

Data-Driven Police Staffing

Weiqing Xu¹, Arthur Delarue²

¹H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology

²Darden School of Business, University of Virginia

Abstract

Large police departments usually operate by assigning regular patrol units to pre-defined regions, with backup units covering multiple regions to handle periods of high demand or replace unavailable regular units. We develop a data-driven approach to determine the optimal number and deployment of these backup units across different shifts, minimizing the expected travel time to incidents. The problem is formulated as a two-stage stochastic mixed-integer program. First-stage decisions determine the number of active backup units per zone and shift, while the second stage optimizes the dispatch of available units (both regular and backup) to the realized incident scenarios. This second-stage dispatch problem, which involves assigning units to incidents subject to incompatibility constraints derived from temporal and spatial conflicts, is shown to be NP-hard and closely related to the Optimal Cost Chromatic Partition problem from graph theory. To solve this challenging formulation, we design an enhanced integer L-shaped method. A key element of our approach is a disaggregated multi-cut strategy that leverages a natural decomposition of the problem by scenario, zone, and shift, leading to significant computational gains. We further accelerate the method with customized caching techniques. We validate our approach through an extensive case study using real data from the Atlanta Police Department, exploring the trade-offs between the number of units, operational costs, and service levels (response time and incident coverage). Our work provides a novel framework for jointly optimizing police staffing and dispatch under uncertainty, offering practical insights for decision-makers balancing resource constraints and response effectiveness.

Keywords: police operations, two-stage stochastic programming, integer L-shaped method, graph coloring.

1 Introduction

A key responsibility of police departments is responding to emergency calls from witnesses or victims of criminal activity. Two key components of a timely response are *dispatching* (i.e., deciding which patrol unit should respond to a particular incident) and *staffing* (i.e., determining how many officers should be on duty and where they should be stationed).

To simplify both problems, police departments typically divide urban areas into *beats*, which are usually the responsibility of a single *patrol unit*. In large cities, beats are further grouped into *zones*, allowing for more efficient resource allocation across beats in the same zone. For instance, the Houston Police Department divides the city into 13 zones (called “patrol divisions”), each containing several beats, enabling efficient resource allocation and incident management (Houston Police Department, 2021). Similarly, the Los Angeles Police Department divides the city into 17 divisions (LA Police Department, 2023), each containing multiple patrol areas. Curtin et al. (2010) point out that this systematic division into zones and beats is designed to optimize coverage and minimize response times, enhancing the overall effectiveness of police operations.

Given this nested districting structure, a typical staffing policy, such as the one followed by the Atlanta Police Department (APD), is to assign one *regular patrol unit* to each beat, as well as some number of backup units in each zone (usually called *umbrella units* because they cover multiple beats). When an incident occurs in a beat, dispatchers will prioritize the corresponding regular unit; if it is unavailable, they can call on either regular units from neighboring beats or umbrella units from the corresponding zone.

The goal of this paper is to determine staffing levels to minimize incident response times. In particular, we develop a data-driven two-stage approach to decide the number and optimal deployment of umbrella units at different times. In the first stage, we determine the number of available umbrella units in each zone; in the second stage, we make dispatch decisions to handle all incidents. We model this two-stage problem using stochastic mixed-integer programming, which we solve using a customized cutting plane approach based on the integer L-shaped method. We apply our model in a case study using data from the Atlanta Police Department.

We summarize our contributions as follows:

- **Modeling innovation:** To our knowledge, we are the first to consider a joint model of police staffing and dispatching based on graph coloring, that trades off response time with resource availability in a

high-demand setting.

- **Customized algorithm:** We identify and exploit a key structural property of the police patrol problem—its decomposability by scenario, zone, and shift. Leveraging this, we design a highly disaggregated multi-cut integer L-shaped method. This approach provides a significantly tighter formulation of the recourse function than standard scenario-based decompositions and forms the foundation for the computational efficiency of our solution methodology.
- **Practical case study:** Using a rich dataset of 911 calls from the Atlanta Police Department, we shed light on the fundamental staffing tradeoffs faced by a large urban police department.

We review related work in Section 2 and introduce our model in Section 3. We develop algorithmic speed-up techniques in Section 4 and evaluate them computationally in Section 5. We then present case study results in Section 6 and conclude in Section 7.

2 Related Work

Police resource allocation. Police operations in general, and resource allocation in particular, have received sustained interest in the literature. One stream of research focuses on districting: how to divide the city into workload-balanced geographic units, or “beats”. Larson (1972) and Ríos-Mercado (2020) provide a comprehensive review. In Atlanta (which is the focus of our case study at the end of the paper), Zhu et al. (2022) developed a queueing-based workload balancing model which was recently used to adjust the city’s beat and zone boundaries.

Our work takes these boundaries as given, and instead focuses on staffing police personnel, i.e., resource allocation. Initial work mostly modeled police workloads using queueing models (Green, 1984). More recently, researchers typically seek to learn a demand function from data (following, e.g., Gerber, 2014), then allocate patrols to maximize predicted coverage (Leigh et al., 2019; Brandt et al., 2022). A related but distinct stream of work instead develops optimization models to proactively deploy deterrence solutions such as cameras (He et al., 2026).

We take a slightly different approach, using two-stage stochastic programming to find a staffing solution that performs well on average over many possible scenarios drawn from historical data. In a recent survey, Samanta et al. (2022) find few instances of stochastic programming use in police operations. The closest work is by Mukhopadhyay et al. (2016), who also formulate staffing as a first stage and dispatch as a

second stage. Because patrols are rarely unavailable in their application, they can model dispatch as a transportation problem, where the key objective is response time. In contrast, we consider a setting where peak incident demand can stretch thin police resources. Our dispatch problem is thus a coloring problem, more specifically a generalization of the Optimal Cost Chromatic Partition Problem (OCCP) introduced by Jansen (1997), itself a generalization of the Chromatic Sum problem (Kubicka and Schwenk, 1989). In OCCP, vertices of a graph are grouped into independent sets, each associated with a cost that reflects the chosen group (or “color”). The goal is to minimize the total cost while ensuring that adjacent vertices are assigned to different groups. This is a more complex, graph-theoretic second-stage model, which therefore requires more algorithmic machinery to solve efficiently.

Two-stage stochastic mixed-integer programming. Our work therefore seeks to improve police resource allocation by leveraging the rich literature on two-stage stochastic programming, a general framework for decision-making under uncertainty. These problems combine “here and now” first-stage decisions, which must be made before uncertainty is realized, and “wait and see” second-stage decisions, which can adapt to the realized uncertainty.

A classical approach for solving two-stage problems with linear recourse is Benders decomposition (Benders, 1962). Its adaptation to the stochastic programming context, widely known as the L-shaped method (Van Slyke and Wets, 1969), leverages dual information from the second-stage problem to generate cutting planes that iteratively approximate the recourse function. The key idea of Benders decomposition is to leverage dual information from the second-stage problem to generate cutting planes in the first-stage problem. Many computational improvements have since been proposed (see Rahmaniani et al., 2017, for a comprehensive review of acceleration techniques). Ntamo and Sen (2005) find empirically that combinatorial problems with high decomposability are most likely to be solved efficiently.

However, many problems, including our dispatch problem, have a nonconvex or NP-hard second stage. As a result, the second-stage recourse function cannot be recovered entirely using Benders cuts applied to the second-stage linear relaxation. Custom schemes have been developed to recover the recourse function, for example in the special case where the second stage can be reformulated as maximizing a concave function (Cormican et al., 1998). In the more general case where the second stage problem includes integer variables, the most commonly used method is to add a special *integer L-shaped cut*, first proposed by Laporte and Louveaux (1993), in addition to the classical Benders cut. The core integer L-shaped method has also been improved by Angulo et al. (2016), who introduce speedups by solving the LP relaxation instead of the

second-stage IP whenever possible, and Zou et al. (2019), who introduce a more general family of Lagrangian cuts.

Applications in logistics. While the use of two-stage stochastic programming is somewhat sparse in police operations, it is a very common tool in logistics. Many works leverage classical Benders decomposition, either because the second stage is convex (Arab et al., 2015) or because it is well-approximated by a linear program (Cordeau et al., 2001; Dror et al., 1989). The most common use case for integer L-shaped cuts is when the second stage is a vehicle routing problem, with examples in freight network design (Miller-Hooks et al., 2012), last-mile delivery (Gendreau et al., 1995), charging networks for electric fleets (Pagnier et al., 2026), bike-sharing (Parada et al., 2025), and just-in-time logistics (Baals et al., 2026). Another common second stage is facility location, particularly in humanitarian logistics (Rawls and Turnquist, 2010; Sanci and Daskin, 2021). Considering a new kind of second stage problem (as we do in this paper with a coloring problem) involves algorithmic tuning to understand which cut regimes and other speedups add the most computational value. For example, Wang and Jacquillat (2020) find that in a more complex air traffic scheduling second stage problem, new cuts need to be designed to enhance the algorithm’s convergence.

3 Model Formulation

We now introduce our model of staffing and dispatch based on two-stage stochastic programming. We consider a city with a set of zones, denoted by $\mathcal{Z}$. Each zone $z \in \mathcal{Z}$ is divided into a set of beats, denoted by $\mathcal{B}_z$. The time horizon is exogenously divided into a set of shifts $\mathcal{T}$ (e.g., in Atlanta, the typical one-week planning horizon we will consider is divided into 21 eight-hour shifts). Let $\mathcal{U}_z$ be the set of umbrella units that could potentially be activated in zone z , indexed by u . We seek to determine the binary vector $q_{z,t}^u$ describing whether the umbrella unit u is active in zone z in each shift t , in a way that minimizes the total response time to (ex ante unknown) incidents. In a slight abuse of notation, we denote by $\mathcal{U}_{z,t} = \{u \in \mathcal{U}_z : q_{z,t}^u = 1\}$ the set of activated umbrella units in zone z and shift t . A summary of notation is provided in Table 1.

3.1 The Dispatch Problem

After we make our staffing decisions $q_{z,t}^u$, we assume the world will be in one of a set of scenarios $\mathcal{S}$. Each scenario $s \in \mathcal{S}$ is drawn with probability p_s and corresponds to a set of *incidents* $\mathcal{I}_s$, where an incident is an event that requires a police response in a given place at a given time. We let $\mathcal{I}_{s,z,t}$ denote the subset of incidents occurring in shift t and zone z under scenario s . To simplify problem dynamics, we represent the

set of incidents as an undirected *conflict graph* $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s)$, where each incident i corresponds to a node, and there is an edge between incident i and i' if they are *incompatible*, i.e., if the two incidents can't be served by the same police unit. Incompatibility can occur when two incidents happen simultaneously or are close enough in time and far enough in space that it is not feasible for a unit to travel from one to the other. Incident incompatibility is thus independent of the dispatching solution.

We also operate under the assumption that incidents in a zone can only be served by regular units assigned to beats within that zone, or umbrella units activated within that zone. We make a similar assumption for incidents happening in a particular shift $t \in \mathcal{T}$. As a result, we can eliminate edges between incompatible incidents occurring in different zones and shifts, since they cannot be served by the same unit by design. This simplification leads to a conflict graph with $|\mathcal{Z}| \times |\mathcal{T}|$ connected components, each written as $\mathcal{G}_{s,z,t} = (\mathcal{V}_{s,z,t}, \mathcal{E}_{s,z,t})$.

Given a specific realization of demand under scenario s , represented by the incident set $\mathcal{I}_{s,z,t}$ and the associated conflict graph $\mathcal{G}_{s,z,t}$ for a particular zone z and shift t , the second-stage problem involves optimally dispatching available patrol units. This *dispatch problem* aims to assign regular beat units and active umbrella units (determined by the first-stage staffing decisions $\mathbf{q}_{z,t}$) to incidents to minimize total system costs. These costs comprise the sum of travel times for assigned units and a penalty representing the operational burden of incidents not immediately covered by this initial dispatch.

Let $t_{b,i}^s$ denote the estimated travel time for the regular unit assigned to beat $b \in \mathcal{B}_z$ to reach incident $i \in \mathcal{I}_{s,z,t}$ under scenario s . Similarly, $t_{u,i}^s$ is the travel time for an active umbrella unit $u \in \mathcal{U}_z$ to incident i . We define binary decision variables $x_{b,i}^s = 1$ if the regular unit from beat b is assigned to incident i , and $y_{u,i}^s = 1$ if umbrella unit u is assigned to incident i ; both are 0 otherwise.

Even though all incidents must be addressed, we do not treat coverage as a hard constraint that might render the problem infeasible under high demand — in practice, police units can be delayed when there are too many incidents to handle. Instead, given a scenario s , if an incident i cannot be optimally assigned to an available unit from $\mathcal{B}_z$ or an active unit from $\mathcal{U}_{z,t}$ (considering travel times and incompatibility constraints), it is marked by a binary variable $w_i^s = 1$. We also include a penalty parameter m in the objective function, which serves as a proxy for the costs associated with operational delays or the engagement of exceptional measures (e.g., reassigning units from distant, unrelated duties or incurring extensive overtime) required to eventually service such incidents.

