Li, Yang, and Kim, A.

Allocating and scheduling resources for a mobile photo enforcement program.

AUTHOR POST PRINT VERSION

Li, Y., & Kim, A. (2021). Allocating and scheduling resources for a mobile photo enforcement program. *Transportation Research Part C: Emerging Technologies*, 125, 1–31.

https://doi.org/10.1016/j.trc.2021.103000

Allocating and scheduling resources for a mobile photo enforcement program

Yang Li*, Amy M. Kim

Department of Civil and Environmental Engineering, University of Alberta

*Corresponding author. Address: 6-362 Donadeo Innovation Centre for Engineering, 9211 - 116 Street NW, Edmonton, Alberta, Canada T6G 1H9.

E-mail addresses: li18@ualberta.ca (Y. Li), amy.kim@ualberta.ca (A. Kim)

Abstract

We present a scheduling model for an urban mobile photo radar speed enforcement (MPE) program. An MPE program utilizes automated photo radar technology to capture speed limit violators, towards an aim to reduce speeding and thus, improve traffic safety. We propose a binary quadratic program that determines where (by location visit tasks) and when (in shifts) to send enforcement resources (operators and equipment) over a month-long schedule. The aim of this program is to minimize violations of enforcement time halos (and thus, efficiency losses) in attaining traffic-safety focused program goals. We solved this problem using a combined Dantzig-Wolfe and column generation optimization approach, after exploring several different types of commonly-used solution methods. The model is applied to a currently operational MPE program in the city of Edmonton, Canada. Using a five-day enforcement time halo, we find that our model results improve resource utilization by 24% over historical program deployment. The scheduling model is the final step of a larger unified MPE program framework that systematically and efficiently connects high-level urban traffic safety goals down shift-level allocations of highly limited enforcement resources. It provides a data-driven and transparent framework that both contributes to cities' pursuits of Vision Zero and combatting negative public perceptions of MPE programs. Additionally, the framework can be easily adapted for, and transferred to, other automated traffic enforcement technologies across jurisdictions.

Keywords: mobile speed enforcement program scheduling, resource allocation, large-scale integer nonlinear programming, mobile photo radar speed detection.

1 INTRODUCTION

We present a scheduling model for an urban mobile photo radar speed enforcement (MPE) program. MPE program planners must decide how to best deploy program resources to achieve program goals, which should align with a city's overall efforts to improve traffic safety. In an MPE program, the automated photo radar technology used to detect and photograph speed violators is not fixed at specific locations. Instead, it is housed in vehicles that operators position at roadside locations throughout a city. Our scheduling model determines where and when to send these resources based on a pre-specified set of urban safety-focused program goals, while minimizing repeat visits that have less efficacy due to an enforcement time halo effect.

Our scheduling model addresses the challenges faced by MPE program managers throughout the world in conducting successful MPE programs. First, enforcement resources (human operators and equipment) are often highly limited. Second, the public associates such programs with municipal revenue generation rather than the goal of improving traffic safety, leading to highly negative public perceptions. Third, there are no systematic methods in place to match limited resources to roadway locations in most need of enforcement. In the Province of Alberta, Canada, it was determined that the 27 MPE programs in operation contributed only 2% towards collision reduction in the province, and nearly 70% of the public opposed such programs (Alberta Transportation, 2018). Although it is documented that some MPE programs are able to achieve collision reductions of 20-50% after implementation (Berkuti & Osburn, 1998; Coleman et al., 1996; Dreyer & Hawkins, 1979), there has been little attention given to MPE resource allocation program design. In fact, there are no systematic, transparent resource scheduling models developed and applied for this highly specialized but widely applied technology.

We solve the MPE deployment problem in three steps, of which the first two are documented in previous papers. The first step involves identifying metrics that serve as quantitative proxies of an MPE program's overall safety goals (Li et al., 2016), while the second step assigns MPE resources to neighborhoods across a city based on the multiple aforementioned safety goals (Li et al., 2019). This paper documents the third and final step, which involves scheduling MPE resources to individual locations in operator shifts. To achieve this, we develop a binary quadratic programming model that minimizes the total cost of scheduling operators working over a planning horizon to complete location visit tasks, previously identified in the second stage. Costs are incurred when enforcement time halos are violated. As our realistic model instances are very large, we solve our problem by first reformulating it using the Dantzig-Wolfe decomposition and column generation approaches. We applied this approach, developed to solve aviation crew scheduling problems, given that other methods did not offer feasible solutions for our problem size.

First, we present the design of a model that minimizes time halo violations in scheduling enforcement site visits to operators – both optimizing resource efficiency while retaining (real-world) operator flexibility within shifts. Second, we explore solution methods for this problem that is significantly larger than most scheduling problems (for which a variety of solution methods exist), but also, smaller than the aviation crew scheduling problem. The proposed framework provides solutions that make efficient use of limited resources in pursuing urban traffic safety goals, which can also be easily communicated to the public.

2 BACKGROUND

2.1 MPE Deployment Framework Overview

This paper focuses specifically on the MPE resource scheduling problem, the final step in a larger MPE program framework (Fig. 1) that is briefly presented here for context and background. As shown in Fig. 1, the framework consists of three stages: 1) goal quantification, 2) spatial allocation, and 3) temporal scheduling.

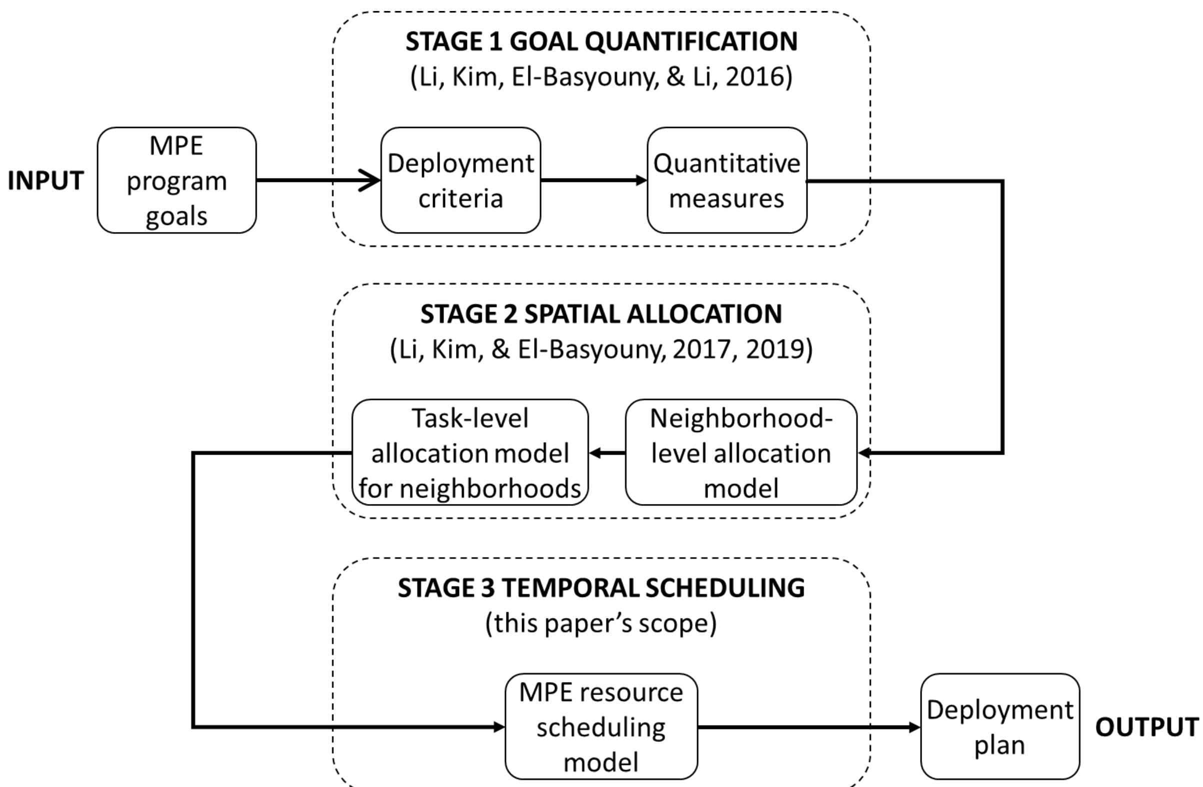

Fig. 1 MPE deployment framework workflow.

The first stage converts qualitative, pre-defined MPE program goals to quantitative deployment criteria and measures (Li et al., 2016). The following deployment priorities associated with MPE program traffic safety goals were identified from a variety of government guidelines on the management of automated speed enforcement schemes (photo radar being one of several technologies used in such programs).

- High collision sites: To reduce accidents;

- High speed violation sites: To reduce speeding vehicles;
- School zones, construction zones, high pedestrian volume sites, and sites with community speeding complaints: To improve pedestrian and cyclist safety.

