Stochastic Route Choice Modelling with an Implicit Acyclic Full Route Set

Jeroen Verstraete (KU Leuven) (FWO SB PhD Fellowship)
Chris M.J. Tampère (KU Leuven)

1. Problem with Route Sets

A route choice model predicts which routes people take to their destination. The model computes probabilities over the subset of all possible routes of a network that the traveller is likely to consider.

In traffic assignment models, two ways exist to handle the route set during equilibration. Either, the route set is determined before the first iteration and remains fixed throughout iterations. Or, alternatively, the route set changes flexibly during equilibration. An example of a widely applied algorithm with flexible route set is Dial's algorithm (Dial and Voorhees, 1971). It defines the route set implicitly by the topological order of the nodes. Each nodes receives a unique ranking and only links that connect a lower to a higher ranked node are considered. Since congestion levels alter the ranking, the topological order and hence also the route set may change during equilibration.

Both flexible and fixed route sets have their advantages and disadvantages. A flexible route set complicates convergence to equilibrium, as not only the route choice probabilities, but also the route set need to converge. A fixed route set does not have this problem. However, it is difficult to define a priori (i.e. before congestion levels at equilibrium are known) which fixed subset of routes is attractive to the traveller and whether all relevant route alternatives are included.

A fixed *full* route set containing *all* existing route alternatives avoids both problems. For example, Recursive Logit (Bell, 1995; Fosgerau, Frejinger and Karlstrom, 2013) considers an implicit full route set for the estimation of route choice models. Verstraete, Himpe, and Tampère (2017) integrated Recursive Logit as a route choice model in a static or dynamic traffic assignment. Their model calculates for each turn the probability of being chosen while traveling towards a destination. The probability of a path is then obtained as the product of probabilities of all turns along this path.

A well-known weakness of logit is that it neglects correlations between overlapping paths (Vovsha and Bekhor, 1998). As a result, it overestimates probabilities along paths that largely coincide but only differ by a small detour. Various solutions exist to handle this correlation problem, for example path-size logit (Ben-Akiva and Bierlaire, 1999). However, the path-size is a non-link-additive cost component and hence not suited for implicit route sets. Full-route-set approaches however are inevitably implicit and hence cannot use the path-size correction. This is problematic, especially since a full set includes many cycles (e.g. an extra turn around a roundabout or city block). Routes differing by just one or two cycles severely overlap (especially if the cycle is short compared to the total length). In addition, routes with cycles are also illogical from a behavioural point of view.

In this paper, we construct full route sets excluding cycles (acyclic full route set). We define a cycle as a part of a path that uses the same link twice. In other words, a cycle occurs when the same possible actions, in this context turns, can be made.

This paper presents an exact method that eliminates all cycles in a directed graph while preserving all acyclic alternatives. We do so by expanding the graph by duplicating and cutting cycles, regardless of the origin-destination patterns or route choice model considered. The resulting algorithm may be exact but is not scalable to networks of relevant practical size. Its role, however, is to serve as a starting point for developing and consequently benchmarking heuristic methods capable of eliminating cycles from practical networks but that may preserve some cycles and/or eliminate some acyclic alternatives.

2. Methodology

This section specifies how to eliminate cyclic routes while preserving all acyclic route alternatives. After a brief introduction, a simple example illustrates the concept of layers. Next, a second example illustrates how nested cycles should be handled as they make the problem more complex. The general method will be described by these two examples.

Ideally, the route choice model should consider all possible routes without a cycle. We achieve this by duplicating (if need be many times) the links of a cycle, and consequently cutting the original cycle and some turns on its duplicates. The result is an extended, acyclic graph in which all non-cyclic paths still exist. The route choice model will then work on this extended graph. Note that there is only one graph regardless of the origins and destinations and of the route choice model, a characteristic that not all heuristic models will have.

