = \begin{cases} a(e) & \text{if } i = o \text{ (the origin node)} \\ a(e) \sum_{e' \text{ in } F_i} w(e') & \text{if otherwise} \end{cases}$$

When the destination node d is reached, the next step is undertaken.

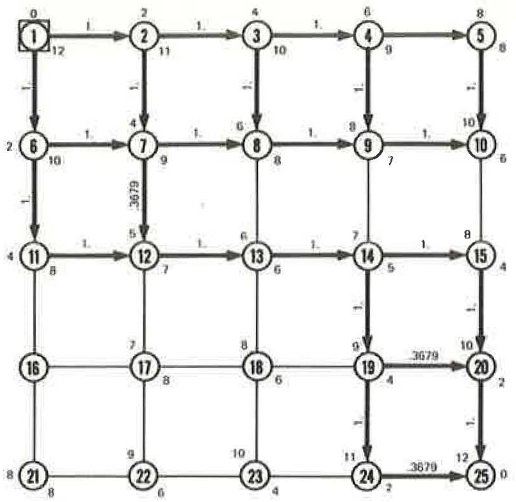
Backward Pass—Starting with the destination node d, we can examine all nodes j in descending sequence with respect to p(j). A trip volume x(e) is assigned to each link e in F_j as follows:

$$x(e) = \begin{cases} y \cdot w(e) \Big/ \sum_{e' \text{ in } F_j} w(e') & \text{if } j = d \text{ (the destination node)} \\ w(e) \sum_{e' \text{ in } I_j} x(e') \Big/ \sum_{e' \text{ in } F_j} w(e') & \text{if otherwise} \end{cases}$$

When the origin node o is reached, the assignment is complete. Notice that, in the algorithm, the funxtions x and w are defined recursively, and thus the order of their calculation must be as specified.

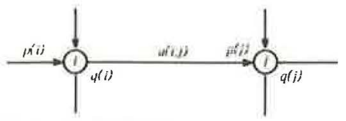
Example—In the simple grid network shown in Figure 1, we assume that all link times are 2, except on the links forming the bisecting horizontal path between node 11 and node 15, where the link times are all unity. We assign 700 cars making the trip from node 1 to node 25 as follows:

1. All pertinent preliminary data are shown in Figure 1: Above and to the left of each node is p(i), the distance the node is from node 1; below and to the right of each node is q(i), the distance the node is from node 25; the exiting link sets I_i; and the entering link sets F_i. Assume that the parameter θ is unity in the definition of a(e). Then the nature of the network allows only 2 values for the exponent of a(e): 0 or -1. Therefore, a(e) is either 0, 1, or 0.3679. The appropriate value for a(e) is posted above those links for which it is nonzero. Where a(e) is zero, no arrows or values appear on the links. Arrowless links will receive no trips because a(e) and, therefore, w(e) are zero.



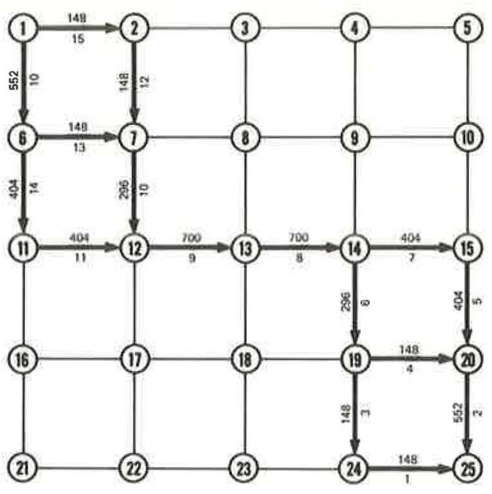
NOTES:

1. ALL ARC TIMES ARE 2, EXCEPT FOR HORIZONTAL ARCS BETWEEN NODES 11 AND 15, WHICH ARE UNITY.
2.



3. a(e) ONLY POSTED WHEN NONZERO.
4. PARAMETER $\theta = 1$.
5. $a(i,j) = \begin{cases} \exp\,[p(j) - p(i) - t(i,j)] & \text{if } p(i) < p(j) \text{ and } q(j) < q(i) \\ 0 & \text{otherwise} \end{cases}$

Figure 1. Preliminary data for Algorithm 1.



