6-2022

Reinforcement learning approach to solve dynamic bi-objective police patrol dispatching and rescheduling problem

Waldy JOE
Singapore Management University, waldyjoe@smu.edu.sg

Hoong Chuin LAU
Singapore Management University, hclau@smu.edu.sg

Jonathan PAN

Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling (ICAPS 2022)

Reinforcement Learning Approach to Solve Dynamic Bi-objective Police Patrol Dispatching and Rescheduling Problem

Waldy Joe,[1] **Hoong Chuin Lau,**[1] **Jonathan Pan**[2]

[1] School of Computing and Information Systems, Singapore Management University
[2] Home Team Science and Technology Agency, Singapore
waldy.joe.2018@phdcs.smu.edu.sg, hclau@smu.edu.sg, Jonathan_PAN@htx.gov.sg

Abstract

Police patrol aims to fulfill two main objectives namely to project presence and to respond to incidents in a timely manner. Incidents happen dynamically and can disrupt the initially-planned patrol schedules. The key decisions to be made will be which patrol agent to be dispatched to respond to an incident and subsequently how to adapt the patrol schedules in response to such dynamically-occurring incidents whilst still fulfilling both objectives; which sometimes can be conflicting. In this paper, we define this real-world problem as a Dynamic Bi-Objective Police Patrol Dispatching and Rescheduling Problem and propose a solution approach that combines Deep Reinforcement Learning (specifically neural networks-based Temporal-Difference learning with experience replay) to approximate the value function and a rescheduling heuristic based on ejection chains to learn both dispatching and rescheduling policies jointly. To address the dual objectives, we propose a reward function that implicitly tries to maximize the rate of successfully responding to an incident within a response time target while minimizing the reduction in patrol presence without the need to explicitly set predetermined weights for each objective. The proposed approach is able to compute both dispatching and rescheduling decisions almost instantaneously. Our work serves as the first work in the literature that takes into account these dual patrol objectives and real-world operational consideration where incident response may disrupt existing patrol schedules.

Introduction

In policing, occurrences of unexpected incidents often disrupt the execution of an existing plan. This paper focuses specifically on how police patrol agents should be dispatched and how patrol schedules need to be adapted to respond to dynamically-occurring incidents while fulfilling the two often conflicting objectives of projecting police presence (*proactive* patrol) and responding to incidents in a timely manner (*reactive* patrol). In addition, such complex decision needs to be made rather quickly and sometimes instantaneously. To add to the complexity and practicality to the problem, changes to the existing schedules need to be kept as minimal as possible.

In this paper, we propose a Reinforcement Learning (RL) approach to enable human planners to derive such decisions quickly and intelligently. We define this real-world problem as a Dynamic Bi-Objective Police Patrol Dispatching and Rescheduling Problem (DPRP) which is a variant of Dynamic Vehicle Routing Problem (DVRP) with some elements of a university time-tabling problem. We formulate the problem as a route-based Markov Decision Process (MDP) (Ulmer et al. 2020).

To solve the MDP, we propose a Deep Reinforcement Learning (DRL) approach that combines neural networks-based Temporal-Difference (TD) learning with experience replay to approximate the value function and a rescheduling heuristic based on ejection chains to learn both dispatching and rescheduling policies jointly. The learnt policies need to be anticipatory meaning that decisions made at a particular moment is not solely based on the present incident but also on future occurring incidents. To manage the trade-offs between the two opposing objectives, we propose a reward function that implicitly tries to maximize the success rate of responding to an incident within a response time target while minimizing the reduction in patrol presence without the need to manually set predetermined weights for each objective. It is intuitive to observe that responding to an incident means that some *proactive* patrol times are being taken up, resulting in reduction in patrol presence.

This paper makes the following contributions:

- We define a new variant of DVRP that incorporates elements of university time-tabling problem called DPRP and formulate it as a route-based MDP.
- We propose a solution approach that combines neural networks-based TD learning with experience replay to approximate value function and our proposed rescheduling heuristic based on ejection chains to learn dispatching and rescheduling policies jointly.
- We address the trade-offs between the opposing patrol objectives by designing a reward function that implicitly maximizes the gain in one objective while minimizing the loss in the other objective at the same time, without the need to manually set predetermined weights.
- We show experimentally on a real-world problem setting that our joint learning approach outperforms existing learning-based approaches which solve similar problem in two stages while is still able to compute the decision in an operationally realistic time.

Related Works

The DPRP belongs to a larger class of problem called police patrol problem. There are many aspects within the scope of police patrol problem ranging from designing of patrol district, resource allocation within a district, route design to combinations of any of these elements (Samanta, Sen, and Ghosh 2021). The discussion in this paper will mainly be on the routing and scheduling aspects of police patrol problem.

Police Patrol Routing and Scheduling Problem

Many existing works on police patrol problem adopt the hotspot patrol strategy which mainly addresses the *reactive* patrol. Such approaches typically begin by predicting crime hotspots spatially and temporally based on past data and then allocating police resources accordingly (Keskin et al. 2012; Leigh, Dunnett, and Jackson 2019; Chase et al. 2021). However, there is a gap in current literature in addressing *proactive* patrol. One such work that addresses both aspects is Wang et al. (2019). The authors addressed the dual objectives by decomposing the problem into two sub-problems. In this paper, we propose a reward function to address the dual objectives without the need to decompose the problem into two smaller components.

Most of the existing works on police patrol routing and scheduling problem assume that planned routes and schedules are fixed and none takes into account the disruption to the existing routes and schedules after an instance of incident response. This paper serves as the first work in addressing the dynamic version of the police patrol routing and scheduling problem while taking into consideration both *proactive* and *reactive* patrolling as discussed above.

