Title
Automatic Generation of School Bus Routes in Los Angeles

Authors
Porter, Mason A.
Spencer, David J.
Hung, Cu Hauw

Publication Date
2020-02-01

RESEARCH REPORT

Institute of Transportation Studies | UNIVERSITY OF CALIFORNIA

Automatic Generation of School Bus Routes in Los Angeles

Mason A. Porter, Professor, Department of Mathematics
David J. Spencer, M.A., Department of Mathematics
Cu Hauw Hung, M.A. Student, Department of Computer Science

February 2020

UCLA Institute of Transportation Studies

Report No.: UC-ITS-2019-40 | DOI: 10.17610/T6V30V

Technical Report Documentation Page

1. Report No.

UC-ITS-2019-40

2. Government Accession No.

N/A

3. Recipient's Catalog No.

N/A

4. Title and Subtitle

Automatic Generation of School Bus Routes in Los Angeles

5. Report Date

December 31, 2019

6. Performing Organization Code

UCLA-ITS

7. Author(s)

Mason A. Porter, Professor, Department of Mathematics

David J. Spencer, M.A., Department of Mathematics

Cu Hauw Hung, M.A. Student, Department of Computer Science

8. Performing Organization Report No.

N/A

9. Performing Organization Name and Address

Institute of Transportation Studies, UCLA

3320 Public Affairs Building

Los Angeles, CA 90095-1656

10. Work Unit No.

N/A

11. Contract or Grant No.

UC-ITS-2019-40

12. Sponsoring Agency Name and Address

The University of California Institute of Transportation Studies

www.ucits.org

13. Type of Report and Period Covered

Final Report
(October 2018–December 2019)

14. Sponsoring Agency Code

UC ITS

15. Supplementary Notes Our program is available at https://github.com/davidspencer6174/schoolbusrouting. DOI: 10.17610/T6V30V	

16. Abstract

The goal of our project is to automatically generate school bus routes for the Los Angeles Unified School District (LAUSD). We examined four algorithms, including two from the existing literature and two new ones that we developed. A major focus of our work was the construction of "mixed-load routes," which transport students from multiple schools. Based on our measurements (whose imperfections we discuss), three of the four algorithms perform at least as well as the existing route plan, and one of those three performs better than the existing route plan. We also delivered a user-friendly routing program to LAUSD that uses one of these algorithms, and we have made our software publicly available. Our insights and results are also applicable to other school districts that permit mixed-load routing.

17. Key Words TBD		**18. Distribution Statement** No restrictions.	
19. Security Classification (of this report) Unclassified	**20. Security Classification (of this page)** Unclassified	**21. No. of Pages** 31	**22. Price** N/A

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized.

About the UC Institute of Transportation Studies

The University of California Institute of Transportation Studies (UC ITS) is a network of faculty, research and administrative staff, and students dedicated to advancing the state of the art in transportation engineering, planning, and policy for the people of California. Established by the Legislature in 1947, ITS has branches at UC Berkeley, UC Davis, UC Irvine, and UCLA.

Acknowledgements

This study was made possible through funding received by the University of California Institute of Transportation Studies from the State of California via the Public Transportation Account and the Road Repair and Accountability Act of 2017 (Senate Bill 1). We thank the State of California for its support of university-based research, and especially for the funding that we received for this project. We also thank the Los Angeles Unified School District for providing data, advice, and domain expertise. We are grateful to William He, Kien Hoac, Janet Hutchinson, Lilia Montoya-Herrera, Ryan Solchenberger, and Ruben Suarez for their efforts. The map data is copyrighted by OpenStreetMap contributors and is available from https://www.openstreetmap.org.

Disclaimer

Automatic Generation of School Bus Routes in Los Angeles

Mason Porter, Professor, Department of Mathematics, UCLA
David Spencer, M.A., Department of Mathematics, UCLA
Cu Hauw Hung, M.A. Student, Department of Computer Science, UCLA

February 2020

UCLA Institute of Transportation Studies

Table of Contents

Table of Contents

List of Tables

List of Figures

Executive Summary

Executive Summary

The goal of our project is to create a program to automatically generate bus routes for the Los Angeles Unified School District (LAUSD). This is an instance of the school bus routing problem (SBRP). We seek to generate route plans with low cost and high service quality. We use number of routes as a proxy for cost and higher mean student ride time as a proxy for lower service quality, allowing us to make comparisons with LAUSD's existing manually generated route plan. Many formulations of the SBRP are NP-hard, so we seek an algorithm to find good route plans, rather than optimal ones.

The Los Angeles case of the SBRP has many features that need to be considered while routing.

- LAUSD is large, servicing about 38000 students and running about 1700 daily routes in the 2018–2019 school year.
- Routes are subject to regulatory constraints, such as a ride-time limit.
- Routing uses a mixed objective: Routers try to minimize both the number of routes and the mean student travel time.
- Many routes pick up students of multiple schools, a practice known as "mixed-load routing."
- About one quarter of the students that are eligible for busing have special needs.
- Students belong to different age groups, and LAUSD prefers not to have high-school students ride with elementary-school students.
- Some students ride infrequently, so LAUSD overbooks many buses, sometimes assigning double or even triple a bus's legal capacity.
- The LAUSD bus fleet is heterogeneous: Its buses have different sizes and different abilities to handle students' special needs (such as spaces for wheelchairs and medical equipment).
- Students who live within a certain radius of their schools are ineligible for busing unless they demonstrate a special need.

Except for the eligibility condition, each of these features has been considered in the SBRP literature. See (Ellegood et al. 2019) for a recent review. However, creating an algorithm that handles all of these features simultaneously requires care and close collaboration with a school district.

Of particular importance is the question of how to handle mixed-load routing. Mixed-load routing is especially beneficial when there is a group of nearby schools and a group of nearby students who attend those schools, but these two groups are far away from each other. Rather than having a bus traverse the long distance from the students to the schools for each school, it is usually more efficient to have a single bus pick up all of the students and take them to each school. Mixed-load routing complicates the SBRP, because (1) it prevents reduction to the smaller single-school case and (2) it necessitates the consideration of school bell times. However, based on discussions with the LAUSD routing team, mixed-load routing is crucial in their system, and making the program capable of it was a high priority.

In our research, we examined four algorithms: two from the existing SBRP literature and two of our own design. Comparing route plans that were generated by the program with the existing route plan was challenging for several reasons, which we will discuss in this report. According to our measurements, the algorithm in the program is able to improve on existing routes (and the other algorithms produce routes that are comparable to existing ones). Our measurements have some issues (which we will discuss), so from LAUSD's perspective, they may or may not constitute improvements over their routes. However, we believe that the routes that are generated by the program that we provided to LAUSD give a useful starting point for their routing team, who can then make tweaks to improve routes that it generates.

