EXPLORING THE PARETO FRONT OF MULTI-OBJECTIVE COVID-19 MITIGATION POLICIES USING REINFORCEMENT LEARNING

A PREPRINT

Mathieu Reymond
Vrije Universiteit Brussel
Brussels, Belgium
mathieu.reymond@vub.be

Conor F. Hayes
National University of Ireland Galway
Galway, Ireland

Lander Willem
University of Antwerp
Antwerp, Belgium

Roxana Rădulescu
Vrije Universiteit Brussel
Brussels, Belgium

Steven Abrams
Hasselt University
Hasselt, Belgium

Diederik M. Roijers
HU University of Applied Sciences Utrecht
Utrecht, the Netherlands

Enda Howley
National University of Ireland Galway
Galway, Ireland

Patrick Mannion
National University of Ireland Galway
Galway, Ireland

Niel Hens
Hasselt University
Hasselt, Belgium

Ann Nowé
Vrije Universiteit Brussel
Brussels, Belgium

Pieter Libin
Vrije Universiteit Brussel
Brussels, Belgium

ABSTRACT

Infectious disease outbreaks can have a disruptive impact on public health and societal processes. As decision making in the context of epidemic mitigation is hard, reinforcement learning provides a methodology to automatically learn prevention strategies in combination with complex epidemic models. Current research focuses on optimizing policies with respect to a single objective, such as the pathogen's attack rate. However, as the mitigation of epidemics involves distinct, and possibly conflicting, criteria (i.a., prevalence, mortality, morbidity, cost), a multi-objective decision approach is warranted to learn balanced policies. To lift this decision-making process to real-world epidemic models, we apply deep multi-objective reinforcement learning and build upon a state-of-the-art algorithm, Pareto Conditioned Networks (PCN), to learn a set of solutions that approximates the Pareto front of the decision problem. We consider the first wave of the Belgian COVID-19 epidemic, which was mitigated by a lockdown, and study different deconfinement strategies, aiming to minimize both COVID-19 cases (i.e., infections and hospitalizations) and the societal burden that is induced by the applied mitigation measures. We contribute a multi-objective Markov decision process that encapsulates the stochastic compartment model that was used to inform policy makers during the COVID-19 epidemic. As these social mitigation measures are implemented in a continuous action space that modulates the contact matrix of the age-structured epidemic model, we extend PCN to this setting. We evaluate the solution set that PCN returns, and observe that it correctly learns to reduce the social burden whenever the hospitalization rates are sufficiently low. In this work, we thus demonstrate that multi-objective reinforcement learning is attainable in complex epidemiological models and provides essential insights to balance complex mitigation policies.

Keywords Multi-objective reinforcement learning · Epidemic control · COVID-19 epidemic models

1 Introduction

Infectious disease outbreaks represent a major global challenge TODO: ref. To this end, understanding the complex dynamics that underlie these epidemics is essential. Epidemiological transmission models allow us to capture and understand such dynamics and facilitate the study of prevention strategies through simulation. However, developing efficient mitigation strategies remains a challenging process, given the non-linear and complex nature of epidemics.

To address these challenges, reinforcement learning provides a methodology to automatically learn mitigation strategies in combination with complex epidemic models [Libin et al., 2021a]. Previous research focused on optimising policies with respect to a single objective, such as the pathogen's attack rate, while the mitigation of epidemics is a problem that inherently covers distinct and possibly conflicting criteria (i.a., prevalence, mortality, morbidity, cost). Therefore, optimizing on a single objective requires that these distinct criteria are somehow aggregated into a single metric. Typically, a decision maker will rely on an expert's assistance to manually design such a metric. However, this is a challenging process, as the decision maker is generally not aware of all possible trade-offs, and as such this may result in a metric that exhibits sub-optimal performance. Furthermore, manually designing such metrics is time consuming, costly and error-prone, as this non-intuitive process requires repetitive and tedious tuning to achieve the desired behaviour [Roijers et al., 2013]. Moreover, taking a single objective approach has several other limiting factors. For example, it reduces the explainability of the learned solution, as we cannot compare the learned behavior with alternatives. Moreover, the preferences of the decision maker might change over time, in which case the optimization process needs to be done again [Hayes et al., 2021].

This challenging process can be circumvented by taking an explicit multi-objective approach that aims to learn the different trade-offs regarding the considered criteria. By assuming that a decision maker will always prefer solutions for which at least one objective improves, it is possible to learn a set of optimal solutions referred to as the *Pareto front*. This enables decision makers to review each solution on the Pareto front before making a decision, thereby being aware of the trade-offs that each solution implies.

