BISOS: Backwards Incremental System Optimum Search Algorithm for Fast Socially Optimal Traffic Assignment

Jordan Ivanchev[1], Daniel Zehe[1], Vaisagh Viswanathan[1], Suraj Nair[1], Alois Knoll[4]

***Abstract*—This paper presents an algorithm, called the Backwards Incremental System Optimum Search (BISOS) for achieving system near-optimum traffic assignment by incrementally limiting accessibility of roads for a chosen set of agents. The described algorithm redistributes traffic volumes homogeneously around the city and converges significantly faster than existing methods for system optimum computation in current literature. Furthermore, as previous methods have mainly been developed for theoretical purposes, the solutions provided by them do not contain all the necessary information for a practical implementation such as explicit paths for the commuting population. In contrast, the BISOS algorithm preserves the information about the exact paths of all commuters, throughout the whole process of computing the system optimum assignment. Furthermore, a realistic traffic scenario is simulated using Singapore as a case study by utilizing survey and GPS traffic data. The BISOS routing method needs 15 times less routing computations to get within 1% of the optimal solution for a simulated scenario compared to conventional methods for system optimum computation.**

I. Introduction

According to [1], in 2014, traffic congestion in U.S. has increased to 6.9 billion hours of delay for a year with 3.1 billion gallons of extra fuel spent; this amounts to 160 billion dollars lost due to congestion. Although congestion's main driver is the large number of commuters and the non-homogeneous distribution in time of traffic (rush-hours), its effects can be mitigated by drivers choosing their routes in a more organized way.

With the advancement in technology, such as GPS devices, increased computational capabilities, and the rise of autonomous vehicles it is natural that researchers should be looking in the direction of employing a centralized routing computation system, which redistributes traffic on the road network and reduces overall congestion levels and travel times. In this work we present an fast and efficient algorithm, which can be used as the backbone of such a system.

Depending on whether the commuting population aims at minimizing the total travel time or each individual finds his/her own fastest path, there are two possible traffic assignments. As described first in [2], user equilibrium (UE) occurs when all commuters have perfect information about the traffic situation in the system and route on it so that no commuter would be willing to change his/her path. This is analogous to the Nash equilibrium also known as the Wardrop's equilibrium, which is used in order to calculate the expected traffic assignment in transportation networks. In [3] various modelling techniques for computing user equilibrium are reviewed including individual choice theory, interacting choice theory, effects on travel information affecting individual choice and interacting behaviour.

On the other hand, system optimum (SO) as defined in [4], aims at minimizing the total travel time of the transportation system. Although subtle, there is a difference between the two formulations. In the case of UE all alternative paths with flows on them have the same length in order to ensure that no one has an incentive to switch their route. This constraint does not exist in the SO formulation, which is concerned with everyone arriving at the destination and minimizing the total travel time of the population. The ideal case would be that the UE and SO solutions coincide. In this case, the commuters are "steered" into taking the most optimal routes by different types of incentives.

The difference between the performance of UE and SO has been thoroughly studied and evaluated in [5], where it has been shown that the total travel time at UE is bounded from above at twice the traffic routed in an optimal way. In [6] an upper bound is given to the inefficiency of the stochastic user equilibrium (SUE). The term "price of anarchy" was coined to characterize this inefficiency in [7]. In literature usually the user equilibrium is targeted since it is easier to compute and more realistic [8], [9], however, the need for a system-wide view of performance has long been recognised [10], [11], [12].

Furthermore, efforts have also been made for the design of networks where the UE and SO coincide. In [13] efficient methods for selfish network design are examined in the case of linear latencies and specific network topologies in polynomial times. In [14] it is shown that bases of matroids are maximal structures in which Braess paradox (an overall increase in congestion levels as a result of the addition of a new road) [15] does not occur.

