Personal and Ubiquitous Computing
https://doi.org/10.1007/s00779-019-01350-w

ORIGINAL ARTICLE

Path complexity and bicyclist route choice set quality assessment

Thomas Koch[1] · Luk Knapen[2] · Elenna Dugundji[3]

Received: 21 August 2019 / Accepted: 26 November 2019

Abstract

Everyday route choices made by bicyclists are known to be more difficult to explain than vehicle routes, yet prediction of these choices is essential for guiding infrastructural investment in safe cycling. Building route choice sets is a difficult task. Even including detailed attributes such as the number of left turns, the number of speed bumps, distance and other route choice properties we still see that choice set quality measures suggest poor replication of observed paths. In this paper we study how the concept of *route complexity* can help generate and analyze plausible choice sets in the demand modeling process. The complexity of a given path in a graph is the minimum number of shortest paths that is required to specify that path. *Complexity* is a path attribute which could potentially be considered to be important for route choice in a similar way. The complexity was determined for a large set of observed routes and for routes in the generated choice sets for the corresponding origin-destination pairs. The respective distributions are shown to be significantly different so that the choice sets do not reflect the traveler preferences, this is in line with classical choice set quality indicators. Secondly, we investigate often used choice set quality methods and formulate measures that are less sensitive to small differences between routes that can be argued to be insignificant or irrelevant. Such difference may be partially due to inaccuracy in map-matching observations to dense urban road networks.

Keywords Route choice generation · Choice sets · Route complexity

1 Introduction

Route *choice models* play an important role in many transport applications and help to understand why people travel the way they do and to predict what they will do in the future. Route *choice set generation* is an essential part of route choice modeling in order to establish the weight of several route attributes in the decision process and to predict chosen routes in simulators. Route choice modeling for bicyclists is a topic of increasing interest as more and more people travel by bicycle for their daily commute, leading to problems with congestion in cycling lanes and at traffic lights as well as parking problems with bicycles. This in turn leads to traffic conflicts with both vehicles and pedestrians, creating unsafe situations. Understanding more about how and why cyclists travel and where they deviate from the shortest path, helps us to propose ways to improve safe cycling infrastructure and to subsequently study the effects of the modifications. Several attributes of a route are significant factors in the choice process: e.g., the number of left turns, the number of speed bumps, distance, slope, scenery etc. This study investigates the use of route complexity as an *additional* attribute. The *complexity* of a *given* (observed) path in a graph is the *minimum* number of shortest paths that is required to specify that path in the network. It can be interpreted as the (minimum) number of intermediate destinations that are connected by shortest subpaths. Note that *complexity* is a graph theoretical property and is not related to geometric properties of the route. *Complexity* is a path attribute which is considered to be important for route choice. The complexity was determined (i) for each route in a large set of routes observed by means of GPS traces and (ii) for routes in the choice sets for the origin-destination pairs corresponding to the observed routes generated by implementations of BFS_LE and DSCSG algorithms in the POSDAP tool[3]. The distributions of observed routes and

✉ Thomas Koch
koch@cwi.nl

[1] Centrum Wiskunde en Informatica, Science Park 123, 1098XG Amsterdam, The Netherlands

[2] Universiteit Hasselt, Agoralaan building D, 3590 Diepenbeek, Belgium

[3] Vrije Universiteit, De Boelelaan 1111, 1081 HV Amsterdam, The Netherlands

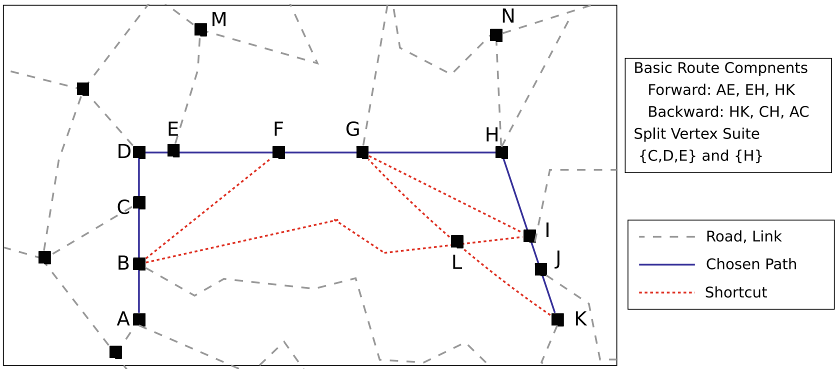


Fig. 1 The blue continuous line visiting vertices A, B, C…I, J, K is the path followed by the traveler. Paths BF, BLI, GLI, GLK, etc represent shortcuts to the chosen path. There are two sets of split vertices: {C,D,E} and {H}. Hence there are three basic path components (BPC). Sample decompositions are ((A,C), (C,H), (H,K)) and (A,E), (E,H), (H,K))

these two route choice generators seem to significantly differ. The complexity of the routes in the generated choice sets does not reflect the traveler behavior we observed in the paths chosen by cyclists. This study looks at two route choice generation techniques and how they compare to the observed routes taken by bicyclists.

The paper is organized as follows: Section 2 briefly reviews the concept of choice set generation and various choice set generators that are described in the literature. Section 3 defines the concept of *route complexity* and describes an algorithm to compute the complexity for a given route. Section 4 describes the data set of chosen bicyclist routes, the distribution for the observed complexity and the relations between route properties. Section 5 shows that the distributions for *route complexity* in generated choice sets significantly differ from the observed one.

2 Background

Choice sets play a crucial role in route choice modeling and prediction. In choice set generation, the universal set U contains all possible routes from the origin to the destination. Such universal set can be infinitely large if it is allowed to include cycles (hence not only graph theoretical *paths* but also *walks*).

In *route based* choice models, finite choice sets are established. Each route in the choice set bears a collection of attributes (distance, number of junctions, scenery etc). A discrete choice model is used to predict the traveler's choice from the attributes. Most models are based on multinomial logistic regression (MNL) and correction factors are introduced to account for correlation between overlapping routes. Model parameters and correction factors are determined using a finite choice set. Recursive logit (RL) models described by Fosgerau[4] and by Mai[9] do not require a choice set for model *estimation*. Conceptually, they are equivalent to MNL models for route choice from an infinite number of alternatives. The model described in Fosgerau[4] allows to compute the ratio of the probabilities of two routes due to the IIA (independence of irrelevant alternatives) property. RL uses link-additive attributes as opposed to route attributes and conceptually applies an MNL at each junction in order to predict the next link. Hence, it can be interpreted as a *link based* choice model.

