Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 151 (2019) 393–400

Procedia Computer Science

www.elsevier.com/locate/procedia

The 10th International Conference on Ambient Systems, Networks and Technologies (ANT)
April 29 - May 2, 2019, Leuven, Belgium

Path complexity for observed and predicted bicyclist routes

Thomas Koch[a,*], Luk Knapen[b,c], Elenna Dugundji[a,c]

[a] *Centrum Wiskunde en Informatica, Science Park 123, 1098XG Amsterdam, The Netherlands*
[b] *Universiteit Hasselt, Agoralaan building D, 3590 Diepenbeek, Belgium*
[c] *Vrije Universiteit, De Boelelaan 1085, 1081 HV Amsterdam, The Netherlands*

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.

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 is considered to be important for route choice in a similar way as the number of left turns, the number of speed bumps, distance and other. 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 seem to significantly differ so that the choice sets do not reflect the traveler preferences. This paper looks at how the observed routes compare to routes generated by Breadth First Search Link Elimination and Double Stochastic Generation Function method.

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

[*] Corresponding author. Tel.: +31 20 592 4132
E-mail address: koch@cwi.nl

10.1016/j.procs.2019.04.054

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 these two route choice generators seem to significantly differ. The complexity of the routes in the generated choice sets of do not reflect the traveler behaviour we observed in the paths we observed 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 ***Background*** briefly reviews the concept of choice set generation and various choice set generators that are described in the literature. Section ***Route Complexity*** defines the concept of *route complexity* and describes an algorithm to compute it a given route. Section ***Case study*** describes the data set of chosen bicyclist routes, the distribution for the observed complexity and the relations between route properties. Section ***Discussion*** shows that the distribution for *route complexity* in generated choice sets significantly differ from the observed ones.

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 a 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 the finite choice set.

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.

There are various choice set generators for the construction of a choice set.

Prato[8] provide a method called Branch and Bound, which looks for paths that satisfy the boundary conditions: directional, temporal, similarity, loop and movement (avoiding left turns). For example with the temporal constraints, a route with only be included if its travel time is not higher than the shortest time by a certain factor.

Rieser[10] 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[7] 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[5] show DSCSG to be faster than the BFS_LE proposed by Rieser[10]. Halldorsdottir[4] show 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] state that the method guarantees, with high probability, that attractive routes are in the choice set, while unattractive routes are not. 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 is investigated for the route *complexity*.

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 edges are connected by a path having a lower cost than the cost to traverse e. Figure 1 shows the minimum decomposition for a sample path p in a graph having complexity $c(p) = 3$. The example shows that multiple decompositions do exist for path p. Knapen et al. [6] 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. [6] present Algorithm 3.1 to determine the complexity of a path (i.e. the minimum number of basic path components). In algorithm 3.1 we have

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

```
Input Graph $G$, Edge costs $c$, $P = (v_0, v_1, \dots, v_l)$ 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 do
        ▷ 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 $k$
```

a graph G with positive edge costs c and a path $P = (v_0, v_1, \dots, v_l)$ with no non-least-cost edges. 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 Knapen et al. [6] 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. Knapen et al. [6] 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.

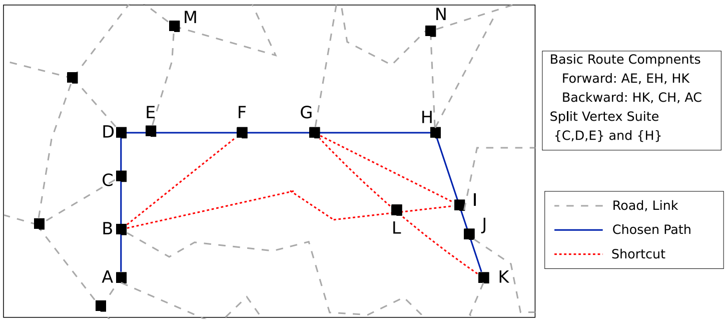

Figure 1. The blue continuous line visiting vertices A, B, C. . . I, J, K is the path followed by the traveler. Paths BF, BLI, GLI, GLK, etc represent shortcuts to the chosen path. There are two sets of split vertices: {C,D,E} and {H}. Hence there are three basic path components (BPC). Sample decompositions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

Figure 2 is taken from Knapen et al. [6] 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 car trips had a complexity lower than 6 basic path components.