It has been shown in [16] that the coefficient of variation of total travel time is extremely small with respect to the choice of a subgroup of vehicles that are rerouted from a given road segment. This result allows the circumvention of the NP hard problem of choosing which vehicles to reroute from congested roads for more homogeneous traffic flow distribution. In other words, if the finding in [16] holds, there is no significant difference between randomly choosing vehicles to be rerouted from a congested road segment and solving the NP hard problem of finding the optimal set of vehicles to be re-directed. In this way using the suggested algorithm based on random

[1]TUM CREATE, 1 CREATE Way, 10-02 CREATE Tower, Singapore 138602 jordan.ivanchev@tum-create.edu.sg

sampling of drivers, which need to find alternative paths, the computation of system optimum becomes more feasible than the UE algorithm from a computational point of view and brings it closer to being utilized in real life and real time applications.

Even if an efficient method is developed for the computing optimal paths in a commuting network, the drivers are required to follow the recommended routes for the approach to work. There are three main techniques in order to achieve this. The first way to “steer” drivers into socially optimal routes is by using reactive [17] (providing traffic information) or anticipatory [18] guidance systems, which predict future demands and accordingly give recommendations. The second way is by using monetary incentives in the form of form taxation such as in [19], [20], [21], [22], [23].

Finally, a third approach is the method of *soft closing* suggested in [16], where information about very congested areas can be supplied to portions of the commuters with the goal that they avoid those routes. In this way, the targeted system optimum traffic assignment state can be achieved. The partial provision of information to the society instead of full knowledge of the system’s state (assumed in the case of UE) makes the difference between minimum commuting time and the more congested UE traffic assignment. This approach can be considered as “fooling” the commuting population, or at least, part of it, however, an analysis of such implications are beyond the scope of this work. The algorithm described in this paper, determines the drivers that are informed of a *soft closing* of a road and need to find an alternative solution and computes the optimal new paths for them thus achieving near system optimum traffic assignment.

II. Existing System Optimum Computation Algorithms

The classic formulation of the system optimum problem is:

$$\min_{\mathbf{F}} T(F_i) = \sum_i t_i(F_i)F_i \qquad (1)$$

subject to

$$\sum_k p_k^{od} = q_{od} \qquad \forall o, d \qquad (2)$$

$$p_k^{od} \geq 0 \qquad \forall k, o, d \qquad (3)$$

where $t_i(F_i)$ is a latency function representing the relationship between the volume of cars F_i that want to utilize a road segment (link) i and the corresponding time it would take for its traversal. Constraint 2 makes sure that the flow is conserved. p_k^{od} is the flow on path k between origin o and destination d and q_{od} is the number of vehicles that belong to this OD pair. Constraint 3 ensures that the flows on all possible paths are non-negative. The definitional constraints linking the path flows with the link flows are:

$$F_i = \sum_o \sum_d \sum_k p_k^{od} \delta_{i,k}^{od} \qquad \forall i \qquad (4)$$

where $\delta_{i,k}^{od}$ is 1 if path k between o and d passes through i and 0 otherwise.

It can be observed that the objective function is convex, provided the latency function is convex, which is the expected case, as road traverse time grows exponentially as the number of vehicles on the road increases. Furthermore, the feasibility space is also convex since all the constraints are linear. Therefore, convex optimisation methods can be utilized. SO problems are usually solved using the convex combination algorithm proposed in [24] or an improved version called Partan, which was suggested in [25] and discussed in [26], [27]. The approach of the convex combination algorithm can be intuitively explained as linearisation of the objective function at the current point followed by computation of optimal step size that needs to be performed and moving along a gradient minimizing direction.

It must be noted that there is no constraint in the formulation, which states that the computed flows on the links must have integer values. In reality, however, drivers cannot be split between alternative paths. The additional integer constraint is taken into consideration in the suggested algorithm. This might not seem as a significant alteration of the problem formulation, provided a significantly large population is considered, however the performance of the convex combination method is severely reduced if it can only work with integer valued flows as will be shown in the results section.

The convex combination method will be implemented and used as a benchmark for the algorithm suggested in this work. According to [4] it converges to a satisfactory solution within 5 iterations. The Partan extension of the convex combination algorithm implementation is demonstrated in [28]. The biggest simulated network there has $50,000$ drivers to be assigned. The Partan method needs more than $320,000$ paths to be computed in order to converge to an optimum solution. This amounts to more than 6 paths computations per vehicle. It must be noted that for very large real networks storing so many paths can lead to memory problems [28].