There is a related stream of works that represents the police patrol routing and scheduling problem as Stackelberg Security Games (SSG) (e.g. (Varakantham, Lau, and Yuan 2013; Brown et al. 2014)). However, Rosenfeld and Kraus (2017) pointed that SSG may not always be applicable to all patrol context. SSG assumes the existence of an attacker-defender relationship which may not always be present in many problem settings. In our context, the police handles a variety of incidents ranging from traffic incidents, maintaining order and crowd control or providing general assistance to public. In security games, the underlying model is a Stackelberg game which hinges on the ability to design an accurate payoff matrix over a finite discrete set of actions, but such a setup is neither possible nor suitable for our problem. The problem presented in this paper is about scheduling the patrol agents across areas in space and time under various complex constraints.

Reinforcement Learning Approach to Solve Routing and Scheduling Problem

We focus our discussion on an offline decision support where policies or values for decision-making are computed prior to the execution of the plan. RL becomes a natural fit to compute such policy offline. Recently, many works have emerged that proposed learning-based approaches to solve combinatorial optimization problems, including routing and scheduling problems. The proposed RL-based approaches solve the problem either through constructing solution node-by-node (Nazari et al. 2018; Kool, van Hoof, and Welling 2018) or through improving an existing solution (Chen and Tian 2019; Lu, Zhang, and Yang 2019).

There also have been many recent works that addressed the dynamic variants of those problems (Chen et al. 2019; Joe and Lau 2020; Li et al. 2021; Chen, Ulmer, and Thomas 2022). Most of these works adopt a two-stage approach. Chen et al. (2019) uses the terms *dispatching-level* and *routing-level* to describe each component of the two-stage approach. In this two-stage approach, the problem is decomposed into two stages and solved one after another. The first stage involves learning the assignment/dispatch policy to determine which agent or vehicle to be selected. After which, routing/scheduling decision is executed based on the decision made in the first stage. In this second stage, the decision is either computed using an exact method or a heuristic. This is done to reduce the action space. Unlike these works, our proposed approach learns the dispatch and rescheduling policies jointly rather than separately.

Notation	Description
I	A set of patrol agents, $I \in \{1, 2, ..., \|I\|\}$
J	A set of patrol areas, $J \in \{1, 2, ..., \|J\|\}$
T	A set of time periods, $T \in \{1, 2, ..., \|T\|\}$
k	Decision epoch
t_k	Time period where k occurs
$\delta_i(k)$	A schedule of patrol agent i at k
$\delta(k)$	A joint schedule of all patrol agents at k where $\delta(k) = (\delta_i(k))_{i \in I}$
$\delta_{-i}(k)$	A joint schedule of all patrol agents except for agent i at k
$\delta^x(k)$	A joint schedule of all patrol agent after executing action x at k
τ_{target}	Response time target
τ_k	Actual response time to incident at k
$D_h(\delta', \delta)$	Hamming distance (in %) between δ' and δ
$d(j, j')$	Travel time from patrol area j to j'
Q_j	Min patrol time (time period) for j
$\sigma(k)$	Patrol statuses of each patrol area in terms of the ratio of the effective patrol time over Q_j where $\sigma(k) = (\sigma_j(k))_{j \in J}$

Table 1: Set of key notations used in this paper.

Problem Description and Model

Table 1 provides key notations and the corresponding descriptions used in this paper.

Problem Description

Our problem is concerned with scheduling patrol agents within a given police sector. There are $|I|$ patrol agents in-charge of patrolling $|J|$ patrol areas within the sector in a shift with a duration of $|T|$ time periods. At the start of the shift, each patrol agent is assigned to an initial patrol schedule. Throughout the shift, incidents occur dynamically and a patrol agent is dispatched to respond to each incident which results in the need to reschedule the initial schedules so as to still fulfill both said objectives. We can observe that

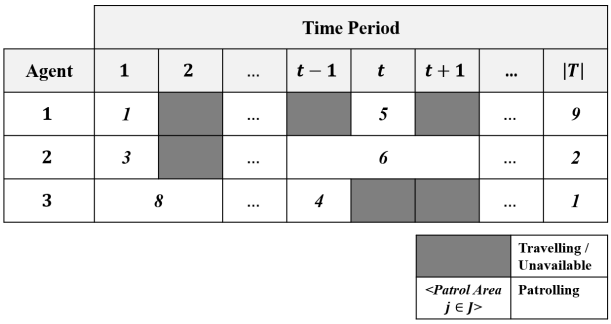

Figure 1: A sample joint schedule, $\delta(k)$ assuming $|I| = 3$.

this problem shares many similarities with DVRP (Dewinter et al. 2020).

Decision Epoch. A decision epoch k occurs whenever an incident occurs and decisions to dispatch agent to respond to the incident and to reschedule the existing schedules are required. $k = 0$ indicates the start of the shift.

Schedule. Each patrol schedule includes the sequence of patrol areas to visit (routes) and when and how long to patrol each areas (schedule). A sample joint patrol schedule can be found in Figure 1. It is similar to a university time-table with an additional key constraint whereby in between two different patrol areas, there must be sufficient time periods to cater for travel time. At the start of the shift, each patrol agent is assigned to an initial patrol schedule, $\delta_i(0)$ that forms a joint schedule, $\delta(0)$. In this paper, we assume that the initial joint schedule is available and computed independently.

Incident. A dynamic incident, ω_k occurs at decision epoch k and is described as the following tuple: $\langle \omega_k^j, \omega_k^t, \omega_k^s \rangle$ where $\omega_k^j \in J$ refers to the location of the incident, $\omega_k^t \in T$ refers to the time period when the incident occurs and ω_k^s refers to the number of time periods needed to resolve the incident. We assume deterministic resolution time.