Contents

Literature Overview

Research on the SBRP may concern any, multiple, or all of several subproblems that are related to school bus routing. Desrosiers et al. (1981) proposed the following list of possible subproblems of the SBRP: data preparation, bus-stop selection, bus route generation, school bell-time adjustment, and route scheduling. A recent SBRP review (Ellegood et al., 2019) included the additional subproblem of strategic transportation policy, which encompasses analysis of how to determine a district's routing policies.

Most existing SBRP research has sought to design algorithms to solve SBRP subproblems in specific applications (which may be real or artificial). One can categorize these applications in several ways. For example, Ellegood et al. (2019) gave a list of possible categories in their review, building on a similar categorization in an earlier SBRP review (Park & Kim, 2010). An SBRP problem can involve multiple schools or only one school, can be within an urban or rural environment, can allow or disallow mixed-load routing, can use a homogeneous or heterogeneous bus fleet, can optimize any subset of a variety of objectives, and can enforce any subset of a variety of constraints. Many SBRP solution approaches use ideas and metaheuristics from the literature for a broader Vehicle Routing Problem (VRP), which is discussed in (Toth & Vigo, 2014).

Very little research thus far has analyzed the SBRP more broadly than within the context of a specific application. One paper that does is (Ellegood, Campbell, and North, 2015), the only paper that is cited in (Ellegood et al., 2019) for the strategic transportation policy subproblem.

Bus Routing in Los Angeles

When designing route-generation algorithms, it is important to keep in mind the specifics of an application. There are many pertinent details in our with work LAUSD that shaped our attempts at the problem over time and underscored the importance of taking a flexible approach.

Let the symbol '~' denote an equivalence relation on the set of students, such that $a \sim b$ if students *a* and *b* attend the same school and wait at the same bus stop. We define a "student group" to be any equivalence class of students. Therefore, when we refer to a bus "picking up" a student group, we mean that the bus is picking up some positive number of students who attend the same school and wait at the same bus stop (and that no other student attends that school and waits at that bus stop).

Bus Assignment

One important consideration is the assignment of buses to routes. The heterogeneity in the available bus sizes makes this a difficult task. While constructing routes, it is helpful to be able to tell whether a route is overcrowded. If all buses have the same size, this is straightforward, as one has advanced knowledge of the capacity of the bus that will handle the route. However, because buses have different sizes, whether a route is overcrowded depends on which bus is assigned to handle it. Assigning buses while constructing routes has a downside, as a bus that one has assigned to a route may be better to employ instead in a later route. However, assigning buses after constructing routes also has a downside, as there may be no feasible assignment of buses to routes and during route construction, it may be unknown whether a route is overcrowded. If all buses have the same size, one does not need to choose when to assign buses.

In the early phases of our algorithm design, our programs were "bus-greedy," in that they added a student group to a route as long as there remained a large enough bus that could handle that route. However, this introduces the problematic feature that one exhausts the largest buses early in the routing process. In our algorithms, there was no reason to assume that the earliest-routed student groups would use the largest buses in the best way. It may be possible to find heuristics that suggest which student groups should be routed first to use these buses in the best way. However, even if we found a successful heuristic to do this, it would force any algorithm that we used to route student groups in this order. This may be undesirable, as some algorithms may benefit from routing student groups in a different order.

In light of the above issues, we elected to use a different approach. Instead of considering buses during the routing phase, our program initially generates routes without considering bus capacity. The sizes of these routes are constrained only by the travel time limit, so some routes have hundreds of students. After the program finishes generating routes, it begins bus assignment. This program is what we call "bus-last," as it does not consider buses until all routes have been created.

We now discuss how bus assignment works. In some cases, sufficiently large buses exist to handle entire routes; in these cases, our program uses the smallest possible bus to do so. In other cases, no single bus can handle a route, so our program will consider ways to split such the route into multiple routes, such that each individual bus handles some of the student groups from the original route.[1] It uses a brute-force method that enumerates assignments of buses to subsets of the set of student groups on the route. It prunes to take advantage of symmetries, prevent duplication of computational work, and avoid computational work in cases where successful assignment is provably impossible.

Usually, it is possible for our program to find the minimum number of buses that are necessary to handle all of a route's student groups within one second. If there are multiple ways to use the minimum number of buses to handle a route, our program breaks ties according to whichever approach gives a smaller mean student ride time. With this in mind, we say that a bus assignment is "optimal" when it uses the smallest possible number of buses and, among such bus assignments, minimizes the mean student ride time. When our program fails to find the minimum number of buses within a second, it uses a simple (but not optimal) approach to get a reasonable answer. Usually, fewer than 1% of routes reach that point, and our program found an optimal assignment of buses for the vast majority of routes in a small fraction of a second. For instance, when we ran the final version of our program on LAUSD's 2018–2019 data on an Intel Core i7-8750H machine, we found an optimal assignment in less than a tenth of a second for about 91% of the routes.

It seems likely that our bus-last programs will result in a more efficient use of the largest buses than our bus-greedy programs. One way to understand this is by considering an idealized example. Suppose that (1) a school has five student groups that each have the same number of students and that (2) these student groups are close enough to each other and to the school that a route through all five of them is within the ride-time limit. Additionally, we also suppose that (3) the largest bus size can handle four of these student groups and the second-largest bus size can handle three of these student groups, but no bus size can handle all five of these student groups. Our bus-greedy programs would build

[1] LAUSD disallows breaking up a student group across two or more buses, because if students from one student group board two buses to the same school, many students who are assigned to the first morning bus are likely to instead try to get on the second bus. In our project, we elected to give this condition priority over the bus capacity constraint. Specifically, if a bus picks up students from only one student group, we allow it to pick up all students from this student group even if this causes the route to exceed the bus's capacity. This occurred in some of our computations, but it was very infrequent.

a route through four of these student groups, but then would not add the fifth student group, because no bus exists that can handle a route with all five student groups. Therefore, our bus-greedy programs would construct a second route that visits only one student group. By contrast, our bus-last programs would build a route through all five student groups. They would find an optimal assignment of buses for this route (which has only five student groups) almost immediately. This assignment would require two buses, because it is possible to handle all of these student groups with two buses (but not with one). This division into two routes may result in the use of two buses of the second-largest size or smaller. If so, the bus-last approach has routed these student groups with the same number of buses as the bus-greedy approach, and the bus-last approach has a mean student ride time that is at most the mean student ride time of the bus-greedy approach (by the definition of an optimal assignment of buses).