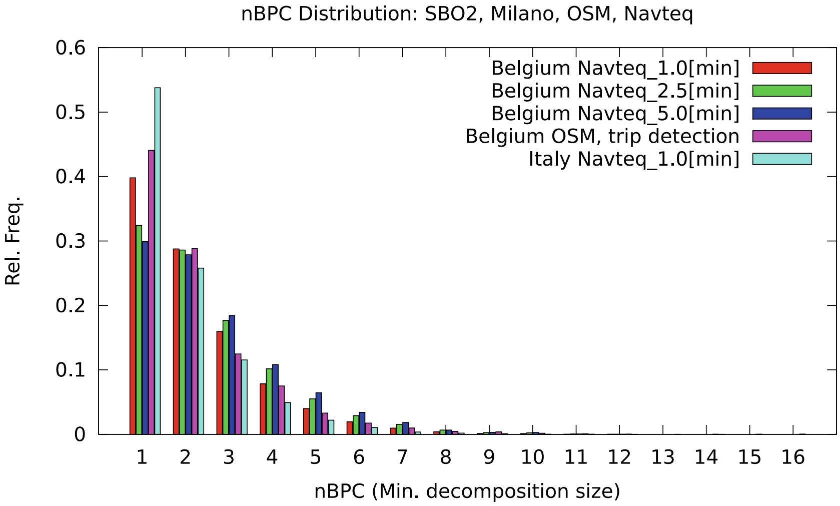
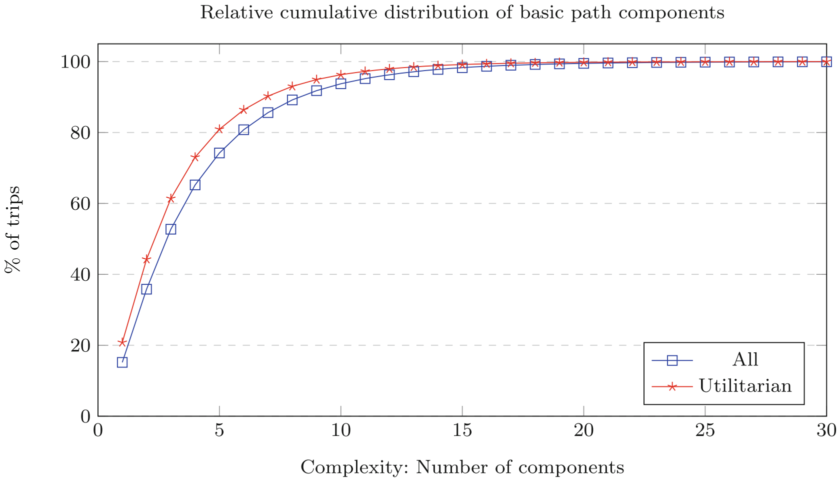


Fig. 2 Relative frequency distribution for the size of the minimum decomposition of paths derived from GPS recordings. The Belgian set consists of *person* traces. It was map-matched using different networks and gap-filling thresholds. The Italian set consists of car traces only (recorded by on-board-unit (OBU)

Fig. 3 Cumulative distribution of the complexity of paths taken by bicyclists. Blue for unfiltered, red for only utilitarian trips with r_d ¡= 1.08



However, in order to *apply* route choice models in stochastic travel simulators, candidate routes need to be generated and compared also after estimating an RL model.

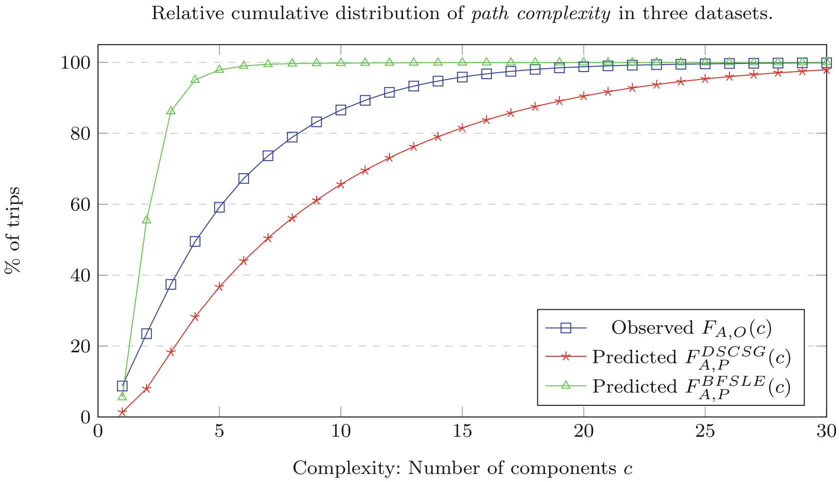
A typical choice set faced by a cyclist can include different paths with detours from the shortest path (i) to avoid dangerous situations such as busy highways, poor pavement conditions, unlighted cycle paths in the dark or unsafe neighborhoods or (ii) because of personal preference for certain areas like a park, slope, signalized junctions or a familiar path. Various choice set generators have been published.

Prato [11] provides a method called Branch and Bound, which recursively constructs paths that satisfy specific conditions i.e. *directional*, *temporal*, *similarity*, *loop and movement* (avoiding left turns). For example with the temporal constraints, a route will only be included if its travel time is not larger than the shortest time multiplied by a given factor.

Rieser [13] came up with a shortest path method, called Breadth First Search Link Elimination (BFS_LE). The BFS_LE method first computes the least cost path from origin to destination. Then links are eliminated in a particular order and a new shortest path is found. BFS refers to the fact that a tree of networks is considered and in each network a shortest path is determined using the A* algorithm. The tree is constructed by consecutively eliminating each element from the shortest path such that each recursively generated network differs in exactly one edge from the parent network in the recursion.

The Double Stochastic Generation Function method (DSCSG) described by Nielsen [10] for public transportation by Bovy [2] produces heterogeneous routes because both the cost and parameters used in the cost function for the links are drawn from a probability function. A possible difficulty of this method is the high computational cost, however Hood [6] shows DSCSG to be faster than the BFS_LE proposed by Rieser [13]. Halldorsdottir [5] shows that DSCSG has a high coverage level of replicating routes taken by bicyclists and that it performs well up to 10 kilometer. Furthermore Bovy [2] states that the method guarantees, with high probability, that attractive routes are in the choice set, while unattractive routes are not.

Fig. 4 Cumulative distributions of number of basic path components of observed bicycling routes in the Amsterdam (blue) and the number of components in paths predicted by POSDAP's implementations of Double Stochastic Generation Function (DSCSG) and Bread First Search Link Elimination (BFS_LE)



In order to generate realistic predictions, the distribution for each route attribute in the choice set needs to comply with the corresponding distribution found in observed sets. This requirement related to route *complexity* is investigated in the current paper.

3 Route complexity

The complexity of a given path in a graph is the minimum number of *Basic Path Components (BPC)* in the decomposition of the path where a basic path component is defined as either a *least cost path* or a *non-least cost edge*. A non-least cost edge is an edge e whose vertices are connected by a path having a lower cost than the cost to traverse e.