Metrics addressing the above criteria were calculated using data from the City of Edmonton (discussed in Section 5.1) and input to the second stage, specifically to a neighborhood-level resource allocation model (Li et al., 2019). This model uses multi-objective optimization to output a set (Pareto front) of MPE resource allocation solutions that consider multiple objectives as tradeoffs. Li et al. (2017) then form location visit tasks (sets of enforcement locations) in neighborhoods chosen for enforcement, and distribute resources to those tasks using weights corresponding to the input metrics. Thus, optimal spatial allocation of MPE resources is completed (comprising Stage 2 of Fig. 1) according to the user-input traffic safety goals of Stage 1.

Stage 3 involves creating a schedule for the task-level MPE resource allocations. Because enforcement tasks can only consist of sites within a single city neighborhood, which cover relatively small areas, the deployment plan output ensures that operators will not travel extensively between sites in a single shift. This paper focuses on the development of the Stage 3 scheduling problem, solution method, and a deployment plan generated for our City of Edmonton case study. The following section (2.2) provides a review of the background literature on scheduling for traffic law enforcement.

2.2 MPE Scheduling Problem

The existing research on MPE scheduling focuses on increasing a program's unpredictability over time, and thus, catching a greater number of offending drivers. A randomized schedule is one where resources are assigned to locations and times randomly, without explicit consideration for site characteristics and visit frequencies, and the frequency and date/times of enforcement activities are not necessarily constant. The purpose of such a scheduling strategy is to make enforcement seem as unpredictable as possible to the public. A fixed, or "static," schedule is defined as one where enforcement operators visit sites at a predictable frequency or at the same time on the same days (for instance, each Tuesday and Thursday mornings). Clearly such an arrangement becomes, over time, easily predicted by the public. Better road safety outcomes are expected with randomized scheduling compared with a fixed scheduling scheme: 30% greater reduction in collisions (Leggett, 1997), and 33% greater reduction in speeding vehicles (Kim et al., 2016). By definition, however, this strategy retains inefficiencies, which is of serious concern when MPE enforcement resources are so limited.

Optimization techniques have been used to improve the efficiency of traffic enforcement resource usage. Yin (2006) introduced a min-max optimization model that efficiently assigns limited traffic service patrols to freeway segments. However, the model only considers the case where patrols spend the longest travel time in handling incidents, and is therefore has nothing to do with time allocation. Adler et al. (2014) developed a police patrol scheduling model (using binary integer programming) that minimized instances of police patrol shifts being continuously allocated to a location under a time halo effect. The time halo effect is a period of several hours to days after enforcement operations end, when drivers' speeding behaviors are reduced, due to the "memory" of having observed prior enforcement at that particular site (Armour, 1986; Cairney, 1988; Gouda & El-Basyouny, 2016; Hauer et al., 1982; Vaa, 1997). Applying enforcement during a time halo is an inefficient use of resources, and thus, the authors aimed to minimize this occurrence. Adler et al.'s model results in an efficient and varied schedule that avoids 75% of unnecessary sequential visits during a two-shift time halo. Li et al. (2017) proposed a similar model, with a resulting schedule that ensured 90% of enforcement visits are not repeated for a two-shift (i.e. one-day) time halo. Nonetheless, both the Adler et al. (2014) and Li et al. (2017) models are unable to solve scheduling cases for which time halos exceed 2-3 shifts. Most commercial integer programming solvers (mainly using branch-and-cut, B&C) run out of memory because the formulation of the two binary integer programming models gives trivial LP bounds, causing the number of branch-and-bound nodes to explode as the model parameters increase. As a different model formulation and solution method is required, the next section reviews major approaches to solving large-scale scheduling problems in transportation applications.

2.3 Approaches to Solving Scheduling Problems

Scheduling problems are often complex integer programming problems, and grow more difficult to solve with size. In particular, airline crew scheduling and rostering are the largest staff scheduling problems across all industries (Ernst et al., 2004). An instance of an airline crew scheduling problem can consist of over 85,000 variables (Hoffman & Padberg, 1993).

Both exact algorithms and metaheuristics have been used extensively in solving many different types of scheduling problems (Ernst et al., 2004). Fig. 2 shows prevalent types of integer programming scheduling problems and their solution methods, referencing studies that clearly mention problem size.

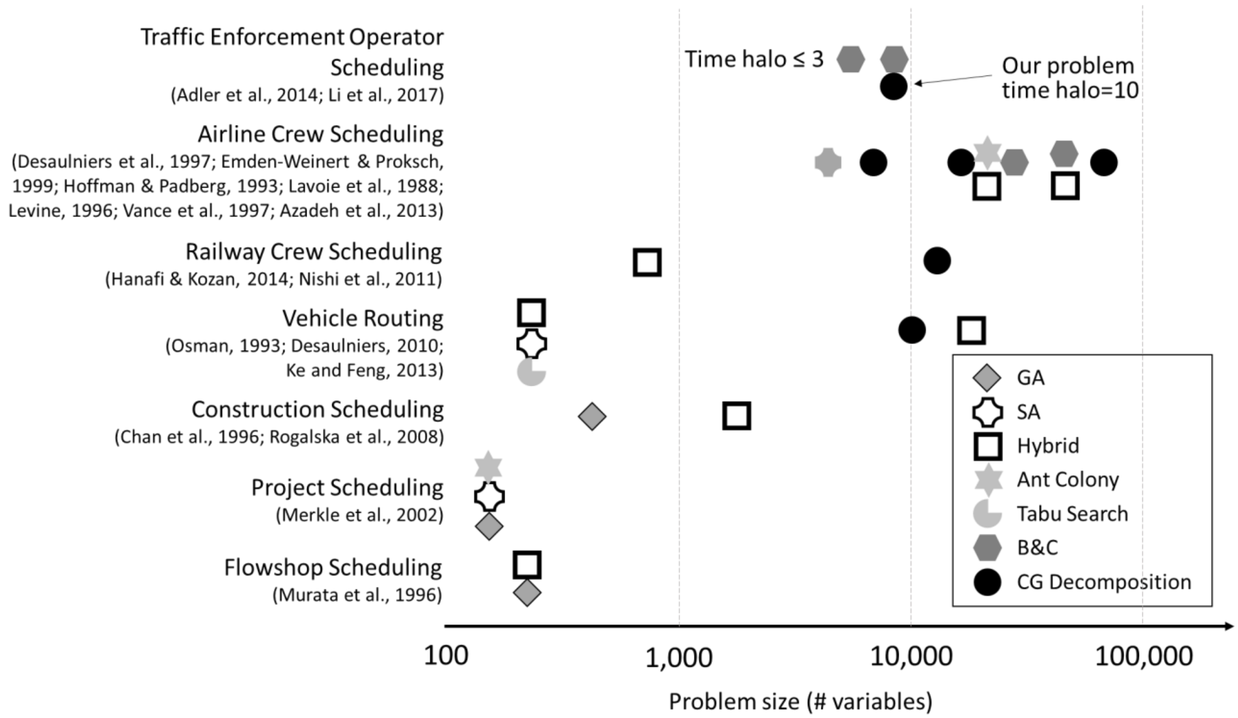

Fig. 2 Overview of scheduling problems and solution methods.

As observed in Fig. 2, genetic algorithm (GA), simulated annealing (SA) and other metaheuristic methods have been used to handle relatively small scheduling problems (less than 1,000 variables). For these problems, hybrid algorithms that add local search on GA and SA have shown higher performance than single metaheuristic algorithms alone (Hanafi & Kozan, 2014; Murata et al., 1996; Rogalska et al., 2008). However, these algorithms have been shown to return solutions that differ by more than 30% from optimal (Hanafi & Kozan, 2014; Merkle et al., 2002).

With larger problems (4,600 to 100,000 variables), the use of exact algorithms increases. These mainly include branch-and-cut (B&C) and column generation (CG) based decomposition methods (Desaulniers, 2010; Desaulniers et al., 1997; Hoffman & Padberg, 1993; Lavoie et al., 1988; Nishi et al., 2011; Vance et al., 1997). Although metaheuristics such as SA (Emden-Weinert & Proksch, 1999) and some hybrid algorithms (Azadeh et al., 2013; Ke & Feng, 2013; Levine, 1996) have been employed to solve large vehicle routing and airline crew scheduling problems, Emden-Weinert and Proksch concede that feasible solutions were difficult to find due to a number of reasons. A combined approach has primarily been used to solve the largest airline crew scheduling problems: one first decomposes a schedule using the Dantzig-Wolfe algorithm, and then solves the reformulation using a column generation algorithm (Desaulniers et al., 2002; Ernst et al., 2004). In transportation scheduling problems, a column or row will represent a path connecting arcs and nodes in vehicle routing problems, or an itinerary connecting flights in airline crew scheduling problems. The Dantzig-Wolfe algorithm (1960) redefines a schedule by treating each column (as opposed to each element) of the schedule as a variable. When branching occurs on these variables, the bounds derived by a branch-and-bound method are closer to the optimal value of the problem objective than branching on a single scheduled element (Ernst et al., 2004). When the number of possible variables (columns) is enormous, the column generation approach (Ford Jr & Fulkerson, 1958) is used in combination with the Dantzig-Wolfe algorithm. Like the simplex method, column generation only identifies new columns that can be entered into the problem's variable basis, rather than enumerating all columns.