In the early phases of the project, our testing indicated that a bus-last approach is a better approach than the bus-greedy approach. Our algorithms and our assumptions about the LAUSD routing application have changed significantly since our early efforts, so although we expect that modifying the current bus-last algorithm to be bus-greedy (and thus not bus-last) would result in worse route plans by our measurements, we have not collected data to justify this expectation.

Student Ages

The ages of the students are provided in the input data. For social reasons, LAUSD strongly prefers that elementary-school students do not ride with high-school students. This occurs in fewer than 10% of LAUSD's existing routes, but it does occur. For simplicity, we forbid our program from putting elementary-school students and high-school students on the same route, except for cases in which elementary-school students and high-school students belong to the same student group.

Determining the feasibility of a bus assignment to an individual route requires some choices about student ages. Because older students tend to take up more space, bus capacities depend on the ages of the students who will be riding on it. LAUSD provided us with the associated bus capacities to take this into account. Typically, the capacity of a bus for elementary-school students is about 150% of its capacity for high-school students, so this difference is significant. Because students in different age groups can ride the same bus, it is necessary to decide how to determine feasibility for mixed-age situations. Let E, M, and H denote a bus's capacities for elementary-school, middle-school, and high-school students, respectively. If a route has e, m, and h students in these age groups, we consider a bus route to be feasible if

$$\frac{e}{E} + \frac{m}{M} + \frac{h}{H} \leq 1.$$

If only one age group rides a bus, this inequality necessitates that the number of students who are on that bus be at most the capacity for that age. If multiple age groups are riding a bus, the extent to which each of E, M, and H matters is larger if more students of that age group are riding.

Overbooking

LAUSD regularly practices overbooking, because there are some students who almost never ride buses. In one extreme example, more than 200 students were assigned to a bus with capacity 65, but the bus was never full. Routers base their decisions on a combination of factors, including student location and the school that they attend. It is our impression that LAUSD's routers are generally successful at determining which students are most likely to ride buses.

If we possessed ridership probabilities for each individual student, we would be able to overbook buses with some confidence that our decisions are reasonable, because we could estimate the daily probability of a given route being over its capacity. However, such data are not available, and although routers generally have good intuition about these situations, it is unreasonable to ask them to produce specific probabilities for tens of thousands of students.

One type of data that is available is ridership data at the bus level (that is, aggregate counts of how many riders were on a bus in a given day), but only for a few days in a given year. We hoped to use this data to estimate ridership rates for each zip code. However, this is challenging because most buses pick up students from multiple zip codes.

We tried using a log-likelihood approach to estimate ridership probabilities for each zip code based on the ridership data for each bus. Taking those probabilities as an input, we can estimate the probability that a given route has a certain number of students on it on a given day (by approximating an underlying Poisson binomial distribution as a normal distribution with the same mean and variance). Adding the logs of these probabilities gives a log-likelihood of observing the entire bus ridership data set, and maximizing this log-likelihood gives an estimate of ridership probabilities by zip code.

Unfortunately, after consulting with LAUSD, we decided that our output was too inaccurate to use. Some zip codes have many students who ride many different buses, whereas others have only a few students that only ride a few buses. Our log-likelihood maximization gave a result in which the more-populous zip codes have a ridership probability that is roughly equal to the citywide mean. The less-populous zip codes can have either very small or large probabilities, depending on how the more-populous zip codes impact the few routes they are in. If the more-populous zip codes' estimated probabilities result in ridership overestimates (respectively, underestimates) for these few routes, then a less-populous zip code is assigned an estimated probability near 0 (respectively, 1).

Ultimately, we decided that our routing program should accept ridership probabilities at the school level as part of its input data. LAUSD expects to be able to come up with estimates of such data for the next time that they do their routing. As we do not possess such data, we have no meaningful tests of the impact of this feature.

Special Needs

About one quarter of bus-eligible LAUSD students attend special-education schools. LAUSD tries to avoid using mixed-load routing for special-needs situations, and it also tries to avoid having students with special needs on buses with students without special needs. We will refer to the latter as "magnet students," although some of them have bus eligibility through non-magnet programs (such as through diversity programs). Therefore, we can consider special-needs students separately from magnet students. We discussed some of the more common special needs with our LAUSD partners and settled on seven relevant types of needs:

1. Need for a wheelchair, which requires additional pickup time, a bus equipped with a lift, and a bus with sufficient space for wheelchairs.

2. Need for a lift bus and additional pickup time due to ambulatory difficulties, but no need for wheelchair space.

3. Need for a reduced route-time limit.

4. Need for the presence of a supervisor (other than the driver) on a bus.

5. Need for an individual supervisor or health-care assistant.

6. Need for a medical machine, which requires a student who needs one to sit in the back of a bus so that students behind them do not fiddle with its controls.

7. Need to be the final student picked up on a route.

These needs complicate several facets of routing. In particular, they make bus assignment more difficult. Several types of students need additional space for equipment or for additional adults on a bus. The buses that are used to pick up these students are challenging to consider, as they have modifiable configurations. For example, in some buses, it is possible to change the number of wheelchair spots that are available to make room for other students. We attempted to perform bus assignment, but we only possessed limited information about the possibilities for each bus, and LAUSD felt that our program was unable to properly evaluate the feasibility of bus assignments for these routes. After some discussion, they suggested that we not assign buses to special-needs routes, as they felt that it was not realistic to mathematically define the possible configurations for these buses in a usable way. Therefore, bus assignment for special-needs routes will need to be done manually by LAUSD.

With this in mind, needs 4 and 5 above are no longer relevant to our routing algorithm. We still must consider needs 1 and 2, because they require longer pickup times. We also must consider need 6, as there cannot be too many students with this need on one route. Needs 3 and 7 are unrelated to bus assignment, so their impact on routing is unaffected if we do not perform bus assignment. After discarding bus routing, it was fairly straightforward to incorporate needs 1, 2, 3, 6, and 7 into our routing program. In our project, we focused on optimizing routes for magnet students, because it requires fewer assumptions and may have a better chance of being useful to LAUSD. However, we did still make the special-needs routing available to LAUSD in our program.

Bell Schedules

When considering the feasibility of a mixed-load route, it is necessary to consider whether its list of schools to visit is feasible. This requires consideration of time. If students arrive at a school too early, no staff will be there to receive them. If they arrive at a school too late, they may miss the beginning of the first class period. Typically, it is acceptable to drop off students at a school between 30 minutes before the school bell and 10 minutes before the school bell (inclusive). Some schools are exceptions, and our program can accept custom time windows for schools in its input specification.