NOTE

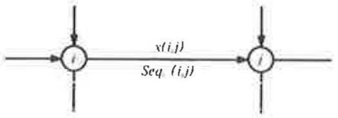
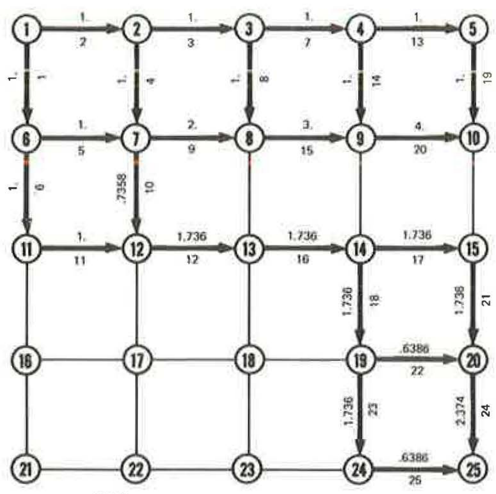


Figure 3. Results of backward pass.



NOTE:

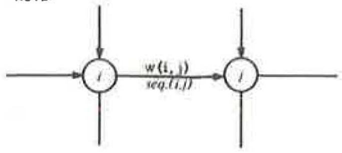


Figure 2. Results of forward pass.

2. Forward pass results are shown in Figure 2. a(e) has been replaced with w(e). The integer below each link indicates the sequence in which the links were processed. For example, link (20,25) was the 24th link processed, and w(20,25) = a(20,25) [w(15,20) + w(19,20)] = 1(1.736 + 0.6386) $\cong$ 2.374.

3. Backward pass results are shown in Figure 3. w(e) has been replaced with x(e), the link volume, and the sequenced numbers have been changed to correspond to the links' computational sequence in the backward pass. For example, link (7,12) was the 10th link processed and x(7,12) = w(7,12) · x(12,13)/[w(7,12) + w(11,12)] = 0.7358 · 700/(0.7358 + 1) $\cong$ 296.

Figure 3 shows the symmetry of the assignment around the dominating freeway axis, which reflects the symmetry of the network. There are 9 efficient paths between nodes 1 and 25. A unique shortest path has a length of 12 units. Four paths are 13 units long. Four have a length of 14. All 9 paths have been simultaneously assigned trips in a single execution of the algorithm.

Figure 4 shows the volume effectively assigned to each of these 9 paths. Each subfigure depicts an efficient path composed of the links indicated by heavy arrows. Posted beneath its graph are the length of the path and the number of trips effectively assigned to it. For example, one finds the shortest path between nodes 1 and 25. Its length is 12 and the number of trips the algorithm effectively assigned to it is equal to 232. Figure 4 shows one of 4 efficient paths between nodes 1 and 25 whose length is 13. The inferred trip volume on this path is 85. Figure 4 shows a 14-unit long path that accommodates 32 trips.