The dispatch problem for zone z , shift t , under scenario s , given the set of active umbrella units $\mathbf{q}_{z,t}$, can

then be formulated as follows:

$$Q_{s,z,t}(\mathbf{q}_{z,t}) := \min \sum_{i \in \mathcal{I}_{s,z,t}} \left(\sum_{b \in \mathcal{B}_z} t_{b,i}^s x_{b,i}^s + \sum_{u \in \mathcal{U}_z} t_{u,i}^s y_{u,i}^s + m w_i^s \right) \quad (1a)$$

$$\text{s.t. } x_{b,i}^s + x_{b,j}^s \leq 1, \quad \forall b \in \mathcal{B}_z, (i,j) \in \mathcal{E}_{s,z,t} \quad (1b)$$

$$y_{u,i}^s + y_{u,j}^s \leq 1, \quad \forall u \in \mathcal{U}_z, (i,j) \in \mathcal{E}_{s,z,t} \quad (1c)$$

$$\sum_{b \in \mathcal{B}_z} x_{b,i}^s + \sum_{u \in \mathcal{U}_z} y_{u,i}^s + w_i^s = 1, \quad \forall i \in \mathcal{I}_{s,z,t} \quad (1d)$$

$$y_{u,i}^s \leq q_{z,t}^u, \quad \forall u \in \mathcal{U}_z, i \in \mathcal{I}_{s,z,t} \quad (1e)$$

$$x_{b,i}^s, y_{u,i}^s \in \{0, 1\} \quad \forall b \in \mathcal{B}_z, i \in \mathcal{I}_{s,z,t}, u \in \mathcal{U}_z. \quad (1f)$$

Constraints (1b) and (1c) ensure that a pair of conflicting incidents cannot be served by the same regular or umbrella unit. Constraint (1d) ensures that each incident $i \in \mathcal{I}_{s,z,t}$ is accounted for: it must either be assigned to exactly one regular patrol unit from $\mathcal{B}_z$ (via $x_{b,i}^s = 1$), assigned to exactly one active umbrella unit from $\mathcal{U}_z$ (via $y_{u,i}^s = 1$), or be designated as operationally delayed (via $w_i^s = 1$), incurring the associated penalty m , and constraint (1e) guarantees that an umbrella unit can only attend to incidents if it is active. The objective minimizes the sum of estimated travel time from regular and umbrella units to incidents. We refer to problem (1) as $\text{Dispatch}(\mathcal{B}_z, \mathcal{U}_z, \mathcal{G}_{s,z,t}, \mathbf{q}_{z,t}, \mathbf{t}, m)$.

This formulation can be strengthened by replacing the incompatibility constraints (1b) and (1c) with stronger clique constraints. We first recall the definition of a clique.

Definition 3.1. *In an undirected graph, a clique designates a complete subgraph, i.e., any subgraph induced by a subset of vertices such that each included pair of vertices is connected by an edge. A clique is maximal if adding any other vertex induces a subgraph which is not a clique.*

For each subgraph $\mathcal{G}_{s,z,t}$, let $\mathcal{C}_{s,z,t}$ designate the set of all maximal cliques. Then we can replace (1b) and (1c) as follows:

$$\sum_{i \in C} x_{b,i}^s \leq 1 \quad \forall b \in \mathcal{B}_z, C \in \mathcal{C}_{s,z,t} \quad (2a)$$

$$\sum_{i \in C} y_{u,i}^s \leq 1 \quad \forall u \in \mathcal{U}_z, C \in \mathcal{C}_{s,z,t} \quad (2b)$$

This is a well-known technique to strengthen integer programming formulations of coloring problems, where

clique constraints are often facet-defining (see, e.g., Kubicka and Schwenk, 1989; Kroon et al., 1997, for more details). However, it may lead to a large number of constraints, as we discuss in the next section. In a slight abuse of notation, future references to formulation (1) indicate the formulation where constraints (1b) and (1c) are replaced with maximal clique constraints. We let $v_1^{\text{IP}}(\mathbf{q})$ and $v_1^{\text{LP}}(\mathbf{q})$ respectively designate its optimal objective value and the optimal objective value of its LP relaxation.

The following result establishes the intractability of the dispatch problem.

Proposition 1. *The dispatch problem is NP-hard.*

Proof. We establish a reduction from the Optimal Cost Chromatic Partition Problem (OCCP) to the dispatch problem. OCCP was proven NP-hard by Kroon et al. (1997).

OCCP involves coloring an undirected graph with n colors, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ such that no two adjacent vertices have the same color. Coloring a node with color c costs k_c and the objective is to find a minimum-cost valid coloring. It can be formulated as follows:

$$\min \quad \sum_{v \in \mathcal{V}} \sum_{c=1}^n k_c x_{v,c} \tag{3a}$$

$$\text{s.t.} \quad x_{v,c} + x_{v',c} \leq 1 \quad \forall (v, v') \in \mathcal{E}, \quad c = 1, \dots, n \tag{3b}$$

$$\sum_{c=1}^n x_{v,c} = 1 \quad \forall v \in \mathcal{V} \tag{3c}$$

$$x_{v,c} \in \{0, 1\} \quad \forall v \in \mathcal{V}, \quad c = 1, \dots, n \tag{3d}$$

The objective function (3a) specifies that we are interested in minimizing the total coloring costs. Constraints (3b) guarantee that two nodes v and v' that are connected by an edge $(v, v') \in \mathcal{E}$ are colored by different colors. Constraints (3c) require each node to be colored exactly once. We see that the formulation above corresponds to a special case of problem (1), which we can write as $\text{Dispatch}([n], \emptyset, \mathcal{G}, \mathbf{0}, \mathbf{k}, \infty)$. $\square$

Table 1: Summary of Notations

Sets	
$\mathcal{Z}$	Zones
$\mathcal{T}$	Shifts
$\mathcal{S}$	Second-stage scenarios
$\mathcal{B}_z$	Beats in zone $z \in \mathcal{Z}$ (one patrol unit per beat)
$\mathcal{U}_z$	Umbrella cars eligible for activation in zone $z \in \mathcal{Z}$
$\mathcal{I}_{s,z,t}$	Set of incidents in zone $z \in \mathcal{Z}$ and shift $t \in \mathcal{T}$ under scenario s
Parameters	
$c_{u,z,t}$	Cost of umbrella car $u \in \mathcal{U}_z$, $\forall z \in \mathcal{Z}$
$t_{b,i}^s$	Travel time from regular patrol unit $b \in \mathcal{B}_z$ to incident i under scenario s
$t_{u,i}^s$	Travel time from umbrella patrol unit $u \in \mathcal{U}_z$ to incident i under scenario s
m	Cost of not serving an incident
Decision variables	
$q_{z,t}^u$	Binary variable indicating the availability of umbrella car $u \in \mathcal{U}_z$ in zone $z \in \mathcal{Z}$ and shift $t \in \mathcal{T}$
$x_{b,i}^s$	Binary variable indicating whether regular unit b serves incident i in scenario $s \in \mathcal{S}$
$y_{u,i}^s$	Binary variable indicating whether umbrella unit u serves incident i in scenario $s \in \mathcal{S}$
w_i^s	Binary variable indicating incident i is not served in scenario $s \in \mathcal{S}$

3.2 The Staffing Problem

Given the dispatch problem formulated above, we can write the staffing problem as

$$\min \sum_{t \in \mathcal{T}} \sum_{z \in \mathcal{Z}} \left(\sum_{u \in \mathcal{U}_z} c_{u,z,t} q_{z,t}^u + \mathbb{E}_{\mathcal{S}}[Q_{s,z,t}(\mathbf{q}_{z,t})] \right) \quad (4a)$$

$$\text{s.t. } q_{z,t}^u \in \{0, 1\}, \quad \forall u \in \mathcal{U}_z, z \in \mathcal{Z}, t \in \mathcal{T}. \quad (4b)$$

We notice that this formulation assumes that each umbrella unit can have unique response times and associated costs. If we treat umbrella units within the same zone and shift as perfectly interchangeable (in particular, $t_{u,i}^s = t_i^s$ for all u), we can then group identical umbrella units together and replace the unit-indexed umbrella clique constraints with aggregated clique constraints. The resulting aggregated relaxation

is presented as follows:

$$\min \sum_{i \in \mathcal{I}_{s,z,t}} \left(\sum_{b \in \mathcal{B}_z} t_{b,i}^s x_{b,i}^s + t_i^s y_i^s + m w_i^s \right) \quad (5a)$$

$$\text{s.t.} \quad \sum_{i \in C} x_{b,i}^s \leq 1, \quad \forall b \in \mathcal{B}_z, C \in \mathcal{C}_{s,z,t} \quad (5b)$$

$$\sum_{i \in C} y_i^s \leq \sum_{u \in \mathcal{U}_z} q_{z,t}^u, \quad \forall C \in \mathcal{C}_{s,z,t}, \quad (5c)$$

$$\sum_{b \in \mathcal{B}_z} x_{b,i}^s + y_i^s + w_i^s = 1, \quad \forall i \in \mathcal{I}_{s,z,t}, \quad (5d)$$

$$x_{b,i}^s, y_i^s, w_i^s \in \{0, 1\} \quad \forall i \in \mathcal{I}_{s,z,t}, b \in \mathcal{B}_z. \quad (5e)$$

Compared to formulation (1), formulation (5) has the same $x_{b,i}^s$ variables, but leverages the interchangeability of umbrella units to keep only one assignment variable for all of them. As a result, the new formulation has $(|\mathcal{U}_z| - 1) \cdot |\mathcal{I}|$ fewer variables, and $(|\mathcal{U}_z| - 1) \cdot (|\mathcal{I}| + |\mathcal{C}|)$ fewer constraints. In the rest of the paper, we refer to formulation (5) as Formulation 2, and denote by $v_2^{\text{IP}}(\mathbf{q})$ and $v_2^{\text{LP}}(\mathbf{q})$ the respective optimal objective values of the IP formulation and its LP relaxation. The following lemma establishes the relationship between the two formulations.

Lemma 1. *Fix the available umbrella cars $\mathbf{q}$, and assume that they are interchangeable. Then*

$$v_1^{\text{LP}}(\mathbf{q}) = v_2^{\text{LP}}(\mathbf{q}) \leq v_2^{\text{IP}}(\mathbf{q}) \leq v_1^{\text{IP}}(\mathbf{q}).$$

Proof. Let FLP1 designate the LP relaxation of formulation (1) and FLP2 designate the LP relaxation of formulation (5). We first show that the LP relaxations of the two formulations have the same optimal value. Without loss of generality, we focus on one zone and one shift and drop the corresponding indices.

Consider any feasible solution $(x_{b,i}^*, y_i^*, w_i^*)$ of FLP2. If $q = 0$, constraint (5c) implies $y_i^* = 0$ for every i , and the two formulations are identical. Suppose that $q \geq 1$. For each available umbrella car $u = 1, \dots, q$, define

$$y'_{u,i} = \frac{1}{q} y_i^*.$$

The regular-unit clique constraints are identical in the two formulations and therefore remain satisfied.

Moreover, for every umbrella car u and maximal clique C ,

$$\sum_{i \in C} y'_{u,i} = \frac{1}{q} \sum_{i \in C} y_i^* \leq 1,$$

where the inequality follows from (5c). Hence, the umbrella-unit clique constraints of **FLP1** are satisfied.

For every incident i ,

$$\sum_{b \in \mathcal{B}_z} x_{b,i}^* + \sum_{u=1}^q y'_{u,i} + w_i^* = \sum_{b \in \mathcal{B}_z} x_{b,i}^* + y_i^* + w_i^* = 1.$$

Thus, the assignment constraints are also satisfied. The variable bounds follow from $0 \leq y_i^* \leq 1$. Finally, because the umbrella cars have identical objective coefficients, the objective value is preserved:

$$\sum_{u=1}^q t_i y'_{u,i} = t_i y_i^*.$$

It follows that $v_1^{\text{LP}}(q) \leq v_2^{\text{LP}}(q)$.

Conversely, consider any feasible solution $(x_{b,i}, y_{u,i}, w_i)$ of **FLP1** and define

$$\bar{y}_i = \sum_{u=1}^q y_{u,i}.$$

For every maximal clique C ,

$$\sum_{i \in C} \bar{y}_i = \sum_{u=1}^q \sum_{i \in C} y_{u,i} \leq q.$$

The assignment constraints, variable bounds, and objective value are also preserved. Therefore, $v_2^{\text{LP}}(q) \leq v_1^{\text{LP}}(q)$, implying that $v_1^{\text{LP}}(q) = v_2^{\text{LP}}(q)$.

Since **FLP2** is the LP relaxation of Formulation 2, we have $v_2^{\text{LP}}(q) \leq v_2^{\text{IP}}(q)$. Furthermore, the preceding projection maps every integer feasible solution of Formulation 1 to an integer feasible solution of Formulation 2 with the same objective value. We conclude that $v_2^{\text{IP}}(q) \leq v_1^{\text{IP}}(q)$. $\square$

We emphasize that formulation 2 is a relaxation of formulation 1, in the sense that the last inequality in Lemma 1 may be strict, as shown in Example 1. We also analyze the gap between the two IP formulations numerically in Section 5 and find that it is very small (and often zero) on realistic instances, justifying the practicality of the relaxation.