Our program determines feasibility of a route with respect to bell schedules by going through the route's school list one school at a time and keeping track of the interval of possible times at which the bus may arrive at a school. Assuming that the first school does not have a custom time window, the interval for the first school is $[t - 30, t - 10]$, where t is the bell time (in minutes) of the school. To compute the interval for the next school, our program has to intersect the time window for the next school with the times at which the bus may arrive at the next school (based on the interval for the previous school). If this interval is empty for any school, the list of schools to visit (and hence the route) is not feasible with respect to the bell schedules.

Routing Algorithms

Mixed-Load Post-Improvement Procedure (Park, Tae, & Kim, 2012)

We sought algorithms that are capable of producing mixed-load bus routes. One algorithm from the literature (Park, Tae, & Kim, 2012) focuses explicitly on this goal. It is a post-improvement procedure, so one can apply it to any route plan and to attempt to produce an improved route plan. In this case, a route plan is judged to be "improved" if it consists of fewer routes than the original plan. At worst, the procedure returns the original route plan. This objective ignores mean ride time, so it differs from the mixed objective in our application.

In Park, Tae, and Kim's post-improvement procedure, one considers each route in sequence. Their procedure applies a "route-removal-attempt process" to each one. When considering a route, this process attempts to delete it without having to add any new routes (to thereby reduce the number of routes). Suppose that morning route r_1 of n routes visits student groups s_1, s_2, ..., s_k and also visits some schools. Park, Tae, and Kim's post-improvement procedure attempts to add s_1 to routes r_2, r_3, ..., r_n and stops early if it finds a route to which it can add s_1; we will call this route a "receiving route." The attempt to add a student group to a receiving route always involves checking whether there exists an insertion of the student group that does not violate the ride-time limit. If the student group's school needs to be added to the receiving route, it is also necessary to check whether the additional travel time violates the ride-time limit and whether the new combination of schools is infeasible because of school bell schedules. For the LAUSD routing application, we also needed to check some additional items, such as whether the students at student group s_1 have sufficiently similar ages to the students on the receiving route and whether the bus that is assigned to the receiving route has sufficient capacity for the additional students.

If the move of student group s_1 to a different route is successful, one then seeks a receiving route for student groups s_2, ..., s_k until either no receiving route exists for a student group or one has moved all of the student groups. (We have not experimented with different orderings of the student groups, but we do not expect different ordering choices to perform significantly worse or significantly better than any other.) If one moves all of the student groups, then r_1 no longer picks up any students, so it can be deleted. This reduces the number of routes. If one cannot move student group s_j for some j, then one returns each of the student groups s_1, s_2, ..., s_{j-1} to r_1. This completes the route-removal-attempt process for r_1. Additionally, this process has determined whether it is possible to remove r_1 by reassigning its student groups to other routes. One repeats the route-removal-attempt process for all routes in some order. In practice, we found sorting the routes in ascending order of the number of students that are assigned to the route to be a successful approach.

Our first implementation of Park, Tae, and Kim's algorithm was as a postprocessing improvement procedure after we first applied a fairly simple single-school routing algorithm. Although this post-improvement procedure was successful at reducing the number of routes relative to the initial single-school routes, the resulting mixed-load routes seemed to be inefficient, as many of them had only one student group for a school or did not include a student group that seemed natural for the route to pick up. We have possible explanations for these observations.

- In many situations, mixed-load routes have several student groups for one of two schools, but only one student group for the other. We believe that this occurs because the postprocessing procedure moves one student group at a time to somewhere that it fits. However, it typically is desirable for a route to have multiple student groups for each school. To understand why, consider a route that visits two schools, *A* and *B* (in that order). The bus spends time traveling from *A* to *B* instead of picking up students, so because of the ride-time limit, mixed-load routes can spend less time picking up students than routes without mixed loads. This is a cost of having multiple schools on a route. However, if a routing algorithm is good, each student group of school *B* that one adds to the route should increase the benefit of having school *B* on the route, because otherwise the algorithm would not put the student group on this route. If the algorithm has only one student group for school *B* on this route, the benefit of having *B* on the route is unlikely to outweigh the cost. However, if it has several for school *B* on this route, the benefit of having *B* on the route is likely to outweigh the cost. The same argument applies to school *A*. Therefore, as this example illustrates, it typically is preferable to have multiple student groups for each school on a mixed-load route.

- Any route that results from Park, Tae, and Kim's post-improvement procedure has what we will call the "super-route property," which means that it is guaranteed to visit all student groups of some input route. Suppose that some efficient route in an (unknown) optimal route plan visits three student groups from each of school *A* and school *B*. If an input route visits all three of those student groups for school *A* and one additional faraway student group for school *A*, there may not be time to add the student groups for school *B* to the route. However, the initial routing procedure may not have a reasonable way to avoid this situation, because it is doing single-school routing and—due to the super-route property—the post-improvement procedure cannot fix this problem.

These observations do not condemn Park, Tae, and Kim's algorithm; instead, they reflect an input route plan that is not sufficiently good. Because Park, Tae, and Kim's algorithm is a post-improvement procedure, it depends on the quality of the route plan with which one starts before applying it, and our initial attempt at generating an input route plan was poor. We decided that it was necessary to generate mixed-load routes in the first phase of routing, instead of generating single-school routes and then modifying them. We then use Park, Tae, and Kim's post-improvement procedure to improve route plans that are generated by other algorithms.

Other Post-Improvement Procedures

As we discussed previously, we use the mixed-load post-improvement procedure from (Park, Tae, & Kim, 2012) on each of the other three routing algorithms that we examined. We will discuss these algorithms in our sections on Savings-Based Route Generation, School Groups, and Pickup Valuation. For each of our algorithms, we also employ two additional post-improvement procedures in a manner that we will specify after discussing the procedures. These are the procedures:

- For each route, we use the so-called "two-opt" procedure of (Croes, 1958) on the list of student groups that are picked up by a route. This procedure is a basic improvement procedure that is used commonly in many different routing problems. In the two-opt procedure, one rewires a path that crosses itself to remove the crossing. In our routing problem, one can view this procedure as reversing the order of a contiguous sublist of the list of student groups in a route and checking whether the overall distance decreases. (By a "contiguous" sublist, we mean that no entries are skipped, so the sublist {4,5,6} of the list {1,2,3,4,5,6,7,8} is contiguous, but the sublist {4,5,7} is not contiguous.)