Patrol Presence. We define patrol presence in terms of the number of time periods each patrol area is being patrolled in a shift. Each patrol area j needs to be patrolled for at least Q_j time periods in a given shift. We define a fitness function, $f_p(\delta(k))$ to quantify the goodness of a given schedule $\delta(k)$ in terms of its ability to project presence. $f_p(\delta(k))$ consists of two components namely patrol utilization (ratio of effective patrol time over total shift time) and penalty cost for failure to meet the minimum patrol time. Thus, a schedule is deemed to have good patrol presence if agents spend most of the time patrolling rather than travelling between patrol areas and each patrol area is being patrolled for at least the required number of time periods.

$$f_p(\delta) = \frac{\sum_{i \in I, t \in T} 1_J(\delta_{i,t}) - \sum_{j \in J}(Q_j(1 - min(\sigma_j, 1)))}{|T| \times |I|} \tag{1}$$

$$\text{where } 1_J(x) = \begin{cases} 1, x \in J \\ 0, x \notin J \end{cases}$$

Response Time. The response time to an incident at decision epoch k, τ_k is computed as the time taken by the assigned agent, x_k^i to act upon the dispatch call from the point where incident occurs ($x_k^t - \omega_k^t$) plus the travel time from its current location to the incident location. A successful incident response happens when $\tau_k \leq \tau_{target}$. We assume that any dispatch call must be acted upon within τ_{max}.

$$\tau_k = (x_k^t - \omega_k^t) + d(j_{t'}, \omega_k^j) \tag{2}$$
$$\text{where } x_k^t - \omega_k^t \leq \tau_{max}, t' = x_k^t$$

Problem Objective. The objective of the problem is to make dispatching and rescheduling decisions at every epoch that maximize the number of successful incident responses while minimizing the reduction in patrol presence.

MDP Formulation

Since the problem that we are addressing is a sequential decision-making problem, we model the problem as a route-based MDP. This modelling framework suits our problem since a rescheduling decision must include the updated patrol schedules instead of simply the next patrol area to visit.

State. A state of this MDP consists of two parts, pre-decision state S_k and post-decision state S_k^x. S_k captures the necessary information required to make the dispatching and rescheduling decisions. S_k is represented as the following tuple: $\langle t_k, \delta(k), \sigma(k), \omega_k \rangle$. Meanwhile, the post-decision state S_k^x captures the changes to the state upon executing a decision. t_k remains the same while the other three components are updated depending on the decision taken.

Action/Decision. x_k is the action of dispatching an agent to an incident and updating the joint schedule at decision epoch k. x_k is represented as the following tuple: $\langle x_k^i, x_k^t, \delta^x(k) \rangle$ where $x_k^i \in I$ is the agent assigned to respond to the incident, $x_k^t \in T$ the time period which the assigned agent starts to act upon the dispatch call and $\delta^x(k)$ the resulting joint schedule after executing action x_k.

In a real-world operational setting, the disruption to the initially-planned schedule must be minimized. We propose the use of Hamming distance to quantify the extent of disruption to the original schedule and this distance must be within a given threshold, P_{max}.

$$D_h(\delta^x(k), \delta(0)) < P_{max} \tag{3}$$

Transition. There are two main transitions in the model namely, from pre-decision state, S_k to post-decision S_k^x and from S_k^x to the next pre-decision state, S_{k+1}. The transition from S_k to S_k^x takes place after executing action x_k. Meanwhile, during transition from S_k^x to S_{k+1}, a realization of a dynamic incident, ω_{k+1} takes place and $S_{k+1} = (S_k^x, \omega_{k+1})$.

Reward Function. The reward function, $R(S_k, x_k)$ is designed in such a way that high reward is given to a successful incident response while minimizing the reduction in patrol presence at the same time. We introduce $f_r(x_k)$ to quantify

the success of an incident response when executing x_k.

$$R(S_k, x_k) = f_r(x_k) \times f_p(\delta^x(k)) - f_p(\delta(k)) \quad (4)$$

$$f_r(x_k) = \begin{cases} 1, \tau_k \leq \tau_{target} \\ 0.5, \tau_k > \tau_{target} \\ 0, \tau_k = \varnothing \end{cases} \quad (5)$$

The last case in Eq. 5 indicates that the incident cannot be responded to due to unavailability of agents within τ_{max} or the existing schedule simply cannot be disrupted beyond P_{max}. As this is beyond the scope of our work, we make the assumption that additional resources may be activated to ensure that all incidents are responded to.

Our proposed reward function implicitly maximizes the success rate of responding to an incident within τ_{target} while minimizing the impact to the patrol presence. Unlike the commonly adopted linear scalarization method to address multiple objectives, there is no need to manually determine a set of weights attached to each objective.

Objective Function. The objective at every decision epoch k is to select action x_k^* which maximizes the immediate reward and the expected future reward from yet-to-realized dynamic events which is represented by the approximated value function, $\hat{V}(S_k^x)$.

$$x_k^* = argmax_{x_k \in X(S_k)}\{R(S_k, x) + \gamma \hat{V}(S_k^x)\} \quad (6)$$

Solution Approach

Our paper adopts the solution framework proposed by Joe and Lau (2020) where we do not reduce the action space by decomposing the problem into two stages or two subproblems; instead we learn the dispatching and rescheduling policies jointly by combining Value Function Approximation (VFA) with a rescheduling heuristic.