Example 1. Consider five incidents A, B, C, D , and E , with duration 2 and respective start times 1,

2, 3, 4, and 5, as shown in Figure 1a. The travel time matrix between each pair of incidents is shown in Figure 1b—note that travel times can violate the triangle inequality because they are time-dependent (see, for reference, Ichoua et al., 2003; Gendreau et al., 2015). The resulting conflict graph is shown in Figure 1c. This forms a 5-cycle (C_5). Suppose that $q = 2$ umbrella cars are available and that all five incidents must be assigned to umbrella cars. In formulation 2, consider a solution in which $y_i = 1$ and $x_{b,i} = 0$ for every $i \in I$. Each maximal clique of C_5 consists of one edge. Therefore, for every maximal clique C ,

$$\sum_{i \in C} y_i = 2 \leq q,$$

and thus the clique inequalities in formulation 2 are satisfied. Thus, this solution is a valid integer solution for formulation 2. For formulation 1, each incident would have to be assigned to exactly one of the two umbrella cars, while constraint (1c) would require the endpoints of every edge to be assigned to different patrol units, which would require at least 3 patrol units. Hence, formulation 1 cannot produce a feasible integer solution given $q = 2$ and $x_{b,i} = 0$ for every $i \in I$.

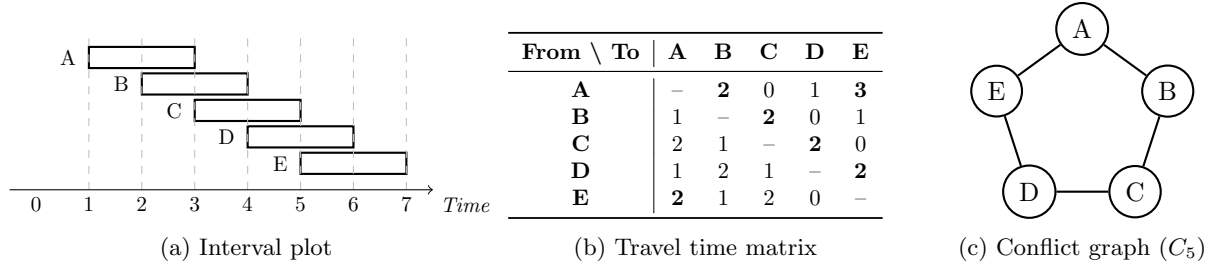


Figure 1: Example of a non-perfect conflict graph due to spatio-temporal constraints.

The conflict graph in Example 1 reveals a subtlety of conflict graph modeling. Even though incidents A and E cannot be served consecutively, it is technically possible for them to be served by the same patrol if that patrol also serves incident C . However, allowing a patrol to serve both incidents would induce a dependence between incident compatibility and unit dispatch, which strongly affects the tractability of our second-stage problem, and introduces practical concerns: for instance, it may not be reasonable to assume the dispatch system is sophisticated enough to identify such compatibility-enhancing decisions. We therefore retain the modeling choice of defining incompatible incidents in a pairwise sense while acknowledging the small downside of inducing a gap between formulation 1 and formulation 2, especially since numerical experiments suggest that this gap is often near zero.

Structural properties and symmetry-breaking constraints. We now establish a simple property of the dispatch problem: the objective value monotonically improves as the number of available umbrella units increases, up to a critical threshold. This property can significantly reduce the computational burden, as we will explore in Section 4.

Lemma 2. *If umbrella units are interchangeable, $Q_{s,z,t}(\mathbf{q}_{z,t})$ is non-increasing as the first-stage decision variable $|\mathbf{q}_{z,t}|$ increases $\forall s \in \mathcal{S}$, where $|\mathbf{q}_{z,t}| = \sum_{z \in \mathcal{Z}} \sum_{t \in \mathcal{T}} q_{z,t}^u$.*

Proof. Let $\mathbf{q}_{z,t}^1$ and $\mathbf{q}_{z,t}^2$ denote the number of umbrella cars such that $|\mathbf{q}_{z,t}^1| < |\mathbf{q}_{z,t}^2|$. We find that $Q(\mathbf{q}_{z,t}^2)$ is a relaxation of $Q(\mathbf{q}_{z,t}^1)$, and thus $Q(\mathbf{q}_{z,t}^2) \leq Q(\mathbf{q}_{z,t}^1)$. $\square$

While we can replace the staffing decision as a binary vector with an integer for the input taken in the second-stage problem, as in (5), the binary vector representation may remain necessary in the first stage of general assignment problems because it is required for the validity of the integer L-shaped method. We can add symmetry-breaking constraints to the first-stage staffing problem to eliminate different solutions with the same objective value:

$$\min \quad \sum_{t \in \mathcal{T}} \sum_{z \in \mathcal{Z}} \left(\sum_{u \in \mathcal{U}_z} c_{u,z,t} q_{z,t}^u + \mathbb{E}_{\mathcal{S}}[Q_{s,z,t}(\mathbf{q}_{z,t})] \right) \quad (6a)$$

$$\text{s.t.} \quad q_{z,t}^i \leq q_{z,t}^j, \quad \forall i > j \quad (6b)$$

$$q_{z,t}^u \in \{0, 1\}, \quad \forall u \in \mathcal{U}_z, z \in \mathcal{Z}, t \in \mathcal{T}. \quad (6c)$$

4 Decomposition Strategy

We now present a decomposition strategy to solve the two-stage problem proposed in the previous section.

Our methodology relies on three components: the first two are a *caching strategy* to store and reuse subproblem solutions (reducing redundant computations) and a fine-grained multi-cut strategy. While generating a separate cut for each of the $|\mathcal{S}| \times |\mathcal{Z}| \times |\mathcal{T}|$ subproblems could be expensive, our caching-informed approach with parallel computing makes it tractable. This combination allows us to realize the benefits of tighter problem approximations from the multi-cut method at minimal computational cost. The third component is a graph-theoretic warm-start heuristic that uses structural insights from the conflict constraints.

We first review the standard integer L-shaped method and its hierarchical cut-generation rule. We then

introduce the caching and multi-cut strategies used in our customized method, followed by the cost-informed warm start.

4.1 The Integer L-shaped Method

Benders decomposition. Our solution methodology is based on the integer L-shaped method, which extends the classical Benders decomposition to problems with mixed-integer recourse. We first describe the implementation of Benders cuts for our problem. Let $\mathbf{q}$ be the aggregated vector of vectors $\mathbf{q}_{z,t}, z \in \mathcal{Z}, t \in \mathcal{T}$. Let $\mathcal{Q}(\mathbf{q})$ denote $\sum_{t \in \mathcal{T}} \sum_{z \in \mathcal{Z}} \mathbb{E}_{\mathcal{S}} [Q_{s,z,t}(\mathbf{q}_{z,t})]$ for convenience of notation, so that $\mathcal{Q}(\mathbf{q})$ represents the value of the second stage objective function given $\mathbf{q}$. Then the model presented in Section 3 is equivalent to the following epigraph reformulation:

$$\min \quad \sum_{t \in \mathcal{T}} \sum_{z \in \mathcal{Z}} \sum_{u \in \mathcal{U}_z} c_{u,z,t} q_{z,t}^u + \theta \quad (7a)$$

$$\text{s.t.} \quad 6b, 6c, \quad q_{z,t}^u \in \{0, 1\}, \forall u \in \mathcal{U}_z, z \in \mathcal{Z}, t \in \mathcal{T} \quad (7b)$$

$$\theta \geq \mathcal{Q}(\mathbf{q}), \quad (7c)$$

$$\theta \geq 0. \quad (7d)$$

Note that in formulation (1), we take the binary vector $\mathbf{q}$ as the input to the second stage, while we sum the binary vector to an integer in formulation 5. Let $\eta_{s,z,t}^{|C|}$ be the optimal dual values associated with Constraints 5c in the linear relaxation, where $|C|$ is the number of maximal cliques for $s \in \mathcal{S}, z \in \mathcal{Z}, t \in \mathcal{T}$. Let $\mathcal{Q}_{\text{LP}}(\mathbf{q})$ be the value of linear relaxation of $\mathcal{Q}(\mathbf{q})$, and let $\bar{\mathcal{Q}}_{\text{LP}}(\mathbf{q})$ denote the optimal value of $\mathcal{Q}_{\text{LP}}(\mathbf{q})$. For every solution $\mathbf{q}^* \in [0, 1]^{|\mathcal{U}_z|}$, we have the following standard Benders optimality cut added to our master problem:

$$\theta \geq \sum_{t \in \mathcal{T}} \sum_{z \in \mathcal{Z}} \left(\sum_{s \in \mathcal{S}} \eta_{s,z,t}(\mathbf{q}_{z,t}^*) \right) (\mathbf{q}_{z,t} - \mathbf{q}_{z,t}^*) + \bar{\mathcal{Q}}_{\text{LP}}(\mathbf{q}^*), \quad (8)$$

The Benders optimality cuts (8) generated in each iteration produce an outer approximation of constraint (7c). Note that the recourse problem is always feasible due to the penalty variables w_i^s (which ensure any incident can be “uncovered” at a cost), thus Benders feasibility cuts are not required.

Integer L-shaped cuts. Benders decomposition is an exact approach when the second-stage problem is a linear program, but not necessarily when it is a mixed-integer program: considering only subgradients

of the LP relaxation may lead to underestimating the second-stage costs. To address this issue, Laporte and Louveaux (1993) proposed the integer L-shaped method for two-stage problems with binary first-stage decisions. We adopt this approach in our problem: given the optimal solution $\mathbf{q}^*$ of the first stage and the associated cost for the second-stage optimization problem $\mathcal{Q}(\mathbf{q}^*)$, we can write the integer optimality cut as follows:

$$\theta \geq (\mathcal{Q}(\mathbf{q}^*) - L) \left(\sum_{i \in S(\mathbf{q}^*)} \mathbf{q}_i - \sum_{i \notin S(\mathbf{q}^*)} \mathbf{q}_i - |S(\mathbf{q}^*)| \right) + \mathcal{Q}(\mathbf{q}^*), \quad (9)$$

where $\mathbf{q}^*$ is the optimal solution to the master problem (MP) in the current iteration, $S(\mathbf{q}^*)$ represents the indices of components of the vectors $\mathbf{q}^*$ that are equal to 1, and $\mathcal{Q}(\mathbf{q}^*)$ is the corresponding cost of the second-stage optimization problem. Note that the integer optimality cuts tend not to be very tight in regions further away from the current solution — this is why in the literature, integer L-shaped cuts are often used in conjunction with Benders subgradient cuts.

Method improvements. Repeatedly solving the second-stage problem (a mixed-integer optimization problem) for different values of the first-stage variables $\mathbf{q}$ can be computationally expensive. Angulo et al. (2016) proposed a simple yet powerful hierarchical strategy, only computing the next integer L-shaped cut when no Benders cuts are currently violated. In our problem, this means that given a candidate solution $(\bar{\mathbf{q}}, \bar{\theta})$, the classical integer L-shaped method evaluates $\mathcal{Q}(\bar{\mathbf{q}})$ exactly and rejects $(\bar{\mathbf{q}}, \bar{\theta})$ if $\bar{\theta} < \mathcal{Q}(\bar{\mathbf{q}})$. Since $\mathcal{Q}_{\text{LP}}(\bar{\mathbf{q}}) \leq \mathcal{Q}(\bar{\mathbf{q}})$, it is sufficient to discard $(\bar{\mathbf{q}}, \bar{\theta})$ when $\bar{\theta} < \mathcal{Q}_{\text{LP}}(\bar{\mathbf{q}})$. Therefore, instead of calculating $\mathcal{Q}(\bar{\mathbf{q}})$, which is an MIP, at each iteration, we begin by evaluating $\mathcal{Q}_{\text{LP}}(\bar{\mathbf{q}})$ and checking whether $\bar{\theta} < \mathcal{Q}_{\text{LP}}(\bar{\mathbf{q}})$. If this condition is met, we add a subgradient cut to eliminate the candidate solution. Otherwise, we compute $\mathcal{Q}(\bar{\mathbf{q}})$ and check whether $\bar{\theta} < \mathcal{Q}(\bar{\mathbf{q}})$; if this condition is met, we apply the integer optimality cut. Otherwise, we accept the solution. In other words, we use the LP relaxation as a proxy to check the feasibility of any candidate solution, reducing the number of times we need to solve the subproblem exactly.

Algorithm 1 gives a high-level overview of our approach to the integer L-shaped method, before we describe the key innovative components in detail. The warm-start procedure is described in Section 4.3. The separation subroutine is presented in Section 4.2.2, and relies on cache-based evaluation in Section 4.2.1. We let $r \in \{\text{LP}, \text{IP}\}$ denote the recourse phase and write $\mathcal{K}_{s,z,t}^r$ for the corresponding phase- r cache. We also let $k_{z,t}(q) := \sum_{u \in U_z} q_{z,t}^u$.