Since cycles should be prevented while preserving all other alternatives, only the turn completing the cycle shall be cut. As the turn completing the cycle depends on where one entered the cycle, we need to remember this entrance. We do so by eliminating the original cycle from the network, and duplicating it in as many ‘layers’ as there are entrances to the cycle. Each entry gives access only to its unique duplicated layer of the cycle. Once on that layer, at any of its nodes, one has three possible choices. First, one can continue on that same layer; this is possible at any turn, except for the last turn of the layer that would make the cycle complete. Second, one can leave the current layer and go ‘down’ to the original ground layer, which consists of links that do not occur on any cycle. Third and last possibility, one can also leave the current layer to go to another layer of another cycle.

We define two kinds of links: the original links on the ground level (not part of any cycle), and links on a layer (that are part of a cycle). There are three different kind of turns that form a connection between the different levels. First, there are entry turns from any layer (ground level or duplicated layer) to a duplicated layer. Second, there are exit turns from a duplicated layer to the ground level. Third and last, there are connections between the different layers of different cycles. A connection turn is a turn that is part of one cycle, but is also an entry turn to another cycle. The set of connection turns is thus a subset of the entry turns.

The first example will explain the concept of layers, while the second example shows the complexity with nested cycles and the corresponding connection turns.

A simple example

To illustrate the idea of duplicating cycles to different layers, imagine the network illustrated in Figure 1. In this network, there is one cycle (3-6-7-8) that has two entry turns (1-2 and 7-10; turn 1-2 represents the turn from link 1 to link 2).

The first step is to duplicate the links of every cycle for each entry turn from the ground level, and thereby deleting the original links. These duplicated links are indicated by a subscript, hence, link 3_2 is link 3 duplicate for someone who has entered the cycle through link 2. Likewise duplicate link 3_{10} is only reachable through link 10. The original link 3 has no meaning anymore, since it can only be travelled on a duplicate layer corresponding to one's entrance. On each layer, the last turn of the cycle is closed, which thus depends on its entrance point. E.g. on the cycle layer from link 2, the turn from link 8_2 to link 3_2 should not exist.

The next step is to connect these layers with the ground layer through entry turns. There should be two entry turns (2-3_2 ; 10-8_{10}). As there are no nested cycles in this example, the method should not check for connection turns and can straight move to the exit turns. In this example, there should be two exit turns (3_2-4 ; 3_{10}-4). The visual representation of the extended graph is given in Figure 2. Note that in the extend graph, the original cycle is gone. Many duplicated links are in this case dead ends and will thus never be used. Therefore, those links can be deleted again.