The key idea of our proposed approach is to learn the approximate value function offline via simulation and use the learned value function to guide the rescheduling heuristic (see Figure 2). The approximate value function represents the value of the state in terms of its expected future rewards. The proposed reward function ensures that the learned values take into account the need to maximize both objectives. In other words, given a state and an incident, we need to choose an action that results in a schedule with better incident response capability with minimal reduction in patrol presence.

Value Function Approximation

As shown in Figure 3, we propose to approximate the value function for each post-decision state, $\hat{V}(S_k^x, \theta)$ with neural networks. Our approach learns the dispatching and rescheduling policies jointly because the value function represents the value of a state after executing both dispatch and reschedule decisions. To learn the parameter θ, we propose the use of on-policy TD learning with experience replay. We refer our readers to Joe and Lau (2020) for the detailed description of the learning algorithm. We propose the use of VFA because the decision variable, x_k is multi-dimensional. Off-policy TD learning method like Q-learning and policy gradient method may not be applicable directly since the action space changes depending on the current state.

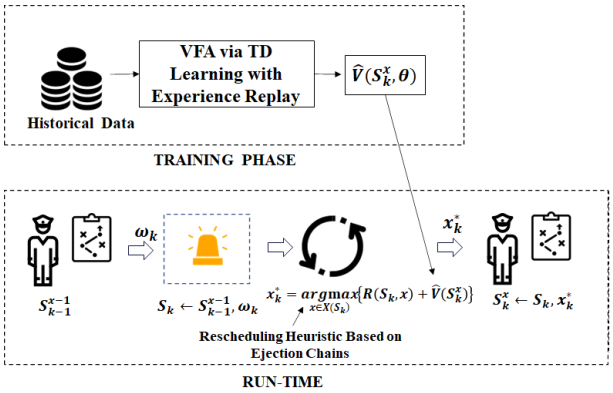

Figure 2: The learned value function approximation is utilised by the rescheduling heuristic to compute rescheduling decisions during run-time.

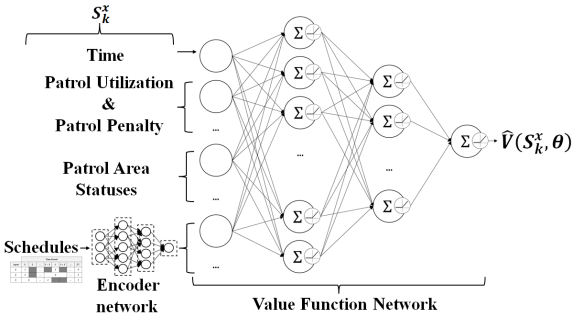

Figure 3: Neural network is used to approximate the value function of a post-decision state. Handcrafted features and encoded schedules are used to reduce the state space.

State Representation

To reduce the state space, we represent the joint schedule by extracting its key features and also encoding it into low-dimensional vector representations. Our encoder network is a multilayer perceptron which takes in the one-hot vector encoding of each agent's schedule. In addition, we introduce the following handcrafted key features to describe a joint schedule:

- **Patrol Utilization.** The ratio of effective patrol time over the total shift time of each agent's schedule.
- **Patrol Penalty.** Penalty cost for failure to meet the minimum patrol time.

Figure 3 shows how our proposed state representation enhances the learning process by supplementing handcrafted features with encoded schedules as inputs to the network.

Rescheduling Heuristic Based on Ejection Chains

To compute the rescheduling decision almost instantaneously, we propose a rescheduling heuristic based on ejection chains. Ejection chain is a complex neighbourhood

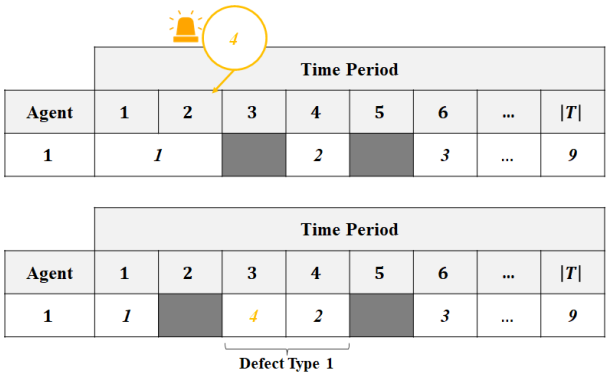

Figure 4: Assuming Agent 1 is dispatched to respond to an incident at patrol area 4 at time period 2 and it takes 1 time period to travel from patrol area 1 to 4 and a resolution time of 1 time unit, direct insertion of the incident into the existing schedule will create a defect to the schedule. This is unlike in the classical VRP where additional stop can be directly inserted into a route.

structure which was first introduced in Glover (1996) and is commonly used as perturbation operator to escape local optimum. Although originally designed for TSP, ejection chain moves have been used by many authors for various optimization problems such as VRP (Rego 1998), knapsack problem (Peng et al. 2016) and time-tabling problem (Kingston 2016).

Ejection chain move consists of a sequence of operators that forms a chain reaction. In our problem, the ejection chain consists of a sequence of defect-checking and repair operations. Insertion of an incident into the schedule potentially introduces a defect to the schedule (see Figure 4). Repairing a defect at one part of a schedule may introduce a defect in another part. Thus, a chain of $CHECK$ and $REPAIR$ operations is formed until termination condition is met or until no defect is present. Two types of defects exist in our problem:

- **Type 1: Patrol Consecutiveness.** There must be sufficient time periods in between two different patrol areas in a schedule. The time periods must be at least able to cater to the travel time between the consecutive patrol areas. This is a hard constraint since the existence of this defect deems the schedule to be infeasible. There are two types of cases for this defect type namely *Insufficient* case and *Excess* case. *Insufficient* case refers to a case where there are insufficient time periods to cater for the required travel time while *Excess* refers to a case where the time periods in between two different patrol areas are more than the required travel time periods.
- **Type 2: Minimum Patrol Presence.** Each patrol area must be patrolled for at least a minimum required amount of time periods, Q_j. This is a soft constraint since the existence of this defect results in a penalty cost that reduces the goodness of a schedule.