Algorithms usually work with path flows instead of link flows and thus employ a method called column generation, which reduces the size of the problem by reducing the the size of the flow vector $\mathbf{F}$. This was first used for user equilibria in [29]. Detailed description of UE and SO algorithms can be found in [4], [30]. Furthermore, computation of UE is discussed rigorously in [31] and algorithmic implementations are suggested in [32]. Flows are shifted between alternative paths until all path costs equalize in [33], [34], [35]. What makes the suggested algorithm different from the pre-existing ones is that it works with link flows and not with paths. In fact, new paths are computed only for certain vehicles and the old ones do not need to be stored, which reduces the memory footprint.

Theoretical work for formulation of SO problems has been done in [36] where a single destination system optimum dynamic traffic assignment is formulated using the cell transmission model as a linear programming problem. It also shows that a sufficient condition for SO is that every unit of flow follows the time-dependent least marginal cost path to the destination. The marginal cost of a road segment is computed as the sum of the traverse time along it and the total added time for all drivers on the road segment by the addition of one more vehicle. The marginal cost can be viewed of the system-aware cost of traversing a link (used in SO computation), as opposed to the travel time cost, which represents the driver-centric routing approach (used in UE computation).

III. Suggested Algorithm: Backwards Incremental System Optimum Search (BISOS)

The algorithm presented in this section aims at resolving the inefficiencies of pre-existing algorithms underlined in the previous section and to provide, feasible and practical solution to the SO traffic assignment problem.

In order to understand the BISOS algorithm it might be helpful to first recollect the incremental assignment algorithm described in [4]. It deals with vehicles one by one, or in chunks, but not with all at once. The incremental assignment method starts off with all weights of the graph equal to the free flow travel times on the links. Each group of vehicles that is assigned on the network computes its paths according to the shortest path algorithm. After each chunk is assigned routes, the weights are recalculated so that each weight represents the current travel time on the respective link. This algorithm is aimed at achieving user equilibrium; however, it is shown in [4] that such a state is not reached by it.

The BISOS algorithm can be viewed to do a similar process but in a reversed fashion. First, all routes are computed based on shortest path algorithm with weights given by the free flow traverse time of the links. After that the most congested road is identified, by looking at the marginal cost of the road segments. A predefined number of vehicles are removed from the road by changing the weight of that road to its current marginal cost. The new routes are computed for those vehicles using the new routing graph with the updated weight. In a way increasing the weight to the marginal cost of the link can be perceived as closing the road segment for the rerouted vehicles since the marginal cost is significantly larger than the free flow cost. Once the link is determined to be unable to give away more vehicles (re-routing vehicles from it leads to an increase in overall population travel time), the next most congested link is explored. When all the links that have a flow higher than their predefined capacity are explored, the iteration is finished. Please note, that the most congested link is chosen by examining the congestion factor of the link, which is proportional to the marginal cost.

In this work, the Bureau of Public Roads (BPR) function is used in order to evaluate the congestion at morning rush hour in a static environment. Rush hour has been chosen since in [4] it has been pointed out that during rush hours, traffic exhibits steady - state behaviour.

Let us introduce the notation that will be used in the formal description of the algorithm:

G - routing graph
R^A - routes of the agent population A
A^m - set of agents that pass through link m
E - list of explored links
α and β - coefficients used in the BPR function
w_i - number of lanes on link i
c_i - congestion factor of link i
U - list of congested links
σ - step size of the algorithm representing the number of agents to be rerouted

The general version of the BISOS algorithm can be formalized in the following way:

Step 0: Initialize flows.
Compute shortest path routes. Get the flows $\mathbf{F}$ and the routes R^A
$T \leftarrow CalculateTravelTimes(G, R^A)$
$E \leftarrow \emptyset$ `// Initialize explored links`
Step 1: Identify most congested link.
The congestion is defined as $c_i = \alpha \left(\frac{F_i}{2000 w_i} \right)^\beta$
$U \leftarrow \forall i : \frac{F_i}{2000 w_i} > 1$ `// identify congested links`
$U \leftarrow U \setminus E$`// Remove explored links.`
$m \leftarrow \max c(U)$`// Identify next link.`
Step 2: Change weight and re-route.
$K \leftarrow RandomSample(A^m, \sigma)$
$SetWeight(G, m, MarginalCost(m))$
foreach $v \in K$ **do**
  $R^v \leftarrow ComputeRoute(G)$`// new route`
  $\tilde{R^A} \leftarrow R^v \cup R^{A \setminus v}$`// store route`
end
Step 3: Recalculate travel times.
$\tilde{T} = CalculateTravelTimes(G, \tilde{R^A})$
if $\tilde{T} < T$ **then**
  $T \leftarrow \tilde{T}$`// Update minimum travel time.`
  $R^A \leftarrow \tilde{R^A}$`// Update routes.`
else
  $E \leftarrow E \cup m$`// Add link to explored list`
  if $U = \emptyset$ **then**
    Go to Step 4
  end
end
Go to Step 1
Step 4: Test for convergence.
$E \leftarrow \emptyset$
$ResetWeights(G)$
if $HasConverged()$ **then**
  Stop
else
  Go to Step 1
end

Algorithm 1: Backwards incremental system optimum search algorithm

The BISOS algorithm presents efficient solutions to both the high computation cost and memory storage issues in current methods. First of all, it does not have to store in memory any of the possible paths for an OD pair. For every user there is exactly one path from origin to destination that needs to be stored at all times. If a large city needs to be analysed in the sense of an SO solution, the path storage can become challenging, when dealing with millions of vehicles that each has several possible paths.

Second, no unnecessary paths need to be computed. Instead of recomputing paths for all users at every iteration step, the proposed algorithm re-routes only a portion of the vehicles that are on the most congested road in the current iteration. This amounts to a drastic reduction of the number of paths to be calculated, which is, in fact, the most time consuming part of any SO algorithm. In this way the column generating methods, which are usually used, can be viewed as a natural consequence of the algorithm since only roads that are congested beyond a certain threshold are inspected.

Furthermore, the logic behind the algorithm is fairly intuitive and more importantly stems from an actual technique for routing managing, namely the selective information dissemination *soft closing* method. Finally, many of the computations can be performed in parallel in order to gain an additional performance boost.

Naturally there are some concerns with this type of greedy optimization. The major one is that converging to the optimal solution is not guaranteed. The random selection of vehicles to be re-routed at every sub-iteration, excludes the possibility of optimal solution. It has been shown in [16] that the variation of the resulting traffic conditions with respect to choosing different set of agents to move from a specific road segment is less than 10^{-4}. Therefore, it can be argued that the optimal set of agents to be moved need not be computed explicitly for practical purposes. In the results section the final result of the suggested algorithm will be compared to the actual SO solution achieved by the convex combination algorithm in order to verify this assumption. The step size of the algorithm σ, which defines the size of the agent set that have to be rerouted, can be varied to mitigate the effects from not explicitly computing the set of agents to be removed from the road segment.

Furthermore, as the set of agents to be re-routed is chosen at random, it is possible (although with small likelihood) that, the randomly sampled agents, which have to be removed from a certain road have no viable alternatives. In this case, the total travel time will be increased and as a result of that, the road segment will not be explored again within the current main iteration. In order to avoid this, one might set a threshold of how many times, the road segment can "fail" in the sense of not producing an improvement to traffic conditions when a set of agents is removed from it. The performance of the algorithm for different values of the threshold, referred to as "failed attempts limit" (FAL) will be examined in the results section.