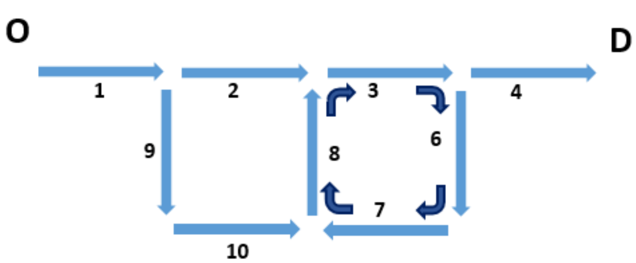

Figure 1 Example network with one cycle

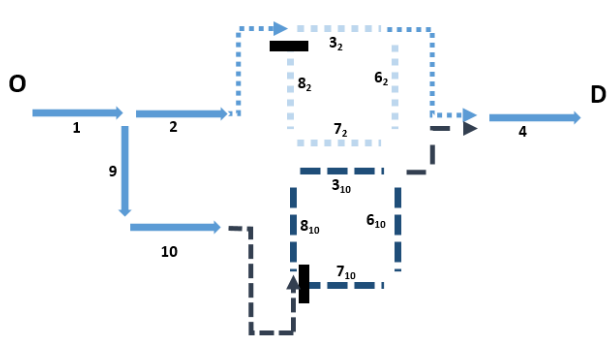

Figure 2 Extended graph of network 1

Example of Nested Cycles

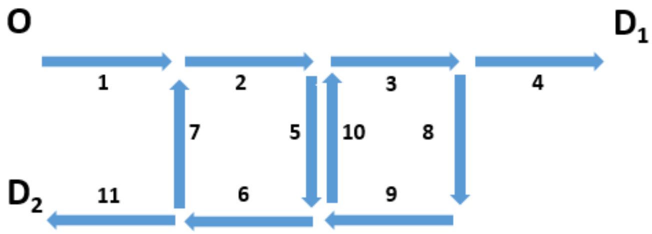

Figure 3 Network with nested cycles

Let us now introduce nested cycles. In the next example, the network, illustrated in Figure 3, contains three cycles (cycle 1: 2-5-6-7 ; cycle 2: 3-8-9-10 and cycle 3: 2-3-8-9-6-7). Before we start the method, we first look into more detail on the network. The first cycle has two entry turns (1-2 and 9-6) and one exit turn (6-11). The second cycle has one entry turn (2-3). There is, however, no entry turn from the ground layer to this cycle, as link 2 is part of a cycle. This means that cycle 2 will not be accessible directly from the ground layer, it is only accessible through another cycle's layer. The second cycle has one exit turn (to link 4). The third cycle does have three entry turns (1-2, 10-3 and 5-6), but only one entry turn from the ground level (from link 1), and two exit turns (3-4 and 6-11).

The first step is to create the layers for each entry turn from the ground level. We define link $2_{1:1}$ as link 2 on the duplicated layer 1:1, which is the layer of cycle 1 with entry turn from link 1. For this example, layer 1:1 and 3:1 are created. Next, all entry turns from the ground level are made. In this example turns 1-$2_{1:1}$ and 1-$2_{3:1}$ are the only entry turns from the ground level. The intermediate graph is plotted in Figure 4. Note that links of the same layer are connected, except for their last turn, which is indicated by the black road block. Turns between different layers have explicitly been drawn.

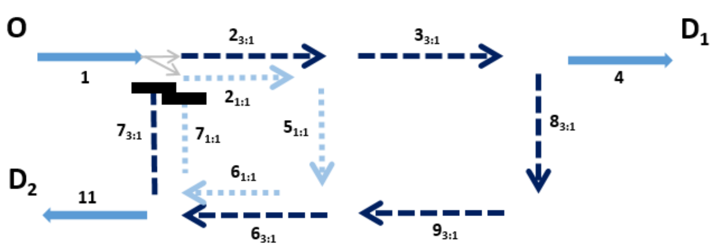

Figure 4 Intermediate step of the nested network

As there are nested cycles, there can be connection turns. The next step consist of going over all the currently created layers and searching for connection turns. A connection turns is a turn part of the cycle, but is also an entry turn to another cycle. In this example, there is no connection turn on layer 1:1. On layer 3:1 there is a connection turn: turn 2-3 is part of cycle 3 and is an entry turn to cycle 2. Layer 2:2 is created together with the connection turn $2_{3:1}$-$3_{2:2}$. Turn 9-6 is also a connection turn, but here an exception comes into play. A connection between $9_{3:1}$-$6_{1:9}$ would make cycles still a possibility, as the turn 7-2 is open on layer 1:9. Prior to reaching layer 3:1, one had also the opportunity to go to a layer 1:1 of cycle 1. Therefore, another connection is not required; actually to avoid cycles layer 3:1 should never reach back to a layer of cycle 1. In order to guarantee this, the method should keep track of all the cycles that a layer should not connect to. Once all created layers have been handled, one should redo this step for all the newly created layers until there are no connection turns unhandled. The result of this step is visualized in Figure 5.