Algorithm 1 Overall Cache-Based Multi-Cut Integer L-Shaped Method

```
1: Initialize  $\mathcal{K}_{s,z,t}^r$  for every  $(s, z, t) \in \mathcal{S} \times \mathcal{Z} \times \mathcal{T}$  and  $r \in \{\text{LP}, \text{IP}\}$ .
2: Warm-start the master problem as in Section 4.3.
3: Solve the master with the following callback, given a candidate solution  $\bar{q}, \bar{\theta}$ :
4: procedure CANDIDATECALLBACK( $\bar{q}, \bar{\theta}$ )
5:    $\mathcal{B}_{\text{LP}} \leftarrow \{\text{SEPARATE}(\text{LP}, s, z, t, \bar{q}_{z,t}, \bar{\theta}_{s,z,t}) : (s, z, t) \in \mathcal{S} \times \mathcal{Z} \times \mathcal{T}\}$ 
6:   if  $\mathcal{B}_{\text{LP}} \neq \emptyset$  then
7:     Add all cuts in  $\mathcal{B}_{\text{LP}}$ .
8:   return
9:   end if
10:   $\mathcal{B}_{\text{IP}} \leftarrow \{\text{SEPARATE}(\text{IP}, s, z, t, \bar{q}_{z,t}, \bar{\theta}_{s,z,t}) : (s, z, t) \in \mathcal{S} \times \mathcal{Z} \times \mathcal{T}\}$ 
11:  if  $\mathcal{B}_{\text{IP}} \neq \emptyset$  then
12:    Add all cuts in  $\mathcal{B}_{\text{IP}}$ .
13:  else
14:    Accept the candidate as optimal.
15:  end if
16: end procedure
17: return the optimal first-stage solution  $q^*$ .
```

4.2 Customized Acceleration Techniques

Beyond the above techniques from the literature, we can further accelerate the integer L-shaped method by leveraging the structure of our problem.

4.2.1 LP and IP Caching Strategy.

The iterative nature of the L-shaped method, particularly when solving for integer first-stage variables $\mathbf{q}$, can lead to redundant evaluations of the second-stage recourse function $Q_{s,z,t}(\mathbf{q}_{z,t})$ and its linear programming relaxation $Q_{\text{LP}(s,z,t)}(\mathbf{q}_{z,t})$. Such redundancies arise when the algorithm revisits identical first-stage sub-solutions for specific zones and shifts (i.e., identical vectors $\mathbf{q}_{z,t}$) across different overall candidate solutions $\mathbf{q}$, or when it explores solutions $\mathbf{q}'_{z,t}$ that are known to yield the same optimal second-stage value as a previously evaluated $\mathbf{q}_{z,t}$ due to problem structure.

Specifically, two sources of computational redundancy are addressed through caching:

1. **Repeated Evaluation of Identical Subproblems:** If a candidate first-stage solution $\mathbf{q}^*$ has components $\mathbf{q}_{z,t}^*$ that are identical to those of a previously evaluated solution for a given (z, t) tuple, then for each $s \in \mathcal{S}$, the corresponding $Q_{s,z,t}(\mathbf{q}_{z,t}^*)$ and $Q_{\text{LP}(s,z,t)}(\mathbf{q}_{z,t}^*)$ will be the same. Storing and retrieving these values from a cache avoids re-solving these subproblems.
2. **Exploitation of Monotonicity:** Lemma 2 establishes that the second-stage cost function, $Q_{s,z,t}(\mathbf{q}_{z,t})$, is non-increasing with respect to the total number of active umbrella units, $k = |\mathbf{q}_{z,t}| = \sum_u q_{z,t}^u$.

Suppose we have evaluated the recourse function for two staffing levels, k_1 and k_2 , corresponding to specific first-stage solutions $\mathbf{q}_{z,t}^{(1)}$ and $\mathbf{q}_{z,t}^{(2)}$ respectively, such that $k_1 < k_2$. If it is found that $Q_{s,z,t}(\mathbf{q}_{z,t}^{(1)}) = Q_{s,z,t}(\mathbf{q}_{z,t}^{(2)})$, then due to the monotonicity established in Lemma 2, it follows that for any staffing level k_3 (corresponding to a solution $\mathbf{q}_{z,t}^{(3)}$) where $k_1 \leq k_3 \leq k_2$, the recourse cost $Q_{s,z,t}(\mathbf{q}_{z,t}^{(3)})$ will also be equal to $Q_{s,z,t}(\mathbf{q}_{z,t}^{(1)})$. This structural property implies that once the recourse function becomes constant over an interval of increasing resource levels, the values for intermediate levels within that interval can be inferred without explicit computation. Caching these constant recourse values for identified intervals significantly reduces redundant evaluations of $Q_{s,z,t}(\cdot)$.

Algorithm 2 describes our cache-first evaluation strategy in pseudocode.

Algorithm 2 Cache-First Evaluation for Subproblem (s, z, t)

```

1: procedure EVALUATE( $r, s, z, t, q_{z,t}$ )
2:    $k \leftarrow \mathbf{1}^\top q_{z,t}$ 
3:   if  $\mathcal{K}_{s,z,t}^r$  has an exact record for  $q_{z,t}$  then
4:     return the exact record.
5:   end if
6:   if  $\mathcal{K}_{s,z,t}^r$  certifies  $k$  from a constant-value interval then
7:     return the certified value (and a certified LP subgradient, if available).
8:   end if
9:    $(v, g) \leftarrow \text{Solve}_r(s, z, t, q_{z,t})$   $\triangleright g = \text{NONE}$  for IP
10:  INSERT( $\mathcal{K}_{s,z,t}^r, q_{z,t}, k, v, g$ )
11:  return  $(v, g)$ 
12: end procedure

```

4.2.2 Multi-Cut Strategy

While classical implementations often aggregate second-stage information into a single cut (e.g., the integer L-shaped cut in Equation (9) representing the expected recourse), a multi-cut strategy, proposed by Birge and Louveaux (1988) can offer a more refined approximation, particularly when subproblem structures are distinct or when seeking computational advantages from parallelization. In a multi-cut framework, separate cuts are generated for the recourse function associated with each scenario or independent subproblem. The multi-cut strategy traditionally generates one set of cuts per scenario $s \in \mathcal{S}$.

However, we observe that our problem possesses a finer-grained decomposability. Due to the assumption that units in a given zone and shift only respond to incidents within that same zone and shift, the recourse function $\mathcal{Q}(\mathbf{q})$ naturally separates not just by scenario s , but also by zone z and shift t .

This structural property allows us to formulate a significantly more disaggregated master problem by in-

roducing auxiliary variables $\theta_{s,z,t}$ for each tuple (s, z, t) , effectively creating $|\mathcal{S}| \times |\mathcal{Z}| \times |\mathcal{T}|$ independent subproblems.

Let $\mathbf{q}$ be the global vector of all first-stage binary decisions $q_{z,t}^u$. The master problem can be formulated to include auxiliary variables $\theta_{s,z,t}$, each representing the cost of the subproblem (s, z, t) :

$$\min_{\mathbf{q}, \boldsymbol{\theta}} \left(\sum_{t \in \mathcal{T}} \sum_{z \in \mathcal{Z}} \sum_{u \in \mathcal{U}_z} c_{u,z,t} q_{z,t}^u + \sum_{s \in \mathcal{S}} p_s \sum_{t \in \mathcal{T}} \sum_{z \in \mathcal{Z}} \theta_{s,z,t} \right) \quad (10)$$

subject to first-stage constraints, and dynamically generated cuts for each $\theta_{s,z,t}$. When a candidate first-stage solution $\mathbf{q}^*$ is obtained, the following types of multi-cuts can be generated for each subproblem (s, z, t) based on its specific input $\mathbf{q}_{z,t}^*$ (the subvector $\mathbf{q}^*$).

Benders Optimality Multi-Cuts. These cuts are derived from the LP relaxation of each subproblem (s, z, t) . Let $\bar{\mathcal{Q}}_{LP_{s,z,t}}(\mathbf{q}_{z,t}^*)$ be the optimal value of this LP relaxation given $\mathbf{q}_{z,t}^*$. Let $\boldsymbol{\eta}_{s,z,t}(\mathbf{q}_{z,t}^*)$ be the vector of subgradients derived from optimal dual multipliers of the LP relaxation of the recourse function $\mathcal{Q}_{LP_{s,z,t}}(\mathbf{q}_{z,t})$ with respect to $\mathbf{q}_{z,t}$, evaluated at $\mathbf{q}_{z,t}^*$. The standard Benders optimality multi-cut for subproblem (s, z, t) is:

$$\theta_{s,z,t} \geq \bar{\mathcal{Q}}_{LP_{s,z,t}}(\mathbf{q}_{z,t}^*) + \boldsymbol{\eta}_{s,z,t}(\mathbf{q}_{z,t}^*)^T (\mathbf{q}_{z,t} - \mathbf{q}_{z,t}^*). \quad (11)$$

These cuts provide valid linear supports for the convex function $\mathcal{Q}_{LP_{s,z,t}}(\mathbf{q}_{z,t})$ which can be added to the master problem.

Integer L-Shaped Optimality Multi-Cuts. Let $\mathcal{Q}_{s,z,t}(\mathbf{q}_{z,t}^*)$ be the true optimal objective value of the integer programming subproblem for tuple (s, z, t) given first-stage decisions $\mathbf{q}_{z,t}^*$. Let $L_{s,z,t}$ be a valid global lower bound for $\mathcal{Q}_{s,z,t}(\mathbf{q}_{z,t})$ such that $\mathcal{Q}_{s,z,t}(\mathbf{q}_{z,t}^*) \geq L_{s,z,t}$. Define the local index sets based on the components of $\mathbf{q}_{z,t}^*$: $S(\mathbf{q}_{z,t}^*) = \{u \in \mathcal{U}_z \mid \bar{q}_{z,t}^u = 1\}$. The classical integer L-shaped optimality cut for subproblem (s, z, t) is formulated as:

$$\theta_{s,z,t} \geq \mathcal{Q}_{s,z,t}(\mathbf{q}_{z,t}^*) + (\mathcal{Q}_{s,z,t}(\mathbf{q}_{z,t}^*) - L_{s,z,t}) \left(\sum_{u \in S(\mathbf{q}_{z,t}^*)} q_{z,t}^u - \sum_{u \notin S(\mathbf{q}_{z,t}^*)} q_{z,t}^u - |S(\mathbf{q}_{z,t}^*)| \right) \quad (12)$$

This cut ensures that if the master problem variables $\mathbf{q}_{z,t}$ take the values of $\mathbf{q}_{z,t}^*$, then $\theta_{s,z,t} \geq \mathcal{Q}_{s,z,t}(\mathbf{q}_{z,t}^*)$. The combinatorial term correctly evaluates to 0 at $\mathbf{q}_{z,t} = \mathbf{q}_{z,t}^*$ and to an integer ≤ -1 if $\mathbf{q}_{z,t} \neq \mathbf{q}_{z,t}^*$.

4.2.3 Parallel Recourse Evaluation and Summary

The fine-grained decomposition in Section 4.2.2 also enables us to solve subproblems in parallel. Conditional on a first-stage staffing vector $\mathbf{q}$, the subproblem $Q_{s,z,t}(\mathbf{q}_{z,t})$ depends only on the local staffing decisions $\mathbf{q}_{z,t}$. Consequently, all the subproblems and their LP relaxations can be evaluated independently.

We exploit this structure using a persistent pool of P worker processes. For each master solution, the parent process first queries the cache for every (s, z, t) block and only dispatches subproblems without cached solutions. Each worker owns a fixed subset of subproblems, constructs the associated LP and IP models once, and reuses them across iterations by updating only the capacity right-hand sides. The workers return objective values and, for LP subproblems, dual multipliers to the parent process, which updates the cache, generates cuts, and resolves the master problem.

We summarize the multi-cut generation step in Algorithm 3.

Algorithm 3 Separation Callback for Subproblem (s, z, t)

```

1: procedure SEPARATE( $r, s, z, t, \bar{q}_{z,t}, \bar{\theta}_{s,z,t}$ )
2:    $(v, g) \leftarrow \text{EVALUATE}(r, s, z, t, \bar{q}_{z,t})$ 
3:   if  $\bar{\theta}_{s,z,t} \geq v$  then
4:     return NOCUT
5:   end if
6:   if  $r = \text{LP}$  then
7:     return BENDERSCUT( $s, z, t, \bar{q}_{z,t}, v, g$ ) ▷ Equation (11)
8:   else
9:     return INTEGERLSHAPEDCUT( $s, z, t, \bar{q}_{z,t}, v, L_{s,z,t}$ ) ▷ Equation (12)
10:  end if
11: end procedure

```

4.3 Cost-Informed Warm-start

We also propose a way of generating an initial “warm-start” solution for the first-stage decision variables which depends on the cost of umbrella units $c_{u,z,t}$ relative to the penalty m for unserved incidents.

In scenarios where the cost of deploying umbrella units is high compared to the delay penalty, the optimal staffing level for umbrella units is likely minimal. For such cases, allowing the L-shaped method to use the “natural” starting point where no umbrella units are initially active is generally effective.

However, a more sophisticated warm-start becomes useful when umbrella units are cheap relative to delay penalties. Under these conditions, the model is incentivized to deploy more umbrella units to ensure high incident coverage. Simply starting with a minimal deployment would require many iterations to build up

the necessary staffing levels. Therefore, for this regime, we employ a warm-start strategy which leverages the *maximum clique number* of the incident conflict graph.