Figure 1 shows the minimum decomposition for an observed sample path p (blue continuous line) in a graph having complexity $c(p) = 3$. The example shows that multiple decompositions do exist for path p.

Knapen et al. [8] define non-cyclic trips as *utilitarian* and formulate the hypothesis that in utilitarian trips, individuals tend to construct their routes as a concatenation of a small number of basic path components. Utilitarian trips have a purpose different from the fun of driving. They are driven with the intention to perform an activity at the destination location. Knapen et al. [8] present Algorithm 3.1 to determine the complexity of a path (i.e. the minimum number of basic path components).

Algorithm 3.1 Algorithm to determine the size of the minimum decomposition of a path into basic path components.

```
Input Graph $G$, Edge costs $c$, $P = (v_0, v_1, \ldots, v_l)$
containing no non-least-cost edges
$start \leftarrow 0$
$k \leftarrow 1$                ▷ $k$ is the minimum decomposition size
while $P(v_{start}, v_l)$ is not a least cost path
        ▷ Find the first vertex $v_j$ in $P(v_{start}, v_l)$ such that
        $lc(v_{start}, v_j) < c(P(v_{start}, v_j))$
$v_j \leftarrow findFirstJoinVertex(P, v_{start})$
$k \leftarrow k + 1.$
$v_{start} \leftarrow v_{j-1}.$
return $k$
```

In algorithm 3.1 we have a graph G with positive edge costs c and a path $P = (v_0, v_1, \ldots, v_l)$ with no non-least-cost edges. Non-least cost edges are easily determined in advance and each of them constitutes a BPC. Variable *start* is the index of the first vertex in a basic path component. Variable k is the minimum decomposition size. In the *while* loop we look for the first vertex v_j for which we can find a shorter path from v_{start} to vertex v_j; such vertices are called *join* vertices because in such vertex the given path and a shortcut *join* (see [8] for details). In a *join* vertex we increment counter k by one. The predecessor of the join vertex is used to continue.

After the loop completes we can split the path at the vertex right before each *join* vertex, the vertex preceding a join vertex is called the split vertex. Using this algorithm, a splitting is found at $k - 1$ vertices, splitting our path P into k basic path components. In [8] the authors proved that the decomposition is minimal but not necessarily unique. For example by running the algorithm in reverse direction of the path we may find a different but minimal decomposition by identifying *fork* vertices.

The technique is illustrated by the example shown in Fig. 1. The algorithm determines the least cost path from A to B and finds out that this coincides with the chosen path. This is repeated for the consecutive vertices and it turns out that the paths up to and including $\langle A, B, C, D, E\rangle$ are least cost paths but $\langle A, B, C, D, E, F\rangle$ is not. Hence, F is a join vertex: the lower cost path $\langle A, B, F$ *joins* the observed path in vertex F. The same algorithm is then applied again starting from vertex E (the predecessor of F in the observed path) and vertex I is found to be a join vertex too. No other joins are found in the *forward* pass.

A *backward* pass is the executed starting from the tail vertex of the observed path. Sub-paths $\langle J, K\rangle$, $\langle I, J, K\rangle$ and $\langle H, I, J, K\rangle$ are least cost paths but $\langle G, H, I, J, K\rangle$ is not because $\langle G, I, J, K\rangle$ and $\langle G, L, K\rangle$ have a lower cost. Vertex G is a *fork* vertex. The set of vertices enclosed between a *fork* and its corresponding *join* is a called a *splitVertexSuite*. Figure 1 shows two *splitVertexSuite*s $\langle C, D, E\rangle$ and $\langle H\rangle$ respectively.

Knapen et al. [8] prove (i) that the same numbers of *join* and *fork* vertices are found, (ii) that the vertices in the *splitVertexSuite*s are potential split vertices (i.e. intermediate destinations that the traveler may have had in mind) and (iii) that each minimum decomposition consists of exactly one vertex from each *splitVertexSuite*.

Not every combination of *splitVertexSuite* members constitutes a valid decomposition. In [7] the authors provide an algorithm to enumerate all valid decompositions. E.g., the set $\{C, H\}$ may constitute a valid decomposition generating three BPC: $\{\langle A, C\rangle, \langle C, H\rangle, \langle H, K\rangle\}$.

Figure 2 is taken from [8] and shows the distribution for the complexity found in several data sets for which the majority (Belgian case) or all (Italian case) trips are car trips. This supports the hypothesis that utilitarian trips are composed of a small number of basic path components. Note that 95% of all trips had a complexity lower than 6 basic path components.

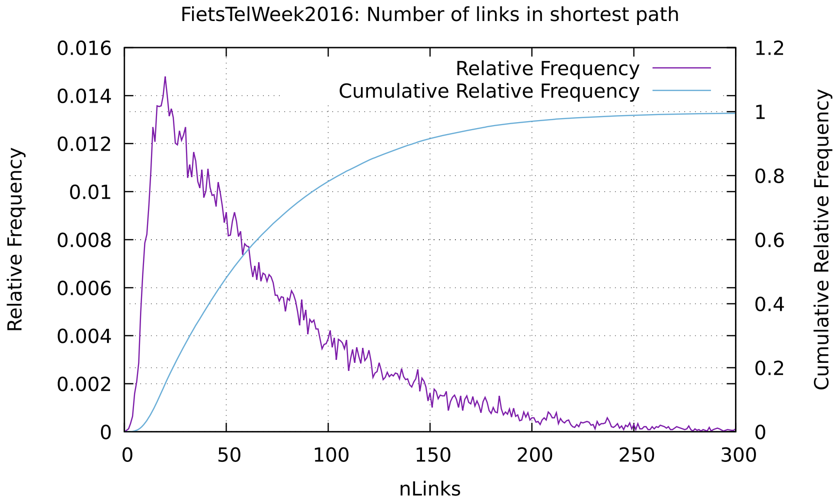


Fig. 5 Distribution for the number of links in the shortest path linking O to D in the observed routes. Only 9.4% consists of at most 16 links

4 Case study

This study considers a trip to be utilitarian if and only if it does not contain a cycle and $r_d = d_{obs}/d_{short} \leq 1.08$ where d_{obs} and d_{short} are the observed and shortest route lengths respectively (details are found in [14]). This definition is stricter than the one used in [8] in which all trips not containing any cycle have been considered.