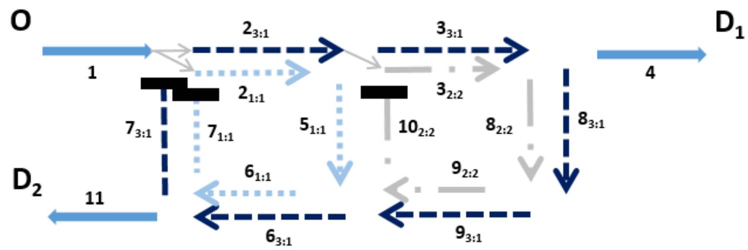

Figure 5 Intermediate step 2 of the nested network

In the last step, the exit turns to the ground level are made. In this example turns, $3_{3:1}$-4, $3_{2:2}$-4, $6_{1:1}$-11 and $6_{3:1}$-11 are exit turns. If we would add them all to the network, there would exist parallel routes on the extended graph that represent the same route on the normal network. For example, route 1-2-3-4 can be represented in the extend graph by route 1-$2_{3:1}$-$3_{3:1}$-4 and by route 1-$2_{3:1}$-$3_{2:2}$-4. These parallel routes would be seen as independent from each other according to a logit choice model, giving each route the same probability of occurring. In this way, the route choice results would not be accurate. To avoid this, the following rule is added to the method to eliminate parallel routes: if there are different layers that overlap from the beginning (of one of them) to an exit, only connection and exit turns on the layers of exactly one of the cycles of the link may be made. The criterion to guarantee picking exactly one is arbitrary, for instance the layers of the cycle with the lowest ID can always be chosen. This means that in this example first layers 3:1 and 2:2 need to be considered. These layers overlap completely from the beginning of one of them (layer 2:2) until the end. Because layer 2:2 has a lower cycle ID than layer 3:1, the exit turn chosen is $3_{2:2}$-4. When considering layers 1:1 and 3:1, we see that these layers do not overlap completely. Therefore, both exit turns are kept as this does not introduce parallel paths. The final extended graph can be seen in Figure 6. It contains all acyclic paths, but cycles are removed.

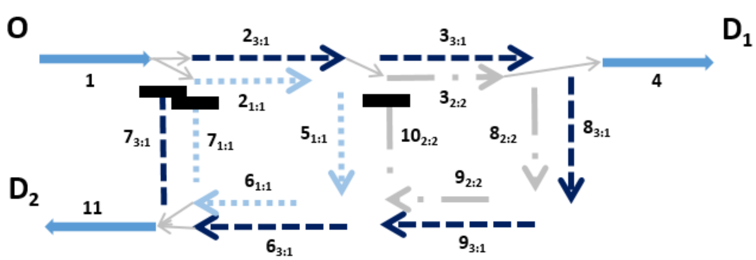

Figure 6 Extended graph of network with nested cycles. With the small arrows indicating a connection between layers.

3. Heuristic Models

By appropriate duplication of cycles and elimination of turns, an extended acyclic graph emerges that still has all acyclic alternatives. However, in this process, the number of (duplicate) links of the graph increases rapidly. In general, the number of additional links needed is $\sum_{c=1}^{C}(M_c - 1)N_c$ (with C the (usually very large) number of cycles in the graph, M_c the number of entries to cycle c and N_c the number of links of that cycle). Note that M_c is not calculated easily beforehand, as it only becomes clear during execution if a connection turn is to be considered as entry or not. This duplicating makes the resulting acyclic graph much larger than the original one and hence not practically usable in a route choice algorithm on any real-sized network. However, based on this exact method, heuristic rules and models can be derived and evaluated.

There are two different kind of heuristic models in this context. First, a heuristic model can contain all acyclic paths, but fails to eliminate all cyclic paths. Second, a heuristic model can eliminate all cyclic paths, but by doing so also eliminates some acyclic paths. Of course, there can also exist a combination of the two, a model that contains some cyclic paths and does not have all acyclic ones.

An example of heuristic models of the first kind is using the exact method but with a budget constraint on the amount of links one can duplicate. Based on some criteria, an order of the cycles is needed to determine which cycles will be eliminated and which ones not. In the full paper, we will discuss candidate criteria, ranging from very simple ones (e.g. based on length) to more complex ones. In general, cycles with high utility (low cost) that are located on a path with high flows (high probability) will influence the assignment the most and thus the criteria should guarantee the limited computational budget to be spent on these worst cases.