The choice of which link to examine is one of the challenging aspects of the suggested algorithm. It is very likely that the most congested link will not be able to give away all its commuters and that at some point, moving a vehicle from it to an alternative path will start to increase the overall population time rather than decrease it. However, the link might still have the highest congestion value. In order to solve this problem, once such a situation occurs, the link in question is excluded from the list of explorable links and its weight is not reset from the marginal cost value assigned to it. Another parameter of the algorithm is the threshold congestion value (TH), which sets the minimum congestion value of examinable links. In the formal description of the algorithm TH is set to 1 since this is the value where the road reaches its maximum throughput.

After traffic redistribution occurs at the lower congestion levels, the initially excluded significantly congested links might become viable redistribution options once again. For this reason, at every iteration of the algorithm, the set of unexplorable links is reset, or in other words, starts as the empty set and all weights are set back to their free flow initial values. Then, for example, at the second iteration of the algorithm, the most congested link that was removed from the explorable set of links can be attempted again after all redistributions on the lower levels of congestion have been done.

IV. Results

This section will examine the performance of the algorithm compared to other pre-existing methods. The performance will be evaluated with respect to the speed of the algorithm and the accuracy of the final solution. Furthermore, the performance of the algorithm is examined while varying the three main parameters, namely; the threshold TH, the step size σ, the failed attempts limit FAL. The city of Singapore will be used as a case study for all experiments. The methodology of generating traffic described in [16] is used, utilizing survey data describing the OD patterns of the population during rush hour and GPS traces used for validation of the simulation results in order to achieve a more realistic scenario.

A. *Quality of Solution Compared to SO Solution and Speed of Convergence*

The SO solution for the traffic assignment was computed using the convex combination method and compared to the solution of the proposed algorithm. The basis of the comparison will be the number of paths to be computed since this is the most time-consuming part of SO computation algorithms and is independent of the machine used unlike the runtime. The speed of convergence comparison of the BISOS algorithm and the convex combination algorithm with and without the integer constraint are shown on Fig. 1.

It can be observed that the BISOS algorithm converges much faster than the convex combination method. In fact, the BISOS algorithm needs only $87,000$ route computations, which is about a quarter of the computations needed for just one iteration of the convex combination algorithm. Altogether, the BISOS algorithm converges 15 times faster, which is a significant result. Since the optimal solution is not guaranteed by BISOS, Fig. 2 examines the difference between the quality of the solutions presented by the BISOS algorithm with different congestion thresholds and the convex combination method with and without the integer constraint.

It can be observed that the BISOS algorithm is not only faster than the convex combination method but also provides a much better solution than the integer constrained version of

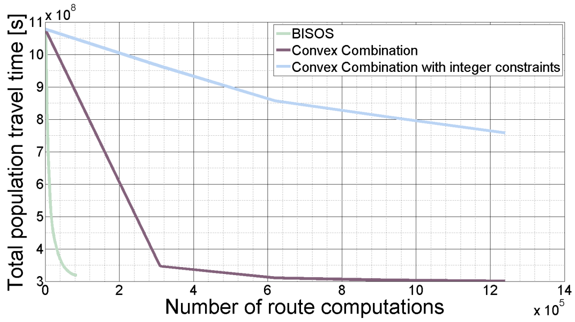

Fig. 1: The convergence speed with respect to needed route computations for convergence for the BISOS algorithm and the convex combination algorithm with and without integer constraints.

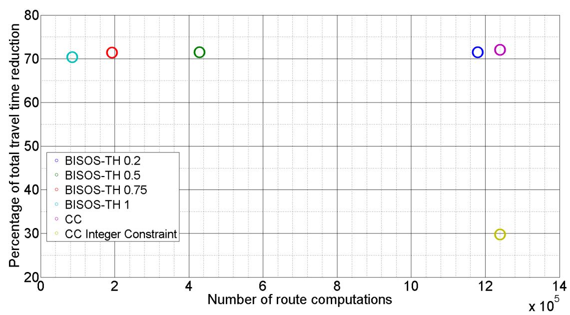

Fig. 2: A demonstration of the trade-off between computing time and quality of system optimum solution and comparison between the final solution of BISOS algorithm and the convex combination method with and without integer constraints.

the convex combination method. For the examined simulated scenario the final solution is only within 1% difference from the theoretical, although infeasible, system optimum solution. The trade-off between computation time and optimality can be observed as well. Decreasing the threshold congestion value, makes the algorithm examine more road segments, thus increasing the number of route computations, however distributing more traffic and further reducing congestion levels. The default value of the congestion threshold is kept at 1 for the rest of the experiments since the throughput of a road is maximized for this value.

