The Costs of Overcrowding (and Release): Strategic Discharges for Isolated Facilities During Epidemiological Outbreaks

Kati Moug
Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, klmoug@umich.edu

Siqian Shen
Corresponding author; Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, siqian@umich.edu

Problem definition: For isolated, densely populated facilities, such as prisons and nursing homes, it is difficult to enact social distancing measures when catastrophic epidemiological outbreaks occur. In such facilities, strategic releases can enhance social distancing, yet have inherent costs, e.g., the potential for recidivism in crime for prisons, or the financial cost of incentives for residents to break contracts in nursing homes. In this paper, we examine how to structure these releases over time to de-densify isolated facilities under several competing objectives. ***Methodology:*** We model the impact of strategic releases on infection transmission with a quadratic function that relates population size and daily interaction rate, which we call the *de-densification function*. Then, we formulate the decision problem as a multi-criteria MDP and develop dynamic solution methods that employ Monte Carlo simulations, k-means clustering, and Q learning with linear function approximation. ***Results:*** We consider a 100-person facility experiencing an outbreak described by a Susceptible-Infectious-Recovered epidemiological model. Under this framework, we derive theoretical conditions for the de-densification function, to ensure it has an intuitive impact on infection transmission. We also test our dynamic solution methods under a number of parameter settings, and demonstrate that our cluster-based method outperforms a static benchmark by up to 13.3% under three different de-densification functions and two priority weights. ***Managerial implications:*** Dynamic release policies can improve long-term cost over single, one-time release actions. The use of k-means clustering in Monte Carlo simulations can improve objective performance while maintaining similar computational time.

Key words: Markov Decision Processes (MDPs), epidemiological modeling, sequential decision making, uncertainty quantification (UQ), multi-objective optimization

1. Introduction

Overcrowding is associated with the fast spread of a variety of infectious diseases, from tuberculosis to rheumatic heart disease to respiratory syncytial virus (RSV). During the initial lockdown necessitated by the SARS-CoV-2 (COVID-19) pandemic, many deadly outbreaks occurred in crowded facilities, including prisons and nursing homes (Jackson and Tanner 2020a,b). In 2020, a study of 618 nursing homes in Ontario, Canada found that deaths from COVID-19 were more than doubled in facilities with high crowding conditions compared to those with low crowding conditions (Brown

et al. 2020). To combat overcrowding and enable social distancing, prisons throughout the United States released individuals with high health risk who were near the end of their sentences early in 2020 (Cohen and Eisen 2022). Other types of densely populated facilities, like nursing homes, also may have benefited from temporary releases – with financial incentives for residents to stay with family or move to annex housing to reduce crowding conditions. While early prison release decisions tend to be ad hoc, determining those eligible for release and at greatest risk, Operations Research tools can help us better plan for future pandemics in our globalized world. This is key, as the world is experiencing an increase in the rate of pandemics, with Severe Acute Respiratory Syndrome (SARS), Influenza A H1N5, H1N1, Middle East Respiratory Syndrome (MERS), and Ebola, for example, all taking place over the decade from 2003-2014. In this paper, we consider how to de-densify a space to best mitigate systemic health and release-related risk, as well as develop convenient sequential decision making tools for broader pandemic containment applications.

1.1. Problem Formulation

There are a few key features to this problem that should be captured by any decision-making tool. First, epidemiological outbreaks are dynamic phenomena with stochastic trajectories. Intuitively, the intervals at which facilities can release individuals, e.g., weekly or monthly, are discrete, each with immediate costs associated with infection and release. The decisions made at the beginning of each interval only need to consider the current state of the system, in terms of the outbreak and population demographics, without knowledge of previous states or decisions, in order to optimize some statistical measure of future cost. Eventually, an outbreak will subside. These features make the problem a good candidate for a discrete, finite-time Markov Decision Process (MDP).

When modeling de-densification for pandemic containment as an MDP, we need to operationalize the state space, action space, and reward structure. One common formulation of epidemiological outbreaks is the Susceptible-Infectious-Recovered (SIR) compartmental model, where each individual in a population is assumed to have S, I, or R status at any given time (Kermack and McKendrick 1927). In our problem, as we balance epidemiological and release-related risk, we consider the state of the system to be a multidimensional vector with each entry the number of individuals with particular status, health risk, and release risk features. The actions are the number of susceptible individuals with particular health risk or release risk features released, and the reward is based on costs associated with release and costs associated with new infections for those who remain.

When we model the transition to a new epidemic state after a certain release action is taken, we use a generic underlying epidemiological model, but adaptively include the effect of the strategic release. Intuitively, reduced overcrowding enhances social distancing, which in turn, reduces daily interactions and finally, new infections. Thus, the epidemiological parameter directly affected by

releases is the daily contact rate. For this reason, we model the release effect indirectly by modeling the contact rate as a function of population size, in what we call the de-densification function. We will formally define the concept and examine its properties for a particular epidemiological model.

While generating the entire state space and transition probability matrix for a large-scale epidemiological MDP would be a Herculean task, constructing what Kearns et al. (2002) call a *generative model* of an epidemic is generally much more straightforward. A generative model is a black box that randomly samples reward and next state from any state-action pair, following the reward and transition probability distributions of the MDP. Any epidemiological simulator can be used as a generative model for the MDP developed later in this paper, with the initial SIR composition of the population given by subtracting the release action from the state. The simulator provides a new state at the end of the epoch, which includes information about the number of new infections, that can be used to calculate epidemiological cost.

This property of the strategic release problem leads us to the two solution approaches we consider in this paper. The first, based on a sparse sampling algorithm developed by Kearns et al. (2002), uses a lookahead tree to estimate the optimal action at a particular state. This method takes advantage of the fact that we have a generative model of the MDP available, sampling repeatedly from a single state, action pair in a way that purely model-free methods cannot, without requiring the full state-action transition probability matrix. This approach also does not require the same level of tuning of hyperparameters that an on-policy neural network would require. The second, Q learning with linear function approximation, estimates the Q function. While this offline method does not take advantage of the generative model in the same way, it has the benefit of producing a full policy, estimating the optimal action at any state. Like the sparse sampling method, it produces actions without having to construct the entire feasible state space or estimate state-action transition probabilities. This approach also can benefit from tailored basis functions that take advantage of domain knowledge. We demonstrate how the basis function can be formulated to mimic the generative model, incorporating the epidemiological dynamics in a different way.

1.2. Methodological Contributions

In this work, in addition to developing an MDP model for the de-densification problem and implementing solution methods, we provide three main methodological contributions.

- First, the sparse sampling algorithm has strong theoretical properties but only provides a single action, rather than the policy for an entire trajectory, which makes it difficult to test the performance of the policy on independent out-of-sample trajectories. To mitigate this, we build an algorithm called *Split-Trajectory Sparse Sampling* that uses k-means clustering and multiple calls to the sparse sampling algorithm to evaluate the policy derived by the lookahead tree on a large number of independent samples.

- Second, the theoretical properties promised by the sparse sampling algorithm require sampling fewer child nodes for each node than algorithms that estimate more than one optimal action, but the number of child nodes is still large in a practical sense. For this reason, we develop an adaptation of the baseline sparse sampling algorithm also using k-means clustering. In our computational tests, we find the *Split-Trajectory Sparse Sampling with Clusters* adaptation that uses this version of the sparse sampling algorithm outperforms the *Split-Trajectory Sparse Sampling* method while taking similar computational time. Both of these methodological advances can be applied to other generic generative MDPs.
- Our final methodological contribution is specific to epidemiological MDPs and involves the definition of basis functions for Q learning with linear function approximation. We demonstrate the effectiveness of our basis functions under particular contact rate assumptions and priority weights.

1.3. Broader Contributions

In addition to methodological advances, we develop open-source R code, freely available on Github, that acts as a generic framework for generative MDP analysis and solution approaches. Our framework allows the public to define their own generative MDPs, including state, action, and reward components and state-action transition simulators. Our open-source code includes implementations of the *Split-Trajectory Sparse Sampling* algorithms with and without clusters that we introduce in Section 5.1, as well as tools to simulate out-of-sample trajectories that result from applying different types of policies.

We also develop a set of epidemiological test instances, adapted from stochastic individual contact models in the peer-reviewed EpiModel package in R (Jenness et al. 2018) that can be used as benchmark for further methodological development in the solution of multi-objective epidemiological MDPs. Our test instances simulate homogeneous mixing of heterogeneous health risk populations with non-health features (that do not impact infection transmission but impact other aspects of the reward function).

Finally, in our numerical studies, we demonstrate the effectiveness of *Split-Trajectory Sparse Sampling with Clusters* at improving long-term cost over a static benchmark. We note the test instances for which the Q learning with linear function approximation method improves the objective over the static benchmark, as well as the method's shorter relative runtime. We delve into the dynamic policies and resulting cost breakdown derived from the cluster method, demonstrating the ability of the priority weights to yield differentiated policies.

2. Literature Review

This work is interdisciplinary, situated in both epidemiological modeling and mathematical optimization fields. In this section, we begin with a review of epidemiological models, including the compartmental framework that we use, the stochastic individual contact model in our test instances,

and their application in pandemic settings. Then, we move into a discussion of optimal control models in epidemiology, and review the related work in the control of pandemics. Finally, we discuss the particular framework that we utilize for our decision-making model, MDPs, in the context of dynamic control, and notable solution approaches previously developed in the literature.

2.1. Epidemiological Models

To use our multi-objective, epidemiological MDP framework, the underlying disease model should include a susceptible compartment and heterogeneous health risk levels, as well as a random simulator. The model should assume homogeneous mixing. In our epidemiological test instances, we utilize agent-based stochastic individual contact models (ICMs). Stochastic ICMs are random analogues of deterministic compartmental models (DCMs), benchmark formulations that date back to Kermack and McKendrick (1927). DCMs use ordinary differential equations (ODEs) to describe disease spread in a population with homogeneous mixing as a function of a set of epidemiological parameters (Brauer 2008). While DCMs are inherently deterministic, they are used in our MDP by randomly drawing epidemiological parameters from a stochastic distribution each epoch.

Many researchers utilize DCMs to model the spread of the COVID-19 pandemic, fitting outbreak data to curves by estimating epidemiological parameters (Fanelli and Piazza 2020), modeling distinct stages of the pandemic (Mishra et al. 2020), and testing the effectiveness of quarantine (Hou et al. 2020). Wang et al. (2020) use deterministic ODEs to describe their model, and then perform stochastic simulations using the τ-leaping method (Keeling and Rohani 2011) to consider the effects of different social distancing measures. Moreover, stochastic ICMs are utilized to model the COVID-19 pandemic by Churches et al. (2020), who extend the stochastic ICM in open-source, peer-reviewed EpiModel (Jenness et al. 2018) from SIR to seven compartments, including compartments representing isolation, hospitalization, and death. In this paper, we also extend the stochastic ICM in EpiModel. However, in line with our objectives, we consider heterogeneous health risk and a simpler generative model, SIR compartments with homogeneous mixing.

2.2. Optimal Control in Epidemiology

While epidemiological models can be utilized to predict disease spread and compare the effects of various interventions, they do not necessarily find optimal courses of action. Work in mathematical optimization aims to bridge this gap, with disease control models that aim to identify strategies to minimize new infections (Mbah and Gilligan 2011) or the exponential decay rate of infections (Nowzari et al. 2015). Two central strains of research in this field are resource allocation models and network interdiction models.