An example of heuristic models of the second kind that eliminate all cyclic paths, but also some acyclic paths, is to close some turns from the graph. Figure 7 shows that closing a turn can have the consequence of losing some acyclic paths. Whichever turn of the cycle is closed, one of the route alternatives will no longer be possible. In general, it is impossible to find turns to close without removing acyclic paths that may be very plausible for some OD-pairs. It is then natural to turn to heuristics that leave turns open or closed depending on the destination. We elaborate that in the full paper. It suffices here to indicate that considering different turns per destination can easily be combined with Recursive Logit, as it is a destination-based route choice model.

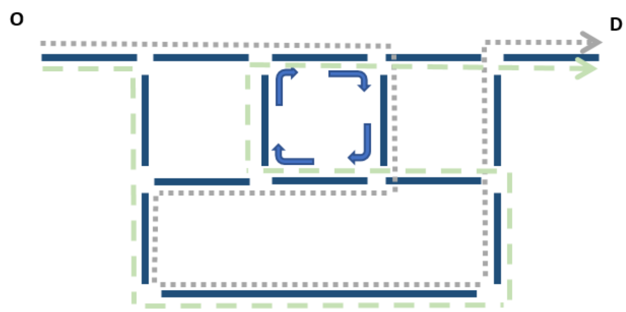

Figure 7 Example network that illustrates that closing a turn can eliminate acyclic paths.

4. Conclusion

When working with an implicit full route set with a logit based route choice model like Recursive Logit, cycles form a threat to the quality of the assignment. This is due to the fact that logit sees two routes as completely independent and is assigns probability only based on the utility of the possibilities. Two identical paths that differ only by a small cycle will have almost the same probability, hence their combined probability increases at the expense of other paths. In other words, cycles attract traffic, whereas in contrast cycles should be behaviourally undesired outputs for a route choice model.

In this paper, an exact method to eliminate cycles while preserving all acyclic routes is proposed. This full acyclic route set is constructed by duplicating each cycle into one or multiple layers. Some of these layers need to be connected to preserve all acyclic routes. By duplicating links and connecting some of them, duplicated routes can occur. To prevent this, an extra rule was added to the method that avoids this by restricting the amount of connections and exit turns. Duplicating each cycle for each of its entry points extends the graph rapidly. This renders the exact method unpractical, hence better scalable approximations are needed. This abstract suggests two kind of heuristic methods that will be elaborated further in the full paper.

Further research will investigate heuristic rules for closing turns on destination-based graphs. It is not always so simple to know the best turn of each cycle to close. Especially when closing a turn also eliminates an acyclic alternative.

References

Bell, M. G. H. (1995) ‘Alternatives to Dial’s logit assignment algorithm’, *Transportation Research Part B*, 29(4), pp. 287–295. doi: 10.1016/0191-2615(95)00005-X.

Ben-Akiva, M. and Bierlaire, M. (1999) ‘Discrete Choice Methods and their Applications to Short Term Travel Decisions’, in. Springer US, pp. 5–33. doi: 10.1007/978-1-4615-5203-1_2.

Dial, R. B. and Voorhees, A. M. (1971) ‘PROBABILISTIC MULTIPATH TRAFFIC ASSIGNMENT WHICH OBVIATES PATH ENUMERATION The problem’, *Transpn Res*.

Pergamon Press, 5, pp. 83–111.

Fosgerau, M., Frejinger, E. and Karlstrom, A. (2013) ‘A link based network route choice model with unrestricted choice set’, *Transportation Research Part B: Methodological*, 56, pp. 70–80. doi: 10.1016/j.trb.2013.07.012.

Verstraete, J., Himpe, W. and Tampère, M. J. C. (2017) ‘Stochastic User Equilibrium with Full Route Set in Dynamic Traffic Assignment’, in Cools, M. and Limbourg, S. (eds) *Proceedings of the BIVEC-GIBET Transport Research Days 2017: Towards an Autonomous and Interconnected Transport Future*, pp. 373–387.

Vovsha, P. and Bekhor, S. (1998) ‘Link-Nested Logit Model of Route Choice: Overcoming Route Overlapping Problem’, *Transportation Research Record: Journal of the Transportation Research Board*. Transportation Research Board of the National Academies , 1645(1), pp. 133–142. doi: 10.3141/1645-17.