3 MPE SCHEDULING MODEL

3.1 Problem Description

Our MPE resource scheduling problem assigns one month of operator visits to enforcement tasks (recall that a task consists of visiting a group of sites within one shift), based on the spatial allocation of enforcement resources from Stage 2 (Fig. 1). We set a one-month MPE schedule because the smallest timeframe for evaluating an MPE deployment is one month (Kim et al., 2016). In addition, site characteristics (i.e. speed compliance figures, collisions, etc.) updated over the month are accounted for. To assign visits, we applied a binary integer programming approach with the goal of minimizing the number of visits to a site within its enforcement time halo. As discussed in Section 2.2, our approach accounts not only for enforcement halos, but also for site characteristics and safety goals (as per Stage 2 (Li et al., 2019)), unlike traditional schemes that simply assign resources to sites in a randomized fashion without consideration of these issues. However, our approach also addresses the need for enforcement (locations and times) to appear as unpredictable as possible to public perception, by starting with the above mentioned randomized assignment, and allowing operators to vary site visit order and duration in each shift. The one exception is when continuous enforcement is deemed necessary at specific hotspot locations.

Because a binary integer program is a combinatorial optimization problem, when encountering complex instances the problem becomes difficult to solve (Smith-Miles & Lopes, 2012). The sizes of the MPE scheduling problems, formulated and solved as binary integer programs by Adler et al. (2014) and Li et al. (2017) use fall somewhere in the horizontal middle of Fig. 2, which exceeds many common scheduling problems discussed in the literature. Therefore, as mentioned previously, these authors generate one-month schedules with enforcement time halos restricted to three shifts (1.5 days, at two shifts per day). As described in Section 2.2, different empirical studies have reported time halos that vary from hours to days. An empirical study by Gouda & El-Basyouny (2016) found that time halos can last 5-7 days (or 10-14 shifts). Thus, we propose a model (Section 3.2) for scheduling shifts over a month with a 5-day time halo and apply the Dantzig-Wolfe decomposition method (Section 4) to solve the model in polynomial time.

3.2 Model Formulation

Our model minimizes consecutive shift visits to the same enforcement task, in consideration of potential enforcement time halo effects. It is a binary integer program (BIP) through which Eqns.

1-4 are used to schedule operator shifts to their corresponding tasks (sets of sites), and Eqn. 5 is used to compute penalty costs based on evaluating time-halo violations at the task level.

3.2.1 Notation

Sets

I	=	set of all shifts in the given month
J	=	set of all tasks identified across all neighborhoods $J = \sum_{n \in N} J_n$
J_n	=	set of tasks included in neighborhood n
N	=	set of all neighborhoods chosen for enforcement in Stage 2
S_j	=	set of feasible shift schedules for task j
K_j	=	set of all possible schedules of distributing x_j visits within I shifts
$\widehat{K}_j$	=	subset of K_j with restricted size

Parameters

i	=	shift index, $i = 1, \ldots, I$
j	=	task index, $j = 1, \ldots, J$
n	=	neighborhood index, $n = 1, \ldots, N$
c_{ij}	=	cost of allocating an operator visit at task j in shift i
dl_i	=	minimum number of visits required for shift i
du_i	=	maximum number of visits allowed for shift i
x_j	=	number of monthly shifts allocated to candidate task j, defined in Stage 2
t	=	number of consecutive shifts over which time halo effects are observed
mod	=	remainder of $1 + (i - 1)$ divided by I
k	=	schedule index, $k = 1, \ldots, K_j$
u_i, v_i, w_j	=	dual solutions corresponding to Eqns. 14-16

Decision Variables

x_{ij}	=	1 if visit occurs at location visit task j in shift i, 0 otherwise
λ_j^k	=	1 if shift schedule x_j^k is selected for task j, 0 otherwise
λ'^k_j	=	LP relaxation of λ_j^k

3.2.2 Formulation

$$min \sum_{i=1}^{I} \sum_{j=1}^{J} c_{ij} x_{ij} \tag{1}$$

Subject to:

$$dl_i \leq \sum_{j=1}^{J} x_{ij} \leq du_i, \quad \forall i = 1 \dots I \tag{2}$$

$$\sum_{i=1}^{I} x_{ij} = x_j, \quad \forall j = 1 \dots J \tag{3}$$

$$x_{ij} \in \{0,1\}, \quad i = 1 \dots I, j = 1 \dots J \tag{4}$$

We use Eqns. 1-4 to construct a table where the rows represent shifts over a month, and columns represent the tasks determined in the Stage 2 model (Fig. 1). Binary variables x_{ij} are set to one if task j is visited by an operator in shift i, and 0 otherwise. Each column of the table, $X_j = (x_{1j}, x_{2j}, \dots, x_{Ij})^T$, shows in which shifts a task will be visited in the month.

Eqn. 1 is used to minimize the cost of allocating operator visits to tasks and shifts. Eqn. 2 limits the total visits that occur in shift i to range $[dl_i, du_i]$, because there are a limited number of operators that can work each shift per day. Eqn. 3 ensures that the total visits assigned to task j is equal to the number of monthly shifts allocated to task j as found in Stage 2 (x_j). Eqn. 4 is the binary constraint on the variable x_{ij}.

Cost c_{ij} (in Eqn. 1) is calculated using Eqn. 5 to penalize task assignments to consecutive shifts (in violation of an enforcement time halo effect). Cost functions on goals other than time halo violations can be specified as well, if there are other program goals of importance.

$$c_{ij} = \sum_{i'=i}^{i+t-1} x_{\{1+[(i'-1) mod\ I]\}j} \quad \forall i, j \tag{5}$$

As shown above, t denotes the duration of the enforcement time halo, represented by number of consecutive shifts. In this t-shift time halo, more than two visits are considered a violation of the time halo (and therefore, inefficient). For a schedule of visits to a specific task, we count the frequency of visits in t shifts as the cost of time halo violation, as illustrated in Eqn. 5. The more frequently the same task (i.e., set of sites) is visited during t shifts, the greater the penalty cost will be. In addition, since Eqn. 5 sums the number of visits allocated within a window of length t over each shift i, higher costs are assigned to successive visits. Note that we penalize the frequency of consecutive visits in t shifts but not the pattern of those visits during the enforcement time halo.

For example, three consecutive shifts in a task incur the same cost compared to three shifts that are spaced a shift apart. Although the intensity of the deterrence effect during an enforcement time halo may not necessarily be constant over its duration, likely being a function of time and enforcement intensity (Gouda & El-Basyouny, 2016), these effects have not been empirically verified in the literature. Therefore, without any empirical evidence otherwise, to maintain model tractability we have assumed that the time halo duration and deterrence effect intensity during the time halo are fixed. Consequently, more than two visits in t shifts are considered a violation of the time halo (and therefore, inefficient as per our (cost) objective function of Eqn. 5).

When calculating c_{ij}, we connect the first and last rows of the schedule to form a closed loop, calculating the cost of the last $t-1$ shifts using a sliding window of length t. Eqn. 5 implements this by expressing the row index as a function of i. The row index is represented as 1 plus the remainder of $i-1$ divided by I, where i iterates from i to $i+t-1$. For example, suppose $t=2$ and $I=60$, the cost of the element in the last row ($i=60$) and the jth column on the schedule, c_{60j}, equals to $x_{60j}+x_{1j}$ from Eqn. 5.

We deliberately did not consider (the cost of) deadheading in the model. First, although our goal in building this model is to introduce systematic efficiencies to MPE programs, we recognize the need to balance this against operator autonomy. Otherwise, the likelihood of implementation is low. In the current City of Edmonton program, although enforcement operators are given general instructions on sites to visit, decisions on how long they spend at locations, where they take their breaks, whether/if they revisit a site, and others, are generally left to operators. Thus, our model results only indicate to which sites operators should go during each shift. Second, the sites in an enforcement task are within a single city neighborhood, which cover relatively small areas (in fact, a region for enforcement can be divided up in any way desired – by neighborhood or otherwise). Thus, deadheading between sites would be a matter of minutes over an eight-hour shift.