Resource allocation models find optimal epidemiological interventions under the constraint of limited resources. The cost of an intervention may be represented as a function of a particular

epidemiological parameter, such as the sufficient contact rate (Brandeau et al. 2003), recovery rate, or rate of movement into a protected state, like vaccination or quarantine (Nowzari et al. 2015). The cost function can also be a function of the number of individuals treated, with that treatment number affecting the epidemiological dynamics (Goldman and Lightwood 2002). The solution methods utilized in epidemiological resource allocation problems are often analytical (see, e.g, Brandeau et al. 2003, Mbah and Gilligan 2011). Zaric and Brandeau (2001) use Taylor series approximations to find solutions for a resource allocation model with an underlying SI model and linear production functions.

Yin et al. (2023) consider allocation of ventilators using a multi-stage stochastic model. Basciftci et al. (2023) compare stochastic programming and distributionally robust optimization approaches to balance cost and quality of service in COVID-19 test kit and vaccine distribution. Generally, problems related to supply, demand, and matching the two in vaccine distribution amenable to operations research models are outlined by Dai and Song (2021). Resource allocation models are also utilized in pandemic settings to ensure resources are used wisely while satisfying social distancing requirements (see, e.g., Barnhart et al. 2022, Navabi-Shirazi et al. 2022, Gore et al. 2022)

On the other hand, network interdiction models focus on the control of disease spread, and assume that the population has a graphical structure. The epidemiological optimization models are an example of general work that mitigates the spread of contagions by removing nodes (Albert et al. 2000) or edges (Kimura et al. 2009) from a network. Enns et al. (2012) formulate the epidemiological edge removal problem as a nonconvex quadratically constrained quadratic program (QCQP) and develop a related QCQP algorithm. Nandi and Medal (2016) use mixed-integer linear programs for formulating their epidemiological network interdiction models. Koch et al. (2013) consider how the basic reproductive number changes when edges are randomly removed from a contact network.

In addition to basic disease control, a number of studies consider multi-criteria decision making in pandemic conditions, generally examining the interplay between economic and health conditions (see, e.g., Akbarpour et al. 2020, Silva et al. 2020, Thunström et al. 2020, Birge et al. 2020, Fajgelbaum et al. 2020). Duque et al. (2020) use a stochastic optimization model with the detailed epidemiological dynamics mentioned in the previous section (Wang et al. 2020) to determine lockdown thresholds, minimizing the total number of days in lockdown while ensuring hospital resources remain available. Some works

2.3. Dynamic Control and Uncertain Parameters

One advantage of our model is its ability to consider policy over time, at different phases of an outbreak. The dynamic nature of this decision making is present in a few different papers in epidemiological resource allocation. For instance, Zaric and Brandeau (2002) consider a finite-horizon problem, with investment decisions made over a finite number of time periods, in order to

maximize quality-adjusted life years or avoided infections, using heuristic methods for the general problem. Blount et al. (1997) use nonlinear programming and dynamic programming approaches to consider control of an SIS epidemic with limited resources. Du et al. (2021) use a data-driven optimization approach for multiperiod resource allocation. At each epoch, they estimate parameters with new information, and make an multiperiod decision based on current parameter estimates, though this decision is only used in the next epoch.

Our MDP model for strategic discharges for de-densification is also dynamic. MDPs have a rich theoretical foundation, with a number of methods to optimize long-run discounted cost when the state space and transition probabilities are known. Unfortunately, due to the size of the state space, these parameters are difficult to estimate. In the literature, a few papers formulate epidemiological problems as MDPs. In a paper by Reluga (2009), continuous-time MDP theory is used to describe individual strategy in one epidemiological game theoretic setting. The emphasis of the work is on equilibrium behavior, rather than systemic decision making, however. Bisset et al. (2009) apply Partially Observable Markov Decision Processes (POMDPs) to infectious outbreaks; on the other hand, their experiments simulate and compare a set of prescribed policies with a main goal of disease control, while we estimate the optimal policy weighing both epidemiological and release costs in this paper. Yaesoubi and Cohen (2011) apply the MDP framework to serialized compartmental models, where individuals flow from disease state to disease state in a single direction (e.g., from Susceptible to Infectious, then from Infectious to Removed).

One application of our multi-criteria MDP framework is de-densification of nursing homes. Control of COVID-19 outbreaks is studied in the nursing home setting by Smith et al. (2020), who simulate the distribution of COVID-19 tests using a variety of strategies, to compare their effectiveness and cost. Barak et al. (2020) also consider the testing policy problem, using network-based SEIR simulations and analysis of an exponential infection model. Another application of our MDP is early prison release. Baycik et al. (2020) apply MDPs in the criminal justice setting, mitigating city-level drug trafficking. However, the strategic release problem and its inherent competing objectives is not considered for the nursing home, cruise ship, or prison case.

3. Strategic Releases and the De-Densification Effect

Central to the formulation of the strategic release problem is the representation of the effects of social distancing. Recall that in this paper, we assume the facility outbreak is represented by a generic epidemiological model with homogeneous mixing and a susceptible compartment. In this section, we introduce the concept of a *de-densification function*, a way to model the effects of reduced overcrowding, in such a model. We derive necessary conditions for a quadratic function to be a de-densification function. Then, we consider a particular epidemiological model – an SIR

DCM with two health-risk levels – and derive sufficient conditions for a quadratic de-densification function to have protective cross-group effects.

DEFINITION 1. Let $\mathcal{E}$ be an epidemiological model with homogeneous mixing, a susceptible compartment, total population size N, and contact rate parameter $\alpha = \alpha(N)$. The function $\alpha(N)$ is a *de-densification function* if $\alpha(N) \geq 0$ and $\alpha(N)$ increases as population size N increases.

Here, increased population size implies decreased social distancing and in turn, higher contact rate. In our numerical studies, we assume the contact rate is a quadratic function of population size. Taking the first derivative, the next theorem about quadratic functions follows directly from the definition.

THEOREM 1. *Suppose the contact rate parameter is a quadratic function of population size,* $N \in [0, N_{\max}]$*, of the form*

$$\alpha(N) = \kappa_2 N^2 + \kappa_1 N + \kappa_0. \tag{1}$$

If the contact rate is a de-densification function, then the following are true.

- $\kappa_0 \geq 0$.
- *If* $\kappa_1 = 0$*, then* $\kappa_2 \geq 0$*. If* $\kappa_2 = 0$*, then* $\kappa_1 \geq 0$*. Otherwise,* $\kappa_1/2\kappa_2 \geq 0$.

While the definition ensures that the de-densification function captures the social distancing effect of strategic releases, not all de-densification functions may lead to reduced infection transmission rates. Consider an SIR DCM with health risk levels $\mathcal{H} = \{h_1, h_2\}$, governed by the following ODEs.

$$\frac{dS_h}{dt} = -\beta_h \alpha(N) \frac{\sum_{h' \in \mathcal{H}} I_{h'}}{N} S_h, \ h \in \mathcal{H}, \tag{2}$$

$$\frac{dI_h}{dt} = \beta_h \alpha(N) \frac{\sum_{h' \in \mathcal{H}} I_{h'}}{N} S_h - \xi_h I_h, \ h \in \mathcal{H}, \tag{3}$$

$$\frac{dR_h}{dt} = \xi_h I_h, \ h \in \mathcal{H}. \tag{4}$$

Here, β_h represents the probability a susceptible person with health risk level h becomes infected, given contact with an infectious person, and ξ_h is the recovery rate for an infectious person with health risk level h. The rate of new infections for health risk level h is $-dS_h/dt$. Intuitively, the proportion of susceptible individuals with health level h that become infected is the product of the average number of contacts per person $\alpha(N)$, the probability a contact is infectious $\sum_{h' \in \mathcal{H}} I_{h'}/N$, and the probability of becoming infected given contact with an infectious person, β_h.

While strategic releases reduce the contact rate $\alpha(N)$ and may reduce the number of susceptible individuals with that health risk level, S_h, they also increase the proportion of the population that is infected, $\sum_{h' \in \mathcal{H}} I_{h'}/N$. (Recall that we assume only susceptible individuals are released in this

paper.) Thus, even if contact rate is a de-densification function, the rate of new infections may not decrease with decreased population size.

We are particularly interested in de-densification functions for which the opposite is true, however. What are outbreaks for which social distancing "slows the spread," in terms of epidemiological model and de-densification function? This is a complex question worthy of further study. We consider a specific model, as well as a particular kind of cross-group spread in the final theorem in this section.

Releasing individuals from a particular health risk level generally has a protective effect for at least some of that subpopulation – the ones that leave the facility. For this reason, we are specifically interested in epidemiological test cases where cross-group protective effects exist. When does releasing individuals from one health-risk level help reduce infection transmission in another health risk level? We consider this question for the SIR DCM with heterogeneous health risk levels, in the following theorem.

THEOREM 2. *Consider an SIR DCM with heterogeneous health risk levels $\mathcal{H}$ and de-densification function $\alpha(N)$, as described in Theorem 1. Suppose there is at least one infection in the facility and at least one susceptible individual with health risk level $h \in \mathcal{H}$. Assume all releases involve only susceptible individuals in other health risk levels. Then, the rate of new infections for health risk level h increases as population size N increases if and only if*

$$\kappa_2 - \kappa_0/N^2 > 0.$$

Proof. Consider

$$\frac{-dS_h}{dt} = \beta_h \alpha(N) \frac{\sum_{h' \in \mathcal{H}} I_{h'}}{N} S_h.$$

Note that the release action does not change the value of the susceptible population with health risk level h, by assumption. In general, the release action does not involve any infectious population. This implies that

$$C = \beta_h \sum_{h' \in \mathcal{H}} I_{h'} S_h$$

is a positive constant, regardless of the action taken, and

$$\frac{-dS_h}{dt} = C \frac{\kappa_2 N^2 + \kappa_1 N + \kappa_0}{N}.$$

Taking the derivative with respect to N and noting that $C > 0$ yields the result. Q.E.D.

The following corollaries are immediate.

COROLLARY 1. *Under the SIR DCM, if de-densification function α is a linear function of population size N and release actions do not involve health risk level h, then rate of new infections for h does not increase with population size.*

COROLLARY 2. *Under the SIR DCM, if de-densification function α is a concave quadratic function of population size N and release actions do not involve health risk level h, then rate of new infections for h does not increase with population size.*

4. Decision Making Models

Now that we have considered how epidemiological dynamics are impacted by strategic releases, we can explore how those decisions should be structured. We do so by formulating the epidemiological and demographic state of the system as an MDP. In this section, we give a brief overview of discrete time MDPs, then present the set of parameters that we utilize in our finite-horizon, multi-criteria formulation, followed by the details of the model. While all notation is described within the text, Table 14 in Online Supplement Section A also provides a quick reference for interested readers.

4.1. Markov Decision Processes

An MDP can be described by a tuple $(\Omega, \mathcal{A}, P, \rho)$. The set Ω contains the states of the system, while $\mathcal{A}(i)$ contains the actions available for each state $i \in \Omega$, with the set of all actions $\mathcal{A} = \bigcup_{i\in\Omega} \mathcal{A}(i)$. For all states, we assume $\mathcal{A}(i) \supseteq \{a_0\}$, where a_0 is the null release action. In the discrete time, finite-horizon setting, we have T epochs, each of length ℓ. Every epoch, $t = 0, \ldots, T-1$, a state $i \in \Omega$ is observed, an action $a \in \mathcal{A}(i)$ is taken, and the system transitions to a new state j with probability $P_{ij}(a)$, defined for all $j \in \Omega$. An immediate cost is accrued at this time, given by function $\rho(i, a) : \Omega \times \mathcal{A} \to \mathbb{R}^-$. This represents the expected cost of taking action a while in state i over the course of one epoch.