Definition 4.1. *The maximum clique number of an undirected graph $\mathcal{G}$, denoted by $\omega(\mathcal{G})$, is the number of vertices of its largest maximal clique.*

It turns out that the maximum clique number is closely related (but easier to compute) to the *chromatic number* of a graph.

Definition 4.2. *The chromatic number of an undirected graph $\mathcal{G}$, denoted by $\chi(\mathcal{G})$, is the minimum number of colors necessary to assign each vertex a color without assigning any of its neighbors the same color.*

In our case, the chromatic number $\chi(\mathcal{G}_{s,z,t})$ of the incident conflict graph represents the minimum number of patrol units required to serve all incidents in subproblem (s, z, t) without conflict. It turns out that the chromatic number is always lower-bounded by the maximum clique number; any graph for which they are equal on every induced subgraph is called a perfect graph.

Unfortunately, we found in practice that the second-stage conflict graph is not guaranteed to be perfect, implying that the chromatic number $\chi(G)$ may be larger than the clique number $\omega(G)$. Thus, $\omega(G)$ can only serve as a lower bound when estimating the minimum number of patrol units required. In Example 1, the clique number is $\omega(C_5) = 2$ and the chromatic number is $\chi(C_5) = 3 \neq \omega(C_5)$, and the resulting conflict graph $G = C_5$ is not perfect.

While computing both chromatic and clique numbers is NP-hard (Boppana and Halldórsson, 1992), the chromatic number is much harder to approximate than the clique number (Zuckerman, 2007). In practice, computing $\omega(G)$ is more tractable than computing or tightly approximating $\chi(G)$, and can be done using classical algorithms available in standard Python graph packages such as `NetworkX`.

Our warm-start strategy sets an initial value for the first-stage variables $q_{z,t}^u$ by estimating the number of umbrella units needed to handle the most demanding scenarios for each zone z and shift t . The procedure is as follows. First, we determine a proxy for the total number of units required in the worst-case for (z, t) by calculating the maximum clique number across all scenarios, denoted $\omega_{\max}(z, t) = \max_{s \in \mathcal{S}} \omega(\mathcal{G}_{s,z,t})$. Given that $|\mathcal{B}_z|$ regular units are already available, the required number of umbrella units is the difference:

$$N_{z,t} = \max(0, \omega_{\max}(z, t) - |\mathcal{B}_z|)$$

We then construct the initial solution, $q^{\text{warm-start}}$, by setting the first $N_{z,t}$ binary variables in the vector $q_{z,t}$

to 1 for each (z, t) , and the remainder to 0.

5 Computational Experiments

In this section, we conduct a series of computational experiments to evaluate the performance of our proposed Cache-Based Multi-Cut (CB-MC) algorithm.

5.1 Experimental Setup

We begin by generating instances of varying size to assess our method’s scalability and effectiveness against benchmark methods.

5.1.1 Instance Generation

Our instance generation process uses a dataset from the Atlanta Police Department (APD). The city of Atlanta is divided into six zones, each subdivided into 10 to 15 beats, for a total of 78 beats. Our dataset contains 911 (emergency) call records from APD, spanning from 2013 to 2017. Each record corresponds to a unique incident that triggered a police response. It includes the geographic location of the incident (including the beat and zone), detailed timestamps of the police response (call receipt, dispatch, arrival at the scene, and clearance), and information about the patrol unit that handled the incident.

Scenario Generation via Stratified Sampling. The generation process follows a profile-based historical stratified sampling approach. We first profile each day in the 2017 dataset along two dimensions: operational intensity (Low, Medium, High) and spatio-temporal pattern (four dominant patterns identified via k-means clustering on zone-shift incident vectors). This creates a library of representative “day types.” Our final scenario sets are then constructed via stratified sampling from this library, ensuring the mix of scenarios in our test instances reflects the observed annual distribution of operational conditions.

Conflict Graph Construction. To implement the incompatibility constraints introduced in Section 3, we construct a conflict graph $\mathcal{G}_{s,z,t} = (\mathcal{V}_{s,z,t}, \mathcal{E}_{s,z,t})$ for each scenario s , zone z , and shift t . Let t_i^{start} denote the start time of incident i , $t_{i,i'}$ represent the estimated travel time between the locations of i and i' , and t_i^{serv} denote the service time associated with incident i .

Historical service durations are computed directly from the dataset as $t_i^{\text{serv}} = t_i^{\text{clear}} - t_i^{\text{arrival}}$, where t_i^{arrival} and t_i^{clear} denote the recorded arrival and clearance times at the incident scene, respectively. An edge (i, i')

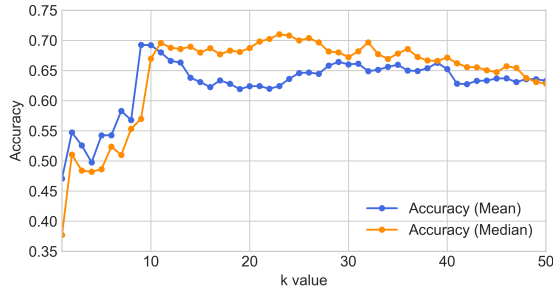
is added to $\mathcal{E}_{s,z,t}$ if a unit that completes service at incident i and then travels to incident i' would not arrive before $t_{i'}^{\text{start}}$, or symmetrically, if the reverse is infeasible. Formally, a conflict is introduced if:

$$t_i^{\text{start}} + t_i^{\text{serv}} + t_{i,i'} > t_{i'}^{\text{start}} \quad \text{and} \quad t_{i'}^{\text{start}} + t_{i'}^{\text{serv}} + t_{i',i} > t_i^{\text{start}}.$$

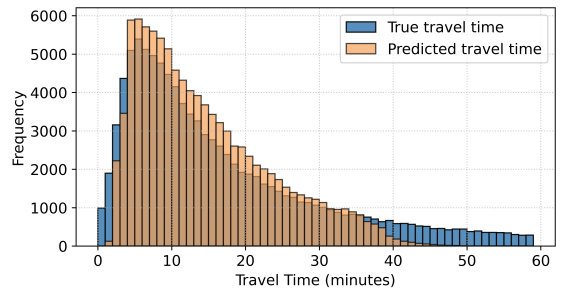
Travel times $t_{i,i'}$ and $t_{i',i}$ are estimated using shortest-path distances on the road network derived from OpenStreetMap (OSM) data, assuming nominal travel speeds.

Parameter Estimation: Unit-to-Incident Travel Times. A critical input to our dispatch model is the estimated travel time from any potentially assignable patrol unit (regular unit b or umbrella unit u) to any given incident i (denoted t_{bi} and t_{ui} , respectively). However, the historical 911 dataset only records the *actual* travel time for the single unit that was dispatched. We lack the information needed to calculate the counterfactual travel times—what the travel time would have been if a different unit had been dispatched. Instead, we estimate counterfactual travel times using a K-nearest neighbors (KNN) approach: To estimate the travel time from the regular unit in beat b to incident i for Regular Units (t_{bi}), we identify the k historical incidents spatially closest to incident i that occurred during the same hour and were handled by a regular unit from beat b . The median of their recorded travel times serves as the estimate for t_{bi} . Similarly, for Umbrella Units, we find the k closest historical incidents handled by an umbrella unit from zone z during the same hour. The median of their travel times provides the estimate for t_{ui} .

We determine the number of neighbors k via 5-fold cross-validation, and select $k^* = 23$ to maximize the coefficient of determination (R^2), as shown in Figure 2a. While this data-driven estimation has inherent limitations, such as difficulty capturing outliers, Figure 2b suggests that the predicted travel times are reasonable.



(a) R^2 as a function of k



(b) Histogram comparison of true vs. predicted travel times

Figure 2: Validation of the KNN-based travel time estimation model.

5.1.2 Benchmark Algorithms and Testbed

We evaluate the performance of our proposed **Cache-Based Multi-cut (CB-MC)** method against four baselines from the literature, as well as ablated versions of the CB-MC method: more precisely, we present the computational performance of solving the entire staffing MIP without decomposition (**Direct**), the classic integer L-shaped method (**Classic**), the improved integer L-shaped method (**Improved**), and a Benders-plus-Lagrangian-cut (**B+L**) method based on Zou et al. (2019). At each outer iteration, **B+L** generates the standard LP-based Benders cuts and solves the integer recourse problems exactly. Instead of integer L-shaped cuts, the method combines standard Benders cuts with the Lagrangian cuts proposed by Zou et al. (2019), computed using a Polyak subgradient algorithm. We also evaluate the three customized methods separately, and the method that combines all three techniques. Methods are summarized in Table 2.

Table 2: Summary of the solution methods used in the computational comparison.

Method	Abbreviation	Description
Direct		Solving the extensive form of the MIP directly with Gurobi.
Classic		Classic integer L-shaped method.
Improved		Improved integer L-shaped method from Angulo et al. (2016).
B+L		Combining Benders cuts with Lagrangian cuts (Zou et al., 2019).
CachedIP		Improved method with only IP Caching enabled.
CachedLP		Improved method with only LP Caching enabled.
F-MC		Improved method with only Fine-grained Multi-Cut enabled.
CB-MC		The fully integrated proposed method with all enhancements.

Summary of Test Instances. To systematically evaluate algorithm performance, we generate test instances of different sizes, letting the number of scenarios $|\mathcal{S}|$ vary in $\{10, 30, 50, 100\}$. The APD planning day is divided into three non-overlapping eight-hour shifts (7am-3pm, 3pm-11pm, 11pm-7am). Across 8,910 zone-shift observations, the 5th percentile, median, and 90th percentile of the number of incidents are 49, 103, and 194, respectively. Therefore, to simulate low-demand, typical, and high-demand zone-shift workloads, we let the number of incidents $|\mathcal{I}|$ vary in $\{50, 100, 200\}$.

All experiments were conducted on a high-performance computing cluster using Python 3.11.7 and Gurobi 12.0.3, with each matched experiment allocated 16 CPU cores and 64 GB of memory. Under the serial computing model the master problem and subproblems are run on a single process. Under parallel computing the master problem is run on one thread and the subproblems are pooled across 6 processes (6×1). A time limit of 3600 seconds was imposed for each run, and the primary performance metrics are total CPU time and the final optimality gap reported by the solver.

5.2 Evaluation of Speed-up Strategies

We present comprehensive results using Formulation 1 in Table 3 (serial) and Table 4 (parallel); and results using Formulation 2 in Table 5 (serial) and Table 6 (parallel). We compare the overall running time in seconds for different instance sizes with a maximum budget of 30 umbrella units. We average running times over 10 repetitions. In each computation, we solve the instance to optimality, i.e., with zero gap. Because we use the **Improved** method as the benchmark for comparison, we show $\frac{\text{Running Time}(\text{Improved})}{\text{Running Time}(\text{CB-MC})}$, the speed-up factor of **Cache-Based Multi-cut** compared to **Improved**.

Table 3: Mean running time (seconds) over 10 replications for Formulation 1 under serial computing. Bold italics identify the fastest baseline; bold identifies the fastest method overall. The speed-up factor is computed as **Improved**/CB-MC.

Instance		Formulation 1: Serial Computing								
$ S $	$ I $	Direct	Classic	Improved	B+L	CachedIP	CachedLP	F-MC	CB-MC	Speed-up factor
10	50	61.4	71.5	<i>39.3</i>	377.3	33.4	25.0	37.1	21.3	1.84
	100	462.7	192.1	<i>100.8</i>	916.8	83.4	64.0	83.4	47.7	2.11
	200	–	979.3	<i>470.4</i>	–	360.4	334.9	338.1	213.7	2.20
30	50	379.6	165.2	<i>91.6</i>	879.3	80.7	56.3	88.9	48.9	1.87
	100	–	799.5	<i>413.2</i>	–	349.6	274.1	356.1	203.5	2.03
	200	–	3511.5	<i>1254.5</i>	–	1123.4	784.0	1163.8	684.4	1.83
50	50	1712.7	261.8	<i>155.6</i>	1419.2	137.7	100.0	151.8	87.7	1.78
	100	–	2321.2	<i>923.8</i>	–	844.0	465.5	895.6	410.0	2.25
	200	–	–	<i>2219.1</i>	–	1951.4	1405.1	2138.9	1198.5	1.85
100	50	–	818.1	<i>334.7</i>	3177.8	296.7	227.8	326.0	193.7	1.73
	100	–	–	<i>1895.5</i>	–	1515.8	1083.4	1350.1	699.5	2.71
	200	–	–	–	–	–	3517.3	–	2398.3	–
Average speed-up factor										2.02

Table 4: Mean running time (seconds) over 10 replications for Formulation 1 under the 6×1 parallel computing setting. **Direct** is reported as the serial computing baseline. Bold italics identify the fastest baseline; bold identifies the fastest method overall. The speed-up factor is computed as **Improved/CB-MC**.