4.1 Collecting data of bicycle movements

The Dutch 2016 FietsTelWeek (Bike Counting Week) data set ([1]) is available at http://www.bikeprint.nl/fietstelweek/. It contains 282,796 unique trips (although the corresponding infographic http://fietstelweek.nl/data/resultaten-fiets-telweek-bekend/ mentions 416,376 trips having a total distance of 1,786,147 kilometers). In order to anonymize the data, parts were stripped from the head an from the tail of the trips; the length of the stripped parts was randomly from the range [0,400] meters using a uniform distribution. Entire road network links were cut away. This process removes short trips and may explain why only 282,796 trips are found in the dataset. It was collected by 29,600 cyclists who voluntarily participated in a week-long survey to track their bicycle movements using a smart-phone app in the week of 19th of September 2016. The application ran in the background to collect the bicycle movements of all participants using the phone's GPS and acceleration sensors. The cyclists involved use their bike in a way often seen in The Netherlands to travel from and to work, supermarket, school, friends, etc. For privacy reasons, the resulting data was further anonymized in addition to the *head/tail stripping* mentioned above by the data provider before making it publicly available (i) by the removal of user information to make it impossible to trace multiple trips to a single person and (ii) by rounding of the trip departure time into one-hour bins to the nearest hour.

4.2 Route complexity in real-life GPS traces

The route complexity for the 282,796 collected by the Dutch FietsTelWeek2016 routes was computed and the distribution is shown in Fig. 3 (blue line).

For Flanders (Belgium) no detailed results for the *bike counting week* are made publicly available; hence, direct comparison is impossible. However, the distribution for the complexity of bicycle routes in The Netherlands significantly differs from the distribution for complexity found in *person traces* for Flanders shown in Fig. 2. Car mode is the prevalent mode in Flanders according to the recurrent OVG travel behavior survey . Hence most *person traces* consist of car trips and, as a consequence, most trips in the sets investigated by [8] are car trips. The difference may result :

- from behavioral difference between car drivers an bicyclists,
- from regional behavior differences and
- from parameters chosen for the map-matching process because some map-matching algorithms fill gaps by connecting positions by the shortest path.

We had no control over the map-matching process because that was performed by the *FietsTelWeek* organizer. Access to raw GPS traces is required to exclude the latter possibility.

4.3 Generating route choice sets

To compare and analyze the conformance of reality, we looked at two route choice set generation methods: *Double Stochastic Generation Function (DSCSG)* by Halldorsdottir [5] and *Breadth First Search Link Elimination (BFS_LE)* by Rieser. [13] and compared their output to the path complexity recorded in the Netherlands by the *FietsTelWeek* data-collection. For each observed trip, the origin and destination were extracted (OD-pair). We used an existing implementation of both algorithms in POSDAP [3] to generate route choice sets for each OD-pair.

Only link length (travel distance) was used in the experiment. POSDAP allows to specify a set of link specific attribute values (like scenery, separate bike lanes etc): this was not used due to lack of data.

As there is no agreement on the size N_0 of the route choice sets, we arbitrarily state that the route choice generator should produce $N_0 = 16$ routes for each origin destination pair. The POSDAP software was slightly modified in order to execute at most a given number of $M = 128$ trials to find N_0 different routes (instead of running for a given duration) so that it behaves identically on different machines. For some origin destination pairs POSDAP is not able to find as many as N_0 routes in M trials, in which case we will use all found routes. The choice sets are written to CSV files for further processing.

We computed the complexity for each route in the choice set generated by POSDAP using the algorithm specified in [8]. The distribution of the path complexity was determined for the set of *predicted* paths (i.e. the paths in the generated choice sets).

5 Discussion

5.1 Run-times

In terms of performance, BFS_LE is significantly quicker than DSCSG, producing 31,000 route-choice sets in 22 minutes for a instance with 6 parallel threads, averaging to approximately 248.3 choice set per minute per instance, on a machine with 2 *Intel Xeon CPU E5440* CPU's (4 cores/socket, 1 thread/core). DSCSG averaged to approximately 2.8 choice set per minute per instance on faster CPU's: 2 *Intel Xeon CPU E5-2660 v4* (14 cores/socket, 2 threads/core).

5.2 Route complexity in generated routes

In Fig. 4 we plotted the different complexity distributions of the routes observed and of the choice sets generated by Double Stochastic Generation Function (DSCSG) and Bread First Search Link Elimination (BFS_LE) respectively. The results are in line with what we expected based on the nature of both algorithms. First we explain the distribution of BFS_LE as follows based on the structure of network in Amsterdam. A road network is said to be *dense* with regard to a set of observed routes if the average length of network links is small relative to the developed length for the observed routes. Equivalently, a network has a high *density* if and only if commonly observed routes contain many links. In the observations for Amsterdam in the `fietstelweek2016` case, the network seems to be dense with regard to the set of shortest paths associated with each observed OD-pair. In most cases the shortest path $SP(o, d)$ for OD-pair $\langle o, d\rangle$ contains many links. This is shown in Fig. 5 for the Amsterdam case. If the required size for the choice set for $\langle o, d\rangle$ is smaller than the number of road links in $SP(o, d)$, each route generated by BFS_LE is derived by finding the shortest path $SP_m(o, d)$ in a modified network where exactly one link link belonging to $SP(o, d)$ was removed. This is easily verified in Algorithm 5.1. Line 1 specifies the recursive BFS_LE procedure. *elimLinkSetsColl* is a collection of *link sets*. Each such link set in turn is used to eliminate links from the network.

Algorithm 5.1 BFS_LE algorithm.

```
Require: $nPathsReqd, O, D, network$
  function $generate(elimLinkSetsColl, paths)$
      $elimLinkSetsCollNextLevel \leftarrow \emptyset$
      for all $elimLinkSet \in elimLinkSetsColl$ do
          $network.removeLinks(elimLinkSet)$
          $sp \leftarrow shortestPath(O, D)$
          if $sp \notin paths$ then                    ▷ New one found
              $paths \leftarrow paths \cup \{sp\}$
              if $|\ paths\ | \geq nPathsReqd$ then
                  return
              else
                  for all $link \in sp$ do
                      $es \leftarrow elimLinkSet \cup \{link\}$
                      if $es \notin elimLinkSetsCollNextLevel$ then
                      $elimLinkSetsCollNextLevel.add(es)$
          $network.addLinks(elimLinkSet)$
      $generate(elimLinkSetsCollNextLevel, paths)$
  $elimLinkSetsColl \leftarrow \emptyset$
  $paths \leftarrow \emptyset$
  GENERATE$(elimLinkSetsColl, paths)$
```

The road network *density* (as defined above) severely affects the distribution for the route complexity in the choice sets generated by BFS_LE. Figure 5 shows the first part of the (fat tail) distribution for the number of links in the shortest path for each observed OD-pair. Only 9.4% of the shortest paths contain at most 16 links. The required choice set size is 16. Hence, in 90.6% of the cases the generated routes are derived from the shortest path by eliminating only one link (belonging to the shortest path) from the network.

In contrast to BFS_LE, DSCSG uses randomness to the cost function to generate new paths and thus subsequently the number of links in the shortest paths has less influence.

Table 1 Coverage and behavioral consistency of path generation techniques

Path generation technique	Coverage (%) for overlap threshold				Behavioral consistency
	100%	90%	80%	70%	
BFS_LE	14.58	28.96	38.37	47.89	0.642
DSCSG	11.01	22.80	33.26	44.69	0.645

5.3 Choice set quality assessment

In order to assess the quality of route choice sets, a technique to evaluate the similarity of two routes is required. Three methods are discussed in this section. All methods use the same concepts of *coverage* and *consistency* defined in Section 5.3.1 but use different similarity (overlap) functions.