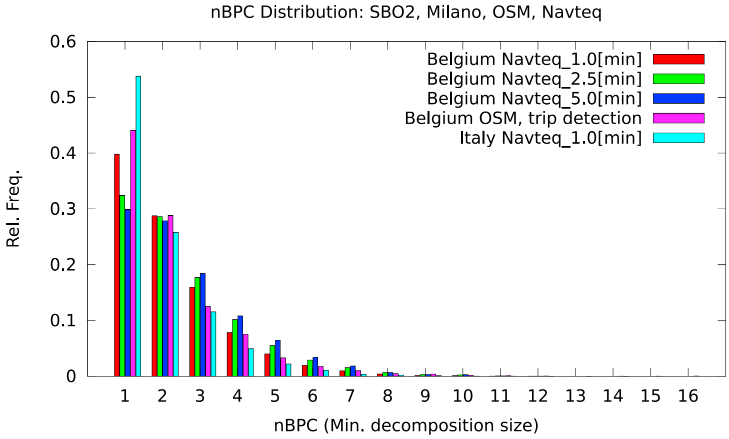

Figure 2. Relative frequency distribution for the size of the minimum decomposition of paths derived from GPS recordings. The Belgian set consists of *person* traces. It was map-matched using different networks and gap-filling thresholds. The Italian set consists of car traces only (recorded by on-board-unit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

4. Case study

This study considers a trip to be utilitarian if and only if $r_d = d_{obs}/d_{short} \leq 1.08$ where d_{obs} and d_{short} are the observed and shortest route lengths respectively (details are found in Wardenier et al. [11]). This definition is stricter than the one used in Knapen et al. [6].

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). 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 as often seen in The Netherlands, using their bike as transportation from and to work, supermarket, school, friends, etc. For privacy reasons the resulting data was anonymized 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 Figure 3 (blue line).

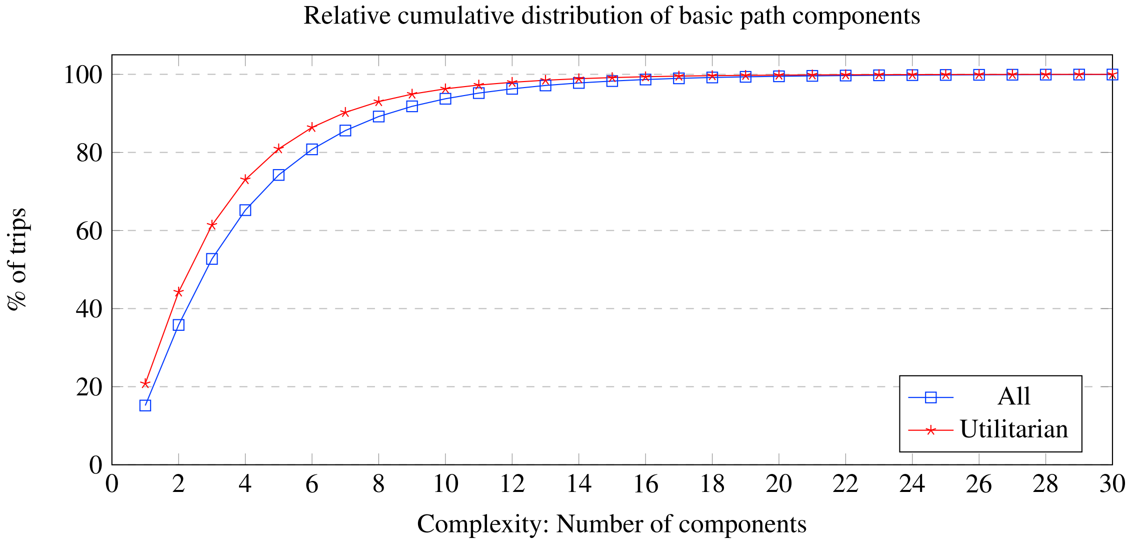

Figure 3. Cumulative distribution of the complexity of paths taken by bicyclists. Blue for unfiltered, red for only utilitarian trips with r_d ¡= 1.08