Instance		Formulation 1: Parallel Computing 6×1								
$ S $	$ I $	Direct	Classic	Improved	B+L	CachedIP	CachedLP	F-MC	CB-MC	Speed-up factor
10	50	61.4	18.7	<i>11.6</i>	81.5	11.6	11.6	11.6	10.9	1.06
	100	462.7	50.6	<i>18.8</i>	231.8	17.4	18.0	17.5	16.0	1.18
	200	–	320.8	<i>151.1</i>	1323.4	137.9	118.6	133.6	93.1	1.62
30	50	379.6	22.2	<i>13.8</i>	168.5	13.3	13.1	13.9	12.4	1.12
	100	–	150.1	<i>83.2</i>	770.6	71.5	83.6	77.8	66.5	1.25
	200	–	–	<i>457.5</i>	–	446.1	449.3	448.5	409.9	1.12
50	50	1712.7	38.7	<i>20.3</i>	984.1	19.9	19.2	20.4	18.6	1.09
	100	–	690.8	<i>211.0</i>	–	204.9	171.7	214.0	160.1	1.32
	200	–	1960.7	<i>934.5</i>	–	894.8	854.0	947.1	818.3	1.14
100	50	–	105.9	<i>41.8</i>	2524.0	37.9	38.7	41.6	35.9	1.17
	100	–	568.1	<i>282.6</i>	–	268.7	251.4	232.4	197.5	1.43
	200	–	–	<i>1547.6</i>	–	1416.0	1368.6	1387.3	1141.5	1.36
Average speed-up factor										1.24

Table 5: Mean running time (seconds) over 10 replications for Formulation 2 under serial computing. Bold italics identify the fastest baseline; bold identifies the fastest method overall. The speed-up factor is computed as **Improved/CB-MC**.

Instance		Formulation 2: Serial Computing								
$ S $	$ I $	Direct	Classic	Improved	B+L	CachedIP	CachedLP	F-MC	CB-MC	Speed-up factor
10	50	<i>6.9</i>	19.4	10.0	51.9	8.5	7.1	9.6	5.3	1.9
	100	<i>26.0</i>	79.9	29.9	162.7	22.7	19.2	22.3	12.3	2.4
	200	290.9	419.2	<i>182.9</i>	741.4	125.8	133.0	109.5	63.5	2.9
30	50	<i>16.3</i>	38.5	22.5	143.8	18.7	15.9	22.3	12.3	1.8
	100	135.2	161.5	<i>112.7</i>	–	81.5	75.8	95.4	42.7	2.6
	200	2831.5	1394.9	<i>366.7</i>	–	302.8	253.9	264.6	126.9	2.9
50	50	<i>27.2</i>	55.9	33.3	149.4	27.1	24.0	32.7	18.9	1.8
	100	654.9	375.3	<i>155.0</i>	1384.3	128.7	108.8	154.3	82.3	1.9
	200	–	1719.8	<i>581.4</i>	–	462.1	451.9	576.6	339.3	1.7
100	50	75.3	138.1	<i>73.6</i>	269.8	60.3	54.1	74.9	41.3	1.8
	100	2210.5	693.9	<i>441.9</i>	1386.1	306.6	310.6	263.7	149.1	3.0
	200	–	–	<i>1785.2</i>	–	1199.2	1377.5	1198.0	697.1	2.6
Average speed-up factor										2.3

Table 6: Mean running time (seconds) over 10 replications for Formulation 2 under the 6×1 parallel computing setting. **Direct** is reported as the serial computing baseline. Bold italics identify the fastest baseline; bold identifies the fastest method overall. The speed-up factor is computed as **Improved/CB-MC**.

Instance		Formulation 2: Parallel Computing 6×1								
$ \mathcal{S} $	$ \mathcal{I} $	Direct	Classic	Improved	B+L	CachedIP	CachedLP	F-MC	CB-MC	Speed-up factor
10	50	6.9	8.5	7.0	23.2	6.4	6.9	7.2	6.7	1.0
	100	26.0	34.5	8.8	145.5	8.5	9.0	8.5	8.2	1.1
	200	290.9	110.2	30.5	686.9	25.3	30.0	22.2	24.8	1.2
30	50	16.3	10.5	7.7	114.5	7.2	7.4	7.7	7.6	1.0
	100	135.2	42.6	23.0	439.9	17.8	23.1	20.6	16.4	1.4
	200	2831.5	454.1	63.2	–	41.3	49.5	49.7	43.6	1.4
50	50	27.2	16.9	8.9	93.9	8.1	8.8	9.0	8.2	1.1
	100	654.9	120.3	24.8	–	20.2	24.2	24.3	21.0	1.2
	200	–	511.3	101.5	–	84.0	100.3	101.0	82.3	1.2
100	50	75.3	23.4	13.4	137.9	12.0	13.1	13.3	11.6	1.2
	100	2210.5	122.9	55.4	902.5	50.0	53.8	33.7	27.8	2.0
	200	–	890.1	345.5	–	217.7	282.9	225.8	150.2	2.3
Average speed-up factor										1.3

In Table 3 through Table 6, we find that the three techniques provide complementary benefits. The **CachedLP** method reduces the number of calls to the linear programming solver, while the **CachedIP** method reduces the number of calls to the integer programming solver. The **F-MC** method improves performance in two distinct ways. First, it decreases the time consumed in each iteration by allowing cuts to be generated independently for each scenario, enabling parallel or more efficient separation routines. Second, it significantly reduces the total number of iterations required for convergence, because multi-cuts provide tighter and more localized approximations of the recourse function θ .

To further clarify the source of the observed computational gains, Table 7 reports a runtime attribution for a representative sampled instance class with $|\mathcal{S}| = 100$ and $|\mathcal{I}| = 100$. The decomposition separates time spent solving the master problem, evaluating LP and IP recourse subproblems, and performing cut generation (model building) and cache management.

Caching IP solutions reduces the time spent solving IP subproblems by about 20%, while caching LP solves reduces the time spent solving LP subproblems by about 12%. The multi-cut approach reduces the time spent solving IP subproblems by about half, at the cost of slower LP solves and cut generation. However, caching particularly helps the multi-cut approach, with another 50% reduction in IP solves without additional LP or cut generation overhead.

Table 7: Runtime attribution for the instance class with $|\mathcal{S}| = 100$ and $|\mathcal{I}| = 100$ under Formulation 2.

Method	Time (s)	Speed-up	Runtime of each component (s)					Outer iter.	Cuts added
			Master	LP subproblem	IP subproblem	Model build	Cache overhead		
Improved	72.0	1.00	3.1	16.4	40.8	11.2	0.6	12	6404
CachedIP	64.9	1.11	3.2	16.4	33.4	11.4	0.5	12	6404
CachedLP	69.9	1.03	3.1	14.3	40.5	11.3	0.6	12	6404
F-MC	53.7	1.34	1.4	18.5	19.3	14.0	0.6	8	4812
CB-MC	36.3	1.98	1.4	14.3	8.5	11.6	0.5	8	4812

Notes: Speed-up is computed relative to the Improved integer L-shaped method. Master time excludes time spent evaluating recourse subproblems. LP and IP recourse times include only subproblems solved explicitly; cached evaluations are accounted for in the cache overhead column. The Direct method is omitted because it does not follow the two-stage decomposition and therefore has no comparable attribution across these categories.

Table 8: Running-time speed-up of Formulation 2 relative to Formulation 1 under serial computing. Values are computed as T_1/T_2 ; values greater than one favor Formulation 2.

$ \mathcal{S} $	$ \mathcal{I} $	Direct	Classic	Improved	B+L	CachedIP	CachedLP	F-MC	CB-MC
10	50	8.9	3.7	3.9	7.3	3.9	3.5	3.9	4.0
10	100	17.8	2.4	3.4	5.6	3.7	3.3	3.7	3.9
10	200	–	2.3	2.6	–	2.9	2.5	3.1	3.4
30	50	23.3	4.3	4.1	6.1	4.3	3.5	4.0	4.0
30	100	–	5.0	3.7	–	4.3	3.6	3.7	4.8
30	200	–	2.5	3.4	–	3.7	3.1	4.4	5.4
50	50	63.0	4.7	4.7	9.5	5.1	4.2	4.6	4.6
50	100	–	6.2	6.0	–	6.6	4.3	5.8	5.0
50	200	–	–	3.8	–	4.2	3.1	3.7	3.5
100	50	–	5.9	4.5	11.8	4.9	4.2	4.4	4.7
100	100	–	–	4.3	–	4.9	3.5	5.1	4.7
100	200	–	–	–	–	–	2.6	–	3.4

Table 9: Running-time speed-up of Formulation 2 relative to Formulation 1 under parallel computing. Values are computed as T_1/T_2 ; values greater than one favor Formulation 2.

$ \mathcal{S} $	$ \mathcal{I} $	Direct	Classic	Improved	B+L	CachedIP	CachedLP	F-MC	CB-MC
10	50	8.9	2.2	1.7	3.5	1.8	1.7	1.6	1.6
10	100	17.8	1.5	2.1	–	2.0	2.0	2.1	2.0
10	200	–	2.9	5.0	1.9	5.5	4.0	6.0	3.8
30	50	23.3	2.1	1.8	1.5	1.8	1.8	1.8	1.6
30	100	–	3.5	3.6	1.8	4.0	3.6	3.8	4.1
30	200	–	–	7.2	–	10.8	9.1	9.0	9.4
50	50	63.0	2.3	2.3	10.5	2.5	2.2	2.3	2.3
50	100	–	5.7	8.5	–	10.1	7.1	8.8	7.6
50	200	–	3.8	9.2	–	10.7	8.5	9.4	9.9
100	50	–	4.5	3.1	18.3	3.2	3.0	3.1	3.1
100	100	–	4.6	5.1	–	5.4	4.7	6.9	7.1
100	200	–	–	4.5	–	6.5	4.8	6.1	7.6

We also compare the two formulations (1 and 2) head-to-head in Table 8 and Table 9, showing that the reformulation of incompatibility constraints provides a significant running time boost independently of other acceleration techniques. Finally, we report the percentage difference between the objective values of Formulation 2 and Formulation 1, together with whether the optimal staffing allocation changes, for each of the 12 combinations of instances. Even though Formulation 2 is a relaxation of Formulation 1, we observe that their objective values are always within 0.01% of each other.

Table 10: Comparison of the objective value and optimal allocation obtained with Formulations 1 and 2.

$ \mathcal{S} $	$ \mathcal{I} $	Mean Relative Objective Change ($\times 10^{-5}$)	Allocation Changed?
10	50	0	No
10	100	0	No
10	200	9.90	No
30	50	0	No
30	100	1.40	No
30	200	1.05	No
50	50	0	No
50	100	0.25	No
50	200	0.89	No
100	50	0	No
100	100	0.11	No
100	200	3.38	No

Pre-computation cost of enumerating maximal cliques. The running times reported above exclude instance pre-computation. Replacing pairwise constraints (1b) and (1c) with maximal-clique constraints (2a) and (2b) requires an additional maximal-clique enumeration step. We report the time required to enumerate the maximal cliques in Table 11. The additional pre-computation time ranges from 0.11 seconds for the smallest instance class to 5.91 seconds for the largest, with an average of 1.47 seconds across the 12 instance classes, which is negligible relative to the total computational time.

Table 11: Average maximal-clique enumeration time (seconds).

$ \mathcal{S} $	$ \mathcal{I} = 50$	$ \mathcal{I} = 100$	$ \mathcal{I} = 200$
10	0.11	0.26	0.65
30	0.31	0.67	1.97
50	0.47	1.19	3.03
100	0.90	2.19	5.91

5.3 Analyzing the behavior of cost-informed warm-start

We then evaluate our warm-start strategy as a function of the amount of uncertainty and the relative cost of unserved incidents and additional staffed umbrella units.

We slightly modify the testbed from the previous subsection, varying the number of scenarios $|\mathcal{S}|$ in $\{30, 50, 100\}$, while holding the number of incidents per scenario ($|\mathcal{I}|$) constant at 100. We then consider seven **(Cost, Penalty)** combinations to vary the problem’s incentives from cost-driven to service-driven, ranging from a *cost-dominant* extreme of $(10^6, 10)$ to a *service-dominant* extreme of $(10, 10^6)$.

For each combination of problem size and economic regime, we compare the total CPU time required for convergence by the L-shaped method with and without our warm-start strategy. The detailed results of this study are presented in Table 12. We observe that as service becomes more and more important, our warm start strategy can improve the runtime by up to 37%.

Table 12: Impact of warm-start strategy on runtime (seconds) across different economic regimes.

(Cost, Penalty)	$ \mathcal{S} = 30, \mathcal{I} = 100$			$ \mathcal{S} = 50, \mathcal{I} = 100$			$ \mathcal{S} = 100, \mathcal{I} = 100$		
	Warm-start		Runtime Change(%)	Warm-start		Runtime Change(%)	Warm-start		Runtime Change(%)
	No	Yes		No	Yes		No	Yes	
$(10^6, 10)$	12.2	17.7	+45.0%	17.0	24.8	+45.9%	36.3	51.9	+43.0%
$(10^5, 10^2)$	12.1	17.5	+44.3%	17.2	25.3	+47.1%	35.7	53.1	+48.7%
$(10^4, 10^3)$	46.2	45.7	-1.1%	66.0	64.5	-2.3%	145.5	142.8	-1.9%
$(5 \times 10^3, 5 \times 10^3)$	43.7	42.6	-2.5%	79.3	75.3	-5.0%	182.1	173.2	-4.9%
$(10^3, 10^4)$	46.4	40.9	-11.9%	77.2	69.1	-10.5%	179.8	156.0	-13.2%
$(10^2, 10^5)$	44.5	27.9	-37.3%	65.1	40.6	-37.6%	130.6	81.1	-37.9%
$(10, 10^6)$	39.6	26.6	-32.8%	61.0	41.2	-32.5%	116.3	78.2	-32.8%

6 Case Study: Atlanta Police Department

We now apply our algorithm to a larger-scale real-world instance, using data from the Atlanta Police Department.