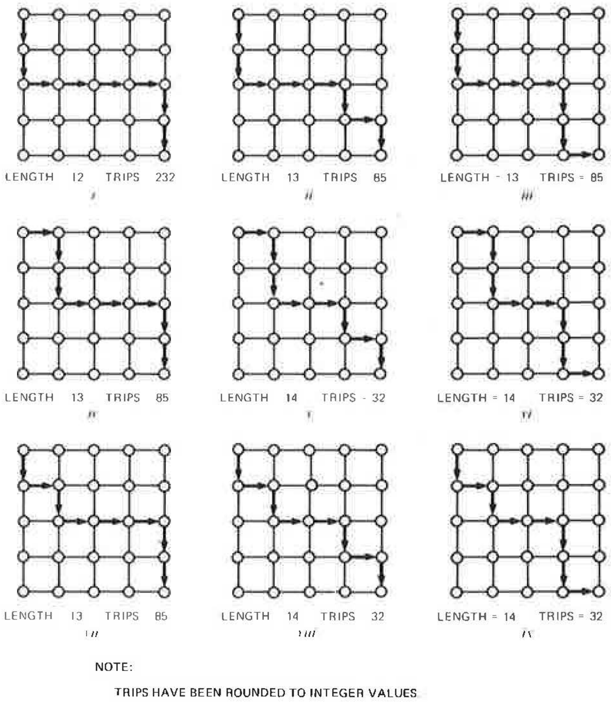


Figure 4. Efficient paths and volumes.

Justification of Algorithm 1

As mentioned previously, detailed justification of the assignment algorithm exists elsewhere. Here only a sketchy proof is given that the algorithm does in fact satisfy the 5 functional specifications given at the beginning of this paper.

In the preliminaries of the algorithm, we define the likelihood of a link e = (i, j) as

$$a(e) = \begin{cases} \exp \theta\, [p(j) - p(i) - t(i,j)] & \text{if } p(i) < p(j) \text{ and } q(j) < q(i) \\ 0 & \text{if otherwise} \end{cases} \tag{1}$$

Notice that the exponent is directly proportional to p(j) - [p(i) + t(i, j)], which is the nonpositive difference between the shortest distance to node j and the length of the shortest path to node j that uses link (i, j). Roughly speaking, a(e) is a kind of "shadow cost" of using link e.

It is assumed that the probability of using a particular (simple) path P is directly proportional to the product of the likelihood of the links in the path; that is,

$$\text{prob}(P) = k \prod_{e \text{ in } P} a(e) \tag{2}$$

Thus, prob(P) is nonzero if and only if the path P is efficient. This verifies specification 1.

By substituting Eq. 1 into Eq. 2, the probability of an efficient path can be written as

$$\text{prob}(P) = k \prod_{e = (i,j) \text{ in } P} \exp \theta\, [p(j) - p(i) - t(i,j)] \tag{3}$$

$$\text{prob}(P) = k \exp \theta \sum_{(i,j) \text{ in } P} [p(j) - p(i) - t(i,j)] \tag{4}$$

$$\text{prob}(P) = k \exp \theta \left[p(d) - \sum_{(i,j) \text{ in } P} t(i,j) \right] \tag{5}$$

The transition from Eq. 4 to Eq. 5 follows from the fact that consecutive links in a path share a node; and, for any given node $i \neq o$, d on the path P, p(i) will appear in the summation in Eq. 4 exactly twice and with opposite sign.

Because p(d) is the shortest path distance from o to d, and θ is a positive constant, Eq. 5 shows that the model satisfied specifications 2 and 3. The value of the summation in Eq. 5 is just the length of path P. Hence, the exponent in Eq. 5 is nonpositive and becomes more negative as the length of P increases. That is to say, prob(P) decreases with increasing path length.

To show that specification 5 is satisfied, we only have to show that the calculated link volumes are obtained in a manner consistent with Eq. 2, because the algorithm obviously does not enumerate paths. This is shown by proving that the algorithm diverts trips from each node according to appropriate conditional link probabilities. A conditional link probability is the probability that a trip between o and d will use a particular line e = (i, j), given that it goes through the link's final node. This probability can be formally stated as

$$\text{prob}[(i,j)|j] = \text{prob}[(i,j), j]/\text{prob}(j) \tag{6}$$

$$\text{prob}[(i,j)|j] = \text{prob}[(i,j)]/\text{prob}(j) \tag{7}$$

$$\text{prob}[(i,j)|j] = \text{prob}[(i,j)] \Big/ \sum_i \text{prob}[(i,j)] \tag{8}$$