B. Performance for Various Step Sizes and Failed Attempts Limits

Technically speaking, the bigger the step size, the higher the opportunity to parallelize the algorithm. For example, if the step size is set to 20 vehicles to be re-routed at one step and there are 20 available cores, the time for one iteration will be virtually the same as re-routing 1 vehicle at a time. The increase in step size, however, might note further speed up the computation after the step size gets bigger than the number of physical cores that are available. Fig. 3 depicts the step size influence on the quality of the final solution and the number of route computations for convergence.

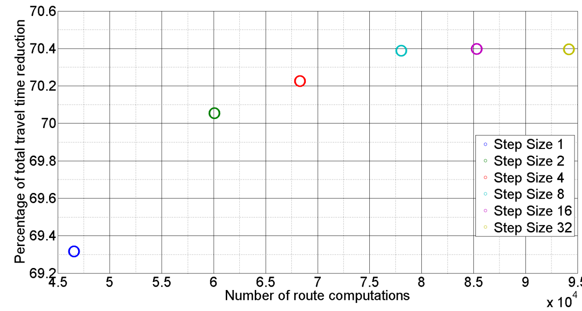

Fig. 3: BISOS algorithm evolution for difference step sizes and convergence speed as a function of required route computations for convergence with respect to BISOS's step size.

A rather unexpected result, which can be observed in Fig. 3 is that the SO solution gets better with increasing of the step size. Intuitively, the opposite is expected as a smaller step size allows for more precise calculation of the critical point where a road cannot give away any more vehicles. The results, however, can be explained by considering the stochastic nature of the BISOS algorithm. Provided that a road is congested, re-routing a single agent from it has a higher probability of increasing the overall travel time (or failed re-routing attempt) than removing a larger group of agents. The effect of the variation of the parameter that deals with failed attempts of re-routing, FAL, is depicted on Fig. 4

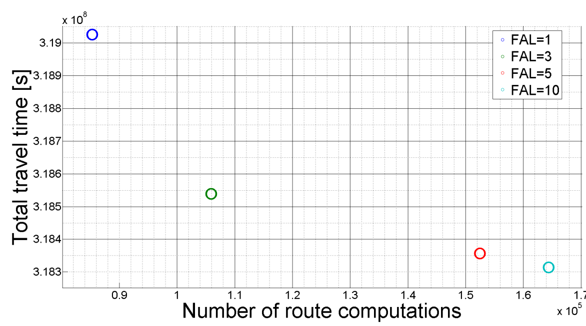

Fig. 4: Trade-off between number of computations and quality of SO solution for different values of the failed attempt limit value.

It can be observed that, as predicted, increasing the FAL improves the quality of the BISOS solution. Furthermore, the very exceedingly small difference between number of route computations for FAL$=5$ and FAL$=10$ indicates that a saturation is reached and no further increase in the FAL value is required.

V. Conclusion

The main contribution of this work is the BISOS algorithm, which aims at computing a system optimum routing solution for all agents in a transportation system. The suggested algorithm converges 15 times faster than current algorithms and furthermore, provides explicit paths for every single driver, which is something that to the best of our knowledge none of existing methods do.

The immense reduction of route computations needed for the algorithm to converge and the practicality of its functionality present a great step in the direction of a centralized routing control system. The key practical aspect of the algorithm besides its speed, is that it can be halted at virtually any point of time, if time constraints require this, and would still be able to produce explicit paths for all agents in the system. This is a highly desired trait for systems, which should be used in real time. Such an algorithm may turn system optimum routing strategies from theoretical measures for estimating the utilization of a road system into practically used strategies for severe reduction of congestion levels.