6.1 Real-world Opportunities for Staffing Optimization

We first describe key features of the dataset in Figure 3, highlighting some of the challenges that data-driven staffing can address. Figure 3a, which summarizes the distribution of daily incident volumes, suggests that the number of incidents per day is relatively concentrated around 2,000-2,500. Figure 3b presents the em-

pirical distribution of travel time (difference between dispatch time and arrival time) across all 911-initiated incidents, differentiating between regular and umbrella units. While most responses occur within 5–15 minutes, the distribution exhibits a significant right tail, indicating opportunities for service improvement. We observe that the average travel time is shorter for umbrella units than regular units — as we will discuss in the next paragraph, this reflects a form of selection bias, as the current staffing policy stations most umbrella units in smaller zones with shorter travel times.

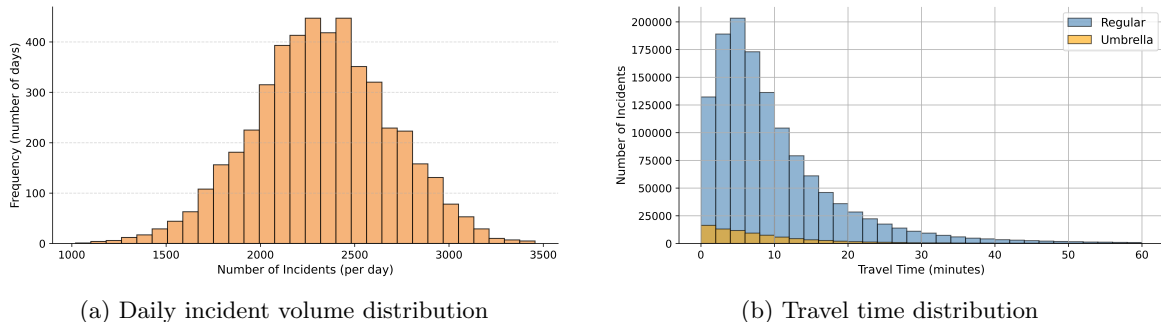


Figure 3: Operational characteristics of APD calls-for-service data.

Figure 4 presents the spatial layout of the Atlanta Police Department. Each beat is colored according to the total number of incidents recorded. Zone 5, which encompasses the central business district, exhibits the highest overall concentration of incidents, with beats 206 and 207 being the most active. These two beats correspond to areas with major commercial infrastructure, including business centers and large shopping malls. By contrast, outer zones such as Zones 1 and 4 experience comparatively lower incident frequencies. This directly motivates the core problem of this paper: determining the optimal number of flexible umbrella units to complement fixed beat resources and efficiently address spatially varying workloads.

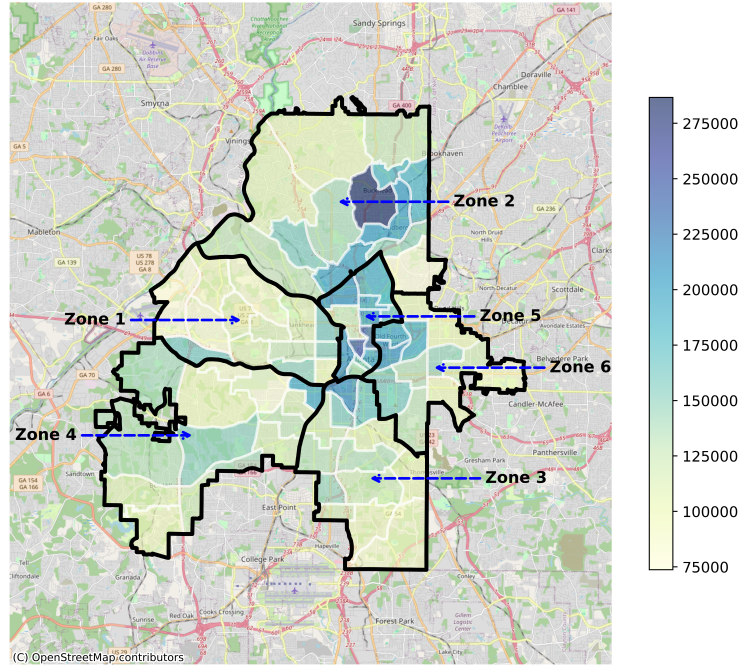


Figure 4: Map of APD Zones and beats, colored by the number of incidents between 2013 and 2017.

Beyond the overall spatial distribution of incidents shown in Figure 4, Figure 5 suggests one way in which the current umbrella unit deployment strategy may be falling short. Figure 5a shows that historical assignment patterns for umbrella units have largely followed incident density (i.e., more units are stationed where more incidents are likely to happen).

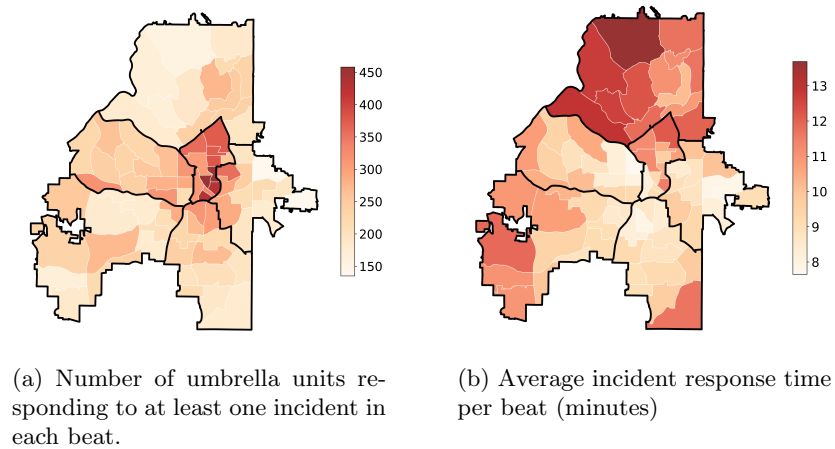


Figure 5: Comparison of incident-to-unit assignments

One adverse effect of this incident-density-driven deployment on service quality across the entire city is

shown in Figure 5b, which presents a heatmap of incident travel times. While central areas, where umbrella units are concentrated, largely benefit from the additional units, the peripheral areas of Atlanta experience substantially longer police response times. These extended travel times in the city’s outer beats and zones suggest that the current de facto deployment logic may underserve areas with lower, yet still significant, incident volumes.

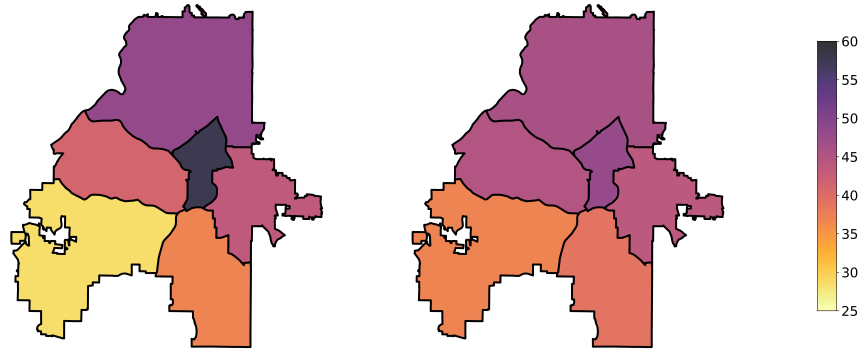
6.2 The Value Proposition of Data-Driven Optimization

We now compare our model’s output against the APD’s estimated historical deployment strategy under an equivalent resource budget. We fix a planning period of one week, divided into three daily shifts according to the current APD schedule (7 AM to 3 PM, 3 PM to 11 PM, and 11 PM to 7 AM).

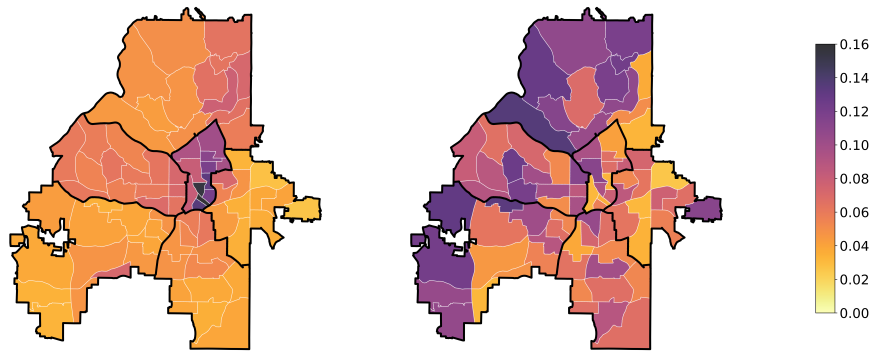
We solve the two-stage stochastic program to determine a weekly staffing policy for the deployment of umbrella units. We determine the optimal staffing levels using a set of 30 scenarios, each corresponding to one week of data from 2017. We then evaluate the performance of this single staffing policy on a separate set of 10 scenarios (a 10-week planning horizon). Figure 6 compares this optimized staffing policy with the status quo.

We observe that the optimized solution reallocates some umbrella units to zones with lower incident volumes rather than concentrating patrol units in Zone 5, where incidents occur most frequently. This shift can be interpreted as a response to spatial efficiency: although Zone 5 experiences a high volume of incidents, it is geographically compact, resulting in relatively short travel times. In contrast, larger zones require more time for units to reach incidents, making it more beneficial to assign additional units to reduce overall response times.

Finally, we demonstrate the reduction in travel time from our optimized staffing policy (using our dispatch model) in Figure 7. We observe that the map exhibits many fewer “dark spots”, which means the optimized staffing policy can improve response times in the most congested beats, without significantly deteriorating response times in the central zone. The average incident response time across the city is 9.41 min under the status quo staffing schedule and 8.53 min under the optimized staffing schedule: on average, optimal staffing can reduce response times by almost one minute.

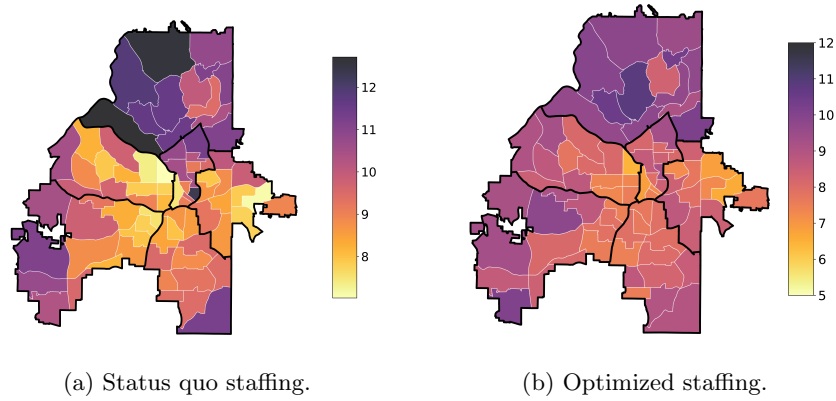


(a) Total umbrella units assigned to each zone under the status quo staffing schedule (left panel) and the optimized schedule (right panel).



(b) Fraction of incidents assigned to umbrella units under the status quo staffing schedule (left panel) and optimized schedule (right panel).

Figure 6: Comparing the status quo and optimized staffing schedules.



(a) Status quo staffing.

(b) Optimized staffing.

Figure 7: Average incident response times (minutes) in each beat under our dispatch model.

6.3 Managerial Insights: A Decision-Support Framework for Staffing

We can also use our model to analyze the fundamental trade-offs between operational costs (i.e., the number of deployed umbrella units) and service levels (i.e., response times and incident coverage). By systematically varying the number of umbrella units under different penalty regimes, we can quantify the costs and benefits of various staffing strategies.

We conduct this analysis under two distinct strategies: (1) a **stringent service policy**, which prioritizes comprehensive incident coverage, and (2) a **balanced service policy**, which reflects a more realistic trade-off between cost and performance.

6.3.1 System Performance Under a Stringent Service Policy

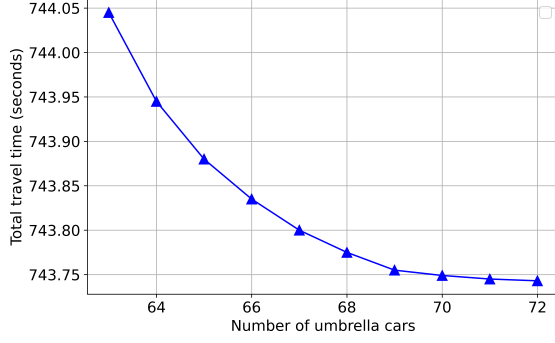
To understand the resources required to achieve near-absolute incident coverage, we first analyze system performance under a stringent service policy. We set the penalty parameter m to a prohibitively large value, which compels the model to assign a unit to every incident (i.e., forcing $w_i^s = 0$ for almost all i, s).