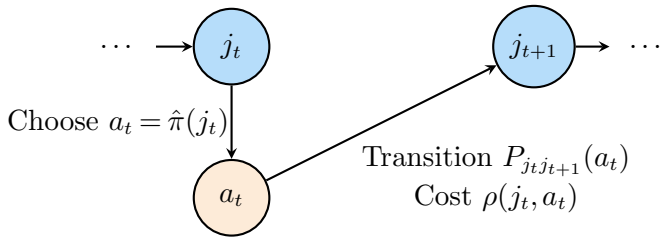


Figure 1 **Example of an MDP transition during epoch $t = 0, \ldots, T-1$, with current state j_t, action a_t chosen by policy $\hat{\pi}(j_t)$, and transition to next state j_{t+1} governed by probability distribution $P_{j_t \cdot}(a_t)$. Cost $\rho(j_t, a_t)$ is accrued immediately.**

The purpose of an MDP is to find a policy function $\pi : \Omega \to \mathcal{A}$ that assigns actions to states in a way that minimizes some cost criterion over time. We consider the expected total discounted cost criterion. Because it is common practice to maximize rewards in the MDP setting, we represent the

costs as negative "rewards," and aim to maximize these rewards. In alignment with the application setting, though, we refer to them as costs when describing the meanings of our parameters.

For any policy $\hat{\pi}: \Omega \to \mathcal{A}$, and all states $i \in \Omega$, the expected total discounted cost over time is given by

$$W_{\hat{\pi}}(i) = \mathbb{E}\left[\sum_{t=0}^{T-1} \gamma^t \rho(j_t, \hat{\pi}(j_t)) | j_0 = i\right],$$

where $\gamma \in (0,1)$ is the discount and j_0 is the initial state at time $t = 0$. The goal is to find a policy π such that for all $i \in \Omega$,

$$W_\pi(i) = \max_{\hat{\pi}} W_{\hat{\pi}}(i).$$

For this optimal policy π, we denote the value function $W: \Omega \to \mathbb{R}^-$ as $W(i) = W_\pi(i)$ for all $i \in \Omega$. We illustrate an example of an MDP transition in Figure 1.

4.2. Model Parameters

In our model, the state space describes the phase and demographic composition of a facility outbreak. Each epoch, the facility is divided into virus compartments $\mathcal{V}$ (e.g., $\{S, I, R\}$), that are further divided by demographic characteristics. Within the facility, we assume that there is a heterogeneous population consisting of individuals with discrete health risk level $h \in \mathcal{H}$ and discrete release category $r \in \mathcal{R}$, with $r^* \in \mathcal{R}$ representing the category that is ineligible for release (e.g., due to length of sentence served, or lack of alternatives to the nursing home).

Table 1 The table below shows an example state $i \in \Omega$ with virus states $\mathcal{V} = \{S, I, R\}$, health risk levels $\mathcal{H} = \{h_1, h_2\}$, and release categories $\mathcal{R} = \{r_1, r_2, r^*\}$. This is the initial state in our numerical studies.

	S			I	R
	r_1	r_2	r^*	$\mathcal{R}$	$\mathcal{R}$
h_1	**15**	**15**	**29**	**1**	**0**
h_2	**10**	**10**	**20**	**0**	**0**

The states of the system, Ω, are a subset of $\mathbb{Z}^{|\mathcal{V}|\times|\mathcal{H}|\times|\mathcal{R}|}$, with each component of state $i \in \Omega$, denoted v_{hr}^i, equal to the number of individuals that have health risk level h and release category r and are currently in virus state v, for all $h \in \mathcal{H}$, $r \in \mathcal{R}$, and $v \in \mathcal{V}$. An example state, later used as the initial state of the system in our numerical studies, can be viewed in Table 1. Notice that some components of the state are aggregated in this representation – with $\sum_{r\in\mathcal{R}} v_{hr}^i$ given for $v \in \{I, R\}$ and $h \in \mathcal{H}$, while the susceptible population is divided into components by both health risk and release risk level. This is because no releases can occur for infectious or recovered individuals, and release category does not affect epidemiological dynamics. Thus, the value function $W(i)$ is

not affected by the release category of infectious or recovered individuals, and the aggregated representation is sufficient.

Release decisions occur at regular intervals of length ℓ. At the beginning of an epoch, the decision maker observes the state of the system, i, and makes a release decision a, in the set of permissible release decisions $\mathcal{A}(i) \subseteq \mathbb{Z}^{|\mathcal{H}|\times(|\mathcal{R}|-1)}$, with each component, denoted a_{hr}, equal to the number of releases in each category $h \in \mathcal{H}$ and $r \in \mathcal{R} \setminus \{r^*\}$. An example action is provided in Table 2 and corresponds to the high health, low recidivism action in our numerical studies.

Table 2 Example action $a \in \mathcal{A}(i)$ for state $i \in \Omega$, given in Table 1. This corresponds to the high health, low recidivism action in our numerical studies.

	r_1	r_2
h_1	**0**	**0**
h_2	**10**	**0**

Expected epidemiological costs, $\rho_{\text{ep}}(i,a)$, and expected release costs, $\rho_{\text{rel}}(i,a)$, for the epoch are accrued. With weight $\lambda_{\text{ep}} \in [0,1]$, we define the overall cost (reward) function as

$$\rho(i,a) = \lambda_{\text{ep}}\rho_{\text{ep}}(i,a) + (1-\lambda_{\text{ep}})\rho_{\text{rel}}(i,a), \quad i \in \Omega,\ a \in \mathcal{A}(i).$$

After cost is incurred, there is a transition from the resulting state after release to next epoch state, j, with probability $P_{ij}(a)$. One possible transition for the state and action given in Tables 1–2 is given in Table 3. This state is taken from a trajectory in our static benchmark simulations.

Table 3 Example next state $j \in \Omega$ with virus states $\mathcal{V} = \{S, I, R\}$, health risk levels $\mathcal{H} = \{h_1, h_2\}$, and release categories $\mathcal{R} = \{r_1, r_2, r^*\}$, taken from a static benchmark simulation trajectory.

	S			I	R
	r_1	r_2	r^*	$\mathcal{R}$	$\mathcal{R}$
h_1	**15**	**14**	**28**	**1**	**2**
h_2	**0**	**7**	**16**	**5**	**2**

4.3. Generative Markov Decision Processes

If the complete state space and state-action transition probability matrix of an MDP is known, a linear program can be utilized to find the optimal policy (Ross 2014). Because it is difficult to construct the entire state space, which suffers from the curse of dimensionality, in this paper, we consider solution approaches for an MDP that includes what Kearns et al. (2002) call a *generative model*, a black box that simulates transitions from any state-action pair in the space, by randomly sampling the next state and reward from the appropriate distributions, and Q learning with linear function approximation. We describe these approaches in Sections 5.1 and 5.2.

5. Solution Methods

We describe the algorithms that we utilize to solve the multi-objective de-densification problem for pandemic containment. We begin with sparse sampling-based methods and conclude with Q learning with linear function approximation.

5.1. Sparse Sampling-Based Methods

The first approach we consider is based on a sparse sampling algorithm developed by Kearns et al. (2002). Their algorithm identifies a near-optimal action at the current state, without constructing a full policy table, by constructing a sparse lookahead tree. By focusing on the current state, this approach provides theoretical guarantees of running time based only on a desired approximation error bound and size of the sampling tree, without respect to the number of states in the MDP.

We first describe the algorithm and discuss the basic adaptations we make for scalability in Section 5.1.1. Then, we describe the *Split-Trajectory Sparse Sampling* algorithm we construct for out-of-sample testing in Section 5.1.2, followed by the cluster adaptation of both algorithms in Section 5.1.3.

5.1.1. Baseline Sparse Sampling Algorithm with Adaptations for Scalability The sparse sampling method constructs a sampling tree with the current state $i \in \Omega$ as the root node. For each action $a \in \mathcal{A}(i)$ available for current state i, the generative model draws a set of samples C from state-action pair (i, a). Each sampled state j becomes a child of the root node, with arc (i, j) labeled with action a and sampled reward ρ_{ij}. This process is repeated for each child node of i, with $|\mathcal{A}(j)||C|$ children of state j at a node are sampled to create new child nodes. This process continues until the tree reaches height H, the horizon size. The value function at the root node is estimated recursively, with the following formulas.

$$\hat{W}_\pi(j) = \begin{cases} 0, \text{ if } j \text{ is a leaf,} \\ \max_{a \in \mathcal{A}(j)} \frac{1}{|C|} \sum_{c \in C(j,a)} (\rho_{jc} + \gamma \hat{W}_\pi(c)), \text{ otherwise.} \end{cases}$$

We incorporate a few changes to the sparse sampling algorithm to reduce computational time. First, we adapt the algorithm to stop branching the sampling tree from any node that represents an absorbing state, reducing the size of the lookahead tree. Second, in our adapted sparse sampling algorithm, we introduce the concept of an action horizon, H_{a}. Here, actions only can be taken during the first H_{a} epochs of horizon T. During the final epochs $H_{\text{final}} = T - H_{\text{a}}$, no actions are taken but the system continues to evolve. This allows us to limit exponential tree growth to horizon H_{a}. Calculating the value function estimate for the leaf nodes then requires sampling a number of trajectories of length H_{final}, rather than simply assuming each action has value 0. Intuitively,

the longer the action horizon, the better the algorithm will estimate Q for different initial actions. When $T = H_{\mathrm{a}}$, we recover the original sparse sampling algorithm.

This adaptation applies to generic generative MDPs. In our problem, this assumption allows us to consider the impact of different timing of dynamic decisions early in an epidemic – as granular as week 1 to 2 to 3 – while taking into account the long-term effects of those release decisions in the final epochs of the horizon. Adapting the algorithms developed by Kearns et al. (2002) to include these features, we obtain the sparse sampling (SpSa) algorithm, Algorithm 1, given in Online Supplement Section B. This algorithm calls a number of helper functions, each an algorithm itself. These algorithms, Algorithms 2–4, also can be found in Online Supplement Section B.

5.1.2. Split-Trajectory Sparse Sampling Algorithm The sparse sampling algorithm is an online method, meant to be applied in real time as a trajectory unfolds. The algorithm produces an estimate of the optimal action to take at a particular time, knowing that more actions can be taken in the future as the state of the system evolves. How do policies derived by this method perform over time, in terms of total mean discounted reward? How does this compare to objective values obtained by other MDP solution methods? Out-of-sample trajectory simulations can help answer these questions. Following the law of large numbers, the greater the number of trajectories N_{traj}, the better the expected estimation. Unfortunately, applying the algorithm to each trajectory for every epoch requires $N_{\mathrm{traj}} H_a$ calls of the SpSa algorithm, with lookahead trees with horizon $H_a - t$ for $t = 0, 1, \ldots, H_a - 1$. In practical terms, this limits the number of trajectories that can be examined, or forces a decision maker to shorten the action horizon H_a.