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 Figure 2. Car mode is the prevalent mode in Flanders according to the recurrent OVG travel behaviour survey https://mobielvlaanderen.be/ovg/ovg52-0.php. Hence most *person traces* consist of car trips and, as a consequence, most trips in the sets investigated by Knapen et al. [6] 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 [4] and *Breadth First Search Link Elimination (BFS_LE)* by Rieser. [10] 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 (OD-pair) were extracted. We used an existing implementation of both algorithms in POSDAP [3] to generate route choice sets for each OD-pair. The distribution of the path complexity was determined for the set of *predicted* paths (i.e. the paths in the generated choice sets). The first option we considered was to include the number of basic path components as extra attribute in the cost function used in the POSDAP DSCSG algorithm, to increase the cost of predicted paths having a complexity that is improbable according to the observed distribution. This way more routes with a lower complexity would end up in the choice set. We did not pursue this option for this paper because of the high cost to adapt the POSDAP algorithm, but it still would be a interesting option for future research.
We decided to run DSCSG in the same way [4] did and to post-process the generated choice sets. 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. Thus we compute the complexity for each route in the choice set generated by POSDAP using the algorithm specified in Knapen et al. [6]. After that we adapt the choice set, keeping in mind the idea that routes with a high number of basic path components are highly unlikely as

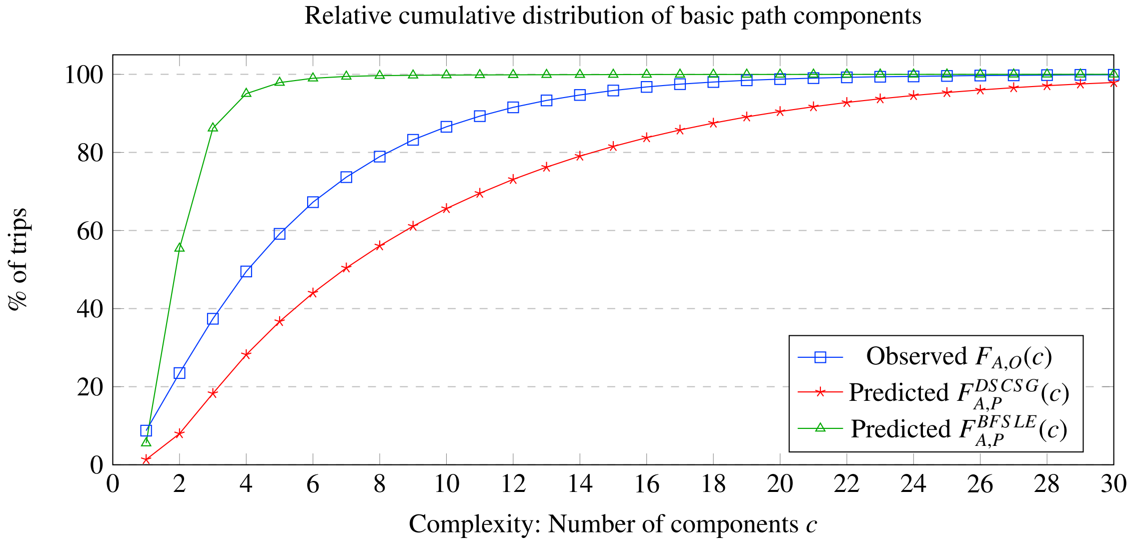

Figure 4. Cumulative distributions of number of basic path components of observed bicycling routes in the Amsterdam (blue) and the number of components in paths predicted by POSDAP 's implementations of Double Stochastic Generation Function (DSCSG) and Bread First Search Link Elimination (BFS_LE)

observed in the recorded 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 $M = 128$ iterations (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 iterations, in which case we will use all found routes. The choice sets are written to CSV files for further processing.

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 figure 4 we plotted the different complexity distributions of the routes observed and the choice sets generated by Double Stochastic Generation Function (DSCSG) and Bread First Search Link Elimination (BFS_LE). The results are in line with what we expected based on the nature of both algorithms. First we try to 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-pairs. In most cases the shortest path $SP(o, d)$ for OD-pair $\langle o, d\rangle$ contains many links. This is shown in Figure 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. 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.