4 DANTZIG-WOLFE DECOMPOSITION AND COLUMN GENERATION (DW-CG) APPROACH

The original BIP of Eqns. 1-4 becomes a binary quadratic program (BQP) with the cost term as shown in Eqn. 5. The methods discussed in Section 2.3 for solving BIP can also be used to solve BQP. However, the solvable BQP problem is several orders of magnitude smaller than BIP (Nowak, 2005). For example, once the size of the problem instance is larger than 30, solving a BQP is inefficient with any algorithm, such that the problem must be reconstructed (Smith-Miles & Lopes, 2012). Restructuring the problem in order to apply a heuristic algorithm requires development of local search algorithms to ensure solution feasibility. Alternately, some relaxation and decomposition methods are widely used because they convert BQP into problems that can be solved by state-of-the-art MIP solvers (Ceselli et al., 2017). For example, to reconstruct a BQP, linearization techniques can be used to reduce the order of the objective function to linear

(Chaovalitwongse et al., 2004; Glover & Woolsey, 1973), quadratic convex reformulation (QCR) to convexify the objective function (Billionnet et al., 2009; Borndörfer & Cardonha, 2012), semi-definite programming (SDP) to replace quadratic terms with semidefinite matrix variables (Rendl et al., 2010; Wang et al., 2016), and Dantzig-Wolfe (DW) to decompose the solution space into sub-problems (Ceselli et al., 2017; Nowak, 2005). Considering the link between Dantzig-Wolfe and the dual simplex algorithm (one of the strongest linear programming algorithms and also widely used in many solvers) (Bixby, 2002), unlike the other methods, most solvers can directly solve the problem remaining after applying Dantzig-Wolfe decomposition and column generation algorithms (Nowak, 2005). Thus, we used this approach to solve Eqns. 1-5.

Fig. 3 shows the process of applying Dantzig-Wolfe decomposition and column generation to solve the MPE scheduling problem introduced in Section 3. The two steps in Fig. 3 are explained in detail in Sections 4.1 and 4.2 respectively. All notations is listed in Section 3.2.1.

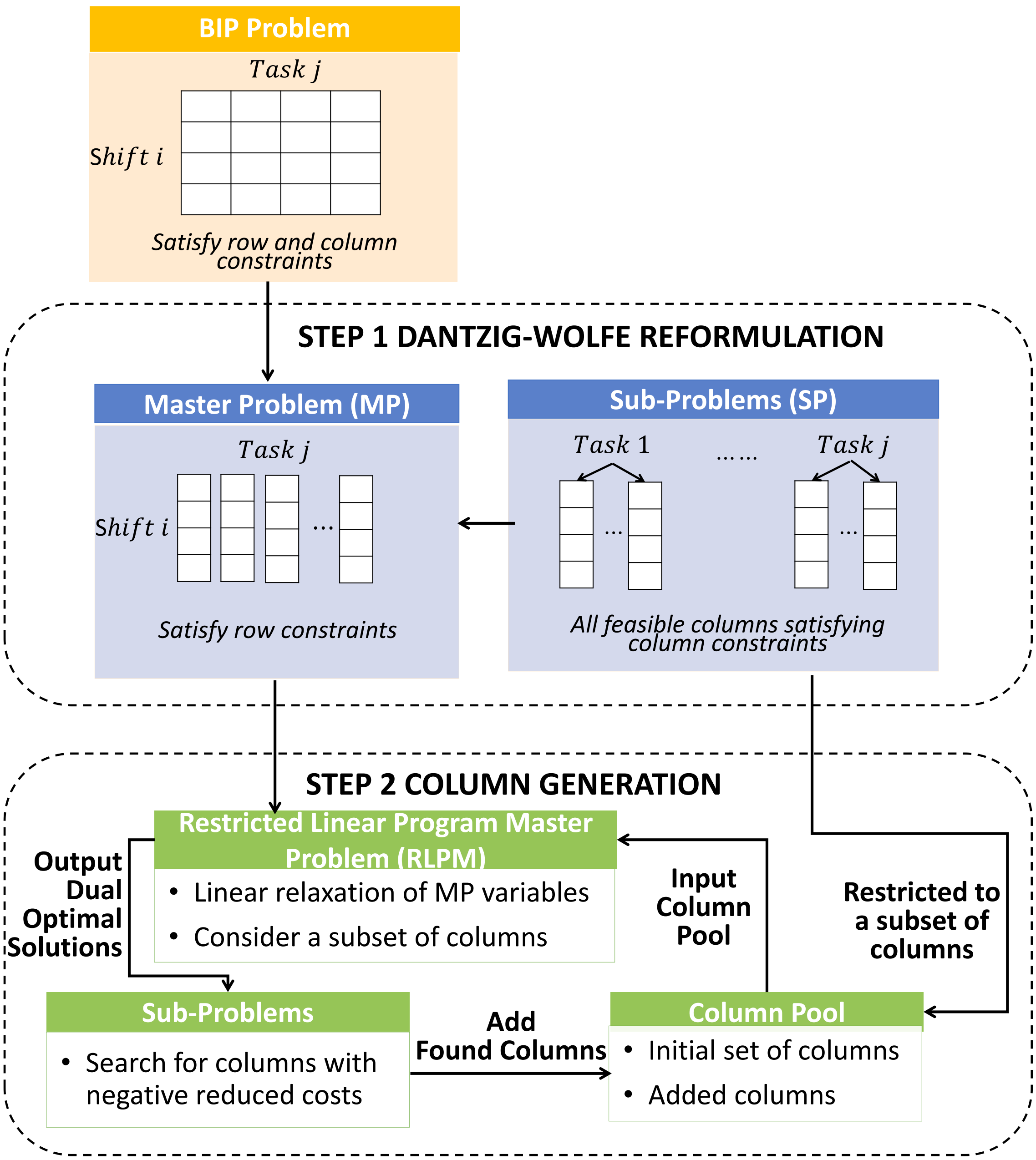

Fig. 3 Implementing Dantzig-Wolfe and column generation algorithms.

4.1 Dantzig-Wolfe Decomposition

Dantzig-Wolfe treats Eqn. 3 and Eqn. 4 as J sub-problems (recall that J is the total number of tasks). For any task $j \in J$, assume all possible schedules (that distribute x_j visits within I shifts) determined by Eqns. 3 and 4 are known, and there are K_j schedules. Let $S_j = \{x_j^1, x_j^2, \ldots, x_j^{K_j}\}$ be a complete enumeration of the feasible shift schedules for task j. Each element of S_j is a feasible solution (column) to Eqns. (3 and 4.

Then, we use Eqns. 6-8 to rewrite x_{ij} from Eqns. 1-4, by introducing a new binary variable λ_j^k. When the k-th column in S_j is selected, $\lambda_j^k = 1$; otherwise $\lambda_j^k = 0$.

$$x_{ij} = \sum_{k=1}^{K_j} \lambda_j^k \, x_{ij}^k \tag{6}$$

$$\sum_{k=1}^{K_j} \lambda_j^k = 1, \quad \forall j = 1 \ldots J \tag{7}$$

$$\lambda_j^k \in \{0,1\}, \quad k = 1 \ldots K_j, j = 1 \ldots J \tag{8}$$

Eqn. 6 represents variable x_{ij} as a convex combination of the elements that are positioned in the i-th row of the feasible columns in S_j. Eqn. 7 is the convexity constraint that forces the sum of λ_j^k over k to be equal to one, and Eqn. 8 states λ_j^k is binary.

Substituting Eqns. 6-8 into Eqns. 1-4 results in Eqns. 9-12, where the original problem (Eqns. 1-4) is rewritten as the Dantzig-Wolfe master problem (MP) using new variable λ_j^k.

Master Problem (MP)

$$min \sum_{j=1}^{J} \sum_{k=1}^{K_j} \left(\sum_{i=1}^{I} c_{ij} \, x_{ij}^k \right) \lambda_j^k \tag{9}$$

Subject to:

$$dl_i \leq \sum_{j=1}^{J} \sum_{k=1}^{K_j} x_{ij}^k \, \lambda_j^k \leq du_i, \quad \forall i = 1 \ldots I \tag{10}$$

$$\sum_{k=1}^{K_j} \lambda_j^k = 1, \quad \forall j = 1 \ldots J \tag{11}$$

$$\lambda_j^k \in \{0,1\}, \quad k = 1 \ldots K_j, j = 1 \ldots J \tag{12}$$

Note x_{ij}^k is no longer a decision variable in Eqn. 9, but rather, is selected from the feasible set (S_j) determined in the sub-problem corresponding to j. Eqn. 11 is the partitioning constraint enforcing that only one column in S_j is selected for task j. When using a classic branch-and-bound integer program algorithm to solve the master problem, branching on λ_j^k is equivalent to branching on the convex combination of feasible columns for j. Therefore, tighter bounds can be obtained from solving the master problem rather than the original problem (branching on a single element of column j), speeding up the solution process.

4.2 Column Generation

Instead of solving the master problem (MP) of Eqns. 9-12 directly, column generation handles a restricted linear programming MP (RLPM), in which the MP decision variable λ_j^k is relaxed to be continuous and the total number of variables for each j, K_j, is restricted to a given number $\widehat{K}_j$. Let $\lambda'^k_j \in [0,1]$ denote the relaxed variable, where $k = 1 \dots \widehat{K}_j < K_j$, and $j = 1 \dots J$. Eqns. 13-17 present the RLPM formulation using variables λ'^k_j.