To consider the performance of the SpSa algorithm on a large number of new trajectories, we develop the *Split-Trajectory Sparse Sampling* (ST-SpSa) algorithm. In ST-SpSa, at time $t = 0$, all trajectories share the same initial state and the algorithm only needs to be performed once. After the estimated optimal action is taken, the transition for each trajectory can produce up to N_{traj} unique states at $t = 1$. Testing the performance of the online method without adaptation would require performing the sparse sampling algorithm for each unique state. This would be the case for $H_{\mathrm{a}} - 1$ action epochs. Instead, to reduce computational time while considering a large number of trajectories, we perform cluster analysis of the current states at each action epoch after $t = 0$ to identify ω representative states. We add an extra dimension to the current states describing which actions are available at that state to ensure clusters include only states that share an action set. We call these clusters *trajectory groups*, to distinguish them from clusters used in the cluster adaptation. We then perform the sparse sampling algorithm on the ω representative states and apply the action to all states in the respective trajectory group. The full description and algorithmic details can be found in Algorithm 5, in Online Supplement Section B.

5.1.3. Cluster Adaptation of Sparse Sampling-Based Methods While trajectories constructed by the ST-SpSa algorithm demonstrate improvement of at least 7.3% over a static benchmark in 4 of 9 test cases in Section 7.2, in five of the test cases, we find the static benchmark performs better. Delving into some of those instances, we notice that the performance of SpSa can be sensitive to the number of absorbing states sampled at the root for a particular action. The children of the root node are random samples, subject to variance. If the number of absorbing states sampled for a particular action at the root is different enough from the expected number of absorbing states, this can impact the action chosen. On the other hand, expanding the number of children near the root of the tree has a multiplicative effect on runtime, even if $|C|$ is only changed at the first level of the tree.

For this reason, we develop an adaptation of SpSa, that utilizes large samples and associated clusters to estimate the children of nodes in the sparse lookahead tree. We demonstrate the effectiveness of this method in our numerical studies, with improvement over SpSa for all 9 instances and improvement over the static benchmark for 8 of 9 instances. We call this algorithm *Sparse Sampling with Clusters* (SpSa-C) and the corresponding out-of-sample test *Split-Trajectory Sparse Sampling with Clusters* (ST-SpSa-C).

SpSa-C works by sampling a large number of next states $\overline{C}$ and corresponding rewards everywhere that SpSa samples smaller number $|C|$ children. The mean immediate reward associated with node i and action a is estimated using set $\overline{C}$. The children of node i, C, are calculated using cluster analysis of the nonabsorbing states in $\overline{C}$. The centers of the $|C|$ clusters become children. Each child c has associated cluster of states $\overline{C}_c$. When calculating the Q function for node i and action a, we use empirical probability associated with each cluster, $\frac{|\overline{C}_c|}{|\overline{C}|}$.

The SpSa-C algorithm steps are the same as those found in Algorithms 1–4 except for a few changes. First, $|\overline{C}|$ becomes an input parameter to any algorithm that requires $|C|$, to satisfy the recursion properties. Second, Algorithm 2, EstimateQ, has the lines given in Algorithm 6, found in Online Supplement Section B, that replace lines 7–8.

5.2. Q Learning with Linear Function Approximation

The second type of approach that we consider is Q learning with linear function approximation, which estimates the state-action value function, or Q function,

$$Q_{\hat{\pi}}(i,a) = \mathbb{E}\left[\sum_{t=0}^{T-1} \gamma^t \rho(i_t, \hat{\pi}(i_t)) | i_0 = i, a_0 = a\right].$$

5.2.1. Background This approach is situated in the field of reinforcement learning. With the large instance size inherent in stochastic epidemiological environments, function approximation (Baird 1995) can be utilized with reinforcement learning, to ensure computationally feasible

algorithms. Libin et al. (2020) apply deep reinforcement learning, which uses neural networks as the function approximators, to an influenza epidemic control problem. While neural networks are powerful approximators, they also are known to be sensitive to hyperparameter tuning and require environment interaction. The Q learning algorithm with linear function approximation (Chen et al. 2019) is a method that utilizes previously simulated trajectories, and is more easily adaptable to epidemiological settings, allowing historical data or separately generated trajectories to be used to learn the optimal policy. The generative model used in the sparse sampling method can also be applied to create trajectories. The basis functions we develop specific to our test instances can be seen in Section 5.2.3.

5.2.2. Overview The Q function represents the total expected discounted reward of taking action a, then following generic policy $\hat{\pi}$. By subtracting the value function from the Q function, we get the advantage of taking action a over following policy $\hat{\pi}$ in state i. The goal of Q learning is to learn the optimal Q function, $Q(i,a) = Q_\pi(i,a)$ for optimal policy π, which satisfies

$$Q(i,a) = \rho(i,a) + \gamma \sum_j P_{ij}(a') \max_{a'} Q(j,a').$$

If we take a sample trajectory $\{(i_k, a_k)\}_{k=0}^{T-1}$ created by following the optimal policy, then we have the following unbiased estimate:

$$\hat{Q}(i_t, a_t) = \rho(i_t, a_t) + \gamma \max_{a'} Q(i_{t+1}, a').$$

This motivates the iterative Q learning algorithm, with Q estimate Q_k at each iteration k. After randomly initializing the function Q_0, we perform the following update based on a sample trajectory generated with randomized behavior policy $\hat{\pi}$.

$$Q_{k+1}(i,a) = \begin{cases} Q_k(i,a) + \zeta((i,a) + \gamma \max_{a'} Q_k(i_{k+1}, a') - Q_k(i,a)), \text{ if } (i,a) = (i_k, a_k), \\ Q_k(i,a), \text{ otherwise.} \end{cases}$$

Here, we take the difference between the newest estimate of Q^* and the previous estimate, and move toward it with step size ζ for the current state-action pair (and do not update any other state-action pair). The ability to only update one state-action pair at a time means that this tabular method requires a large number of samples.

Linear function approximation addresses this computational challenge by assuming that $Q(i,a)$ is a weighted sum of basis functions, $\phi(i,a)$, with components $m = 1, \ldots, M$ (Chen et al. 2019). With weight vector $\theta \in \mathbb{R}^M$, we have $Q \approx \phi(i,a)^T\theta$. In Q learning with linear function approximation, rather than learning the Q function, we learn the weight parameters θ. One benefit of this method is that the basis functions can incorporate domain knowledge.

We present the algorithmic details of Q learning with linear function approximation in Algorithm 7 in Online Supplement Section B. We add a convergence criterion that ensures if the distance between θ_k and θ_{k+1} falls below parameter ϵ times the magnitude of θ_k, the algorithm stops. In the algorithm, again, we have the difference between the current Q estimate and the previous one, and step size ζ. But now, because we are ascending on θ, we multiply by the gradient of Q_θ, which equals $\phi(s_k, a_k)$.

5.2.3. Basis Functions Incorporating domain knowledge into the basis functions in the Q learning method with linear function approximation can improve the algorithm's performance, allowing the functions to better approximate Q. Regardless of epidemiological outbreak or facility type, the Q function for the strategic release problem should include aspects related to epidemiological cost and release cost. In this section, we give an example of how to develop basis functions for the strategic release problem that incorporate these two cost types. We consider the prison release problem and SIR stochastic ICM with homogeneous mixing that we utilize in our computational tests. First, we describe the epidemiological basis components, and then we give the recidivism-related components.

Suppose the state of the facility at the start of epoch t is i, and we plan to take action a. Because the basis function is meant to help us estimate the Q function for the entire trajectory, not just immediate reward, we construct epidemiological components that estimate the number of new infections in each future epoch $t + \mathrm{w}$ for $\mathrm{w} = 0, \ldots, w-1$, given that action a is taken at time t, and the null action is taken later. Number of epochs to estimate w is a hyperparameter that has to be tuned.

In order to construct these components, we develop a set of recursive functions that estimate the number of new infections for health risk level $h \in \mathcal{H}$ over $w\ell$ days of the epochs, denoted $\tau = 0, \ldots, w\ell - 1$. We define $S^h(\tau)$ and $I^h(\tau)$ to be the number of susceptible and infectious individuals in the facility at the start of day τ and $I^h_{\text{new}}(\tau)$ to be the number of new infections that take place that day. These three functions all depend on state i and action a, though we omit them from the notation for ease of reading.

At the start of the first epoch, the first two functions are defined by the population after release – the number of susceptible and infectious individuals that remain.

$$S^h(0) = \sum_{r \in \mathcal{R}} S^i_{hr} - \sum_{r \in \mathcal{R}_{\text{rel}}} a_{hr} \tag{5}$$

$$I^h(0) = I^i_h. \tag{6}$$

The number of new infections during day $\tau = 0, \ldots, w\ell - 1$ is estimated using a formula based on the deterministic compartmental analogue of the stochastic ICM, given in equations (2)–(4), as follows.

$$I^h_{\text{new}}(\tau) = \min\left\{\beta_h\left(\kappa_2 N(i,a)^2 + \kappa_1 N(i,a) + \kappa_0\right)\frac{\sum_{h' \in \mathcal{H}} I^{h'}(\tau)}{N(i,a)}, 1\right\} S^h(\tau). \tag{7}$$

Recall that $\kappa_2 N(i,a)^2 + \kappa_1 N(i,a) + \kappa_0$ is the contact rate for the current facility, while β_h is the probability of transmission for health risk level $h \in \mathcal{H}$. Intuitively, $\beta_h\left(\kappa_2 N(i,a)^2 + \kappa_1 N(i,a) + \kappa_0\right)\frac{\sum_{h' \in \mathcal{H}} I_{h'}}{N(i,a)}$ estimates the expected number of infection transmission events one individual will have in the next time period – the product of their probability of infection given contact, the number of contacts they expect to have, and the probability that a contact will be infected. If this value is less than 1, it is multiplied by the total number of susceptible individuals present in the facility with that health risk level, h, to estimate the total expected number of infectious contacts. Otherwise, we estimate that all susceptible individuals with health risk level h are infected.

To update the number of susceptible individuals for the next day, we subtract those we estimated were infected:

$$S^h(\tau) = S^h(\tau - 1) - I^h_{\text{new}}(\tau - 1). \tag{8}$$

To update the number of infectious individuals in the next day, we add those we estimated were infected and then multiply by the probability of not recovering that day.

$$I^h(\tau) = (1 - \xi_h)(I^h(\tau - 1) + I^h_{\text{new}}(\tau - 1)). \tag{9}$$

Then, for each epoch, $t + \text{w}$ for $\text{w} = 0, \ldots, w - 1$, and each health risk level $h \in \mathcal{H}$, we construct the following basis components.

$$\phi^{\text{w}}_h(i,a) = \frac{1}{\sqrt{M}N}\sum_{\tau=\text{w}\ell}^{(\text{w}+1)\ell-1} I^h_{\text{new}}(\tau), \tag{10a}$$

$$\phi^{\text{w,quad}}_h(i,a) = \sqrt{M}\left(\phi^{\text{w}}_h(i,a)\right)^2. \tag{10b}$$

Our recidivism-related components, linear and quadratic functions of the number released, and bias component are as follows.

$$\phi^{\text{lin}}_r(i,a) = \frac{1}{\sqrt{M}N}\cdot\sum_{h \in \mathcal{H}} a_{hr}, \quad r \in \mathcal{R} \setminus \{r^*\} \tag{11a}$$

$$\phi^{\text{quad}}_r(i,a) = \sqrt{M}\left(\phi^{\text{lin}}_r(i,a)\right)^2, \quad r \in \mathcal{R} \setminus \{r^*\} \tag{11b}$$

$$\phi_{\text{bias}}(i,a) = \frac{1}{\sqrt{M}}. \tag{11c}$$

Recall that M is the number of components of ϕ. Here, the leading coefficients ensure that $||\phi|| \leq 1$.

6. Benchmark De-Densification Test Instances

To test our models, we examine the early prison release problem described in Section 1, with two health risk levels and three recidivism risk levels (including one ineligible for release), as well as virus states $\mathcal{V} = \{S, I, R\}$.