Algorithm 1 describes how this rescheduling heuristic works for every (x_k^i, x_k^t) pair. A tabu list is introduced to

Algorithm 1: Rescheduling Heuristic Based on Ejection Chains

Input : Joint schedule $\delta(k)$, incident ω_k, agent x_k^i, action time x_k^t
Output: Post-decision joint schedule $\delta^x(k)$

1 $\delta_i^x(k) :=$ Insert incident ω_k^j into agent x_k^i's schedule at time period x_k^t
2 $\delta^x(k) := (\delta_i^x(k), \delta_{-i}(k))$
3 **while** $\delta^x(k)$ *is defective* **do**
4 **if** $\delta^x(k)$ *in TabuList* **then**
5 **if** $\delta^x(k)$ *is feasible* **then**
6 **return** $\delta^x(k)$
7 **else**
8 **return** None
9 **end**
10 **end**
11 Add $\delta^x(k)$ into TabuList
12 $defects := CHECK(\delta^x(k))$
13 **if** $D_h(\delta^x(k), \delta(0)) > P_{max}$ **then**
14 **return** None
15 **end**
16 **if** *defects is not empty* **then**
17 Select a defect, d from $defects$
18 $\delta^x(k) := REPAIR(\delta^x(k), d)$
19 **end**
20 **end**
21 **return** $\delta^x(k)$

avoid cycling (lines 4-9). To speed up the heuristic, we propose to explore only one ejection chain instead of exploring multiple ejection chains at the same time with priority given to repair Type 1 defect (line 17). This simplified approach results in 5x speed-up on average with minimal impact to the solution quality.

REPAIR Operation. Each ejection chain consists of a series of $CHECK$ and $REPAIR$ operations (lines 4-19). We omit the implementation details for $CHECK$ to simplify the discussion as it is fairly straightforward to implement it. Meanwhile, Algorithm 2 describes how the $REPAIR$ operation works. For each type of defect, we introduce corresponding $repairOperator$.

- **Repair operators for Type 1 defect.** We introduce two repair operators namely *Insert* and *Replace*. Figure 5 illustrates how these operators repair a schedule with Type 1 Defect (*Insufficient* case). Meanwhile, for the *Excess* case, the *Insert* operator will insert either the origin or destination patrol area to the excess time period.
- **Repair operator for Type 2 defect.** We introduce *Replace* operator which simply selects patrol areas that have extra patrol time and replace them with patrol areas that require more patrol time.

Each repair operator is akin to a local neighbourhood search which explores the neighbouring schedules by repairing the defect and return the neighbouring schedule with the highest $getScore$ value (line 7). However, we introduce an

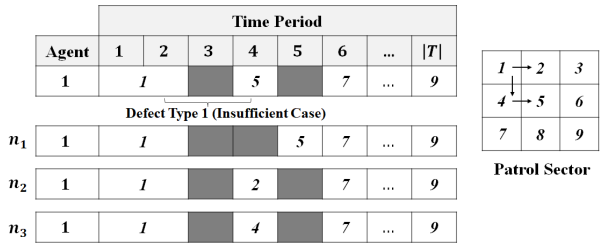

Figure 5: Assuming a patrol sector in the form of 3x3 grid, *Insert* operator inserts the necessary travel time into the schedule (see n_1). Meanwhile, *Replace* operator replaces the infeasible destination with the feasible ones (see n_2 and n_3).

Algorithm 2: $REPAIR$

Input : Joint schedule $\delta^x(k)$, defect d
1 Create a set of neighbouring schedules, $\mathbf{N} := \{\}$
2 **for** *each repair operator* **do**
3 $\quad n := repairOperator(\delta^x(k), d)$
4 $\quad \mathbf{N} \cup \{n\}$
5 **end**
6 With probability ϵ, $bestSolution \sim U(\mathbf{N})$
7 Otherwise,
$\quad bestSolution := argmax_{n \in N} getScore(n)$
8 **return** $bestSolution$

ϵ-greedy policy to explore chain that results from possibly selecting a poorer solution so as to escape possible local optimum (line 6). The learned value function is being utilized in $getScore$ function to determine the exact neighbourhood move to take. In fact, $getScore$ is equivalent to Eq. 4 plus the learned value function and a penalty term for infeasible solution since the neighbouring schedule may still contain some defects. For example, in Figure 5, only n_3 has zero penalty term since n_1 and n_2 still contain Type 1 defect.

Experiments

The objective of the experiment is to evaluate the solution quality and the computational time of our proposed approach against a real-world problem setting. While we try to mimic the real-world problem setting as close as possible, we evaluate our proposed approach with synthetically-generated data due to classified nature of the data.

Experimental Setup

Environment Setup. Hexagonal grids of diameter 2.22 km each are drawn over the local police sectors. Each grid represents a patrol area. We assume that patrol agents have the flexibility to plan their own routes within a patrol area.

To evaluate the robustness of our approach, we consider 4 patrol sectors with different parameter settings and profiles to represent different problem complexities (see Table 2 and Figure 6). Sectors A and B represent a slightly less complex problem with most of the patrol areas having relatively homogeneous patrol density while C and D represent a more complex problem with more diverse patrol areas. We define patrol density as the number of minimum patrol time required in a given hexagonal grid. Meanwhile, B and D have added complexity of lower agent-to-area ratio.