4.2.1 Restricted Linear Programming Master Problem (RLPM)

$$min \sum_{j=1}^{J} \sum_{k=1}^{\widehat{K}_j} \left(\sum_{i=1}^{I} c_{ij}\, x_{ij}^k \right) \lambda'^k_j \tag{13}$$

Subject to

$$\sum_{j=1}^{J} \sum_{k=1}^{\widehat{K}_j} x_{ij}^k\, \lambda'^k_j \geq dl_i, \quad \forall i = 1 \dots I \tag{14}$$

$$-\sum_{j=1}^{J} \sum_{k=1}^{\widehat{K}_j} x_{ij}^k\, \lambda'^k_j \geq -du_i, \quad \forall i = 1 \dots I \tag{15}$$

$$\sum_{k=1}^{\widehat{K}_j} \lambda'^k_j \geq 1, \quad \forall j = 1 \dots J \tag{16}$$

$$\lambda'^k_j \geq 0, \quad k = 1 \dots \widehat{K}_j < K_j, j = 1 \dots J \tag{17}$$

To find feasible solutions using the simplex algorithm, we rewrite the constraint matrix of RLPM to single-sided inequality constraints. Eqns. 14 and 15 are the single-sided inequality constraints restated from the two-sided inequality constraint of Eqn. 10. Eqn. 16 relaxes the partitioning constraint in Eqn. 11 (that requires selection of exactly one column for j) to a covering constraint (at least one column); the LP relaxation of set covering constraints requires less computational effort to solve (Albers, 2009) but still produces the same optimal solution.

4.2.2 Initialization of a Feasible Schedule

To solve the RLPM, we begin by initializing with a schedule where we assign resources to locations and shifts in a random fashion (Section 2.2). This creates a starting subset of columns in the column generation process (see Fig. 3).

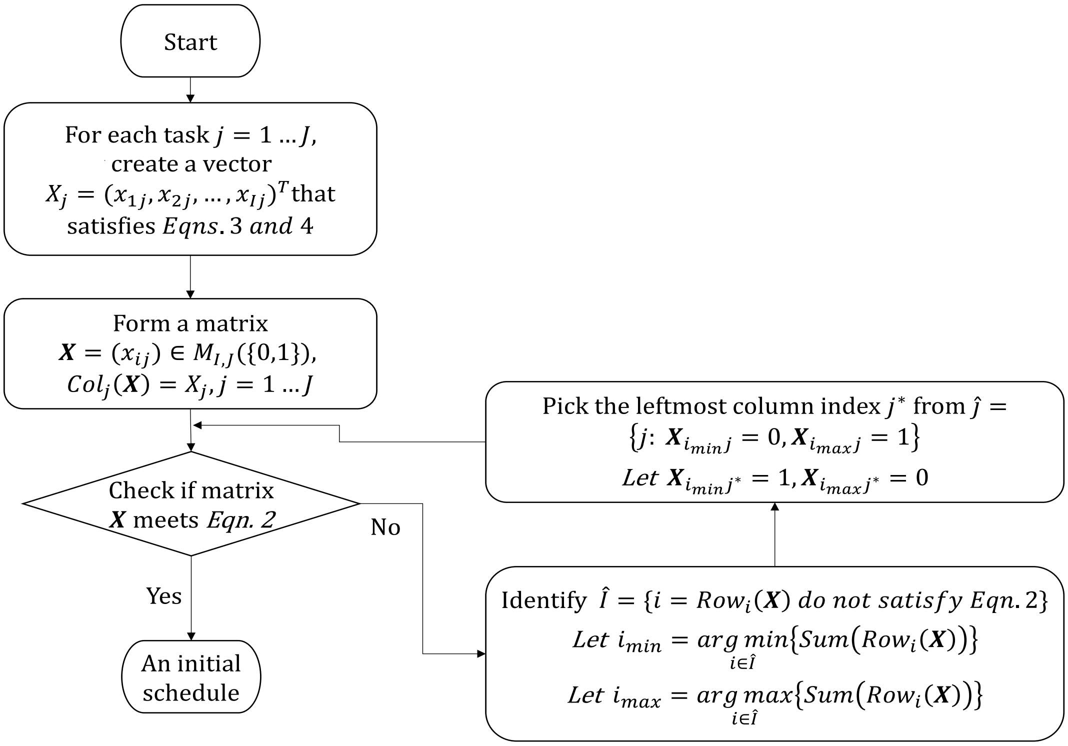

Fig. 4 Generating an initial subset of columns in RLPM.

As described above, for each task $j = 1 \dots J$, we randomly generate a column vector of binary variables, $X_j = (x_{1j}, x_{2j}, \dots, x_{Ij})^T$, whose sum is equal to x_j (derived from the Stage 2 of Fig. 1). This column vector contains feasible solutions to Eqns. 3 and 4. A matrix $\boldsymbol{X}$ is created by combining all the columns generated, which can be a feasible schedule if Eqn. 2 is satisfied. To check this is true, we sum each row of $\boldsymbol{X}$. Rows that do not satisfy Eqn. 2 are identified. The next step is to iteratively swap some zeros and ones of these identified rows, until they all comply with Eqn. 2. Specifically, we start with the rows with minimum and maximum sums, because they have the largest differences from the lower and upper limits of Eqn. 2 (dl_i and du_i). We search for a leftmost column with a zero in the minimum row and a one in the maximum row, and we swap these two elements. By doing this, the x_j value of this column will not change, thus ensuring Eqn. 3 is still satisfied. In this iterative swapping process, a schedule (matrix $\boldsymbol{X}$) feasible to the RLPM is created.

4.2.3 Sub-Problems (SPs)

After a feasible initial schedule is found, we use simplex to solve the RLPM. To obtain a final optimal solution and schedule, we search the columns that can replace the columns in the current pool that are inferior such that the RLPM objective function value decreases until there are no remaining columns. This search process (green boxes of Fig. 3) can be written as the following sub-problems (Eqns. 18-20).

$$min \sum_{i=1}^{i=I} (c_{ij} - \mu_i + v_i)\, x_{ij} - \omega_j, \qquad \forall j = 1 \dots J \tag{18}$$

Subject to

$$\sum_{i=1}^{I} x_{ij} = x_j, \quad \forall j = 1 \dots J \tag{19}$$

$$x_{ij} \in \{0,1\}, \quad i = 1 \dots I, j = 1 \dots J \tag{20}$$

Eqn. 18 minimizes the reduced cost of a feasible column $(x_{1j}, \dots, x_{Ij})^T$ for task j. The reduced cost of a column is the marginal unit change in RLPM objective function value when this column is added to the solution. Therefore, we use Eqn. 18 to find a column with the lowest reduced cost value, and if it is negative, it is entered to the next solution. Eqn. 18 consists of the dual solutions corresponding to the RLPM constraints (Eqns. 14-16) using μ_i, v_i, and ω_j, respectively. The optimal dual solution obtained each time RLPM is solved with the current matrix. Eqns. 19 and 20 ensure that $(x_{1j}, \dots, x_{mj})^T$ satisfies the conditions for a feasible shift schedule.

The optimal solution to the RLMP is usually fractional but does provide a tight bound to the master problem. We re-optimize RLPM containing the final column pool using integer variables to quickly find an approximate solution to the BIP formulation.

4.3 Computational Experiments

We compared the DW-CG approach against B&C in commercial solvers CPLEX. Usually, large BQPs are solved by decomposing the original problem (Smith-Miles & Lopes, 2012). Here we show how the DW-CG decomposition algorithm performs against that of algorithms used without decomposing the problem, as the problem size becomes larger. B&C is one of the most commonly used exact algorithms for solving mixed integer programming in commercial solvers. We compare these two approaches to demonstrate the ability of DW-CG to solve large problem instances. We note that various algorithms can be redeveloped to decompose and simplify the problem, in addition to DW-CG, but that doing so to compare the results of DW-CG is out of the scope of this paper. We created 14 test problems with 5-60 rows and 15-150 columns (and time halo $t = \{3,5,7,10\}$. Results are shown in Table 1.

Table 1 Results of DW-CG and B&C

Prob. no.	Problem Set			DW-CG		B&C (CPLEX)	
	Rows	Cols	Time Halo, t (shifts)	Optimal Solution	Total Time (s)	Optimal Solution	Total Time (s)
1	5	15	3	92	0.5	92	0.1
2	5	15	5	142	0.7	142	0.1
3	10	20	3	271	11.6	271	0.2

4	10	20	5	445	5.3	445	2.3
5	10	20	7	610	12.6	610	42.1
6	10	20	10	879	0.9	*	-
7	40	100	3	613	169.6	613	0.3
8	40	100	5	670	314.5	*	-
9	40	100	7	788	553.9	*	-
10	40	100	10	1060	2048.8	*	-
11	60	150	3	785	291.8	785	539.5
12	60	150	5	785	400.9	785	1435.3
13	60	150	7	828	2049.9	*	-
14	60	150	10	966	4470.7	*	-

** Out of memory.*

We set x_j to be random numbers between 1 and 10. For two problems containing only five rows (first two rows of the table), $t = \{7,10\}$ are not considered and the range of x_j is reduced to $[1, 5]$. For feasibility, we assign random numbers to dl_i, du_i that fall respectively in the intervals $\left[0.6 \times \frac{\sum_j x_j}{I}, \frac{\sum_j x_j}{I}\right]$ and $\left[\frac{\sum_j x_j}{I}, 1.4 \times \frac{\sum_j x_j}{I}\right]$. Each problem was solved ten times by the two methods on a computer with Intel® Core™ i7-8700 CPU (3.20GHz) and 48GB RAM. The average optimal solution values and computation times are reported.