This algorithm can also be used as a first step of a hybrid system optimum computation since it reaches a good solution an order of magnitude faster than the existing one. In cases where the theoretical minimum should be computed, the first few iterations of the standard algorithms can be speeded up by using the BISOS algorithm final solution as a starting point for the conventional methods.

Finally, in contrast to current methods of reducing congestion, which encompass expensive additional road construction or alteration, the system optimum routing approach presents a strategy that can both ease traffic conditions and is cost free in the sense that no construction of infrastructure is necessary. In a way the functioning of the BISOS algorithm can be viewed as empowering the previously static road infrastructure to present itself in a different way to each of the traffic participants, thus making it dynamical and able to adapt to all possible traffic demand changes.

Acknowledgement

This work was financially supported by the Singapore National Research Foundation under its Campus for Research Excellence And Technological Enterprise (CREATE) program.

References

[1] D. Schrank, B. Eisele, T. Lomax, and J. Bak, "Urban mobility scorecard," Technical Report August, Texas A&M Transportation Institute and INRIX, Inc, Tech. Rep., 2015.

[2] M. Beckmann, C. McGuire, and C. B. Winsten, "Studies in the economics of transportation," Tech. Rep., 1956.

[3] M. van Essen, T. Thomas, E. van Berkum, and C. Chorus, "From user equilibrium to system optimum: a literature review on the role of travel information, bounded rationality and non-selfish behaviour at the network and individual levels," *Transport Reviews*, pp. 1–22, 2016.

[4] Y. Sheffi, "Urban transportation network," *Pretince Hall*, 1985.

[5] T. Roughgarden and É. Tardos, "How bad is selfish routing?" *Journal of the ACM (JACM)*, vol. 49, no. 2, pp. 236–259, 2002.

[6] X. Guo, H. Yang, and T.-L. Liu, "Bounding the inefficiency of logit-based stochastic user equilibrium," *European Journal of Operational Research*, vol. 201, no. 2, pp. 463–469, 2010.

[7] C. Papadimitriou, "Algorithms, games, and the internet," in *Proceedings of the thirty-third annual ACM symposium on Theory of computing*. ACM, 2001, pp. 749–753.

[8] H. Mahmassani, T. Hu, S. Peeta, and A. Ziliaskopoulos, "Development and testing of dynamic traffic assignment and simulation procedures for atis," ATMS applications. Technical Report DTFH61-90-R-0074-FG, Center for Transportation Research, University of Texas at Austin, Tech. Rep., 1994.

[9] M. Ben-Akiva, M. Bierlaire, J. Bottom, H. Koutsopoulos, and R. Mishalani, "Development of a route guidance generation system for real-time application," in *Proceedings of the IFAC Transportation Systems 97 Conference, Chania*. Citeseer, 1997.

[10] C. Charbonnier, J. Farges, and J. Henry, "Models and strategies for dynamic route guidance. part c, optimal control approach," in *DRIVE Conference (1991: Brussels, Belgium). Advanced telematics in road transport*, 1991.

[11] G. Beccaria and A. Bolelli, "Modelling and assessment of dynamic route guidance: the margot project," in *Vehicle Navigation and Information Systems Conference (3rd). Vehicle navigation & information systems: conference record of papers*, 1992.

[12] I. Kaysi, M. Ben-Akiva, and A. de Palma, "Design aspects of advanced traveler information systems," in *Urban Traffic Networks*. Springer, 1995, pp. 59–81.

[13] D. Fotakis, A. C. Kaporis, and P. G. Spirakis, "Efficient methods for selfish network design," *Theoretical Computer Science*, vol. 448, pp. 9–20, 2012.

[14] S. Fujishige, M. X. Goemans, T. Harks, B. Peis, and R. Zenklusen, "Matroids are immune to braess paradox," *arXiv preprint arXiv:1504.07545*, 2015.

[15] P.-D. D. Braess, "Über ein paradoxon aus der verkehrsplanung," *Unternehmensforschung*, vol. 12, no. 1, pp. 258–268, 1968.