5.3.1 Coverage based on link matching

In order to compare our routes generated with what we found in the literature, we computed *coverage (COV)* and behavioral *consistency (CON)* as found in Prato[12], Halldorsdottir [5] and others.

In general, multiple observed routes may share a single OD-pair. Let S^{odp} denote the set of OD-pairs and $S^{obs}(p)$ denote the set of observations for $p \in S^{odp}$.

Definition 1 (coverage) The *coverage* realized by a particular choice set generator for a given set of observed routes is the fraction of observed routes for which the generated choice set contains *at least one* route for which the similarity to the observed one exceeds a given threshold.

Let $I(.)$ be the *coverage indicator function* which equals one in case its argument is true and zero otherwise. Let S^{OD} denote the set of observed OD-pairs. Let $O(r^{obs}, \rho)$ denote the fraction of overlap between the observed route r^{obs} and a route ρ in the choice set $S^{CS}(r^{obs})$ for r^{obs}. Let δ denote a threshold value. Then the coverage *COV* is defined by:

$$O(r^{obs}, \rho) = \frac{(length(overlap(r^{obs}, \rho))}{length(r^{obs})} \tag{1}$$

$$COV = \frac{1}{\mid S^{odp} \mid} \cdot \sum_{p \in S^{odp}} \left(\max_{r \in S^{obs}(p), \rho \in S^{CS}(p)} I(O(r, \rho) \geq \delta) \right) \tag{2}$$

Note that

$$\max_{\rho \in S^{CS}(r^{obs})} I(O(r^{obs}, \rho) \geq \delta) \in \{0, 1\} \tag{3}$$

$$COV \in [0, 1] \tag{4}$$

and that (3) was used to speed up the computation.

The *index of behavior consistency (CON)* considers each observed route r and compares it to *all* routes in the appropriate choice set in order to find the largest similarity value (i.e. finding the best match).

Definition 2 (consistency) The *index of behavior consistency (CON)* is the average value, computed over all observations, of the maximal similarity between the observed route and *any* of the generated routes.

The *index of behavior consistency (CON)* measure compares a path generation method with an algorithm that would replicate *all* observations. It is formally defined by

$$CON = \frac{1}{\mid S^{odp} \mid} \cdot \sum_{p \in S^{odp}} \left(\frac{1}{\mid S^{obs}(p) \mid} \cdot \sum_{r \in S^{obs}(p)} \left(\max_{\rho \in S^{CS}(p)} O(r, \rho) \right) \right) \tag{5}$$

where: CON is the consistency index and $O(r, \rho)$ is the overlap between the routes r and ρ (see (1)).

In the experiment using `fietstelweek2016` data, each OD-pair has exactly one observation. In that case the expressions for *coverage* and *consistency* reduce to

$$COV = \frac{1}{\mid S^{obs} \mid} \cdot \sum_{r \in S^{obs}} \left(\max_{\rho \in S^{CS}(r)} I(O(r, \rho) \geq \delta) \right) \tag{6}$$

$$CON = \frac{1}{\mid S^{obs} \mid} \cdot \sum_{r \in S^{obs}} \left(\max_{\rho \in S^{CS}(r)} O(r, \rho) \right) \tag{7}$$

Table 2 Statistics for the Hausdorff distance between an observation and the route with lowest Hausdorff distance in the corresponding choice set

	Min	Median	Mean	Max	Std Dev
BFS_LE	0	93.491	297.009	25372.660	622.889
DSCSG	0	66.997	232.946	25372.660	584.124

Table 3 Statistics for the minimum Fréchet distance between an observation and the routes in the corresponding choice set

	Min	Median	Mean	Max	Std Dev
BFS_LE	0	84.904	243.549	25897.228	583.795
DSCSG	0	107.04	306.603	25897.228	624.756

What we see in low values for coverage and behavioral consistency in Table 1 is similar to what we see in the route complexity distributions in Fig. 4: the predicted routes have a low conformance to reality.

5.3.2 Coverage based on geometric distance

Different *geometric distances* are proposed as alternatives for assessment of similarity by *matching complete links*. Hausdorff and Fréchet distance are evaluated as similarity measures to compare routes.

Firstly we looked at Hausdorff distance between the observed and each generated path, which verifies whether every point on the route is close to the some other point in the other route. The Hausdorff distance is the longest distance between any point p_0 on the observed route to the point p_1 on the generated route that is nearest to p_0. We use the euclidean distance $d_E(.,.)$ between points as the base metric to define the Hausdorff distance $d_H(.,.)$ between routes as follows:

$$d_H(r_0, r_1) = \max\left(\max_{p_0 \in r_0} \min_{p_1 \in r_1} d_E(p_0, p_1), \max_{p_1 \in r_1} \min_{p_0 \in r_0} d_E(p_0, p_1)\right) \tag{8}$$

Table 2 shows that in general the minimum Hausdorff distance between an observed route and any of the routes in its associated choice set is smaller for DSCSG than for BFS_LE.

On the other hand the Fréchet distance measures the similarity between curves, taking into account the location and order of points along the curves, which gives more useful information about similarity. For example if we take a route and shuffle the order, that route will still have the same Hausdorff distance however the Fréchet distance will signal the difference.

Imagine two bicyclists cycling across town from the same origin to the same destination but over two different paths, the Fréchet distance would be equal to the shortest rope that would necessary to connect both cyclists. For example if the two bicyclists would take a nearly parallel path but the first bicyclist on the left bank of a river and the other bicyclist on the right bank of a winding river, the coverage and consistency would be nearly 0 but the Fréchet distance would be limited to the width of the river.

Table 3 lists the Fréchet distances for both generation methods. It shows that BFS_LE on average is able to produce routes more similar to the observed one than DSCSG does. This is opposite to the evaluation by Hausdorff distance, meaning that routes generated with DSCSG might be Hausdorff-closer to the observed routes but when taking into account curves with Fréchet the routes generated with BFS_LE have a lower Fréchet distance.

5.3.3 Coverage based on geometry buffering

Additionally we looked at a different way to measure overlap and coverage, by allowing a given margin around one of the routes in the comparison.

The GIS (Geographical Information System) *buffer* concept is used. A buffer with radius R around a geometry g is a geometry $b(g, R)$ that contains all points at a distance from g that is less than or equal to R.