Although it is obvious that the probability of using a link (i, j) is just

$$\text{prob}[(i,j)] = \sum_{\{P:(i,j) \text{ in } P\}} \text{prob}(P) \tag{9}$$

It is useful to write Eq. 9 in a more elaborate form, to facilitate cancellation of common factors in the numerator and denominator of Eq. 8. To this end, notice that an efficient path through link (i, j) can be partitioned into 3 sets of links: (a) P_i = {all links topologically preceding link (i, j)}; (b) {(i, j)}; and (c) P_j = {all links topologically following link (i, j)}. Now if $\mathbb{P}_i$ is the family of all P_i representing partition 1 of any efficient path from o to d, and $\mathbb{P}_j$ is the family of partition 3 of all efficient paths between o and d, then

$$\sum_{\{P:(i,j) \text{ in } P\}} \text{prob}(P) = k\, a(i,j) \left[\sum_{P \text{ in } \mathbb{P}_i} \prod_{e \text{ in } P} a(e) \right] \left[\sum_{P \text{ in } \mathbb{P}_j} \prod_{e \text{ in } P} a(e) \right] \tag{10}$$

Equation 10 follows from the fact that all efficient paths can be constructed by independently choosing a member from each of $\mathbb{P}_i$ and $\mathbb{P}_j$ and putting link (i, j) in between and that all such combinations constitute efficient paths. Substituting Eq. 10 into Eq. 8, we see that

$$\text{prob}[(i,j)|j] = \frac{k\, a(i,j) \left[\sum_{P \text{ in } \mathbb{P}_i} \prod_{e \text{ in } P} a(e) \right] \left[\sum_{P \text{ in } \mathbb{P}_j} \prod_{e \text{ in } P} a(e) \right]}{\sum_i \left\{ k\, a(i,j) \left[\sum_{P \text{ in } \mathbb{P}_i} \prod_{e \text{ in } P} a(e) \right] \left[\sum_{P \text{ in } \mathbb{P}_j} \prod_{e \text{ in } P} a(e) \right] \right\}} \tag{11}$$

$$\text{prob}\,[(i,j)|j] = \frac{k\, a(i,j)\left[\sum_{P \text{ in } \mathbb{P}_i} \prod_{e \text{ in } P} a(e)\right]\left[\sum_{P \text{ in } \mathbb{P}_j} \prod_{e \text{ in } P} a(e)\right]}{k \sum_{P \text{ in } \mathbb{P}_j} \prod_{e \text{ in } P} a(e) \left\{\sum_i \left[a(i,j) \sum_{P \text{ in } \mathbb{P}_i} \prod_{e \text{ in } P} a(e)\right]\right\}} \tag{12}$$

$$\text{prob}\,[(i,j)|j] = \frac{a(i,j) \sum_{P \text{ in } \mathbb{P}_i} \prod_{e \text{ in } P} a(e)}{\sum_i \left[a(i,j) \sum_{P \text{ in } \mathbb{P}_i} \prod_{e \text{ in } P} a(e)\right]} \tag{13}$$

Arguing by induction that the forward pass of algorithm 1 calculates link weights, we can show that w(e) in the algorithm is equal to the numerator of Eq. 13 when e = (i,j). Hence,

$$\text{prob}\,[e|j] = w(e) \Big/ \sum_{e' \text{ in } F_j} w(e') \tag{14}$$

is just a rewriting of Eq. 13. The right side of Eq. 14 is precisely the quantity the algorithm uses to divert trips from node j to link e. Using Eq. 14 and again arguing inductively, but this time in the order to the backward pass of algorithm 1, we can show that, at the time that trips of node j are diverted, the quantity

$$y(j) = \sum_{e \text{ in } I_j} x(e) \tag{15}$$

does comprise all trips from o to d that are expected to go through node j. This completes the proof that the diversion volumes are indeed those implied by the path probability defined in Eq. 2.