Sector	Parameter	Description
A	$\|I\| = 7$, $\|J\| = 14$	High agent-to-area ratio, relatively homogeneous patrol densities (medium)
B	$\|I\| = 4$, $\|J\| = 23$	Low agent-to-area ratio, relatively homogeneous patrol densities (low)
C	$\|I\| = 3$, $\|J\| = 6$	High agent-to-area ratio, more diverse patrol densities (low to high)
D	$\|I\| = 7$, $\|J\| = 23$	Low agent-to-area ratio, more diverse patrol densities (low to high)

Table 2: Different patrol sectors representing different problem structures and complexities.

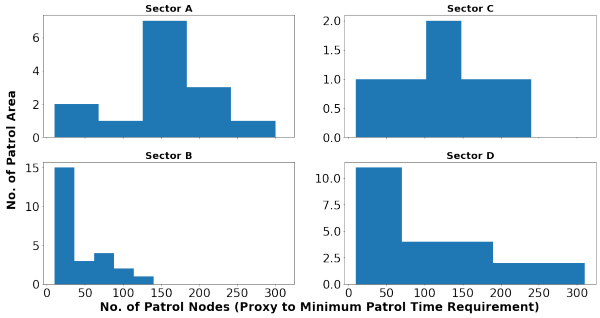

Figure 6: Histogram showing the structure of each patrol sector in terms of the distribution of its patrol areas by their patrol density.

We define T as a 12-hour shift discretized to 10-minute time periods. We model the inter-arrival time of the dynamic incident using a Poisson process with λ as the rate of occurrences of dynamic incidents per hour.

Model Parameters. The value function network is represented as fully connected neural network with 2 hidden layers with 64 nodes and 32 nodes respectively. Here are the values of some key hyperparameters used in the model: batch size $= 64$, learn and update frequencies of once in every 10 steps, learning rate $= 0.005$, $\gamma = 0.99$ and $\epsilon = 0.3$. We set the number of training episodes for both our approach and DQN to be 10000 episodes. We observe that the cumulative rewards (represented as % improvement over myopic) stabilize after around 5000 training episodes (see Figure 7).

Training & Testing Setup. In the learning phase, we synthetically generate 100 samples of initial joint schedules. Each sample represents a joint schedule of a single shift in a given day. 75% of the samples are used as training data while 25% withheld as testing data. We derive the initial schedule samples by formulating and solving a mathematical model based on the static version of the problem (which

Figure 7: Average cumulative rewards over last 250 training episodes.

is in essence a set cover model). During the testing phase, we run 30 experiments to simulate a month's worth of patrol schedules. For each experiment, we run 20 different realizations of dynamic incidents to simulate different scenarios that may take place in a given day.

Performance Measure. We propose to evaluate the approach based on the % improvement over myopic in terms of the success rate i.e. number of incidents that are successfully responded within τ_{target} and patrol presence in terms of its $f_p(\delta)$ value. The myopic approach is simply choosing a decision that gives the maximal immediate reward using the proposed heuristic. Instead of presenting mean value as a point estimate to represent the solution quality, we also use 95% Confidence Interval (CI) of the mean since the problem is stochastic in nature.

Baseline Models. Given that there is no prior work in solving DPRP, we propose to the following two common approaches in solving similar problem as baseline models:

- **Two-Stage.** This corresponds to the earlier-mentioned technique commonly used in many learning-based approaches. We implemented DQN as the learning algorithm to learn the dispatch policy with slight differences in the state representation since DQN approximates Q-value around S_k rather than S_k^x.
- **Greedy.** A commonly adopted dispatching policy by human decision-makers is to dispatch the nearest available agent to an incident.

The same reward function and rescheduling heuristic are used in the experiment for fairer comparison. Here, we do not include the commonly-used online approaches to solve DVRP such as Multiple Scenario Approach (MSA) (Bent and Van Hentenryck 2004) as baseline because such sampling-based approaches takes longer computation time and may not be operationally suitable (see experiments done in Joe and Lau (2020)).

Experiment Results and Analysis

Solution Quality. There are 3 main observations that we can gather from the experiments.

1. *Our proposed approach is statistically able to produce decisions that result in higher success rate as compared to the other two baseline models in 3 out of 4 sectors* (see Table 3). However, our approach performs poorer than myopic in Sector C. This is because C represents a very small police sector with very small numbers of agent and patrol areas. Thus, a myopic approach will suffice given that the decision space is rather limited.
2. *Our proposed approach outperforms the Two-Stage approach in more complex problems.* For ease of illustration, we intentionally present the results as line charts in Figure 8. We observe that when the problem scenarios are less complex (A and B), the performances of both approaches do not differ much. However, the performance gap widens with our approach outperforming the Two-Stage approach when it comes to more complex problems (C and D). Furthermore, our approach has slight edge over Two-Stage approach when the agent-to-area ratio is low (B and D). In fact, the Two-Stage approach performs rather poorly and even worse off than myopic when the problem is both diverse and the agent-to-area ratio is low (D). This may be due to the fact that in our joint learning approach, the rescheduling heuristic is also learning and adapting while, in Two-Stage approach, the rescheduling heuristic provides the reward signal to learn the dispatch policy but the learned policy does not inform how the heuristic works. Thus, there is in fact no learning in the second stage. The joint learning mechanism in our approach may be crucial in addressing more complex problem scenarios.
3. *Greedy approach is outperformed by the other models in almost all the scenarios except for B.* This shows that the common notion of dispatching the nearest available agent may be sub-optimal for most cases except for some specific cases.