In order to evaluate the *nearness* of a route r_0 to route r_1 we define the buffer $b(r_1, R)$. We transform the concatenation of the geometries of all links in the route r_0 into a single line-string which is then subdivided into patches of a predefined length (except for the last one). A patch s_i in the reconstitution of route r_0 is counted to be *near to* r_1 if and only if $s_i \cap b(r_1, R)$ (the patch intersects the buffer). In case of intersection, the total length of the patch is assumed to be *near* r_1. Let ρ_0 denote the reconstitution of r_0. The similarity is defined as:

$$sim(r_0, r_1) = \frac{\sum\limits_{s \in \rho_o | s \cap b(r_1, R) \neq \emptyset} len(s)}{\sum\limits_{s \in \rho_0} len(s)} \tag{9}$$

When evaluating choice sets, the *observed* route is subdivided into short patches and the buffers are defined around the *generated* routes.

By dividing up the complete geometry of the observed route into small patches of d meter, we can use GIS to determine whether any point on each patch intersects with the generated path. The total distance of all patches on the observed path that intersect the generated path, divided by the total length of the observed path gives us a overlap fraction. that is used to determine an approximation for the observed route covered by the buffer. The fraction of the observed route length covered is used as a quality measure. The smaller we make d patch size, the higher accuracy we get as we define intersection as two geometries touching each other at all possible points. However this is a trade-off

Table 4 Average of $maxOverlap(U, CS(U))$ over all observed routes for two different resolutions (5 and 50 meters). A value of 1 indicates that *each* choice set contains *at least one* route that *coincides* (using a given distance threshold) with the observed route). A value of 0 indicates that for all choice sets there is no route that has *any* link near to the observed route (because we consider *maxOverlap*$(U, CS(U))$)

	50 meter patches		5 meter patches	
	BFS_LE	DSCSG	BFS_LE	DSCSG
Using 1 meter buffer	0.674	0.701	0.646	0.653
Using 5 meter buffer	0.689	0.722	0.661	0.680
Using 50 meter buffer	0.767	0.825	0.748	0.806
Using 250 meter buffer	0.846	0.894	0.834	0.884
Using 500 meter buffer	0.892	0.925	0.889	0.923
Using 1000 meter buffer	0.923	0.944	0.932	0.951
Using 2500 meter buffer	0.940	0.953	0.959	0.966

Table 5 Coverage with 1 meter buffer

	Coverage (%) for overlap threshold				
Path generation technique	100%	90%	80%	70%	Behavioral consistency
BFS_LE	11.50	29.59	39.06	48.62	0.646
DSCSG	8.71	23.69	34.52	46.19	0.654

Table 6 Coverage with 5 meter buffer

	Coverage (%) for overlap threshold				
Path generation technique	100%	90%	80%	70%	Behavioral consistency
BFS_LE	12.06	31.68	41.40	51.05	0.661
DSCSG	9.65	27.04	38.71	51.31	0.680

Table 7 Coverage with 50 meter buffer

	Coverage (%) for overlap threshold				
Path generation technique	100%	90%	80%	70%	Behavioral consistency
BFS_LE	12.77	48.62	56.57	63.79	0.748
DSCSG	13.47	54.12	63.77	72.23	0.806

Table 8 Coverage with 250 meter buffer

	Coverage (%) for overlap threshold				
Path generation technique	100%	90%	80%	70%	Behavioral consistency
BFS_LE	13.32	64.00	71.44	76.95	0.834
DSCSG	14.11	72.21	79.73	85.29	0.884

Table 9 Coverage with 500 meter buffer

	Coverage (%) for overlap threshold				
Path generation technique	100%	90%	80%	70%	Behavioral consistency
BFS_LE	13.49	73.95	82.02	86.46	0.889
DSCSG	14.27	81.65	88.10	91.45	0.923

with computation time as the work load multiplies, in this study we used a patch size of 50 and 5 meters.

By defining a *buffer* around the geometry of each observed route we can apply a *spatial tolerance* while comparing two paths (link sequences). If a path p_0 is within the buffer $b(p_1, r)$ we consider paths p_0 and p_1 to overlap (and hence to be equivalent). For example: the prediction p_{pred} follows a path parallel to the observation p_{obs} but over the cycle-way on the other side of the road.

For each observed route U we have the generated choice set $CS(U)$ that contains $N(U)$ routes. For each generated route in $CS(U)$ we determine a buffer with radius R meters and look at the total length of the patches on the observed routes that intersect with the buffer of the generated path. For each observed route and for both route choice set generation methods (BFS_LE and DSCSG) we determine the generated path with the highest overlap fraction. The corresponding maximum overlap is called $maxOverlap(U, CS(U))$ and is used as a quality measure for the choice set $CS(U)$. Table 4 lists the average (over all observations U) of the $maxOverlap(U, CS(U))$ for both choice set generation methods (BFS_LE and DSCSG). We experimented using two different patch sizes of 50 and 5 meters respectively to evaluate the effect of the resolution on the precision; as expected we see a lower maximum overlap when using a smaller patch size since we are able to look at a closer level at the distance between generated and observed path. Intersection of the observed route with a 5 meter patch S_5 implies intersection of the observed route with each 50 meter patch S_{50} that contains S_5 but the inverse is not true.

Tables 5, 6, 7, 8, 9, 10, and 11 show coverage values calculated in a similar way as in Table 1 but using different *overlap* functions. We used a large range of buffer radius values in order to verify to what extent the difference between the generated and observed routes was limited to being on the different side of the street, river, city block or neighborhood. For 1[m] and 5[m] buffers, BFS_LE has a larger coverage than DSCSG in most cases. Starting at 50[m] BFS_LE consistently reports a lower coverage than DSCSG.

Using larger *patches* measures coincidence of sub-routes less accurately than using smaller patches; furthermore, the coincidence is consistently overestimated. This is because a large patch indicates large overlap length even if only a single point of the observed route is near to the generated one.

The over-estimation grows with the *patch* length. In cases where the observed and generated routes *share* some nodes but the sub-routes connecting consecutive shared nodes are spatially not near to each other, the use of longer patches generates a larger over-estimation. Therefor, the coverage increases with the buffer radius (explaining difference between tables).

Regarding the differences within the tables, observe that the ratio $COV_{bfsle}/COV_{dscsg} < 1$ for buffer radius values up to 5[m] and $COV_{bfsle}/COV_{dscsg} > 1$ for the other cases. This reveals that the shortest routes selected by BFS_LE (which are not necessarily the *K shortest paths*) on average have less points near to the observed route than the (possibly more complicated) routes generated by DSCSG.

Table 1 is taken as the reference case. It represents the case where network links need to be replicated entirely (*overlap* is equivalent to *identification*). In the buffer based methods, *overlap* is defined in terms of *geometric neighborhood*: this leads to a relaxed requirement (identification is not a necessary condition) and hence, for each given choice set, the *coverage* and *consistency* quality values can be

Table 10 Coverage with 1000 meter buffer

	Coverage (%) for overlap threshold				
Path generation technique	100%	90%	80%	70%	Behavioral consistency
BFS_LE	13.67	82.93	90.47	93.54	0.932
DSCSG	14.52	88.38	93.48	95.75	0.951

Table 11 Coverage with 2500 meter buffer

	Coverage (%) for overlap threshold				
Path generation technique	100%	90%	80%	70%	Behavioral consistency
BFS_LE	13.71	89.51	95.87	97.84	0.959
DSCSG	14.63	92.09	96.35	98.01	0.966

expected to grow with the buffer radius used in the overlap function.