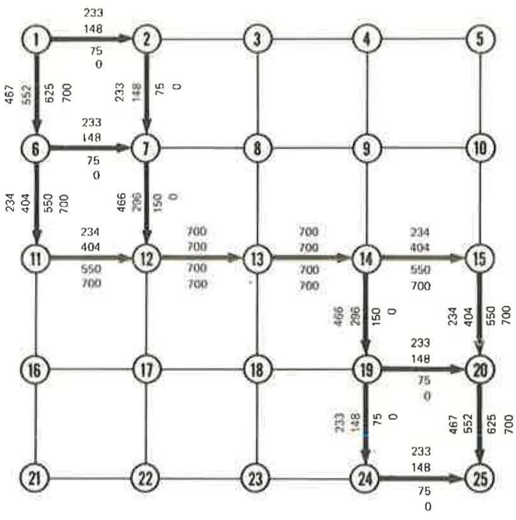
The termination of the algorithm is obvious. Each link is processed twice, at most, and there are a finite number of links in the network.

The Parameter θ

To affect diversion probabilities and thus satisfy specification 5, the user sets the value of the parameter θ appearing in the exponent of the link likelihood a(e). As shown earlier, as θ varies from zero to infinity, the probability of using a particular path, which is Δt longer than the shortest path, is directly proportional to exp $(-\theta \cdot \Delta t)$. Thus, as θ increases, the probability that a trip will use a shortest path also increases. When θ is zero, all efficient paths are considered equally likely; the topological significance of a link in an efficient path is its sole criterion for attracting trips. At the other extreme, when θ is large, i.e., 10 or larger, the effect is a multiple shortest-path assignment, which assigns trips to all and only shortest paths. This allows the network designer to perform the equivalent of an all-or-nothing assignment that appropriately considers parallel routes.

Between these 2 (useful) extremes, presumably, there is a value for θ that does the best job in duplicating the results of human behavior. This writer does not know what this value is, or even whether a unique value would suffice for all o-d pairs, trip purposes, or geographic locations. This is a good subject for future experimentation. Given route selection data, we could estimate θ directly by using numerical curve fitting techniques. Alternatively, screenline interviews, in which trip-makers crossing particular links are asked the origin, destination and purpose of their trips, could provide a target toward which an iterative calibration procedure would aim. Or finally, θ would be tinkered with as the network analyst now tinkers with link times until the assigned volumes satisfactorily duplicated the observed ground counts.

Figure 5 shows inferred link volumes by using various values of θ in the preceding example, where we assigned 700 trips between nodes 1 and 25. The 4 numbers posted on each link are, from top to bottom, the link volumes obtained with the parameter θ equal to 0, 1, 2, and 10. Notice how, as θ increases, trips are drawn to the links on the shortest path.



NOTES:

1. ALL LINK TIMES ARE 2, EXCEPT FOR HORIZONTAL LINKS BETWEEN NODES 11 AND 15, WHICH ARE UNITY.

2.

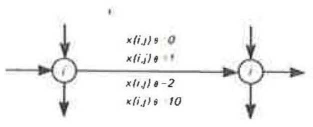


Figure 5. Arc volumes using various values of parameter θ

MODEL 2—PARALLEL PROBABILISTIC ASSIGNMENT

This second model redefines an efficient path to exclude constraint 2 above. In this alternative model, a path is efficient if all of its links satisfy constraint 1. This eliminates the need to know the shortest path distance from each node to the destination node d, and all trips originating at the origin node o to all destinations can be assigned simultaneously, in a single execution of algorithm 2. Thus, this second algorithm, a minor variation of the first, is effectively orders of magnitude more efficient than the first, but it is less discriminating in its selection of probable paths.

Efficient Path Redefined

In this alternative model, we discard one of the conditions for path efficiency and let a path be efficient if and only if all of its links have an initial node closer to the origin node than is its final node. Recall that this property was a necessary condition in the original definition of an efficient path. We now let it constitute a sufficient condition. Thus, all of the efficient paths under the original definition are contained in the set of efficient paths as presently redefined. We now have a larger number of efficient paths between a given origin and destination node pair than before. Thus, trips will generally be spread over more links than before.