Sector	Our Approach	Two-Stage	Greedy
A	$\mathbf{18.4 \pm 1.9}\%$	$16.0 \pm 1.8\%$	$-11.7 \pm 1.7\%$
B	$36.6 \pm 9.1\%$	$24.1 \pm 10.6\%$	$\mathbf{48.5 \pm 10.4}\%$
C	$\mathbf{-5.1 \pm 1.7}\%$	$-22.4 \pm 2.5\%$	$-6.2 \pm 2.0\%$
D	$\mathbf{33.9 \pm 5.3}\%$	$0.8 \pm 4.0\%$	$10.1 \pm 5.6\%$

Table 3: Our approach statistically outperforms the other two baseline models in terms of % improvement (success rate) over myopic across 30 experiments in most sectors.

Computational Time. Although our proposed approach is slower than the two baseline models, it is still able to compute the decision within operationally realistic time of < 10 seconds on average across all sectors (see Figure 9).

The computational time may seem to be relatively long given the simplicity of the rescheduling heuristic. This is because the heuristic needs to be run for every (x_k^i, x_k^t) pair. We allow x_k^t to take values other than ω_k^t (see Eq. 2) which means that agent may not need to respond to the incident immediately. The intuition behind this waiting strategy which myopically may not make sense, is that it allows agent to respond to another more urgent incident in the meantime or by responding late to this current incident, it may result in more incidents being responded successfully later on.

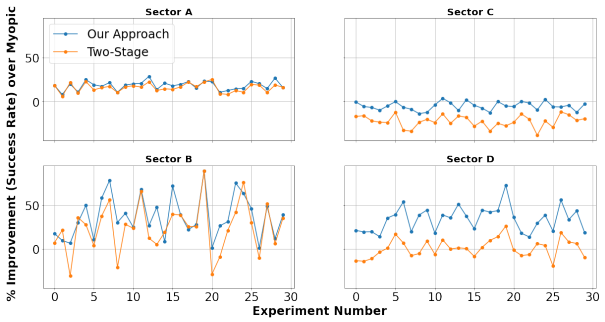

Figure 8: The gap in performance between our approach and the Two-Stage approach widens with more complex problem scenarios.

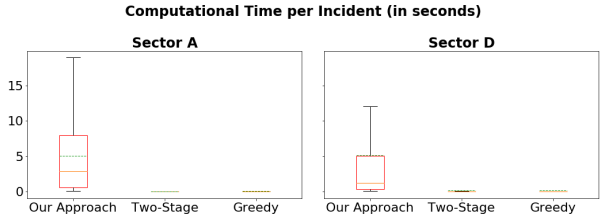

Figure 9: Our proposed approach, although slower, is still able to compute decision within seconds. Only results from 2 largest sectors, A and D are shown as results from the other sectors also exhibit similar trends.

Response Vs. Presence In all 3 approaches, we observe that an increase in success rate will correspondingly result in a reduction in patrol presence. To evaluate how our reward function performs in managing this trade-off, we run our approach on the same set of experiments data but with a modified reward function that is based on linear scalarization of the two objectives, $R'(S_k, x_k) = w_1 \times f_r(x_k) + w_2 \times f_p(\delta^x(k))$. In this experiment, we select two sets of weights $(w_1, w_2) = (0.5, 0.5)$ and $(0.7, 0.3)$ to represent the notions of "balance" and greater emphasis on incident response respectively; and Sectors A and D (2 largest sectors with one representing a more complex problem structure than the other).

Table 4 summarizes the experiment results. Comparing against the linear scalarization method, our reward function enables our proposed approach to respond to 2 and 4 more incidents while trading-off $< 5\%$ and $< 10\%$ less *proactive* patrol time in A and D respectively. This translates to an average of about 15 minutes less *proactive* patrol time per agent for 1 more successful incident response. This represents a reasonable trade-off operationally. In addition, our proposed reward function avoids the difficulty of manually assigning weights which have been shown to produce differing outcomes depending on the problem structures.

Experiment Discussion and Path to Deployment

We have shown experimentally that our proposed joint learning approach outperforms the popular Two-Stage approach especially when the problem scenario becomes more complex within an operationally realistic time. We have also shown empirically how our reward function is able to implicitly maximize one objective while minimizing the loss in the other without the need to manually assign weights to each objective. More importantly, our proposed approach is performing better than the greedy and myopic policies which are commonly adopted by human decision-maker in most of the problem scenarios.

Sector	Reward Function	No. of Successful Incident Response	Patrol Presence f_p
A	R	**16.3**	0.77
	$R'(0.7, 0.3)$	14.2	0.80
	$R'(0.5, 0.5)$	14.0	**0.81**
D	R	**14.1**	0.71
	$R'(0.7, 0.3)$	13.1	0.74
	$R'(0.5, 0.5)$	9.5	**0.80**

Table 4: Comparison of our approach with proposed vs. modified reward functions over 30 experiments. Results presented are average values across 30 experiments.

In the following paragraphs, we present several challenges in deploying our proposed solution and show how they can be addressed to ease the path towards deployment.

Data & Training. Learning value function based on sample realization of dynamic incidents can be done directly from real data (Powell 2011). This means that there is no need to model the underlying probability distribution of the dynamic incidents and learning can take place directly from available historical data, which eases the deployment effort.

Weighing Dual Objectives. Our proposed reward function, unlike the usual linear scalarization approach to multi-objective problem, does not require human to set predetermined weights (which may be subjective and difficult to compute and justify).

Generality. We assume one generic learned policy that is applicable to any given shift in any given day within a police sector. This may not always be true as a certain day of the week may have different incident pattern. Thus, for deployment, distinct policies may need to be learnt from different set of data and applied to different problem scenarios.

Scalability. Scalability can be addressed through several engineering means such as decomposing the problem into smaller police sectors or parallelizing the rescheduling heuristic. Instead, we have evaluated the robustness of our approach against different problem complexities which may not necessarily be determined solely by scale.