Figure 8a illustrates the relationship between the total number of active umbrella units and the resulting total system travel time. We observe that requiring 100% incident coverage is not practical. The minimum number of total umbrella units to ensure full incident coverage is high (63 units) but the total travel time reduction from each additional unit is nearly zero.

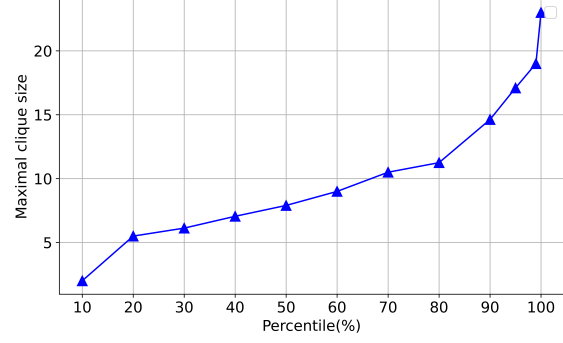
This inefficiency occurs because ensuring 100% incident coverage in all scenarios implies staffing a worst-case number of umbrella units. These extra units are mostly not utilized, except in the most challenging scenarios. These extreme cases correspond to scenarios with large, complex *cliques* in the incident conflict graph $\mathcal{G}_{s,z,t}$. We show the distribution of maximal clique size across scenarios in Figure 8b. We can see that cliques of size 20 or more occur only about 1% of the time. This finding provides a key insight for decision-makers: achieving the final few percentage points of coverage under all possible circumstances is disproportionately expensive.

6.3.2 System Performance Under a Balanced Service Policy

While the stringent policy analysis provides a theoretical upper bound on resource requirements, a more operationally relevant perspective involves balancing service costs against a finite penalty for unassigned incidents. In Figure 9a, we show the average number of unserved incidents and the average response time for different values of m and the total number of umbrella units. These curves provide a number of actionable



(a) Average incident response time as a function of the number of staffed umbrella units.

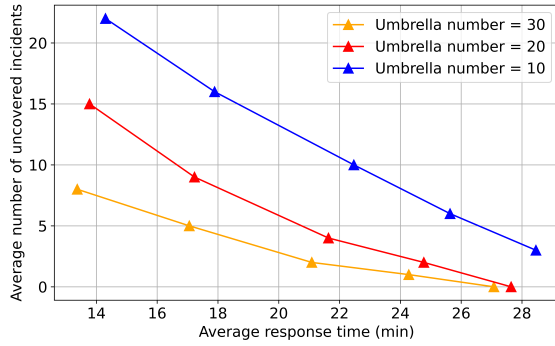


(b) Cumulative distribution of maximal clique size in incident conflict graph.

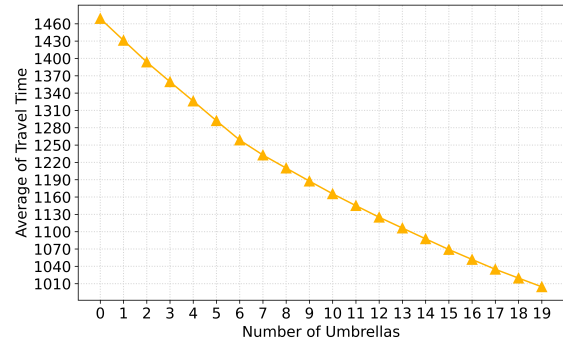
Figure 8: System Performance under Stringent Service.

insights. For instance, staffing 20 instead of 10 umbrella units can reduce the number of unserved incidents by about 7 without affecting the average response time, or reduce average response time by around 5 minutes for the same coverage.

In Figure 9b, we calibrate the penalty parameter m to be equivalent to the system-wide average travel time (essentially capturing a moderate additional delay to respond to the incident). Under this parameter choice, each additional umbrella unit roughly reduces average travel time by 20 to 30 seconds.



(a) Average number of unserved incidents versus average response time, for different numbers of available umbrella units.



(b) Response time (seconds) as a function of staffed umbrella units, for the calibrated coverage penalty.

Figure 9: Visualizing tradeoffs between coverage, response time, and staff size.

7 Conclusion

This paper addresses the challenge of optimizing the deployment of flexible police patrol resources, specifically the “umbrella” units used by the Atlanta Police Department to supplement regular beat patrols and manage

fluctuating demand.

We formulated this problem as a two-stage stochastic mixed-integer program, with staffing in the first stage and dispatch in the second, and developed a tractable methodology to solve it. Computationally, we show that our combination of several speedup techniques leads to substantial runtime improvements. We can use our tractable method to compute clear tradeoffs between key stakeholder objectives.

We also acknowledge some limitations to our study. In particular, counterfactual travel times were not available due to the structure of our dataset; future work could explore more sophisticated estimation techniques if richer data becomes available. The model also assumes fixed beat structures and does not currently incorporate dynamic unit redeployment during a shift.

Ultimately, the paper aims to shed light on a real-world challenge faced by large police departments – and provide a tractable methodology that can empower leadership to use their limited resources as efficiently as possible.

References

- Gustavo Angulo, Shabbir Ahmed, and Santanu S. Dey. Improving the integer l-shaped method. *INFORMS J. Comput.*, 28:483–499, 2016. URL <https://api.semanticscholar.org/CorpusID:6563641>.
- Ali Arab, Amin Khodaei, Suresh K Khator, Kevin Ding, Valentine A Emesih, and Zhu Han. Stochastic pre-hurricane restoration planning for electric power systems infrastructure. *IEEE Transactions on Smart Grid*, 6(2):1046–1054, 2015.
- Julian Baals, Simon Emde, and Ola Jabali. Optimal assignment of shipments to milk runs for just-in-time part supply under uncertain due dates. *Transportation Science*, 60(4):652–674, 2026.
- Jacques F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962. URL <https://api.semanticscholar.org/CorpusID:120664321>.
- John R Birge and Francois V Louveaux. A multicut algorithm for two-stage stochastic linear programs. *European Journal of Operational Research*, 34(3):384–392, 1988.
- Ravi Boppana and Magnús M Halldórsson. Approximating maximum independent sets by excluding subgraphs. *BIT Numerical Mathematics*, 32(2):180–196, 1992.
- Tobias Brandt, Oliver Dlugosch, Ayman Abdelwahed, Pieter L van den Berg, and Dirk Neumann. Prescriptive analytics in urban policing operations. *Manufacturing & Service Operations Management*, 24(5):2463–2480, 2022.
- Jean-François Cordeau, Goran Stojković, François Soumis, and Jacques Desrosiers. Benders decomposition for simultaneous aircraft routing and crew scheduling. *Transportation Science*, 35(4):375–388, 2001.
- Kelly J Cormican, David P Morton, and R Kevin Wood. Stochastic network interdiction. *Operations Research*, 46(2):184–197, 1998.

- Kevin M. Curtin, Karen L Hayslett-McCall, and Fang Qiu. Determining optimal police patrol areas with maximal covering and backup covering location models. *Networks and Spatial Economics*, 10:125–145, 2010. URL <https://api.semanticscholar.org/CorpusID:9917441>.
- Moshe Dror, Gilbert Laporte, and Pierre Trudeau. Vehicle routing with stochastic demands: Properties and solution frameworks. *Transp. Sci.*, 23:166–176, 1989. URL <https://api.semanticscholar.org/CorpusID:41621458>.
- Michel Gendreau, Gilbert Laporte, and René Séguin. An exact algorithm for the vehicle routing problem with stochastic demands and customers. *Transportation science*, 29(2):143–155, 1995.
- Michel Gendreau, Gianpaolo Ghiani, and Emanuela Guerriero. Time-dependent routing problems: A review. *Computers & operations research*, 64:189–197, 2015.
- Matthew S. Gerber. Predicting crime using twitter and kernel density estimation. *Decis. Support Syst.*, 61:115–125, 2014. URL <https://api.semanticscholar.org/CorpusID:1669156>.
- Linda Green. A multiple dispatch queueing model of police patrol operations. *Management science*, 30(6):653–664, 1984.
- Long He, Xiaobo Li, and Yue Zhao. Proactive policing: Resource allocation for crime prevention with deterrence effect. *Operations Research*, 74(2):700–721, 2026.
- Houston Police Department. Police stations and neighborhood storefronts, 2021. URL <https://www.houstontx.gov/police/contact/substations.htm>.
- Soumia Ichoua, Michel Gendreau, and Jean-Yves Potvin. Vehicle dispatching with time-dependent travel times. *European journal of operational research*, 144(2):379–396, 2003.
- Klaus Jansen. The optimum cost chromatic partition problem. In *Proceedings of the Third Italian Conference on Algorithms and Complexity*, CIAC '97, page 25–36, Berlin, Heidelberg, 1997. Springer-Verlag. ISBN 3540625925.
- Leo G. Kroon, Arunabha Sen, Haiyong Deng, and Asim Roy. The optimal cost chromatic partition problem for trees and interval graphs. In *Graph-Theoretic Concepts in Computer Science - 22nd International Workshop, WG 1996, Proceedings*, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pages 279–292, 1997. ISBN 3540625593. doi: 10.1007/3-540-62559-3.23. 22nd International Workshop on Graph-Theoretic Concepts in Computer Science, WG 1996 ; Conference date: 12-06-1996 Through 14-06-1996.
- Ewa Kubicka and Allen J Schwenk. An introduction to chromatic sums. In *Proceedings of the 17th conference on ACM Annual Computer Science Conference*, pages 39–45, 1989.
- LA Police Department. Los angeles police department - unit numbering system, 2023. URL <https://www.ibiblio.org/jimmy/mcguinn/lapd.html>.
- Gilbert Laporte and François V. Louveaux. The integer l-shaped method for stochastic integer programs with complete recourse. *Oper. Res. Lett.*, 13:133–142, 1993. URL <https://api.semanticscholar.org/CorpusID:117468276>.
- Richard C. Larson. *Urban Police Patrol Analysis*. MIT Press, 1972. URL <https://api.semanticscholar.org/CorpusID:108730111>.
- Johanna Leigh, Sarah Dunnett, and Lisa Jackson. Predictive police patrolling to target hotspots and cover response demand. *Annals of Operations Research*, 283(1):395–410, 2019.
- Elise Miller-Hooks, Xiaodong Zhang, and Reza Faturechi. Measuring and maximizing resilience of freight transportation networks. *Computers & Operations Research*, 39(7):1633–1643, 2012.

- Ayan Mukhopadhyay, Chao Zhang, Yevgeniy Vorobeychik, Milind Tambe, Kenneth Pence, and Paul Speer. Optimal allocation of police patrol resources using a continuous-time crime model. In *International conference on decision and game theory for security*, pages 139–158. Springer, 2016.
- Lewis Ntaimo and Suvrajeet Sen. The million-variable “march” for stochastic combinatorial optimization. *Journal of Global Optimization*, 32:385–400, 2005. URL <https://api.semanticscholar.org/CorpusID:16753856>.
- Céline Pagnier, Tord Gunnar Holen, Thomas Haugen de Lange, Patrick Levin, Steffen JS Bakker, and Peter Schütz. Charging station location planning for electric trucks under demand and grid uncertainty. *arXiv preprint arXiv:2603.01782*, 2026.
- Lucas Parada, Jean-François Côté, and Michel Gendreau. A disaggregated integer l-shaped method for the bike sharing rebalancing problem with stochastic demands. *European Journal of Operational Research*, 2025.
- Ragheb Rahmaniani, Teodor Gabriel Crainic, Michel Gendreau, and Walter Rei. The benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3):801–817, 2017.
- Carmen G Rawls and Mark A Turnquist. Pre-positioning of emergency supplies for disaster response. *Transportation research part B: Methodological*, 44(4):521–534, 2010.
- Roger Z Ríos-Mercado. *Optimal districting and territory design*. Springer Nature, 2020.
- Sukanya Samanta, Goutam Sen, and Soumya Kanti Ghosh. A literature review on police patrolling problems. *Annals of Operations Research*, 316(2):1063–1106, 2022.
- Ece Sanci and Mark S Daskin. An integer l-shaped algorithm for the integrated location and network restoration problem in disaster relief. *Transportation Research Part B: Methodological*, 145:152–184, 2021.
- Richard M. Van Slyke and Roger J.-B. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *Siam Journal on Applied Mathematics*, 17:638–663, 1969. URL <https://api.semanticscholar.org/CorpusID:26352538>.
- Kai Wang and Alexandre Jacquillat. A stochastic integer programming approach to air traffic scheduling and operations. *Operations Research*, 68(5):1375–1402, 2020.
- Shixiang Zhu, He Wang, and Yao Xie. Data-driven optimization for atlanta police-zone design. *INFORMS Journal on Applied Analytics*, 52(5):412–432, 2022.
- Jikai Zou, Shabbir Ahmed, and Xu Andy Sun. Stochastic dual dynamic integer programming. *Mathematical Programming*, 175:461–502, 2019.
- David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(6):103–128, 2007. doi: 10.4086/toc.2007.v003a006. URL <https://theoryofcomputing.org/articles/v003a006>.