Although the new definition of an efficient path yields more efficient paths, it surprisingly provides immense computational benefit. We assign to more paths, but the new definition permits us to do it a great deal faster. For example, in a network with 1,000 origins and 1,000 destinations, the number of executions of algorithm 2 would be 1,000 times fewer than the number of algorithm 1 with the old definition. Algorithm 2 would execute 1,000 times; algorithm 1 would execute 1,000,000 times. With the new definition, all trips from a given origin zone are assigned in roughly the same amount of time as was previously required for a single, widely separated origin and destination node pair. This fact becomes apparent in the following algorithm description.

The algorithm for the parallel assignment model is obtained by using the new definition of an efficient path and slightly modifying the multipath algorithm described earlier. The principal difference is that this parallel multipath assignment algorithm maintains node volumes. It is described in 3 major steps.

Algorithm 2

Preliminaries—To simultaneously assign all trips from origin node o to all destination nodes requires that the following 4 items be known for each node: y(i) = the number of trips from node o terminating at node i; p(i) = the shortest path distance from node o to node i; I_i = the set of all links starting at node i; and F_i = the set of all links ending at node i. Again, letting link e = (i, j) have length t(i, j), we can calculate the link likelihood of each link e.

$$a(e) = \begin{cases} \exp \theta\,[p(j) - p(i) - t(i,j)] & \text{if } p(i) < p(j) \\ 0 & \text{if otherwise} \end{cases}$$

(Notice that the old constraint q(j) < q(i) has been dropped from the definition of a(e), thus we eliminate the need to know q(j), the distance between j and d.)

Forward Pass—By examining all nodes i in ascending sequence with respect to p(i), their distance from the origin, we can calculate for each link e in I_i its link weight

$$w(e) = \begin{cases} a(e) & \text{if } i = o \text{ (the origin node)} \\ a(e) \sum\limits_{e' \text{ in } F_i} w(e') & \text{if otherwise} \end{cases}$$

When the destination node most distant from the origin node o is reached, the next step is undertaken (notice that this step is identical to the first step in algorithm 1).

Backward Pass—Starting with the most distant destination node, we can examine all nodes j in descending sequence with respect to p(j). For each link e in F_j, the following 2 steps are undertaken:

1. A trip volume x(e) is assigned to each link e:

$$x(e) = y(j)\, w(e) \Big/ \sum_{e' \text{ in } F_j} w(e')$$

2. The node volume at e's initial node i is increased by e's link volume:

$$y(i) \leftarrow y(i) + x(e)$$

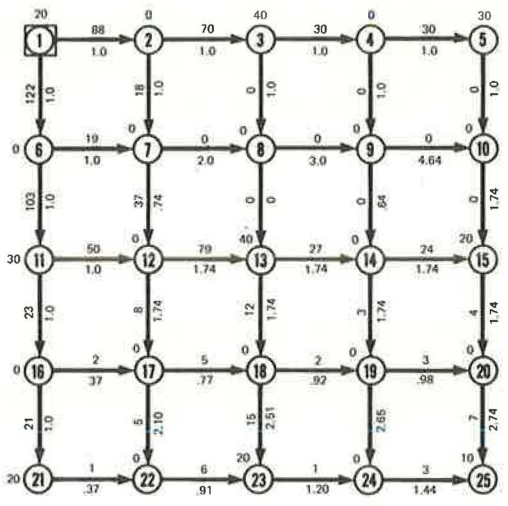
When the origin node o is reached, the assignment is complete. All trips originating at node o have been assigned. (At this time, the node volume of the origin, y(o), should equal the total trips originating at o. This constitutes a good error check in a computer implementation.)