- We use a "greedy-moves" procedure that searches for instances in which one can move a student group from one route to another such that one decreases the mean student ride time. This procedure is slow if there are too many student groups. Using the current version of our program on LAUSD's 2018–2019 magnet data (which has 1102 student groups), this procedure required 57 seconds on an Intel Core i7-8750H machine. For routing students with special needs, many students are picked up at home, so the number of student groups is much larger, even though the number of students is smaller than is the case for other students. Therefore, we turn off the greedy-moves procedure when we route students with special needs, but we leave it on for other situations.

In total, we have three total post-improvement procedures at our disposal for non-special-needs routing and two such procedures for special-needs routing. As our postprocessing for our algorithms, we cycle these procedures—first using the greedy-moves procedure, then the two-opt procedure, and finally the mixed-loads procedure—until none of them results in an improvement. Applying these procedures in a different order and then cycling them usually produces a different route plan, but we have not tested whether one order is better than another.

Savings-Based Route Generation (Campbell, North, & Ellegood, 2015)

The savings-based route-generation algorithm is based on a classical algorithm (Clarke & Wright, 1964), known as "savings-based routing," for more general vehicle-routing problems. A recent book (Campbell, North, & Ellegood, 2015) discussed their adaptation from (Campbell, Ellegood, & North, 2013)) of savings-based routing to mixed-load school bus routing. The approach of savings-based routing is based on an idea that is similar to agglomerative clustering. In the algorithm from (Campbell, North, & Ellegood, 2015), each of the S student groups begins on its own route. For each pair of routes, r_i and r_j, one then checks whether concatenating the list of student groups in r_i and the list of student groups in r_j (and combining their school lists) results in a combined route that is feasible and has a shorter length than the sum of the lengths of r_i and r_j. If so, one records the amount of time that is saved in entry (i,j) of an S-by-S matrix A (in which one assigns a large negative score to infeasible combinations). After one fills the matrix entries, the algorithm finds i and j to maximize A_{ij} and then, if A_{ij} is positive, combines routes i and j. It updates the entries in the rows and columns i and j and continues making combinations and updates until A has no positive entries. Our implementation of savings-based routing from (Campbell, North, & Ellegood, 2015) works fairly well. Specifically, according to our measurements, it produces slightly fewer routes (see Table 1) than LAUSD's existing route plan.

In some respects, the savings-based routing approach of (Campbell, North, & Ellegood, 2015) is similar to that of Park, Tae, and Kim's mixed-load post-improvement procedure. It tries to eliminate routes by moving their student groups onto other routes, and this occurs in such a way that new routes always contain other routes as subroutes. However, this savings-based approach starts from a very poor initial route plan (specifically, from a route plan with one route per student group). There are two important differences from our implementation of the method of (Park, Tae, & Kim, 2012) that help explain its success:

- Although the initial route plan is very poor in absolute terms, it is also extremely flexible, as any possible route is a combination of the initial routes. Such flexibility is absent when one starts with a nontrivial initial route plan.
- The aforementioned flexibility is dangerous, as Park, Tae, and Kim's mixed-load post-improvement procedure alone makes the first feasible combination that it finds, rather than looking for good combinations. However, because a savings-based approach always chooses a combination that maximizes the savings, it is able to take advantage of this flexibility to produce efficient mixed-load routes.

School Groups

One algorithm (which we call "School Groups") that we developed was inspired by the fact that, in their routes, LAUSD treats small groups of nearby schools as individual schools. Their routing team makes many mixed-load routes for schools in these groups, but it makes very few mixed-load routes across these groups. In our School Groups approach, we initially route each school on its own. We then consider possible ways to combine two schools into one group, and we see whether routing for that pair results in fewer total routes than routing the schools individually. We repeat this process until we do not find any additional combining to further reduce the number of routes.

It is necessary to choose how to create routes for a small number of schools. In theory, one can do this with any mixed-load routing algorithm. In practice, however, because one will try many combinations of schools (including all pairs of schools), it is necessary to consider the computational efficiency of algorithms. We use an algorithm that is fast and simple. Specifically, we collect the list of student groups for these schools, create a nearest-neighbor path that goes through all of these student groups, divide these paths into multiple routes that satisfy the ride-time limit, and then try to combine the small routes that sometimes remain after dividing these paths. This algorithm also produces results of similar quality to LAUSD's existing routes.

Like savings-based routing, our School Groups approach also uses ideas from agglomerative clustering. However, instead of combining routes, the latter combines schools. An upside of our School Groups approach is that it produces explicit groups of schools, which helps LAUSD from an administrative standpoint (based on our discussions with them).

Pickup Valuation

Our final method (which we call "Pickup Valuation") is based on a notion of value for choosing pickups of student groups. Suppose that we are building a route and are deciding between two choices of which student group to add. Suppose further that each of these two student groups is 5 minutes out of the way of the route. If one student group is far from the school and the other student group is near the school, it makes sense to add the former student group, because it is less likely that other routes for that school will go near that student group. Based on this idea, our method builds one route at a time, deciding which student group to add to a route based on a combination of how much time it would add to the route and a utility score (i.e., a value of the benefit) for choosing a pickup at that student group. The most important contribution to this utility score is the distance from the student group to the school, but there is also a smaller contribution that depends on the distance from the student group to other student groups for the same school. This Pickup Valuation approach produces a smaller number of routes than the existing LAUSD routes, although the mean student ride time is slightly longer than that of the LAUSD routes. This approach seems to perform the best among those that we have considered, so we used it to test the impact on route efficiency of changes in routes and policies (see Figures 2–3).

Because our Pickup Valuation approach was the most successful of the employed approaches, we provide a more formal description of it than we have for the other approaches. Throughout our description, an "*a*" always represent a school and an "*s*" always represents a student group.

Let *S(a)* be the set of all student groups for school *a*. Fix some number *t* of minutes; the value of *t* is typically 10–12 for LAUSD bus routing. For each school *a*, let *N(a,t)* denote the set of schools that are at most *t* minutes from *a*. (Note that school a is part of the set *N(a,t)*.)

First, repeat the following procedure until all student groups are routed:

1. Choose some student group *s**. (We describe the choice of student group in the "Pickup Valuation: Specific Choices" section.) Instantiate a route *R* that visits only *s** and its school *a**. Mark *s** as routed. Let $S=\bigcup_{a \in N(a^*,t)} S(a)$. That is, *S* is the set of all student groups for schools near *a**. Then repeat the following:

1.1 For each unrouted student group *s* in *S*, approximate the optimum time cost c_s of adding student group *s* to route *R*. This may require adding *s*'s school if it is not already in *R*. If so, the program checks time costs for all permutations of the schools which are valid with respect to bell schedules. (See the subsection "Bus Routing in Los Angeles" for a discussion of LAUSD bus schedules.) We assume that a bus visits all student groups before it visits any schools. If adding student group *s* would exceed the ride-time limit, define c_s to be infinite.