6.1. Epidemiological Outbreak Test Instances

We develop benchmark epidemiological outbreak test instances that can be used in future multi-objective epidemiological MDP work using an extension of the basic stochastic ICM built in to EpiModel. We adapt the existing one-group homogeneous and two-group heterogeneous mixing SIR models to a two-group homogeneous mixing SIR model with option to track individual non-health features.

6.1.1. One-Group SIR Model In the basic one-group model included in EpiModel, each day, random encounters occur between individuals in the facility, labeled $\{1, \ldots, N\}$, based on the contact rate α (e.g., 5 contacts per person). The total expected number of contact pairs $\bar{z}$ is calculated using the following formula:

$$\bar{z} = \left\lceil \frac{\alpha N}{2} \right\rceil .$$

Then, two random samples of size $\bar{z}$ are taken from individuals $\{1, \ldots, N\}$, with replacement, to form vectors $\mathbf{e}^1$ and $\mathbf{e}^2$ of length $\bar{z}$. If any component $i = 1, \ldots, \bar{z}$ of vectors $\mathbf{e}^1$ and $\mathbf{e}^2$ has the property that $e_i^1 = e_i^2$, entry e_i^2 is resampled until that is no longer the case. The ith component of each vector then represents an encounter between individual e_i^1 and e_i^2, for $i = 1, \ldots, \bar{z}$.

For each encounter, the current status of each individual is checked. If one individual is susceptible and another infectious, a random binomial draw determines whether the susceptible individual is infected. Finally, among the infected, random binomial draws determine whether recovery takes place. The random encounter, infection, and recovery steps repeat for each day of the simulation.

This one-group model assumes that probability of transmission and recovery rate are the same for the whole population. This is not ideal for our problem, in which we are interested in balancing various levels of health risk and recidivism risk.

6.1.2. Two-Group Heterogeneous Mixing SIR Model The two-group heterogeneous mixing model built into EpiModel expands upon the one-group model. The basic structure remains the same, with each day composed of random encounters, infection transmission, and recovery. In contrast to the basic model, however, the extension assumes the facility has two distinct groups, each with their own probability of infection and recovery rate. (In particular, when a susceptible individual in group 1 has contact with an infectious individual in group 2, the random binomial draw uses the group 1 probability of infection.) This property is a better fit for our problem, where

we are interested in considering the effect of releasing individuals from a facility with different health and recidivism risk levels.

On the other hand, this built-in extension assumes the groups mix purely heterogeneously, meaning a person from group 1 only interacts with people in group 2, and vice versa. The EpiModel programmers mention this type of model can be appropriate for a sexually transmitted disease under simplifying assumptions about gender and sexual orientation (EpiModel 2022).

For our problem, on the other hand, where we are interested in exploring the effects of de-densification of crowded facilities on epidemiological outbreaks, the homogeneous mixing used in the one-group model is a better fit. For this reason, we adapt the code from the two models to create a new two-group homogeneous mixing model.

6.1.3. Two-Group Homogeneous Mixing SIR Model with Non-Health Features In our adaptation of the stochastic ICM, we simulate random encounters using the one-group model method described in Section 6.1.1. The random binomial draws that simulate infection and recovery, however, use group-specific probability and rate, respectively, like the two-group heterogeneous mixing model described in Section 6.1.2. This code can be used to simulate two-group homogeneous mixing in general.

In addition, more specific to our problem, the adaptation includes the option to label each initial susceptible individual with one of three release categories. Then, as these labeled individuals randomly interact and are infected, we keep track of how many susceptible individuals remain in each health and release category. Because release category is no longer relevant for infected and recovered individuals, who are not eligible for release, we only keep track of release category for susceptible individuals. This ensures that we do not consider two states to be different when, in essence, they are the same.

In our code, we refer to release category more generically as a feature. In this way, we distinguish non-health feature from group, which in EpiModel refers to a designation that impacts epidemiological parameters.

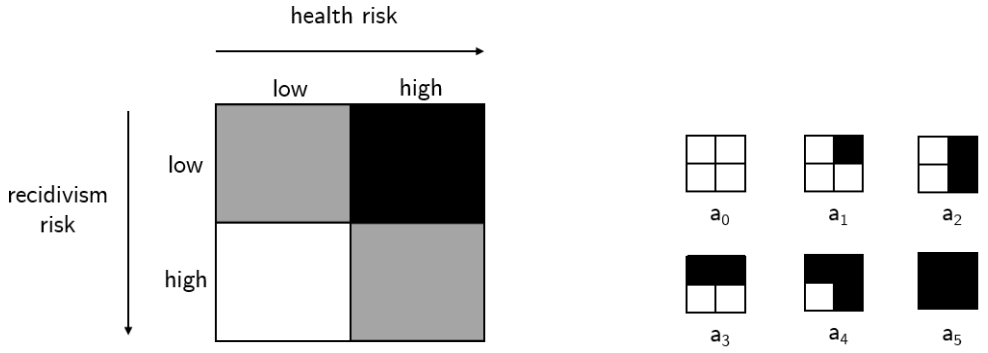
6.2. Epidemiological Parameters

Because we have low- and high-risk health levels in our experiments, we assume that the high-risk susceptible population has a higher probability of becoming infected when they come into contact with the pathogen than the low-risk susceptible population. We also assume that the rate of recovery (or the inverse of recovery time) is smaller for the high-risk infected population, as shown in Table 4.

We assume 60% of the 100-person facility is low-health risk level (h_1), and 50% are ineligible for release (r_3). Of those eligible for release, we assume an equal proportion are low recidivism risk

Table 4 Epidemiological Parameters

	Low health-risk group h_1	High health-risk group h_2
Probability of infection	0.01	0.1
Recovery rate	0.5	0.125



(a) Release Priority

(b) Action Space

Figure 2 (a) Priority for prisoner release. Darker boxes are higher priority. (b) Available actions in each epoch. For each action, all currently susceptible prisoners in the shaded categories are released.

(r_1) and high recidivism risk (r_2). We assume that one person is initially infected. Table 1 depicts the full initial state of the system. We assume that at full capacity, the average daily contact rate is 10.

6.3. Action Space Definition

In order to test the multi-criteria MDP with these test instances, we need to define an action space for releases. To limit the size of the lookahead tree in the sparse sampling algorithm, we restrict our action space to six possible release actions, illustrated in Figure 2. We choose these actions to reflect a prioritization of higher health risk and lower recidivism risk individuals for release, as shown in Figure 2a. In each of actions $\{a_0, \ldots, a_5\}$, we release all currently susceptible individuals in particular (h, r) risk categories, as pictured in Figure 2b. In action a_0, no one is released; in a_1, only the highest priority individuals are released – those with high health risk and low recidivism risk. In actions a_2 and a_3, we allow either all high health risk or all low recidivism risk populations to be released, which allows us to compare the weight of health versus recidivism risk in the resulting optimal policy. Finally, in a_4, we release all but the lowest priority individuals – those with high recidivism and low health risk, and in a_5, we release all eligible individuals who are currently susceptible.

Note that not all actions are available in every state. In order to release susceptible individuals in a certain category, there must be currently susceptible individuals in that category. In Table 5,

Table 5 **Categories that must have currently susceptible individuals for each action to be taken.**

Action	Category
a_0	
a_1	(high health, low rec)
a_2	(high health, high rec)
a_3	(low health, low rec)
a_4	(low health, low rec), (high health, high rec)
a_5	(low health, high rec)

we list the categories that must have currently susceptible individuals in order for an action to be available. For instance, to take action a_4, both (low health, low rec) and (high health, high rec) must have currently susceptible individuals. Otherwise, the appropriate equivalent action to a_4 is either a_2 or a_3. This table ensures that the effect of each action is unique for every state.

Our choice of action space takes advantage of a natural prioritization of individuals for release, and allows us to limit the size of lookahead tree. There are two key characteristics of this action space that help with this. First, there are only six actions, which limits the action-driven growth of the sparse sampling algorithm tree at each epoch to multiplication by six. Second, and more importantly, by releasing *all* susceptible individuals in a particular (h, r) category at state i, we further limit the set of actions available in the next epoch.

To maintain consistency across methods, we use these six actions in our benchmark tests and Q learning with linear function approximation tests as well.

6.4. Contact Rate Assumptions

The final epidemiological parameter that we need to define is the *de-densification function*, or how the contact rate changes as the crowding level decreases. We consider three functions with quadratic form (1), with coefficients given in Table 6. Notice that at full capacity, the contact rate is 10. All three satisfy the conditions in Theorem 1 for de-densification functions, and the first two satisfy the conditions given by Theorem 2, ensuring that release actions affecting one health risk level lower the new infection rate for the other health risk level.

Table 6 **Quadratic coefficients for de-densification functions.**

Contact Rate Assumption	κ_2	κ_1	κ_0
Polynomial	0.00044	0.046	1
Quadratic	0.001	0	0
Proportional	0	0	0.1

6.5. Multiobjective Parameters

A key aspect of the strategic release problem is the balancing of competing objectives. We test three reward functions, that either (a) prioritize health, (b) prioritize crime reduction, or (c) give equal priority.

We define health cost $\rho_{\text{ep}}(i,a)$ as the expected number of long-term health consequences from newly infected individuals. Given probability of long-term health effect p_h for $h \in \mathcal{H}$ and number of newly infected individuals I^h_{new}, we have

$$\rho_{\text{ep}}(i,a) = \sum_{h \in H} p_h I^h_{\text{new}}. \tag{12}$$

We define release cost $\rho_{\text{rel}}(i,a)$ as the expected number of crimes committed by newly released individuals. Given probability of recidivism p_r for $r \in \mathcal{R} \setminus \{r_3\}$ and number of newly released individuals a^r_{new}, we have

$$\rho_{\text{rel}}(i,a) = \sum_{r=r_1,r_2} p_r a^r_{\text{new}}. \tag{13}$$

We assume the probability is larger for high risk than low risk in both cases, as seen in Table 7.

Table 7 Reward Parameters

	Low risk	High risk
Probability of recidivism (p_{r_1}, p_{r_2})	0.1	0.25
Probability of long-term health effect (p_{h_1}, p_{h_2})	0.1	0.25

For each of the different cost functions, the cost is defined using priority weights λ^{rel} and λ^{health} as

$$\rho(i,a) = -\lambda^{\text{rel}}\rho_{\text{rel}}(i,a) - \lambda^{\text{health}}\rho_{\text{health}}(i,a).$$

The weights we use are shown in Table 8. Here, we assume when we prioritize health that one

Table 8 Weight Parameters

Prioritize	γ^{rel}	γ^{health}
Health	1	2
Crime Reduction	2	1
Both Equally	1	1

long-term effect is as costly as two crimes, while one crime is as costly as two long-term effects when we prioritize crime reduction.

7. Computational Experiments

In this paper, we introduce three methods, from sparse sampling-based approaches with and without clusters to Q learning with linear function approximation. We also develop nine test instances, taking into account de-densification and reward functions. In this section, we describe the experimental setup and delve into the results, comparing the reward and computational performance associated with each of the three methods across the nine test cases. In general, we find that the cluster method outperforms the baseline SpSa in all cases, as well as Q learning with linear function approximation methods for crime reduction prioritization. The Q learning method yields similar objective values to ST-SpSa-C for the other two priority weights, with shorter runtimes.