[16] J. Ivanchev, S. Litescu, D. Zehe, M. Lees, H. Aydt, , and A. Knoll, "Determining the most harmful roads in search for system optimal routing," Technical University Munich, Tech. Rep. TUM-I1632, 2016.

[17] M. Ben-Akiva, A. de Palma, and I. Kaysi, "The impact of predictive information on guidance efficiency: An analytical approach," in *Advanced methods in transportation analysis*. Springer, 1996, pp. 413–432.

[18] I. A. Kaysi, "Framework and models for the provision of real-time driver information," 1992.

[19] M. Zangui, Y. Yin, S. Lawphongpanich, and S. Chen, "Differentiated congestion pricing of urban transportation networks with vehicle-tracking technologies," *Transportation Research Part C: Emerging Technologies*, vol. 36, pp. 434–445, 2013.

[20] W. S. Vickrey, "Congestion theory and transport investment," *The American Economic Review*, vol. 59, no. 2, pp. 251–260, 1969.

[21] E. J. Gonzales and C. F. Daganzo, "Morning commute with competing modes and distributed demand: user equilibrium, system optimum, and pricing," *Transportation Research Part B: Methodological*, vol. 46, no. 10, pp. 1519–1534, 2012.

[22] T. Harks, I. Kleinert, M. Klimm, and R. H. Möhring, "Computing network tolls with support constraints," *Networks*, vol. 65, no. 3, pp. 262–285, 2015.

[23] H. Yang, W. Xu, B.-s. He, and Q. Meng, "Road pricing for congestion control with unknown demand and cost functions," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 2, pp. 157–175, 2010.

[24] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval research logistics quarterly*, vol. 3, no. 1-2, pp. 95–110, 1956.

[25] L. J. LeBlanc, R. V. Helgason, and D. E. Boyce, "Improved efficiency of the frank-wolfe algorithm for convex network programs," *Transportation Science*, vol. 19, no. 4, pp. 445–462, 1985.

[26] M. Florian, J. Guálat, and H. Spiess, "An efficient implementation of the partan variant of the linear approximation method for the network equilibrium problem," *Networks*, vol. 17, no. 3, pp. 319–339, 1987.

[27] Y. Arezki and D. Van Vliet, "A full analytical implementation of the partan/frank–wolfe algorithm for equilibrium assignment," *Transportation Science*, vol. 24, no. 1, pp. 58–62, 1990.

[28] O. Jahn, R. H. Möhring, A. S. Schulz, and N. E. Stier-Moses, "System-optimal routing of traffic flows with user constraints in networks with congestion," *Operations research*, vol. 53, no. 4, pp. 600–616, 2005.

[29] T. Leventhal, G. Nemhauser, and L. Trotter Jr, "A column generation algorithm for optimal traffic assignment," *Transportation Science*, vol. 7, no. 2, pp. 168–176, 1973.

[30] M. Patriksson, *The traffic assignment problem: models and methods*. Courier Dover Publications, 2015.

[31] J. R. Correa and N. E. Stier-Moses, "Wardrop equilibria," *Wiley encyclopedia of operations research and management science*, 2011.

[32] R. B. Dial, "A path-based user-equilibrium traffic assignment algorithm that obviates path storage and enumeration," *Transportation Research Part B: Methodological*, vol. 40, no. 10, pp. 917–936, 2006.

[33] M. Florian, I. Constantin, and D. Florian, "A new look at the projected gradient method for equilibrium assignment. cd-rom," *Transportation Research Board of the National Academics, Washington, DC*, 2009.

[34] G. Gentile, "Linear user cost equilibrium: a new algorithm for traffic assignment," *submitted to Transportation Research B*, 2009.

[35] H. Bar-Gera, "Traffic assignment by paired alternative segments," *Transportation Research Part B: Methodological*, vol. 44, no. 8, pp. 1022–1046, 2010.

[36] A. K. Ziliaskopoulos, "A linear programming model for the single destination system optimum dynamic traffic assignment problem," *Transportation science*, vol. 34, no. 1, pp. 37–49, 2000.