We observed that both methods can find true optimal solutions for the first five problems, while DW-CG is relatively slower than B&C. For the remaining 9 problems, B&C failed to find optimal solutions for problems #6, 8-10, 13, and 14 as CPLEX ran out of memory after running 2-16 hours. DW-CG found optimal solutions for all problems and the longest solution time was 1.24 hours.

5 APPLICATION AND RESULTS

This section discusses model solutions for a real-world instance. We determined a schedule of 449 shifts assigned to 145 tasks (i.e., site groups) within one month (30 days), considering a five-day time halo ($t = 10$ consecutive morning and afternoon shifts as per Eqn. 5). The data used is described below.

5.1 Data Description

We generated a shift schedule for a September 2014 candidate neighborhood-level deployment plan. This particular neighborhood-level plan is one solution from the set of optimal solutions determined using the neighborhood-level allocation model by Li et al. (2019). This plan consisted of an entire month of shifts assigned to 44 neighborhoods containing 130 enforcement sites. Three years of geocoded data (2012-2014) from the City of Edmonton consisting of 18,198 speed-related midblock collisions, 893 speed survey reports, and 296 school locations were used to calculate metrics representing three important enforcement needs (reducing collisions and speed violations, and increasing enforcement presence in school zones). The chosen plan returned objective function values of 3441, 248, and 797 associated with the three metrics.

Based on the average number of sites per shift observed in September 2014 (three sites), a total of 145 tasks from the 44 neighborhoods chosen for enforcement in Stage 2 (Fig. 1) were created. In addition, 449 shifts were split between tasks; thus, these $J = 145$ tasks and their received shift allocations $x_{j \in J}$ are input into the model.

The specification of a five-day time halo is based on a result of a study on enforcement halos measured on arterial and collector roads in the City of Edmonton (Gouda & El-Basyouny, 2016). The time halo effects of enforcement were observed to reach an average of five days at nine locations monitored over five weeks. Enforcement intensity levels were found to impact the duration of the time halo; however, a relationship between the enforcement intensity and time halo was not determined. Therefore, we will assume a five-day enforcement time halo duration, regardless of the intensity of the applied enforcement. Future work may include modifying the function (Eqn. 5) to account for the effects of enforcement intensity on time halo durations.

5.2 Results

We solved the instance in 2.1 hours on a Windows 10 with a 3.2 GHz Intel® Core i7 processor, and an optimal solution for RLPM was found. The solution contained fractional values, so the RLPM was re-optimized by solving integer variables in the last column pool generated. We found an optimal integer solution to the re-optimization problem in 0.1 seconds. Although reducing computational time should be a future priority, our model does solve the MPE scheduling problem for an entire month. Shorter time halos allow for faster solution times.

Fig. 5 shows a sample of the resulting schedule. The Fig. 5 schedule has been transposed, with tasks on the rows and daily shifts (M = morning, A = afternoon) on the columns. "X" indicates a shift during which visits were made to a neighborhood task (included sites are identified). For example, task #53 was completed twice in a month, in the morning shifts of Day 4 and Day 27. This task consists of three enforcement sites (#10073, #10091, and #10866) located in the neighborhood of Queen Mary Park.

Task No.	Neighborhoods	Days & Shifts / Enforcement Site No.			D1		D2		D3		D4		D5		…	D26		D27		D28		D29		D30	
					M	A	M	A	M	A	M	A	M	A	…	M	A	M	A	M	A	M	A	M	A
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
53	Queen Mary Park	10073	10091	10866							X				…			X							
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
134	Terrace Heights	5185	10214	10534					X						…										
135		5185	10218	10534				X							…										
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…

Fig. 5 Sample of the resulting schedule.

We note that the resulting scheduling contains a variety of task visit frequencies, which is important for contributing to the public perception of unpredictability in MPE presence. The most frequently visited task occurs daily (clearly violating the five-day time halo, discussed further in 5.3), whereas the most infrequent tasks are visited only once per month. We do not instruct operators to visit sites for equal durations or in the same order during each shift; allowing and encouraging variations in both can contribute to the perception that enforcement lacks a predictable order or plan. Also, recall that the final resulting schedule (which optimizes resource utilization and minimizes time halo violations) is based on an initial randomly assigned schedule described in Section 4.2.2 (and motivated in 2.2); thus, some aspects of that initial schedule are retained.

5.3 Time Halo Violation Analysis

The schedule shown in Fig. 5 has an objective function value (Eqn. 13) of 1,263, meaning that it contains 1,263 cost units due to violations of the five-day time halo. This value is 14% lower than the cost of the initial solution.

Visits assigned to 80 of the 130 sites (included in the 145 tasks) violated the five-day time halo. Fig. 6 shows the locations of these 80 sites (i.e. roadway segments) and their neighborhoods. Most violations are observed in neighborhoods receiving medium and high enforcement intensities (marked as yellow and red polygons) in the previous stage neighborhood-level resource allocation. Only three low intensity neighborhoods (green polygons) contain sites with time halo violations. Because neighborhoods warranting high enforcement attention were allocated a large number of visits in Stage 2 (Fig. 1), these neighborhoods are also more likely to have time halo violations in the operator schedule.

The most common violation scenario is when a site is visited twice within 10 shifts (5 days). This occurs 610 times in the month-long solution (501 non-successive visits and 109 successive visits), mainly at 48 sites represented by grey lines in Fig. 6. Each site was assigned seven visits on average over the month. In scheduling seven visits in 60 shifts, it is unavoidable that some violations will occur with a 5-day (10-shift) time halo.

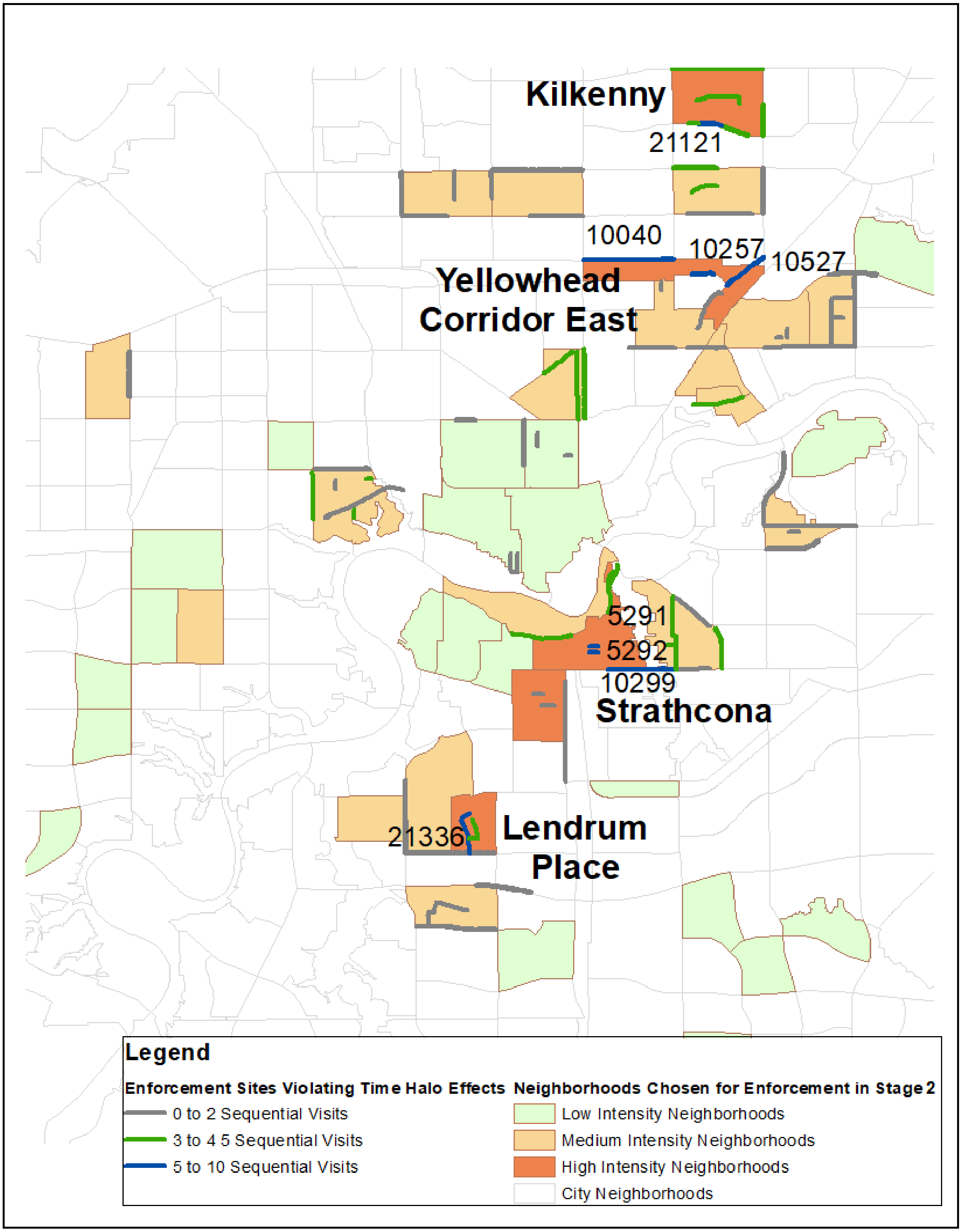

Fig. 6 Task locations with shift schedule violating the time halo effect, City of Edmonton.