1.2 If no student group has a finite cost, skip to step 2, because we cannot add more student groups to the route.

1.3 For each student group *s* in *S*, compute the "value" v_s of picking up student group *s* for some definition of value. The idea is that it may be more beneficial to pick up some student groups than others if their time costs are similar. (We give our definition of "value" in the "Pickup Valuation: Specific Choices" section.)

1.4 For some function $f(v,c)$ that is increasing in v and decreasing in c, pick up a student group s_{new} that maximizes $f(v_s,c_s)$ across *s* in *S* with finite c_s and such that $f(v_s, c_s)$ exceeds some constant parameter $score_{min}$. (We choose the function $f(v,c)=v-c$, which is simple and seems to perform well.) If no such s_{new} exists because $f(v_s, c_s)$ does not exceed $score_{min}$ for any *s*, go to step 2. Otherwise, add s_{new} to the route. Remove s_{new} from *S* and mark it as routed. Return to step 1.1. (We require $f(v,c)$ to be strictly decreasing in c so that, when considering two student groups with equal value and different costs, one adds the student group with a lower cost to the route if one adds either student group to the route. For an analogous reason, we require $f(v,c)$ to be strictly increasing in v.)

2. Add *R* to the set of routes.

Once we have routed all student groups (by repeating steps 1 and 2 until this is the case), our program runs the postprocessing, assigns buses, and runs the postprocessing again. Running our postprocessing both before and after bus assignment seems to perform better than running it only once. Typically, 70–80% of our program's computation time comes from running the postprocessing.

Pickup Valuation: Specific Choices

The choice of which student groups to use to initialize routes in step 1.1 is important. In our observations, different choices of how to do this can yield variations of more than 5% in the final number of routes.

In our implementation of our Pickup Valuation approach, we sort all student groups in decreasing order of travel time from each student group to its school. We call this list the "student-group-processing list." Whenever we reach step 1.1, we choose the first unrouted student group in the student-group-processing list. This approach is sensible because it seems reasonable to pick up student groups that are "along the way" from other student groups to their schools, and it is more likely that student groups will be along the way if we begin from a faraway student group.

In the implementation of the Pickup Valuation method in the program that we provided to LAUSD, a user specifies an amount of time that they wish the program to run. After the program selects parameters, runs our Pickup Valuation method, runs our postprocessing, assigns buses, and runs our postprocessing again, it uses any remaining time to try to find better route plans. The program does this by perturbing the student-group-processing list and running the steps that we listed in the previous sentence. If the perturbation yields an improvement, we keep it; otherwise, we discard it. If we allow the aforementioned steps to run for a sufficiently long time (by considering a larger number of student-group-processing lists), we were able to obtain an improvement of over 1% in comparison to only running them once. See the final row of Table 1.

The definition of the "value" of a student group is also important. We want to reward picking up student groups that are far from their schools and picking up student groups that are far from other student groups for their schools, because it may be costly to add these student groups to routes. Let $t_1(s)$ denote the travel time from a student group s to its school, and let $t_2(s)$ denote the travel time from student group s to the nearest location among its school and its school's unrouted student groups. We define the value of a student group to be a linear combination of these quantities: $v(s)=b_1t_1(s) + b_2t_2(s)$. Note that $t_2(s)$ may change when one routes another student group.

In our Pickup Valuation algorithm, we have not yet specified three important parameters: $score_{min}$, b_1, and b_2. One chooses these parameters during an initial parameter-selection phase, in which our program runs without postprocessing on various random triples (with each entry sampled uniformly from an interval, where we chose the interval based on empirical testing) for these three parameters and chooses the best triple. It runs without postprocessing because it would significantly slow down this parameter-selection phase.

We define the "best triple" in terms of the mixed objective of number of routes and mean ride time. One of the data files that is used by our program has an input parameter w that specifies the weighting of the mean ride time relative to the number of routes. The program only uses w for determining this triple; it selects a triple that minimizes the linear combination of these quantities with the weights from the input file. The quantity $score_{min}$ controls the trade-off between route length and mean student ride time. Increasing the value of $score_{min}$ causes the maximum value of $f(v_s, c_s)$ to drop below $score_{min}$ at least as soon (and possibly sooner) than it would with a smaller value of $score_{min}$. Because of step 1.4, this causes route construction to terminate at least as soon (and possibly sooner) as it would with a smaller value of $score_{min}$. We chose the interval from which we sample $score_{min}$ to be wide enough that route construction often terminates earlier for the largest possible value of $score_{min}$ than it would for the smallest possible value of $score_{min}$. Increasing the value of $score_{min}$ results in more routes, which are shorter.

Overall Program Flow

To recap, the overall program flow of the route-generation algorithm for magnet students in the implementation in the program that we gave to LAUSD works as follows:

1. User data selection: The user chooses which input files they want to use to generate routes. One of these input files allows the user to specify the amount of time that they want the program to run.

2. Parameter selection: To determine parameter values, run the Pickup Valuation procedure many times with no bus assignment or postprocessing.

3. Create a student-group-processing list: Sort the student groups in decreasing order of distance to their schools. Whenever it is time to begin a new route, the program uses the earliest unrouted student group in this student-group-processing list.

4. Run the Pickup Valuation procedure, and then apply the postprocessing to the resulting routes. Run the bus assignment procedure. Apply the postprocessing again. If the resulting route plan is not the best-known route plan, undo the most recent perturbation of the student-group-processing list; if it is the best-known route plan, store the route plan.

5. If sufficient time remains for another iteration of step 4, perturb the student-group-processing list and return to step 4. Otherwise, return the best route plan that has been found.

Discussion and Results

To compare our route plans to the existing LAUSD route plan, it is necessary to make some assumptions. The most dubious one is related to overbooking. Because LAUSD has knowledge about ridership probabilities, they overbook their routes. Our program is capable of accepting ridership probabilities at the granularity of schools, but this information is not currently available. It is almost meaningless to compare a route plan that overbooks to a route plan that does not overbook. Without overbooking, the buses that are used by LAUSD in the existing magnet routes would not be sufficient to pick up all of the magnet students, even if ride-time limits were ignored altogether, so no routing algorithm that does not overbook can obtain a result that is close to LAUSD's actual number of routes.