Example—Figure 6 shows the results of a single execution of the multiterminal, multipath algorithm. The origin node is 1. The number of trips from node 1 destined for each node appears posted above the destination node. These numbers constitute one row of a travel demand matrix or "trip table." For example, the number of trips from node 1 to node 3 is 40; from node 1 to node 7, there are 0 trips; and at node 1 there are 20 intranodal trips, which will not be assigned to any links, but are included to show that they are judiciously ignored.

Posted below each line e is w(e), its link weight as calculated in the forward pass. Above each link e is x(e), its assigned link volume as calculated in the backward pass. Figure 6 shows the way all trips are spread from a given origin node through the network. The reader may attempt to duplicate these results by playing algorithm 2. Or he may prefer to refer to Dial's paper (70) that discusses and exemplifies the algorithm in much finer detail.

In a single execution of the algorithm, all trips originating at node 1 have been assigned to a total of 151 different paths. Of these 151 paths, only 57 end at nodes whose terminal volume is nonzero. Thus, 94 paths were "loaded" with zero trips. Of the 151 different paths originating at node 1, exactly 3 paths terminate at node 13. We may, for example, approximate the effective path assignments for the 40 trips from node 1 to node 13, as follows:

1. 23 trips were assigned to the shortest, 6-min path through nodes 1, 6, 11, 12, and 13;
2. 8.5 trips were assigned to the 7-min path through nodes 1, 6, 7, 12, and 13; and
3. 8.5 trips were assigned to the 7-min path through nodes 1, 2, 7, 12, and 13.



NOTES:
1. ALL LINK TIMES ARE 2, EXCEPT FOR HORIZONTAL LINKS BETWEEN NODES 11 AND 15, WHICH ARE UNITY.
2.

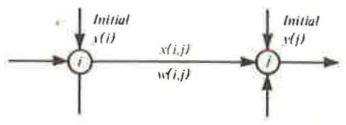


Figure 6. Parallel multipath assignment.

Table 1 gives the number of efficient paths to each destination node. General path-counting algorithms are given elsewhere; however, in this case, the preceding path volumes can be readily verified by referring to Figure 6, where the link weights are posted. Notice that all efficient paths from 1 to 13 go through node 12. Furthermore, only one of these paths, the shortest path, uses link (11, 12). Therefore, the probability of using the shortest path is identical to the conditional probability of using link (11, 12), given that node 12 is used:

$$\begin{aligned}
\text{prob [shortest path]} &= \text{prob } [(11, 12) | 12] \\
&= w(11, 12) \Big/ \sum_{e \text{ in } F_{12}} w(e) \\
&= w(11, 12)/[w(11, 12) + w(7, 12)] \\
&= 1/1.74
\end{aligned}$$

Thus, the expected number of trips from 1 to 13 that will use the shortest path is 40/1.74 = 23. The remaining 17 trips must split equally among the 2 remaining equal length, and therefore equiprobable, efficient paths. In the general case, of course, this calculation is not so simple, and reference should be made to Eq. 5 to determine individual path probabilities.

Justification of Algorithm 2

The effective difference between the faster, parallel assignment algorithm and its predecessor discussed in the preceding is in the parallel procedure's larger set of reasonable paths. For example, Figure 6 shows that, of the 10 trips from node 1 to node 25, 1 trip used paths that crossed link (23,24). Under the prior, more restrictive definition of a reasonable path, this link received no trips at all; there were no reasonable paths that used it. When we drop the explicit constraint that all nodes in a path must progress closer to the destination, we allow greater divergence of a path as it approaches its destination. Perhaps the tremendous computational advantage of the parallel multipath algorithm could be outweighed by an undesirable admission of too

TABLE 1

TRAVEL DEMAND AND EFFICIENT PATHS ORIGINATING AT NODE 1 AND TERMINATING AT NODE i

Destination Node i	Travel Demand[a]	Efficient Paths
1	20	0
2	0	1
3	40	1
4	0	1
5	30	1
6	0	1
7	0	2
8	0	3
9	0	7
10	0	11
11	30	1
12	0	3
13	40	3
14	0	3
15	20	3
16	0	1
17	0	4
18	0	7
19	0	10
20	0	13
21	20	1
22	0	5
23	20	12
24	0	22
25	10	35
Total	230	151

[a]Intranodal trips.

many paths. On the other hand, the parameter θ in the calculation of link likelihood may be used to effectively restrict path diversion. Judicious use of this parameter could render the multiterminal model useful and allow the economic benefit of its tremendous computational efficiency.