7.1. Experimental Setup

For each test instance, we compare the performance of the three different types of policies on a set of 1000 trajectories of 10 epochs each. These trajectories share an initial state, and each has a fixed seed associated with every epoch transition. This ensures that when an action is shared across policies for a specific trajectory, epoch, and state, the transition to the next state is also the same. We consider discount $\gamma = 0.9$.

As a benchmark to evaluate the policies generated by our dynamic methods, we consider static, single-action policies that take place before the first epoch. We consider each of the six actions, running 1000 trajectories after each action is applied, and choose the action with the best mean discounted reward as the benchmark for each test.

We implement ST-SpSa with and without clusters in R, using a generic Markov simulation framework we describe in Online Supplement Section C. For our experiments, we use the following parameters, shown in Table 9.

Table 9 ST-SpSa Parameters

Parameter	Value
Action horizon H_a	3
Number of trajectories to estimate leaf node Q N_{final}	10
Trajectory groups ω	10
Baseline Children $\lvert C\rvert$	10
Clusters Children $\lvert C\rvert$	5
Clusters Samples $\lvert\overline{C}\rvert$	100

Like the sparse sampling methods, we implement the trajectory sampling portion of Algorithm 7 in R, using the generic Markov simulation framework. The randomized policy works as follows. At the start of each new trajectory, we choose three random epochs in which an action can be taken. Then, when an action epoch arrives, an action is randomly chosen from available actions.

In some cases, only the null action will be available at future action epochs. We use this policy to encourage the agent to "wait and see" during epoch 0 in a high number of trajectories. The number of environmental interactions that we require is 100,000.

We implement the remainder of Q learning with linear function approximation in Python with convergence criteria $\epsilon = 1 \times 10^{-6}$. For each test instance, we consider the set of hyperparameters seen in Table 10.

Table 10 Q **learning with linear function approximation Hyperparameters Tested**

Initial θ Seed	Step Type	Step Size	w
0	div_root	0.5	1
1	div_linear	0.75	2
		1	3

The step ζ is a function of algorithm iteration k and step size. When the step type is div_root and step size is .75, $\zeta = 0.75/\sqrt{k}$; when step type is div_linear, $\zeta = 0.75/k$. The initial weight θ_0 is randomly initialized to values in $[-1, 0]$ using random seeds 0 or 1.

Because reinforcement learning methods are known to be sensitive to hyperparameters, we tune them for each of the 9 test instances separately, and choose the resulting θ that gives the best out-of-sample value.

7.2. Comparison Across Methods

Table 11 **Percentage improvement in total mean discounted reward for different methods over static benchmark, for different contact assumptions and priority weights.**

	Polynomial			Quadratic			Proportional		
Method	**Health**	**Equal**	**Crime**	**Health**	**Equal**	**Crime**	**Health**	**Equal**	**Crime**
ST-SpSa	-22.9	-10.3	9.0	-27.0	-10.5	11.6	9.7	-2.2	7.3
ST-SpSa-C	13.3	1.3	10.0	5.0	-0.4	13.0	11.4	0.2	8.7
QLLFA	13.3	0.9	-0.9	5.7	0.5	2.5	11.6	0	-0.6

Table 11 shows how the out-of-sample objective changes when the policies given by each method are followed, rather than the best single release action. Notice that the sparse sampling algorithm with clusters and Q learning method lead to the highest percentage improvement in the health prioritization cases, as well as similar performance to the static benchmark in the equal priority cases. When crime reduction is highest priority, ST-SpSa-C performs best. In Section 7.3, we delve into the polynomial case and demonstrate that the dynamic ST-SpSa-C method chooses a largely static policy for equal priority weight, explaining its lack of improvement.

Without the cluster adaptation, ST-SpSa yields varying results – with improvement as high as 11.6% and decrement as high as 27%. Incorporating k-means clustering to the algorithm improves consistency. In Table 12, we explore why this may be the case. Here, we note the probability of the epidemic dying out in the first epoch after a particular release action is taken, over the 1000 out-of-sample benchmark trajectories. Because of the large number of trajectories, we expect that this empirical probability is our best estimate of the true probability. We then examine the sparse

Table 12 Probability of epidemic dying out after one epoch for each action in static benchmark trajectories and sparse sampling lookahead trees, with and without clusters.

	Polynomial			Quadratic		
Action	**Benchmark**	**ST-SpSa-C**	**ST-SpSa**	**Benchmark**	**ST-SpSa-C**	**ST-SpSa**
a_0	0.616	0.590	0.600	0.616	0.590	0.600
a_1	0.722	0.710	0.600	0.702	0.780	0.400
a_2	0.801	0.780	0.900	0.839	0.870	0.900
a_3	0.729	0.730	0.900	0.764	0.750	0.700
a_4	0.806	0.820	0.800	0.857	0.870	1
a_5	0.814	0.840	0.700	0.890	0.880	0.900

lookahead trees for each of the two methods, comparing the sampled child nodes at the initial state root. For the quadratic contact rate assumption and high health, low recidivism action a_1, for example, 702 out-of-sample trajectories had 0 infections after 1 epoch. For the baseline ST-SpSa sparse lookahead tree, after action a_1 was taken, 4 of 10 sampled child nodes had 0 infections, yielding a probability of .400 of the epidemic dying out. With the cluster adaptation, on the other hand, 78 of 100 sampled children had 0 infections, yielding a much better estimation of the empirical probability of the epidemic dying out, of .780. Here, we see that using a larger sample size with k-means clustering to generate the child nodes better estimates this probability. In turn, the clustering method more consistently outperforms a static benchmark in out-of-sample mean total discounted cost.

Table 13 Runtime (hours) for each computational method.

	Polynomial			Quadratic			Proportional		
Method	**Health**	**Equal**	**Crime**	**Health**	**Equal**	**Crime**	**Health**	**Equal**	**Crime**
ST-SpSa	7.04	15.89	16.52	6.76	12.58	12.92	10.6	11.98	13.19
ST-SpSa-C	9.42	9.31	11.63	9.16	9.35	10.99	10.03	10.08	11.87
QLLFA	8.39	8.48	8.35	7.82	7.70	7.68	5.99	6.08	6.03

At the same time, the computational runtime of ST-SpSa-C is generally similar or better than ST-SpSa, as seen in Table 13. Only in the cases where ST-SpSa strongly underperforms in terms of

objective – the polynomial and quadratic prioritize health instances – is the runtime slightly shorter. Thus, adding the cluster adaptation generally improves objective performance while maintaining runtime. The runtime for Q learning with linear function approximation is shorter than both other methods, though the objective performance is more variable for this method across priority weights.

7.3. Comparison Across Policy Weights

To analyze how the policy changes for different priority objectives, we consider the polynomial contact rate assumption. Because the cluster method outperforms the baseline sparse simulator algorithm and shows more consistent performance over Q learning with linear function approximation across priority weights, we consider the policies given by this method.

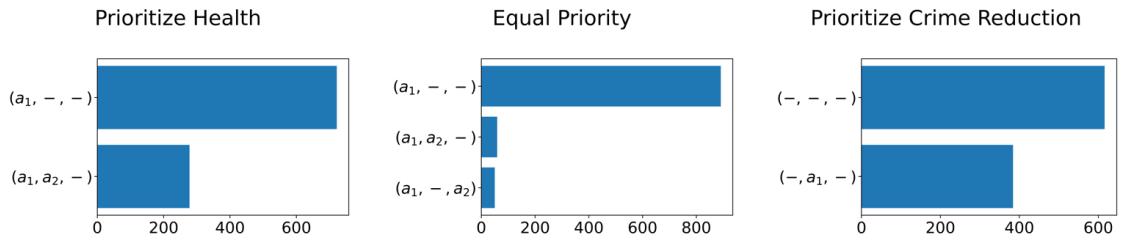


Figure 3 **Number of 1000 out-of-sample trajectory policies given by cluster method of each type, for polynomial contact rate assumption and three priority weights.**

We illustrate the distribution of the 1000 out-of-sample trajectory policies for each priority weight in Figure 3. Notice that the instances with greater improvement over the static benchmark, prioritize health and crime reduction, benefit most from the dynamic nature of the cluster method, with at least 20% of trajectories utilizing a release action after one or two epochs. In line with intuition, the initial action is most aggressive for the health prioritization and equal priority cases, a_1, where all individuals with high health and low recidivism risk are released, as opposed to no initial action taken for crime reduction prioritization.

Another point of interest in this figure is that releasing all low recidivism risk individuals (a_3) is never a chosen action. When more individuals than the high health and low recidivism risk group are released, the policies release all high health risk (a_2).

Next, we examine the costs associated with the chosen policies for each prioritization type. Figure 4 shows the mean infection and crime cost each epoch for out-of-sample tests of the cluster algorithm policy for the polynomial contact rate assumption, under different priority weights. Notice that long-term health effects exceed crime throughout the planning horizon when we prioritize crime reduction. For the equal priority weight, crime exceeds long-term health effects for one epoch, and when we prioritize health, this is the case for two epochs.

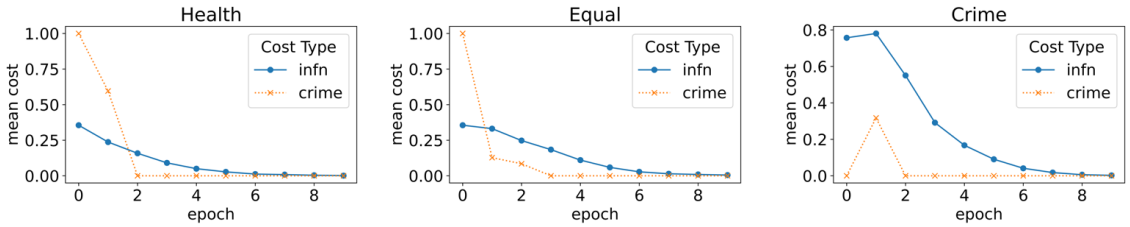


Figure 4 **Mean cost by cost type over planning horizon for different priority weights under polynomial contact rate assumption for out-of-sample tests using cluster algorithm policy.**

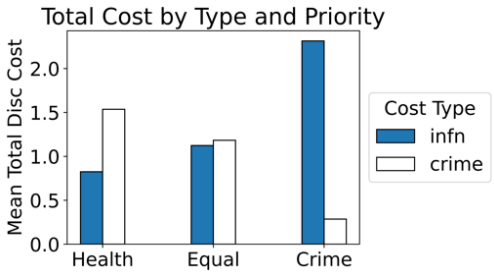


Figure 5 **Total mean discounted cost by cost type for different priority weights under polynomial contact rate assumption for cluster algorithm out-of-sample tests.**

In Figure 5, we illustrate the mean total discounted cost for each priority weight for the entire planning horizon. The total discounted crimes committed exceed total discounted long-term health effects for the health prioritization and equal priority objectives. This trend is reversed for the crime reduction prioritization. Here, the priority weights are able to produce distinct policies with the desired differentiated effects.

8. Conclusion

In this paper, we considered the novel de-densification for pandemic containment problem, developing models and solution methods to implement strategic releases that weigh epidemiological and release-related risk. We introduced the problem, noted its key features, and constructed a multi-criteria, multidimensional MDP to model facility population, decisions, and associated costs. We modeled the social distancing effect of reduced crowding with the concept of a de-densification function, and derived theoretical properties under particular assumptions. We also developed general approaches with open-source software implementations for epidemiological MDPs.

In particular, we formulated a cluster-based algorithm to perform large out-of-sample tests of an online sparse sampling policy. We also improved the online sparse sampling algorithm itself, using k-means clustering to create more representative sample nodes in the lookahead tree. These two contributions allowed us to develop a policy that demonstrably improves objective performance

over a static benchmark on a large set of multi-epoch trajectories under a range of contact rate assumptions and two priority weights.