A high number of time halo violations (three to four consecutive visits) occurred at 24 sites (green lines in Fig. 6), at an average of two such visits per site in a month. Each of these 24 sites was assigned an average of 16 visits, which is twice that of the 48 locations shown above. These locations were allocated high intensity enforcement attention because they are in neighborhoods with average SVI (% of vehicles violating speed limits) of 62% and SZD (number of schools per sq.km) of 1.7 – 30% higher than the neighborhoods containing the 48 sites above.

There are eight sites (blue lines in Fig. 6) where five to ten consecutive visits were assigned. Sites #5291, #5292, #10299 and Sites #10040, #10257, #10527 are the only pre-designated enforcement sites in Strathcona and Yellowhead Corridor East, respectively. Strathcona, containing a mix of residential and commercial land uses, is a central Edmonton neighborhood with an average of 12.2 EPDO/km (EPK) and 1.9 SZD. The Yellowhead Corridor East neighborhood, which contains both residential and industrial land uses, includes a section of the Yellowhead Trail Expressway on which drivers frequently exceed posted speed limits. It has an average EPK of 16.6 and SVI of 75%. Strathcona and Yellowhead Corridor East were assigned 60 and 40 visits respectively, which were all allocated to their three enforcement sites (grouped into a single location visit task for each neighborhood). Consequently, visits assigned to Strathcona and Yellowhead Corridor East repeated 10 and six times on average during each 10 shifts. Two other sites, one each in Kilkenny and Lendrum Place, received high intensity enforcement attention, at 31 visits (Site #21121) and 27 visits (Site #21336) in one month, respectively. The highest number of consecutive visits was six. To put these high site visit assignments in context, Kilkenny was assigned a total of 56 visits over the month due to average SZD and SVI values of 2.9 and 56%, while Lendrum Place had 40 visits with SZD and SVI of 3.5 and 64% respectively.

Visits at sites during consecutive shifts are unavoidable with the results of the Stage 2 neighborhood-level resource allocation model. As mentioned in the formulation of BIP (Section 3), our model schedules visits for tasks and calculates time halo violations at the task rather than individual site level. Hence, it does not consider cases when two tasks that contain several of the same enforcement sites are scheduled for visits in successive shifts. For instance, the Terrace Heights neighborhood in Fig. 6 contains sets #134 and #135, both of which contain enforcement sites #5185 and #10534. These sites are assigned visits in the Day 2 afternoon shift and Day 3 morning shift, violating the five-day enforcement time halo at the site level. This type of violation occurred at 35 sites (nine neighborhoods), and they constitute about half the 109 two-shift consecutive visits discussed above. Future work should involve improvements in the way tasks in a neighborhood are selected, to reduce duplications of sites in different tasks. However, Edmonton neighborhoods are quite small in area, and MPE operator presence may be felt throughout a neighborhood through an enforced site's distance halo effect (halo effects have been observed in distance as well as time).

Finally, we compared time halo violations that actually occurred in September 2014 versus model results, and found that the time halo violation costs of the model results are 24% lower than those assessed for the actual deployment of September 2014.

6 SENSITIVITY ANALYSIS

The enforcement time halo duration (t, in shifts) is critical for model outcomes and computational time. Thus, we explore the results of varying t using the same instance described in Section 5.

Table 2 Results for Varying Enforcement Time Halo Shift Duration t

Time Halo Duration t (shifts)	Days represented by t	Objective Value of RLMP Z			Total Computation Time (min)
		Initial Solution λ^0	Optimal Solution λ	Diff (%)	
2	1	558	529	-5%	1.3
4	2	775	696	-10%	2.2
6	3	1000	877	-12%	5.5
8	4	1209	1069	-12%	23.2
10*	5	1435	1262	-12%	126.3

* Time halo shift duration used for previous results (Section 5.3).

Columns 3 and 4 of Table 2 show objective function value Z for the initial solution λ^0 and the approximate optimal solution λ, while Column 5 shows the percent difference between λ^0 and λ. The final column shows the total computation time to obtain solutions. The Z values in Columns 3 and 4 and the computation time in the last column are the average results after each time halo is calculated 10 times (on the same PC used for obtaining results in Sections 4 and 5).

As t increases from 2 to 10 shifts, the difference between $Z(\lambda^0)$ and $Z(\lambda)$ increases from 5% to 12%. This suggests that the solution λ is an improvement over the initial (randomly generated) schedule (i.e., the initial solution λ^0) for all time halo values applied.

The computational time is highly sensitive to t, growing quickly with t as expected. When $t = 2$ shifts, we can obtain an approximate integer solution in about one minute. When $t = 8$ shifts, solution times are below 25 minutes. However, solution time exceeds two hours when $t = 10$, which is a significant growth from the 25 minutes of $t = 8$. This relationship between t and solution time is because the model enumerates all possible visits to be made in the next $t - 1$ shifts after each visit assignment to a shift. More t means larger enumerations.

7 CONCLUSIONS

This paper describes a resource scheduling model for an urban mobile photo speed enforcement (MPE) program, which relies on the use of automated photo radar technology. The model was demonstrated using data from a MPE program currently in place in Edmonton, Canada, providing a month-long schedule that assigns enforcement operators and equipment to locations throughout the city, in shifts. The overall aim of the scheduling model is to optimize attainment of a set of high level safety-related goals (reducing speeding and collisions) identified in a previous modeling stage. We use a binary integer programming model to generate a shift schedule that minimizes repeat visits to sites where an enforcement time halo is active. In order to solve for larger enforcement time halos (i.e., greater than two-shift), we apply the Dantzig-Wolfe decomposition and column generation algorithm. A set of computational tests demonstrate that the performance of the algorithm is superior to CPLEX when dealing with large scheduling problems. Results suggest that this model can improve resource utilization (measured by time halo violations) by an average of 5-12% over schedules where resources are randomly assigned to locations and times, and by 24% over the MPE schedule currently implemented in Edmonton.

We believe this research comes at an important time when many cities are focusing on improving traffic safety, often through adoption of the Vision Zero goals. For cities that also operate MPE programs, this research provides a unified framework by which traffic safety-related goals can be systematically incorporated, and limited enforcement resources efficiently used. Also, the framework can be easily adapted for, and transferred to, other automated traffic enforcement technologies across jurisdictions, by tailoring to jurisdiction-specific program goals (with corresponding metrics) and technologies in place.

There are several directions by which the proposed MPE scheduling model can be improved and expanded upon. A key methodological improvement would involve accounting for violations to specific sites within tasks, rather than only at the task level. We also recommend further investigating how to reduce model solution times. Furthermore, we recommend addition of a post-processing step that analyzes trade-offs between enforcement goals and resource efficiency. Through this step, visits that are redundant within enforcement time halos are reassigned to other sites, while ensuring that the optimized enforcement goals values (from Pareto solution in Stage 2) are maintained at acceptable levels. Furthermore, the Stage 2 multi-objective optimization model (currently only considering traffic safety objectives), may be modified to include enforcement efficiency as a goal. This way, when the program manager chooses a Pareto optimal solution in Stage 2, they will also choose a tradeoff between resource utilization and safety outcomes. Finally, to measure program efficacy in terms of collision and speed violation reductions, a pilot will be conducted to compare it against an existing deployment scheme. To effectively capture the program's effects on collision counts, the evaluation period should be set to one year (Kim et al., 2016).

8 ACKNOWLEDGEMENT

The data used in this study was obtained from the Office of Traffic Safety at the City of Edmonton, Alberta, Canada. The authors also thank Zeeshan, a University of Alberta Research Experience undergraduate intern, for his assistance in running computational test problems.