Because LAUSD expects to make its school-wide ridership probability estimates available internally, our program will be able to overbook when their routing team needs to run it. However, as part of our project, we sought to make comparisons with the existing LAUSD routes. To do so, we interpreted LAUSD's existing routes as lacking bus assignments and ran them through our own bus-assignment procedure. We then used these routes, which no longer implement overbooking, for our comparisons. These comparisons are not fair to LAUSD's routes, because our choices during the development of our program sought to improve the measurements for the bus-assigned version of the routes (where the bus assignment process is defined by our program), whereas LAUSD's routes were not developed with our bus-assignment procedure in mind. Therefore, although the numbers in Table 1 appear to indicate that our program is outperforming LAUSD's existing routes, the situation is not clear. It is essentially impossible to quantify the impact of this approximation without access to the ridership probability data.

There are other, smaller respects in which the comparisons in Table 1 are not fair. As we discussed previously, some LAUSD routes have both elementary-school and high-school students. This is rare in our program, because we allow such mixtures only when such students share a student group. In other words, our program is adhering to stricter restrictions than LAUSD's routes, disadvantaging it relative to the existing routes. Similarly, the mapping data (OpenStreetMap contributors, 2018) that we used with the aid of the Open Source Routing Machine (Luxen & Vetter, 2011) suggests that some existing LAUSD routes are infeasible with respect to the ride-time limit (occasionally by tens of minutes). This may be due partially to discrepancies between our mapping data and LAUSD's mapping data, but one can approximately infer LAUSD's expected ride times by looking at pickup times and school bell times, and it seems that some of their routes are also significantly longer than the limits that are supposed to be imposed. This is another respect in which our program is disadvantaged in our comparisons. Finally, LAUSD occasionally adds a special-needs student to a magnet route in situations in which it is particularly difficult to incorporate that student into a special-needs route. Our program does not include these students in the magnet routing, so this disadvantages LAUSD.

Overall, although the Pickup Valuation numbers in Table 1 appear better than the numbers for LAUSD's existing routes, we are unable to conclude whether our program's routes improve on LAUSD's existing routes. However, it does seem likely that our program's routes can provide a good starting point for LAUSD's routing team, especially after ridership probability estimates become available. We hope that they will use our program as a starting point and then tweak the resulting routes to fix any issues with them.

Implementation and Final Product

Based on the data in Table 1, we chose to use the Pickup Valuation method in the program that we gave to LAUSD. Additionally, we developed a user interface so that non-technical members of their routing team can also use the software. Users navigate to the data files in their file system and click a button to generate a route plan. Route generation finishes in under half an hour for LAUSD's problem size even on modest hardware. (Allowing it to run for roughly eight additional hours yielded routes with over a 1% improvement in the mixed objective.) Users can view these routes with some useful metadata within our program, or they can click a button to open a route in Google Maps.

As we have discussed in this report, we expect that the routes that are generated by our program will require tweaking from LAUSD's routing team. They have significant domain knowledge that is difficult (or perhaps impossible) to impart through our program's input files. Additionally, our program has the following issues (which can yield poor or infeasible routes that LAUSD's routing team likely will be able to improve or fix):

- It uses a constant travel-time multiplier to account for traffic, instead of using detailed traffic data. This may affect whether routes exceed ride-time limits.
- It uses only very limited data on student ridership probabilities, which inform LAUSD's policy of overbooking.
- It may not have reliable data on which pairs of schools are acceptable for mixed-load routes. Our program determines acceptability based on ride times and bell schedules, but this determination is not always accurate (especially in the presence of localized traffic near schools around bell times).
- It does not assign buses for special-education ("special ed") routes, as our discussions with LAUSD concluded that there is no reasonable way for our program to define the feasibility of assignment of buses to these routes.

For each of these issues (except for special-education bus assignment), there is some way to account at least partially for the issue by modifying the input data. However, not all users of our program will possess the necessary knowledge to perform such modifications. Therefore, we also included an editing utility as part of the program. This utility allows users to move student groups from one route to another or to change the order of student groups or schools in a route.

Table 1. Comparisons of routing approaches

We compare three algorithms to LAUSD's existing routes (both with and without the use of the postprocessing). We incorporated the first algorithm (Park, Tae, & Kim, 2012) that we considered as a part of our postprocessing. The final row gives statistics for a version of our Pickup Valuation algorithm that repeatedly reruns (until it exhausts the time that is specified by a user) the entire algorithm with small modifications to the initial conditions and keeps those modifications if the resulting route plan is better.

Routing Approach	Number of Magnet Routes	Mean Student Ride Time (minutes)
Manually Generated by LAUSD	656	25.6
Route Combining (no postprocessing)	704	24.5
Route Combining (with postprocessing)	650	25.6
School Groups (no postprocessing)	740	26.4
School Groups (with postprocessing)	647	26.0
Pickup Valuation (no postprocessing)	664	25.7
Pickup Valuation (with postprocessing)	640	25.8
Pickup Valuation (with postprocessing, and after rerunning the algorithm with different initial conditions to search for further improvements)	629	25.7