This phenomenon is explained as follows. The coverage indicator function used in (2) uses *overlap* as a similarity measure (see (1)). The *overlap* function used to generate Table 1 considers *entire links*. In the case of Table 5, the overlap function (i) considers *patches* (limited length parts of the links in the observed route) and (ii) considers a predicted route to cover a *patch* if it is *sufficiently nearby* (i.e. coincidence is not even required).

5.3.4 Distance measures and quality assessment

Several measures for dissimilarity between routes have been discussed in previous sections. These are based on euclidean distance and hence on the notion of nearness in space.

All of them can be used a quality measures for a choice set.

Assume an OD-pair $\langle O, D \rangle$, its associated choice set $C(\langle O, D \rangle)$ and a set $S^{obs}(\langle O, D \rangle)$ of observed routes for the OD-pair. The choice $C(\langle O, D \rangle)$ set is of high quality *only if* for each observed route $r^{obs} \in S^{obs}$ it contains a route r having a small dissimilarity $d_X(r, r^{obs})$. This is a *necessary but not sufficient* condition for quality.

Another requirement for high quality of a choice set is *diversity* of the routes with regard to their attributes that are assumed relevant in the choice process. Such attributes may be rated differently by different categories of travellers. Insufficient variation makes it harder to understand why a particular participant took the observed route. This variety may require routes not similar to the observed one in terms of the evaluation methods presented above (Figs. 6 and 7).