9 REFERENCES

Adler, N., Hakkert, A. S., Raviv, T., & Sher, M. (2014). The traffic police location and schedule assignment problem. *Journal of Multi-Criteria Decision Analysis*, *21*(5–6), 315–333.

Albers, M. (2009). *Freight railway crew scheduling: Models, methods, and applications*. Logos Verlag Berlin GmbH.

Alberta Transportation. (2018). *Automated Traffic Enforcement – Program Review* (Summary Report, p. 170). MNP LLP. https://www.alberta.ca/assets/documents/trans-ate-program-review.PDF

Armour, M. (1986). The effect of police presence on urban driving speeds. *ITE Journal*, *56*(2), 40–45.

Azadeh, A., Farahani, M. H., Eivazy, H., Nazari-Shirkouhi, S., & Asadipour, G. (2013). A hybrid meta-heuristic algorithm for optimization of crew scheduling. *Applied Soft Computing*, *13*(1), 158–164.

Berkuti, C., & Osburn, W. (1998). PHOTO ENFORCEMENT IN THE WILD WEST: NATIONAL CITY'S EXPERIENCE WITH PHOTO RADAR ENFORCEMENT PROGRAM. *1998 Compendium of Technical Papers*. http://trid.trb.org/view.aspx?id=581327

Billionnet, A., Elloumi, S., & Plateau, M.-C. (2009). Improving the performance of standard solvers for quadratic 0-1 programs by a tight convex reformulation: The QCR method. *Discrete Applied Mathematics*, *157*(6), 1185–1197.

Bixby, R. E. (2002). Solving real-world linear programs: A decade and more of progress. *Operations Research*, *50*(1), 3–15.

Borndörfer, R., & Cardonha, C. (2012). A Binary Quadratic Programming Approach to the Vehicle Positioning Problem. In *Modeling, Simulation and Optimization of Complex Processes* (pp. 41–51). Springer.

Cairney, P. T. (1988). The effect of aerial enforcement on traffic speeds. *AUSTRALIAN ROAD RESEARCH BOARD PROCEEDINGS*. https://trid.trb.org/view.aspx?id=339835

Ceselli, A., Létocart, L., & Traversi, E. (2017). *Dantzig Wolfe decomposition and objective function convexification for binary quadratic problems: The cardinality constrained quadratic knapsack case.*

Chaovalitwongse, W., Pardalos, P. M., & Prokopyev, O. A. (2004). A new linearization technique for multi-quadratic 0–1 programming problems. *Operations Research Letters*, *32*(6), 517–522.

Coleman, J. A., Paniati, J. F., Cotton, R. D., Parker Jr, M. R., Covey, R., Pena Jr, H. E., Graham, D., Robinson, M. L., MaClauley, J., Taylor, W. C., & Morford, G. (1996). *FHWA Study tour for speed management and enforcement technology*. Federal Highway Administration, US Department of Transportation.

Dantzig, G. B., & Wolfe, P. (1960). Decomposition principle for linear programs. *Operations Research*, *8*(1), 101–111.

Desaulniers, G. (2010). Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. *Operations Research*, *58*(1), 179–192.

Desaulniers, G., Desrosiers, J., Dumas, Y., Solomon, M. M., & Soumis, F. (1997). Daily aircraft routing and scheduling. *Management Science*, *43*(6), 841–855.

Desaulniers, G., Desrosiers, J., & Solomon, M. M. (2002). Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems. In *Essays and surveys in metaheuristics* (pp. 309–324). Springer.

Dreyer, C. B., & Hawkins, T. E. (1979). *Mobile ORBIS III speed enforcement demonstration project in Arlington, Texas: Program Evaluation*.

Emden-Weinert, T., & Proksch, M. (1999). Best practice simulated annealing for the airline crew scheduling problem. *Journal of Heuristics*, *5*(4), 419–436.

Ernst, A. T., Jiang, H., Krishnamoorthy, M., & Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, *153*(1), 3–27.

Ford Jr, L. R., & Fulkerson, D. R. (1958). A suggested computation for maximal multi-commodity network flows. *Management Science*, *5*(1), 97–101.

Glover, F., & Woolsey, E. (1973). Further reduction of zero-one polynomial programming problems to zero-one linear programming problems. *Operations Research*, *21*(1), 156–161.

Gouda, M., & El-Basyouny, K. (2016). Investigating time halo effects of mobile photo enforcement on urban roads. *Submitted for Presentation at the 96th Annual Meeting of Transportation Research Board and Publication in Transportation Research Record: Journal of Transportation Research Board*.

Hanafi, R., & Kozan, E. (2014). A hybrid constructive heuristic and simulated annealing for railway crew scheduling. *Computers & Industrial Engineering*, *70*, 11–19.

Hauer, E., Ahlin, F. J., & Bowser, J. S. (1982). Speed enforcement and speed choice. *Accident Analysis & Prevention*, *14*(4), 267–278.

Hoffman, K. L., & Padberg, M. (1993). Solving airline crew scheduling problems by branch-and-cut. *Management Science, 39*(6), 657–682.

Ke, L., & Feng, Z. (2013). A two-phase metaheuristic for the cumulative capacitated vehicle routing problem. *Computers & Operations Research*, *40*(2), 633–638.

Kim, A. M., Wang, X., El-Basyouny, K., & Fu, Q. (2016). Operating a mobile photo radar enforcement program: A framework for site selection, resource allocation, scheduling, and evaluation. *Case Studies on Transport Policy*, *4*(3), 218–229.

Lavoie, S., Minoux, M., & Odier, E. (1988). A new approach for crew pairing problems by column generation with an application to air transportation. *European Journal of Operational Research*, *35*(1), 45–58.

Leggett, L. (1997). Using police enforcement to prevent road crashes: The randomised scheduled management system. *Policing for Prevention: Reducing Crime*, *Public Intoxication and Injury*. *Crime Prevention Studies, 7*, 175–97.

Levine, D. (1996). Application of a hybrid genetic algorithm to airline crew scheduling. *Computers & Operations Research*, *23*(6), 547–558.

Li, Y., Kim, A., & El-Basyouny, K. (2019). INTERACTIVE ALLOCATION OF MOBILE PHOTO ENFORCEMENT RESOURCES WITH MULTIPLE PROGRAM OBJECTIVES. *Sustainable Cities and Society*, 101572.

Li, Y., Kim, A., & El-Basyouny, K. (2017). Scheduling resources in a mobile photo enforcement program. *Transportation Information and Safety (ICTIS), 2017 4th International Conference On*, 645–652.

Li, Y., Kim, A. M., El-Basyouny, K., & Li, R. (2016). Using GIS to interpret automated speed enforcement guidelines and guide deployment decisions in mobile photo enforcement programs. *Transportation Research Part A: Policy and Practice*, *86*, 141–158. https://doi.org/10.1016/j.tra.2016.02.008

Merkle, D., Middendorf, M., & Schmeck, H. (2002). Ant colony optimization for resource-constrained project scheduling. *IEEE Transactions on Evolutionary Computation*, *6*(4), 333–346.

Murata, T., Ishibuchi, H., & Tanaka, H. (1996). Genetic algorithms for flowshop scheduling problems. *Computers & Industrial Engineering*, *30*(4), 1061–1071.

Nishi, T., Muroi, Y., & Inuiguchi, M. (2011). Column generation with dual inequalities for railway crew scheduling problems. *Public Transport*, *3*(1), 25–42.

Nowak, I. (2005). *Relaxation and decomposition methods for mixed integer nonlinear programming* (Vol. 152). Springer Science & Business Media.

Rendl, F., Rinaldi, G., & Wiegele, A. (2010). Solving max-cut to optimality by intersecting semidefinite and polyhedral relaxations. *Mathematical Programming*, *121*(2), 307.

Rogalska, M., Bożejko, W., & Hejducki, Z. (2008). Time/cost optimization using hybrid evolutionary algorithm in construction project scheduling. *Automation in Construction*, *18*(1), 24–31.

Smith-Miles, K., & Lopes, L. (2012). Measuring instance difficulty for combinatorial optimization problems. *Computers & Operations Research*, *39*(5), 875–889.

Vaa, T. (1997). Increased police enforcement: Effects on speed. *Accident Analysis & Prevention*, *29*(3), 373–385.

Vance, P. H., Barnhart, C., Johnson, E. L., & Nemhauser, G. L. (1997). Airline crew scheduling: A new formulation and decomposition algorithm. *Operations Research*, *45*(2), 188–200.

Wang, P., Shen, C., van den Hengel, A., & Torr, P. H. (2016). Large-scale binary quadratic optimization using semidefinite relaxation and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(3), 470–485.

Yin, Y. (2006). Optimal fleet allocation of freeway service patrols. *Networks and Spatial Economics*, *6*(3–4), 221–234.

Yang Li: Methodology, Software, Data curation, Visualization, Investigation, Writing - Original draft preparation and Modification.

Amy Kim*:* Conceptualization, Supervision, Writing - Reviewing and Editing.