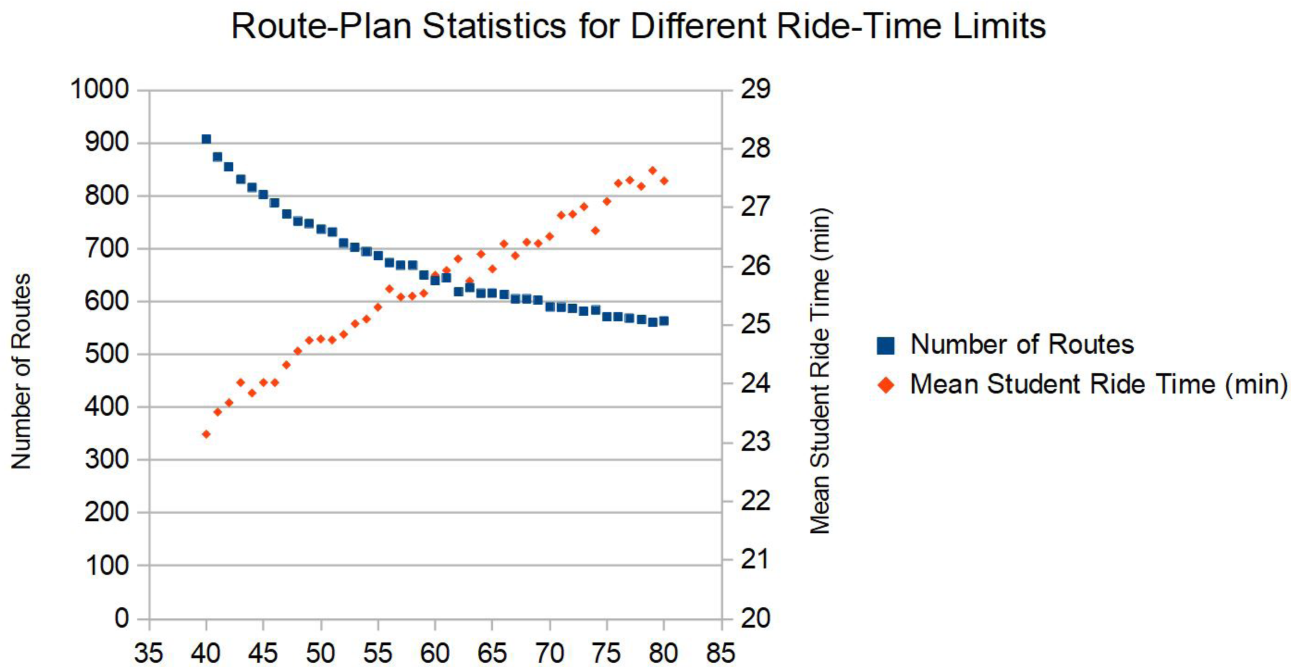

Figure 1. Varying the time limit of rides

LAUSD currently allows routes in which students can ride buses for up to 60 minutes (not taking traffic into account). We evaluated the impact of changing the maximum permissible ride time.

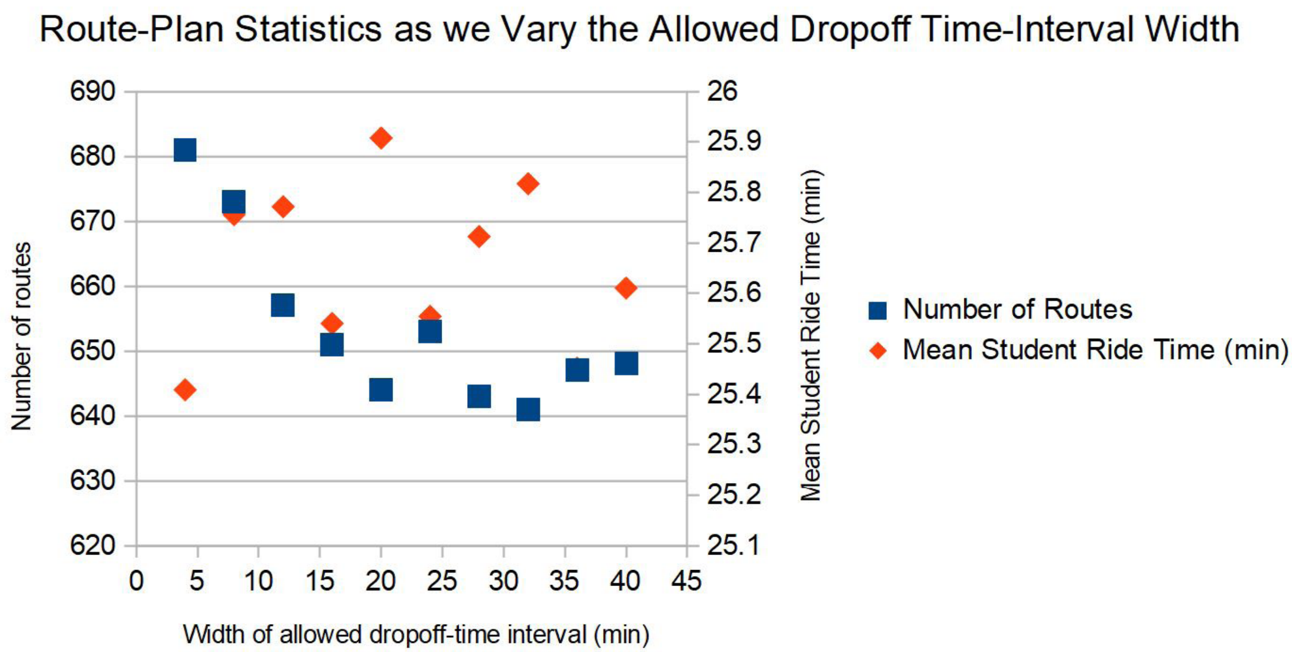

Figure 2. Varying the width of the dropoff time interval

LAUSD currently allows students to be dropped off between 30 minutes and 10 minutes before their school's bell time. We evaluated the impact of changing the length of this interval. To better test this change, we removed a constraint on the maximum distance between schools in a single mixed-load route.

References

Campbell, J. F., Ellegood, W. A., & North, J. W. (2013). Optimizing school bus routing in Missouri. Final report for the University of Missouri Research Board, 7 August 2013.

Campbell, J. F., North, J. W., & Ellegood, W. A. (2015). Modeling mixed load school bus routing. In A. Klose, M.G. Speranza, & L. N. V. Wassenhove (Eds.), *Quantitative Approaches in Logistics and Supply Chain Management* (pp. 3–30). Cham, Switzerland: Springer International Publishing.

Clarke, G., & Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, *12*(4), 568–581.

Croes, G. A. (1958). A method for solving traveling-salesman problems. *Operations Research*, *6*(6), 791–812.

Desrosiers, J., Ferland, J.A., Rousseau, J. M., Lapalme, G., & Chapleau, L. (1981). An overview of a school busing system. In N.K. Jaiswal (Ed.), *Scientific Management of Transport Systems* (pp. 235–243). Amsterdam, The Netherlands: Elsevier.

Ellegood, W. A., Solomon, S., North, J., & Campbell, J. F. (2019). School bus routing problem: Contemporary trends and research directions. *Omega*. Advance online publication. Available at https://doi.org/10.1016/j.omega.2019.03.014.

Ellegood, W. A., Campbell, J. F., & North, J. (2015). Continuous approximation models for mixed load school bus routing. *Transportation Research Part B: Methodological*, *77*, 182–198.

Luxen, D., & Vetter, C. (2011). Real-time routing with OpenStreetMap data. In *Proceedings of the 19th ACM SIGSPATIAL international Conference on Advances in Geographic Information Systems* (pp. 513–516). Chicago, Illinois, USA: Association for Computing Machinery.

OpenStreetMap Contributors. California dump. Retrieved 6 November 2018 from http://download.geofabrik.de/north-america/us/california.html .

Park, J., & Kim, B. I. (2010). The school bus routing problem: A review. *European Journal of Operational Research*, *202*(2), 311–319.

Park, J., Tae, H., & Kim, B. I. (2012). A post-improvement procedure for the mixed load school bus routing problem. *European Journal of Operational Research*, *217*(1), 204–213.

Toth, P., & Vigo, D. (Eds.). (2014). *Vehicle Routing: Problems, Methods, and Applications*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.