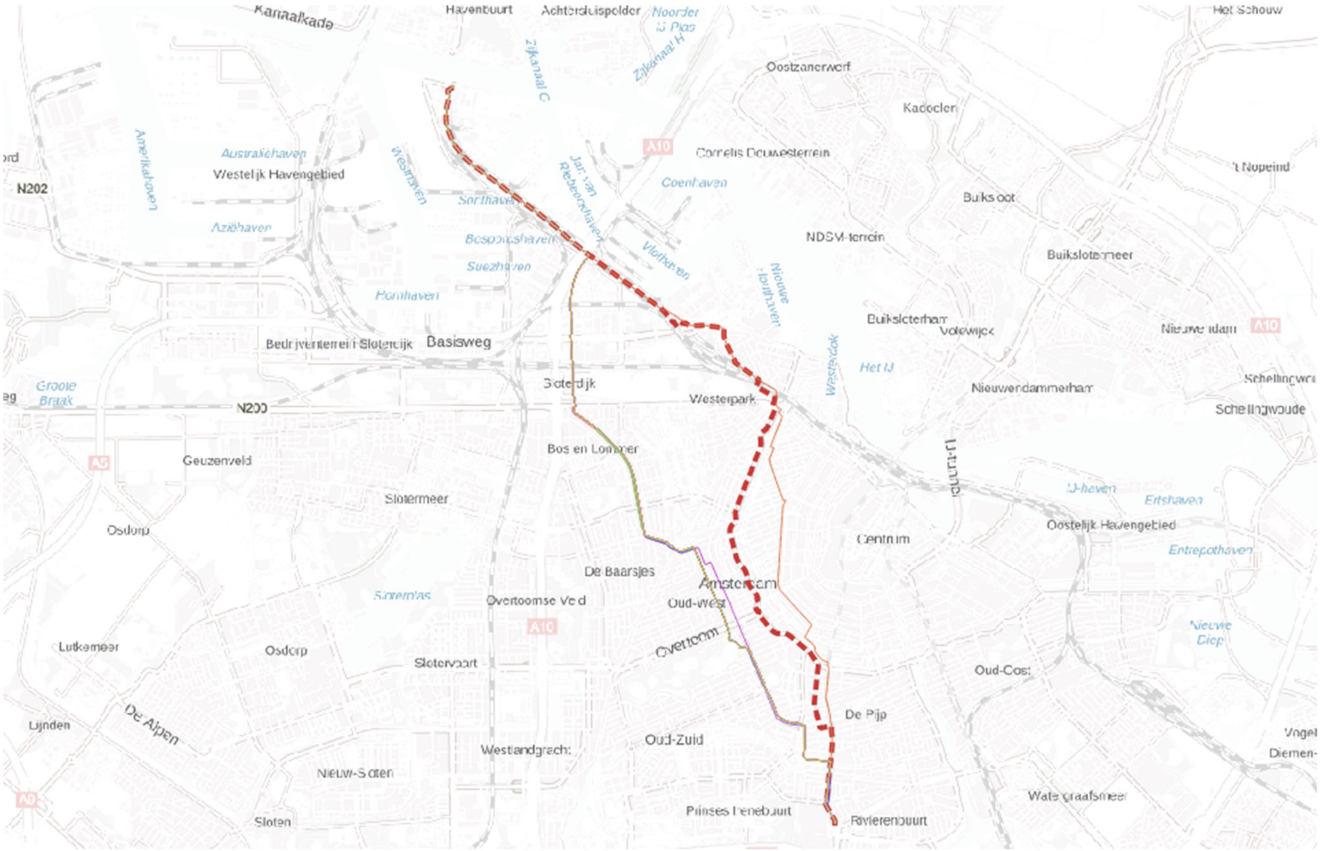


Fig. 6 Generated paths by BFS_LE, red-dashed path indicates the observed route

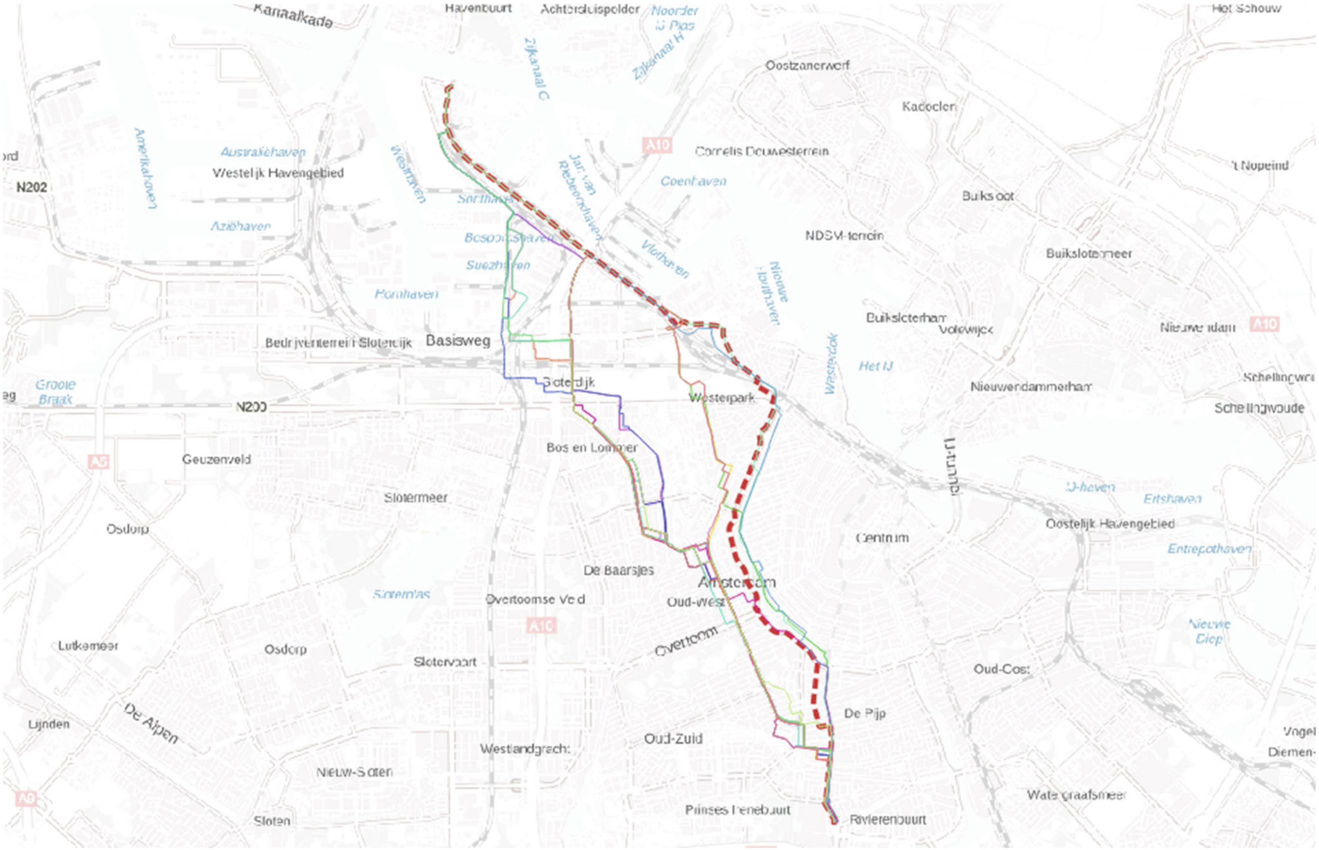


Fig. 7 Generated paths by DSCSG, red-dashed path indicates the observed route

An additional quality measure is the compliance of the distribution for route complexity (discussed in Section 3) in the choice set with the one observed in reality (discussed in Section 4.2).

6 Conclusions and recommendations

There are various methods to generate route choice sets. In this paper we used two of them. Double Stochastic Generation Function (DSCSG) is evaluated because it generates heterogeneous routes, because it is reported to perform well for trips up to a length of 10 kilometers and because it puts more attractive routes in the choice set. The problem with this kind of route choice generation is that the generated route can be over-complicated and unrealistic. Secondly we used Breadth First Search Link Elimination (BFS_LE) to compare run-times and output.

This study formally defines the concept of *route complexity* and computes complexity distributions for both a set of observed routes by bicyclists and a set of routes generated by the POSDAP implementations of BFS_LE and DSCSG. The distributions of the generated paths are shown to be significantly different from reality.

We propose two options to solve the problem. Firstly we could attempt to find a technique where we integrate route complexity inside route choice generation algorithms such as BFS_LE and DSCSG in order to filter out the most inappropriate routes. However this may turn out to be impossible since the quality criterion applies to the distribution of a particular route property as opposed to a route specific property. A second option consists of a two-step method that first applies classical route choice-set generators like BFS_LE and DSCSG to generate an initial choice-set that we subsequently reduce in order to make sure the complexity distribution is more realistic, as described in prior work by Wardenier [14]

Finally resulting choice-sets can be evaluated by their complexity distribution and by their coverage and consistency quality indicators making use of the proposed relaxed similarity functions as in this paper.

Funding information This research received funding from “Stochastics - Theoretical and Applied Research” (STAR) in the Netherlands.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Bikeprint: Download bestanden Nationale Fietstelweek 2015 en 2016 (2017). http://www.bikeprint.nl/fietstelweek/
2. Bovy PH, Fiorenzo-Catalano S (2007) Stochastic route choice set generation: behavioral and probabilistic foundations. Transportmetrica 3(3):173–189
3. ETH-Zurich: Position data processing (2012). https://sourceforge.net/projects/posdap/
4. Fosgerau M, Frejinger E, Karlstrom A (2013) A link based network route choice model with unrestricted choice set. Transp Res B 56:70–80. https://doi.org/10.1016/j.trb.2013.07.012
5. Halldórsdóttir K, Rieser-Schüssler N, Axhausen KW, Nielsen OA, Prato C (2014) Efficiency of choice set generation techniques for bicycle routes. European Journal of Transport and Infrastructure Research 14(4):332–348
6. Hood J, Sall E, Charlton B (2011) A gps-based bicycle route choice model for San Francisco, California. Transportation Letters 3(1):63–75
7. Knapen L, Hartman IBA, Bellemans T (2017) Using path decomposition enumeration to enhance route choice models. Future Generation Computer Systems pp. –. https://doi.org/10.1016/j.future.2017.12.053. https://www.sciencedirect.com/science/article/pii/S0167739X17321866
8. Knapen L, Hartman IBA, Schulz D, Bellemans T, Janssens D, Wets G (2016) Determining structural route components from gps traces. Transp Res B Methodol 90:156–171
9. Mai T, Fosgerau M, Frejinger E (2015) A nested recursive logit model for route choice analysis. Transportation Research Part B: Methodological 75(Supplement C):100–112. https://doi.org/10.1016/j.trb.2015.03.015. http://www.sciencedirect.com/science/article/pii/S0191261515000582
10. Nielsen OA (2000) A stochastic transit assignment model considering differences in passengers utility functions. Transp Res B Methodol 34(5):377–402
11. Prato C, Bekhor S (2006) Applying branch-and-bound technique to route choice set generation. Transportation Research Record: Journal of the Transportation Research Board (1985):19–28
12. Prato C, Bekhor S (2007) Modeling route choice behavior: How relevant is the composition of choice set?. Transportation Research Record: Journal of the Transportation Research Board (2003):64–73
13. Rieser-Schüssler N, Balmer M, Axhausen KW (2013) Route choice sets for very high-resolution data. Transportmetrica A: Transport Science 9(9):825–845
14. Wardenier N, Knapen L, Koch T, Dugundji E (2019) Improving bicycle route choice set generation using route complexity in GPS traces. In: TRB 2019 Annual Meeting. Transportation Research Board, Washington, D.C.