As part of our numerical studies, we also constructed freely available benchmark epidemiological test instances, that can be used in future research. We developed basis functions for Q learning with linear function approximation specific to these test instances and demonstrate their effectiveness in health prioritization cases.

This area of research is new, and provides many opportunities for innovation. One area that could benefit from future study is the de-densification function. The basis functions could also be adapted to try to better approximate the Q function in crime reduction prioritization cases. Additionally, it could be interesting to apply neural networks to the problem, to see how computational runtime and out-of-sample objective performance compare to the online sparse sampling algorithm.

References

Akbarpour M, Cook C, Marzuoli A, Mongey S, Nagaraj A, Saccarola M, Tebaldi P, Vasserman S, Yang H (2020) Socioeconomic network heterogeneity and pandemic policy response. *University of Chicago, Becker Friedman Institute for Economics Working Paper* (2020-75).

Albert R, Jeong H, Barabási AL (2000) Error and attack tolerance of complex networks. *Nature* 406(6794):378–382.

Baird L (1995) Residual algorithms: Reinforcement learning with function approximation. *Machine Learning Proceedings 1995*, 30–37 (Elsevier).

Barak B, Nitzan M, Tannenbaum NR, Yuval J (2020) Optimizing testing policies for detecting COVID-19 outbreaks. *arXiv preprint arXiv:2007.04827* .

Barnhart C, Bertsimas D, Delarue A, Yan J (2022) Course scheduling under sudden scarcity: applications to pandemic planning. *Manufacturing & Service Operations Management* 24(2):727–745.

Basciftci B, Yu X, Shen S (2023) Resource distribution under spatiotemporal uncertainty of disease spread: Stochastic versus robust approaches. *Computers & Operations Research* 149:106028.

Baycik NO, Sharkey TC, Rainwater CE (2020) A Markov Decision Process approach for balancing intelligence and interdiction operations in city-level drug trafficking enforcement. *Socio-Economic Planning Sciences* 69:100700.

Birge JR, Candogan O, Feng Y (2020) Controlling epidemic spread: Reducing economic losses with targeted closures. *University of Chicago, Becker Friedman Institute for Economics Working Paper* (2020-57).

Bisset KR, Feng X, Marathe M, Yardi S (2009) Modeling interaction between individuals, social networks and public policy to support public health epidemiology. *Proceedings of the 2009 Winter Simulation Conference (WSC)*, 2020–2031 (IEEE).

Blount S, Galambosi A, Yakowitz S (1997) Nonlinear and dynamic programming for epidemic intervention. *Applied Mathematics and Computation* 86(2-3):123–136.

Brandeau ML, Zaric GS, Richter A (2003) Resource allocation for control of infectious diseases in multiple independent populations: Beyond cost-effectiveness analysis. *Journal of Health Economics* 22(4):575–598.

Brauer F (2008) Compartmental models in epidemiology. *Mathematical Epidemiology*, 19–79 (Springer).

Brown KA, Jones A, Daneman N, Chan AK, Schwartz KL, Garber GE, Costa AP, Stall NM (2020) Association between nursing home crowding and COVID-19 infection and mortality in Ontario, Canada. *JAMA Internal Medicine* .

Chen Z, Zhang S, Doan TT, Maguluri ST, Clarke JP (2019) Finite-time analysis of Q-learning with linear function approximation. *arXiv preprint arXiv:1905.11425* .

Churches T, Jorm L, et al. (2020) Flexible, freely available stochastic individual contact model for exploring COVID-19 intervention and control strategies: Development and simulation. *JMIR Public Health and Surveillance* 6(3):e18965.

Cohen A, Eisen LB (2022) Reducing jail and prison populations during the COVID-19 pandemic. URL `https://www.brennancenter.org/our-work/research-reports/reducing-jail-and-prison-populations-during-covid-19-pandemic`.

Dai T, Song JS (2021) Transforming covid-19 vaccines into vaccination: Challenges and opportunities for management scientists. *Health care management science* 24(3):455–459.

Du M, Sai A, Kong N (2021) A data-driven optimization approach for multi-period resource allocation in cholera outbreak control. *European Journal of Operational Research* 291(3):1106–1116.

Duque D, Morton DP, Singh B, Du Z, Pasco R, Meyers LA (2020) Timing social distancing to avert unmanageable COVID-19 hospital surges. *Proceedings of the National Academy of Sciences* 117(33):19873–19878.

Enns EA, Mounzer JJ, Brandeau ML (2012) Optimal link removal for epidemic mitigation: A two-way partitioning approach. *Mathematical Biosciences* 235(2):138–147.

EpiModel (2022) Basic DCMs with EpiModel. URL `https://www.epimodel.org/tut/BasicDCMs.html#SI_Model_with_Demography`, accessed August 22, 2022.

Fajgelbaum P, Khandelwal A, Kim W, Mantovani C, Schaal E (2020) Optimal lockdown in a commuting network. Technical report, National Bureau of Economic Research.

Fanelli D, Piazza F (2020) Analysis and forecast of COVID-19 spreading in China, Italy and France. *Chaos, Solitons & Fractals* 134:109761.

Goldman S, Lightwood J (2002) Cost optimization in the SIS model of infectious disease with treatment. *Topics in Economic Analysis & Policy* 2(1):1007.

Gore AB, Kurz ME, Saltzman MJ, Splitter B, Bridges WC, Calkin NJ (2022) Clemson university's rotational attendance plan during covid-19. *INFORMS Journal on Applied Analytics* 52(6):553–567.

Hou C, Chen J, Zhou Y, Hua L, Yuan J, He S, Guo Y, Zhang S, Jia Q, Zhao C, et al. (2020) The effectiveness of quarantine of Wuhan city against the Corona Virus Disease 2019 (COVID-19): A well-mixed SEIR model analysis. *Journal of Medical Virology* 92(7):841–848.

Jackson A, Tanner K (2020a) Active COVID-19 cases in Michigan prisons are down, but experts say threat isn't over. `https://www.freep.com/story/news/local/michigan/2020/07/24/michigan-prisons-coronavirus-cases/5470839002/`.

Jackson A, Tanner K (2020b) Coronavirus cases at Michigan prison surge as widespread testing begins. `https://www.freep.com/story/news/local/michigan/2020/04/25/coronavirus-cases-michigan-prison-surge-widespread-testing-prisoners/3002811001/`.

Jenness SM, Goodreau SM, Morris M (2018) EpiModel: An R package for mathematical modeling of infectious disease over networks. *Journal of Statistical Software* 84.

Kearns M, Mansour Y, Ng AY (2002) A sparse sampling algorithm for near-optimal planning in large Markov Decision Processes. *Machine Learning* 49(2):193–208.

Keeling MJ, Rohani P (2011) *Modeling Infectious Diseases in Humans and Animals* (Princeton University Press).

Kermack WO, McKendrick AG (1927) A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character* 115(772):700–721.

Kimura M, Saito K, Motoda H (2009) Blocking links to minimize contamination spread in a social network. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 3(2):1–23.

Koch D, Illner R, Ma J (2013) Edge removal in random contact networks and the basic reproduction number. *Journal of Mathematical Biology* 67(2):217–238.

Libin P, Moonens A, Verstraeten T, Perez-Sanjines F, Hens N, Lemey P, Nowé A (2020) Deep reinforcement learning for large-scale epidemic control. *arXiv preprint arXiv:2003.13676* .

Mbah MLN, Gilligan CA (2011) Resource allocation for epidemic control in metapopulations. *PLoS one* 6(9):e24577.

Mishra BK, Keshri AK, Rao YS, Mishra BK, Mahato B, Ayesha S, Rukhaiyyar BP, Saini DK, Singh AK (2020) COVID-19 created chaos across the globe: Three novel quarantine epidemic models. *Chaos, Solitons & Fractals* 138:109928.

Nandi AK, Medal HR (2016) Methods for removing links in a network to minimize the spread of infections. *Computers & Operations Research* 69:10–24.

Navabi-Shirazi M, El Tonbari M, Boland N, Nazzal D, Steimle LN (2022) Multicriteria course mode selection and classroom assignment under sudden space scarcity. *Manufacturing & Service Operations Management* 24(6):3252–3268.

Nowzari C, Preciado VM, Pappas GJ (2015) Optimal resource allocation for control of networked epidemic models. *IEEE Transactions on Control of Network Systems* 4(2):159–169.

Reluga TC (2009) An SIS epidemiology game with two subpopulations. *Journal of Biological Dynamics* 3(5):515–531.

Ross SM (2014) *Introduction to stochastic dynamic programming* (Academic Press).

Silva PC, Batista PV, Lima HS, Alves MA, Guimarães FG, Silva RC (2020) COVID-ABS: An agent-based model of COVID-19 epidemic to simulate health and economic effects of social distancing interventions. *Chaos, Solitons & Fractals* 139:110088.

Smith DR, Duval A, Pouwels KB, Guillemot D, Fernandes J, Huynh BT, Temime L, Opatowski L (2020) Optimizing COVID-19 surveillance in long-term care facilities: A modelling study. *BMC Medicine* 18(1):1–16.

Thunström L, Newbold SC, Finnoff D, Ashworth M, Shogren JF (2020) The benefits and costs of using social distancing to flatten the curve for COVID-19. *Journal of Benefit-Cost Analysis* 1–27.

Wang X, Pasco RF, Du Z, Petty M, Fox SJ, Galvani AP, Pignone M, Johnston SC, Meyers LA (2020) Impact of social distancing measures on coronavirus disease healthcare demand, central Texas, USA. *Emerging Infectious Diseases* 26(10):2361.

Yaesoubi R, Cohen T (2011) Generalized Markov models of infectious disease spread: A novel framework for developing dynamic health policies. *European Journal of Operational Research* 215(3):679–687.

Yin X, Büyüktahtakın İE, Patel BP (2023) Covid-19: Data-driven optimal allocation of ventilator supply under uncertainty and risk. *European journal of operational research* 304(1):255–275.

Zaric GS, Brandeau ML (2001) Resource allocation for epidemic control over short time horizons. *Mathematical Biosciences* 171(1):33–58.

Zaric GS, Brandeau ML (2002) Dynamic resource allocation for epidemic control in multiple populations. *Mathematical Medicine and Biology* 19(4):235–255.