The utility of this alternative model rests on the practicality of the revised definition of an efficient path and the user's valuation of computer time. Otherwise, its properties are identical to the probabilistic assignment described previously. The new definitions do not invalidate the statements proved previously. Inefficient paths still receive no trips. Among efficient paths, the shorter ones still get more trips than the longer ones. All the highly desirable properties discussed earlier still hold; only the meaning of efficient path has changed. Formal proof of these statements is contained in the justification of algorithm 1. The only needed addition to the proof is an inductive argument showing that the node volume y(i) is properly maintained and is complete before being distributed to the links in F_i.

CONCLUSION AND RECOMMENDATION

This paper has briefly described and justified the mechanics of 2 efficient multipath traffic assignment models. It is shown elsewhere that the algorithms have extended utility. They can be readily modified to perform all-or-nothing assignment or all-shortest-path assignment. They also can yield path statistics, revealing, for example, the length distribution of all competing efficient paths.

The models also appear promising in areas of "capacity-restraint" assignment and modal split. In capacity restraint, the model is an excellent tool to perform an incremental assignment, where in each iteration a fraction of the total trips are assigned and the speeds of the links are decreased to reflect their increased volumes. In modal split, it is intriguing to imagine using the model to divert trips among modes. In such a model, the network would be multimodal, and the diversion rules would appropriately reflect fare and transfer "costs" as well as preclude illegal mode changes en route. These 2 extensions are both interesting topics for further research.

As yet, the proposed model is an untested hypothesis. While laboratory experimentation using artificial networks has been encouraging, the model has not been tested by using full-scale, real-life transportation planning input. Therefore, it is recommended that such a test be undertaken to ascertain the model's utility.

While we prefer the aesthetics of the symmetric envelope of efficient paths of algorithm 1, we feel that most computers are not yet fast enough for its blanket substitution for all-or-nothing assignment in large networks. (It would, however, be quite feasible to restrict its employment to selected key origins and destinations, where it is felt that its path discrimination would be significant.) On the other hand, algorithm 2 is more efficient than some all-or-nothing model computer implementations. It would be practical to use it on a full-scale transportation network of 4,000 to 8,000 nodes. To this end, Alan M. Voorhees and Associates, Inc., under the sponsorship of the U.S. Department of Transportation, has completed a computer code for the parallel probabilistic assignment algorithm and the capacity restraint alluded to earlier.

This computer program is completely compatible with the Federal Highway Administration transportation planning programs for the IBM 360 computer. Its output include all the detailed information, e.g., turn volumes, that the planner has come to expect from such a program. With the program's input and output compatibility, the model can be put to work in a manner imposing no compromises on its user. He will have all the analytical and comparative capabilities present in the rest of the programs and, thus, possess an ideal mechanism to observe the model's performance in the field. The actual application of the model in the planner's workaday environment is, therefore, a feasible subject for further experimentation.

ACKNOWLEDGMENTS

There is a long list of individuals who have helped the writer in the development and description of these algorithms. At the top of the list are Walter Hansen, Howard Bevis, and Edgar Horwood. Support has also come from the U.S. Department of Transportation, the University of Washington, and Alan M. Voorhees and Associates, Inc.

REFERENCE

1. Dial, R. B. Probabilistic Assignment: A Multipath Traffic Assignment Model Which Obviates Path Enumeration. Univ. of Washington, Seattle, PhD dissertation 1970.