Future Works

We are working with relevant security agencies to further evaluate our approach's applicability and performance on specific real-world patrol problem scenarios. In addition, one interesting and challenging direction for future works will be to look at a multi-agent setting where police agents from different sectors collaborate to balance between maintaining patrol presence and response capability across all the sectors and within their own sectors.

Ethics Statement

AI in policing may bring about significant benefits to society and also serious ethical and adverse societal impacts. In fact, any policing initiative that is driven by data is inevitably susceptible to data manipulation and abuse by human operators. In addition, the existing public perception of the law enforcement agency would also have significant impact on how the proposed initiative is perceived and the subsequent impact to the society after its implementation. We acknowledge that these factors exist with or without AI and they differ in both form and extent from country to country.

Notwithstanding the above-mentioned factors, the adverse ethical/societal impact resulted from the research work presented in this paper is minimal because our work (1) aims to achieve efficiency and better service quality for the public, (2) covers wide-ranging incidents beyond crimes, (3) takes into account both reactive and proactive patrol and (4) learns directly from raw data. We believe that AI does not replace human operators entirely but it complements human decision-makers in making informed decision. Thus, human operators still have the responsibility to use data with integrity for public good.

References

Bent, R. W.; and Van Hentenryck, P. 2004. Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research*, 52(6): 977–987.

Brown, M.; Saisubramanian, S.; Varakantham, P.; and Tambe, M. 2014. Streets: game-theoretic traffic patrolling with exploration and exploitation. In *Twenty-Sixth IAAI Conference*.

Chase, J.; Phong, T.; Long, K.; Le, T.; and Lau, H. C. 2021. GRAND-VISION: An Intelligent System for Optimized Deployment Scheduling of Law Enforcement Agents. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 31, 459–467.

Chen, X.; and Tian, Y. 2019. Learning to Perform Local Rewriting for Combinatorial Optimization. *Advances in Neural Information Processing Systems*, 32: 6281–6292.

Chen, X.; Ulmer, M. W.; and Thomas, B. W. 2022. Deep Q-learning for same-day delivery with vehicles and drones. *European Journal of Operational Research*, 298(3): 939–952.

Chen, Y.; Qian, Y.; Yao, Y.; Wu, Z.; Li, R.; Zhou, Y.; Hu, H.; and Xu, Y. 2019. Can Sophisticated Dispatching Strategy Acquired by Reinforcement Learning? In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, 1395–1403.

Dewinter, M.; Vandeviver, C.; Vander Beken, T.; and Witlox, F. 2020. Analysing the police patrol routing problem: A review. *ISPRS International Journal of Geo-Information*, 9(3): 157.

Glover, F. 1996. Ejection chains, reference structures and alternating path methods for traveling salesman problems. *Discrete Applied Mathematics*, 65(1-3): 223–253.

Joe, W.; and Lau, H. C. 2020. Deep reinforcement learning approach to solve dynamic vehicle routing problem with stochastic customers. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, 394–402.

Keskin, B. B.; Li, S. R.; Steil, D.; and Spiller, S. 2012. Analysis of an integrated maximum covering and patrol routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 48(1): 215–232.

Kingston, J. H. 2016. Repairing high school timetables with polymorphic ejection chains. *Annals of Operations Research*, 239(1): 119–134.

Kool, W.; van Hoof, H.; and Welling, M. 2018. Attention, Learn to Solve Routing Problems! In *International Conference on Learning Representations*.

Leigh, J.; Dunnett, S.; and Jackson, L. 2019. Predictive police patrolling to target hotspots and cover response demand. *Annals of Operations Research*, 283(1): 395–410.

Li, X.; Luo, W.; Yuan, M.; Wang, J.; Lu, J.; Wang, J.; Lü, J.; and Zeng, J. 2021. Learning to Optimize Industry-Scale Dynamic Pickup and Delivery Problems. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, 2511–2522. IEEE.

Lu, H.; Zhang, X.; and Yang, S. 2019. A learning-based iterative method for solving vehicle routing problems. In *International Conference on Learning Representations*.

Nazari, M.; Oroojlooy, A.; Takáč, M.; and Snyder, L. V. 2018. Reinforcement learning for solving the vehicle routing problem. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 9861–9871.

Peng, B.; Liu, M.; Lü, Z.; Kochengber, G.; and Wang, H. 2016. An ejection chain approach for the quadratic multiple knapsack problem. *European Journal of Operational Research*, 253(2): 328–336.

Powell, W. B. 2011. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, volume 842. John Wiley & Sons.

Rego, C. 1998. A subpath ejection method for the vehicle routing problem. *Management science*, 44(10): 1447–1459.

Rosenfeld, A.; and Kraus, S. 2017. When security games hit traffic: optimal traffic enforcement under one sided uncertainty. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 3814–3822.

Samanta, S.; Sen, G.; and Ghosh, S. K. 2021. A literature review on police patrolling problems. *Annals of Operations Research*, 1–44.

Ulmer, M. W.; Goodson, J. C.; Mattfeld, D. C.; and Thomas, B. W. 2020. On modeling stochastic dynamic vehicle routing problems. *EURO Journal on Transportation and Logistics*, 9(2): 100008.

Varakantham, P.; Lau, H. C.; and Yuan, Z. 2013. Scalable randomized patrolling for securing rapid transit networks. In *Twenty-Fifth IAAI Conference*.

Wang, W.; Dong, Z.; An, B.; and Jiang, Y. 2019. Toward Efficient City-Scale Patrol Planning Using Decomposition and Grafting. *IEEE Transactions on Intelligent Transportation Systems*.