Online Supplement of the Paper: "The Costs of Overcrowding (and Release): Strategic Discharges for Isolated Facilities During Epidemiological Outbreaks"

Appendix A: Notation

Category	Symbol	Meaning
MDP	Ω	MDP State Space
	$\mathcal{A}$	MDP Action Space
	$P_{ij}(a)$	Probability of transition from i to j if action a is taken
	$\rho_{\text{ep}}(i,a)$	Expected epidemiological cost of $i \in \Omega$, $a \in \mathcal{A}(i)$
	$\rho_{\text{rel}}(i,a)$	Expected release cost of $i \in \Omega$, $a \in \mathcal{A}(i)$
	λ_{ep}	Weight in [0,1] of epidemiological costs
	$\rho(i,a)$	Weighted cost function
	ℓ	Length of epoch
	T	finite-horizon length
	$\mathcal{H}$	Set of health risk levels
	$\mathcal{R}$	Set of release categories
	$r^* \in \mathcal{R}$	Category ineligible for release
	$\mathcal{V}$	Set of virus states, e.g., S, I, R
	v^i_{hr}	Number of individuals in population with $h \in \mathcal{H}$, $r \in \mathcal{R}$, $v \in \mathcal{V}$
	$W(i)$	Value function for $i \in \Omega$
	$Q(i,a)$	State-action value function for $i \in \Omega$, $a \in \mathcal{A}(i)$
	$\pi(i)$	Optimal action taken in state $i \in \Omega$
SpSa	G	Generative model of the MDP
	d	Current depth of lookahead tree
	H_a	Action horizon length
	H_{final}	Number of epochs no action is taken
	$\lvert C\rvert$	Number of children for each node, action pair
	N_{final}	Number of trajectories to estimate value function of leaves
SpSa-C	$\lvert\overline{C}\rvert$	Number of samples to form representative children
ST-SpSa(-C)	ω	Number of trajectory groups
QLLFA	θ	Weight parameters learned by the algorithm
	k	Iteration of the algorithm
	θ_k	Estimate of θ during iteration k
	$\phi(i,a)$	Basis functions to estimate $Q(i,a)$ for $i \in \Omega$, $a \in \mathcal{A}(i)$
	ϵ	Convergence criterion
Epi Instance	α	Average contact rate per day, whole population
	β_h	Probability of transmission for invidividual with health risk $h \in \mathcal{H}$
	ξ_h	Rate of recovery from infection for invidividual with health risk $h \in \mathcal{H}$
	N	Total facility population
	$N(i,a)$	Population size after release action a occurs in state i
	κ_2, κ_1, κ_0	Constants for quadratic de-densification function (1)
Release Action	a_0	none
	a_1	high health, low recidivism only
	a_2	high health only
	a_3	low recidivism only
	a_4	only exclude low health, high recidivism
	a_5	all eligible

Table 14 Summary of Notation and Key Parameter

Appendix B: Algorithms

Algorithm 1 Sparse Sampling (SpSa)

Input: Action horizon H_{a}, initial state i, discount γ, MDP generative model G, number of children $|C|$, number of final epochs H_{final}, number of trajectories to estimate value function of leaves N_{final}.

Output: Action that maximizes estimated Q function at state i.

Define $\{(a, \hat{Q}_{H_{\mathrm{a}}}(i,a)) : a \in \mathcal{A}(i)\}$=**EstimateQ**($H_{\mathrm{a}}$, i, γ, G, $|C|$, H_{final}, N_{final})

return $\arg\max_{a \in \mathcal{A}(i)} \{\hat{Q}_{H_{\mathrm{a}}}(i,a))\}$

Algorithm 2 EstimateQ

Input: Current depth of lookahead tree d, current state i, discount γ, MDP generative model G, number of children $|C|$, number of final epochs H_{final}, number of trajectories to estimate value function of leaves N_{final}.

Output: Estimated Q function for current state i and all allowable actions.

if State i is absorbing **then**

  return $\{(a_0, 0)\}$

else if Depth $d = 0$ **then**

  return EstimateFinalW(i, γ, G, H_{final}, N_{final})

else

  for Action $a \in \mathcal{A}(i)$ **do**

    Using generative model G, generate $|C|$ samples of next states C and calculate mean sampled cost $\hat{\rho}(i,a)$

    Estimate $\hat{Q}_d(i,a)$ as follows:

$$\hat{Q}_d(i,a) = \hat{\rho}(i,a) + \gamma \frac{1}{|C|} \sum_{j \in C} \mathbf{EstimateW}(d-1, j, \gamma, G, |C|, H_{\mathrm{final}}, N_{\mathrm{final}})$$

  end for

end if

return $\{(a, \hat{Q}_d(i,a)) : a \in \mathcal{A}(i)\}$

Algorithm 3 EstimateFinalW

Input: Current state i, discount γ, MDP generative model G, number of final epochs H_{final}, number of trajectories to estimate value function of leaves N_{final}.

Output: Estimated value function for state i during final H_{final} epochs.

1: Using generative model G, generate N_{final} trajectories of H_{final} epochs each from state i, taking null action a_0 each epoch t and noting reward r_{nt}, for $n = 1, \ldots, N_{\text{final}}$.

2: Calculate mean total discounted reward

$$\hat{W}_0(i, a_0) = \frac{1}{N_{\text{final}}} \sum_{n=1}^{N_{\text{final}}} \sum_{t=0}^{H_{\text{final}}-1} \gamma^t \hat{\rho}_{nt}$$

3: **return** $\{(a_0, \hat{W}_0(i, a_0))\}$

Algorithm 4 EstimateW

Input: Current depth of lookahead tree d, current state i, discount γ, MDP generative model G, number of children $|C|$, number of final epochs H_{final}, number of trajectories to estimate value function of leaves N_{final}.

Output: Estimated value function for current state i.

1: Define $\{(a, \hat{Q}_d(i, a)) : a \in \mathcal{A}(i)\}$=**EstimateQ**($d$, i, γ, G, $|C|$, H_{final}, N_{final})

2: **return** $\max_{a \in \mathcal{A}(i)}\{\hat{Q}_d(i, a))\}$

Appendix C: Open-Source Generic Markov Simulation Code

To test our methods, we develop a generic Markov simulation framework in R that can be used for other generative MDP problems. This framework includes a number of objects that work together to simulate and analyze discrete Markov chains, including the following.

- **GlobalMarkovParameters.** This class contains attributes that describe the Markov chain, such as num_epochs (length of horizon) and state_components (labels for components associated with a state, e.g., *Infectious,* h_1).
- **TransitionSimulator.** This class simulates the transition from a state, action pair, generating a reward and next state.
- **ActionGenerator.** This class generates the set of actions available at a particular state.
- **ActionSimulator.** This class determines the action to take at a particular state.
- **MarkovTrajSimulator.** This class generates a Markov trajectory of particular length, using GlobalMarkovParameters, TransitionSimulator, and ActionSimulator objects.
- **MarkovResults.** This class analyzes a dataframe of Markov trajectories, calculating useful metrics such as mean total discounted reward for a given set of discounts and weights.

We also implement generic subclasses of ActionSimulator that can be used with discrete Markov chains.

Algorithm 5 Split-Trajectory Sparse Sampling (ST-SpSa)

Input: Action horizon H_{a}, initial state i, discount γ, MDP generative model G, number of children $|C|$, number of final epochs H_{final}, number of trajectories to estimate value function of leaves N_{final}, number of out-of-sample trajectories N_{traj}, number of trajectory groups ω.

Output: Set of N_{traj} trajectories of length $H_{\mathrm{a}} + H_{\mathrm{final}}$ with actions taken in first H_{a} epochs.

1: Obtain estimated optimal initial action with **SpSa**(H_{a}, i, γ, G, $|C|$, H_{final}, N_{final}).
2: Generate reward and next states $\hat{\Omega}_1$ from i and estimated optimal action for N_{traj} trajectories.
3: **for** $t = 1, \ldots, H_{\mathrm{a}} - 1$ **do**
4: Identify actions available in each state $j \in \hat{\Omega}_t$. Add dimension with index to separate states with different actions available.
5: Perform cluster analysis on $\hat{\Omega}_t$ to determine ω representative states and corresponding trajectory groups (clusters).
6: Obtain estimated optimal action for each representative state j with **SpSa**($H_{\mathrm{a}} - t$, j, γ, G, $|C|$, H_{final}, N_{final}).
7: Generate reward and next state for each trajectory using the estimated optimal action from its trajectory group. Record set of next states $\hat{\Omega}_t$.
8: **end for**
9: For each trajectory, generate reward and next state for H_{final} final epochs.
10: **return** Trajectories.

Algorithm 6 Cluster Adaptation of EstimateQ – Lines 7–8

1: Using generative model G, generate $|\overline{C}|$ samples of next states $\overline{C}$ and calculate mean sampled cost $\hat{\rho}(i, a)$
2: Perform cluster analysis of nonabsorbing states in $\overline{C}$ to construct $|C|$ children C
3: For $c \in C$, denote the states in its cluster as $\overline{C}_c$
4: Estimate $\hat{Q}_d(i, a)$ as follows:

$$\hat{Q}_d(i,a) = \hat{\rho}(i,a) + \gamma \frac{|\overline{C}_c|}{|\overline{C}|} \mathbf{EstimateW}(d-1, j, \gamma, G, |C|, H_{\mathrm{final}}, N_{\mathrm{final}}, |\overline{C}|)$$

- **ZeroActionSimulator.** This class assigns a zero action, regardless of state. For example, the subclass for our experiments, EpiRelZeroActionSimulator, releases no one from the facility.
- **FixedActionSimulator.** This class takes as input a list of fixed actions to take at particular epochs, regardless of state. For any unspecified epoch, the zero action is taken.

Algorithm 7 Q Learning with Linear Function Approximation for Epidemiological Control.

Input: Total environmental actions K, initial state i_0, randomized behavioral policy $\hat{\pi}$, convergence criterion ϵ, step size ζ, generative model for the MDP G.

Output: Estimate of weight parameter θ.

Initialize $j := 1$, $k := 1$, $i_1 := i_0$. Generate randomized trajectory as follows.

while $j \leq K$ **do**

if i_k is absorbing state **then**

Define $a_k := a_0$, the null action. Define $i_{k+1} := i_0$ and $\rho(i_k, a_k) = 0$.

Increment k by 1.

else

Sample action a_k from $\hat{\pi}(i_k, \cdot)$. Sample reward $\rho(i_k, a_k)$ and next state i_{k+1} from G.

Increment j and k by 1.

end if

end while

Let $\bar{K}$ denote the total number of state-action pairs in the trajectory.

Initialize parameter $\theta = \theta_0$.

while $k \leq \bar{K}$ and and $\|\theta_{k+1} - \theta_k\| \geq \epsilon \|\theta_k\|$ **do**

Update θ with gradient ascent as follows (Chen et al. 2019)

if i_k is not absorbing **then**

$\theta_{k+1} = \theta_k + \zeta \phi(i_k, a_k) \left(\rho(i_k, a_k) + \gamma \max_a \phi(i_{k+1}, a)^T \theta_k - \phi(i_k, a_k)^T \theta_k\right)$

else

Set $\theta_{k+1} = \theta_k$.

end if

end while

Set $\theta = \theta_k$.

- **RandomActionSimulator.** This class chooses a random action from available actions. The class requires an ActionGenerator object to determine available actions. Options to limit the epochs when actions can be taken or to set a probability of no action taken are available.
- **QActionSimulator.** Given a function $\phi : \Omega \times \mathcal{A} \to \mathbb{R}^n$ and weights $\theta \in \mathbb{R}^n$, this class determines the action that maximizes $\phi(i, a)^T \theta$ for particular state i.

The benefit of this generic framework, available on Github, is that it allows practitioners to adapt the dynamic methods that we utilize to other problems that can be modeled with MDPs. For instance, we implement Algorithm 5, ST-SpSa, in R, taking these generic classes as input. By creating subclasses specific to new MDPs, the ST-SpSa can be easily utilized for new problems. In addition, MarkovTrajSimulator and RandomActionSimulator can be used to create random trajectories, to be utilized in the Q learning algorithm

with linear function approximation, Algorithm 7. When an optimal θ is determined for particular function ϕ, out-of-sample tests can implemented using MarkovTrajSimulator and QActionSimulator.

As an aside, because Algorithm 7 does not require online generation of trajectories via EpiModel, an R program, we implement this Q learning with linear function approximation algorithm in our preferred language, Python. While this code is more tailored to our problem, slight adjustments to the functions that read trajectories and define phi can be made to utilize the algorithm in other contexts.