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$(elimLinkSetsColl, paths)$
```

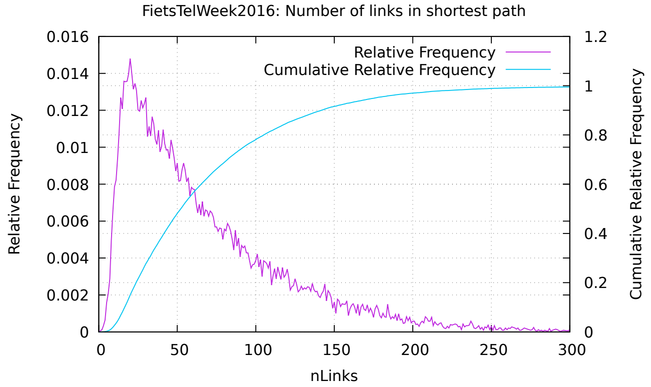

Figure 5. Distribution for the number of links in the shortest path linking O to D in the observed routes. Only 9.4% consists of at most 16 links.

5.3. Coverage

To compare our routes generated with what we found in the literature, we computed coverage and behavioral consistency as found in Prato[9], Halldorsdottir [4] and others. Coverage measures the percentages of the observations for

Table 1. Coverage and Behavioral Consistency of Path Generation Techniques

	Coverage (%)	for	Overlap	Threshold	
Path Generation Technique	100%	90%	80%	70%	Behavioral Consistency
BFS_LE	14.17	22.40	33.87	46.84	0.661
DSCSG	20.25	28.62	39.11	48.98	0.658

which the path generation technique reproduces the observation at a threshold. A path generation technique with a higher overlap is more capable at producing at least one path that is similar to the path observed.The index of behavior consistency measure compares a path generation method with the ideal algorithm that would show 100% overlap for all observations, an algorithm that would replicate all observations. 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 figure 4: the predicted routes have a low conformance to reality.

6. Conclusion

There are various methods to generate route choice sets. In this paper we used two. Double Stochastic Generation Function (DSCSG), because it generates heterogeneous routes, performs well for trips up to a length of 10 kilometers and puts the 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 runtimes and output.

This study formally defines the concept of *route complexity* and computes complexity distributions for both a set of observed routes and for routes generated by the POSDAP software. The distributions are shown to significantly differ and a technique is proposed to enhance the generated choice set w.r.t. complexity. Finally we looked at the route complexity in BFS_LE and DSCSG and reason why they deviate from the route complexity observed in the GPS data.

7. Acknowledgements

This research received funding from 'Stochastics - Theoretical and Applied Research' (STAR) in the Netherlands.

References

[1] Bikeprint, 2017. Download bestanden Nationale Fietstelweek 2015 en 2016. URL: http://www.bikeprint.nl/fietstelweek/.

[2] Bovy, P.H., Fiorenzo-Catalano, S., 2007. Stochastic route choice set generation: behavioral and probabilistic foundations. Transportmetrica 3, 173–189.

[3] ETH-Zurich, 2012. Position data processing. https://sourceforge.net/projects/posdap/.

[4] Halldórsdóttir, K., Rieser-Schüssler, N., Axhausen, K.W., Nielsen, O.A., Prato, C.G., 2014. Efficiency of choice set generation techniques for bicycle routes. European journal of transport and infrastructure research 14, 332–348.

[5] Hood, J., Sall, E., Charlton, B., 2011. A gps-based bicycle route choice model for san francisco, california. Transportation letters 3, 63–75.

[6] Knapen, L., Hartman, I.B.A., Schulz, D., Bellemans, T., Janssens, D., Wets, G., 2016. Determining structural route components from gps traces. Transportation Research Part B: Methodological 90, 156–171.

[7] Nielsen, O.A., 2000. A stochastic transit assignment model considering differences in passengers utility functions. Transportation Research Part B: Methodological 34, 377–402.

[8] Prato, C., Bekhor, S., 2006. Applying branch-and-bound technique to route choice set generation. Transportation Research Record: Journal of the Transportation Research Board , 19–28.

[9] 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 , 64–73.

[10] Rieser-Schüssler, N., Balmer, M., Axhausen, K.W., 2013. Route choice sets for very high-resolution data. Transportmetrica A: Transport Science 9, 825–845.

[11] 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.