In this work, we investigate the use of *multi-objective reinforcement learning* (MORL) to learn a set of solutions that approximate the Pareto front of multi-objective epidemic mitigation strategies. We consider the first wave of the Belgian COVID-19 epidemic, which was mitigated by a strict lockdown [Willem et al., 2021]. When the incidence of confirmed cases was steadily decreasing, epidemiological experts were tasked to investigate deconfinement strategies, to reduce the severe contact and mobility reductions. Here, we consider an epidemiological model that was constructed to describe the Belgian COVID-19 epidemic, and was fitted to hospitalisation incidence data and serial sero-prevalence data Abrams et al. [2021]. This model constitutes a stochastic discrete-time age-structured compartmental model that simulates mitigation strategies by varying social distancing parameters concerning school, work and leisure contacts. Based on this model, we contribute a novel mutli-objective epidemiological reinforcement learning environment (MOBelCov), in the form of a multi-objective Markov decision process (MOMDP) [Roijers et al., 2013]. MOBelCov encapsulates the epidemiological model developed by Abrams et al. [2021] to implement state transitions, with an action space that combines a proportional reduction of school, work and leisure contacts at each time step, Furthermore, it defines a reward function based on two objectives: the attack rate (i.e., proportion of the population affected by the pathogen) and the social burden that is induced by the mitigation measures.

To learn and explore the trade-offs between the attack rate and social burden we present a state-of-the-art MORL approach based on Pareto Conditioned Networks (PCN) [Reymond et al., 2022]. PCN uses a single neural network to learn the policies that belong to the Pareto front. As PCN is an algorithm designed for discrete action-spaces, we extend it towards continuous action-spaces to accommodate MOBelCov's action-space. With this continuous action variant of PCN, we explore the Pareto front of multi-objective COVID-19 mitigation policies.

By evaluating the solution set of mitigation policies learned by PCN, we observe that PCN minimises the social burden in scenarios where hospitalization rates are sufficiently low TODO: update (challenges). Therefore, in this work we illustrate that multi-objective reinforcement learning can be utilised to provide important insights surrounding the trade-offs between complex mitigation polices in real-world epidemiological models.

2 Background

In the following, we use bold notation $\mathbf{V}$ to denote a vectorial value, while non-bold notation V for its scalar counterpart.

2.1 Multi-Objective Reinforcement Learning

Real-world decisions problems typically consider multiple and possibly conflicting objectives. Multi-objective reinforcement learning (MORL) can be used to find optimal solutions for sequential decision making problems with multiple objectives [Hayes et al., 2021]. Multi-objective sequential problems are typically modelled as a multi-objective Markov decision process (MOMDP), i.e., a tuple, $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, \boldsymbol{\mathcal{R}} \rangle$, where $\mathcal{S}$, $\mathcal{A}$ denote the state and action spaces respectively, $\mathcal{T}\colon \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ denotes a probabilistic transition function, γ is a discount factor determining the importance of future rewards and $\boldsymbol{\mathcal{R}}\colon \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^n$ is an n-dimensional vector-valued immediate reward function, where n corresponds to the number of objectives.

While MOMDPs define the actions an agent can take, it does not define the agent's behavior, i.e., which actions are taken in each state. Given this MOMDP, an agent follows a policy π, that expresses the probability to take action $a \in \mathcal{A}$ when in state $s \in \mathcal{S}$: $\mathcal{S} \times \mathcal{A} \to [0, 1]$. We measure the performance of π through the expected sum of discounted rewards (denoted the *Value*) it achieves from the initial state s_0 until the end of the decision problem:

$$\mathbf{V}^{\pi} = \mathbb{E}\left[\sum_{t=0}^{h} \gamma^t \mathbf{r}_t | \, \pi, s_0\right], \tag{1}$$

where $\mathbf{r}_t$ corresponds to the multi-objective reward observed at time t, by following policy π, given by the reward function $\mathcal{R}(s_t, a_t, s_{t+1})$, where s_t is the current state, a_t corresponds to the action that was taken, and s_{t+1} signifies the state reached by taking action a_t.

For single-objective RL, where $n = 1$, the goal is to find the policy π^* that maximizes the V-value:

$$\pi^* = \arg\max_{\pi} V^{\pi}. \tag{2}$$

In contrast, for MORL, $n > 1$ which leads to vectorial returns. In this case, there can be policies for which, without any additional information, it is impossible to know if one is better than the other. For example, it is impossible to decide which policy between π_1, π_2 is optimal if both policies lead to expected returns $\mathbf{V}^{\pi_1} = (0, 1), \mathbf{V}^{\pi_2} = (1, 0)$ respectively. We call these solutions *non-dominated*, i.e., solutions for which it is impossible to improve an objective without hampering another. The set that contains all the non-dominated solutions of the decision problem is called the *Pareto front* $\mathcal{F}$. Our goal is to find the set of policies that lead to all the V-values contained in the Pareto front $\Pi^* = \{\ \pi | \mathbf{V}^{\pi} \in \mathcal{F}\}$. In general, we call any set of V-values a *solution set*. When a solution set contains only non-dominated V-values, it is referred to as a *coverage set*. In the case that no π exists that has a $\mathbf{V}^{\pi}$ dominating any of the solutions in a coverage set, then this coverage set is the Pareto front.

2.2 Multi-Objective Metrics

Comparing the learned coverage sets produced by different algorithms is a non-trivial task, as one algorithm's output might dominate the other in some region of the objective-space, but be dominated in another. Intuitively, one would generally prefer the algorithm that covers a wider range of decision maker preferences. In this work we use several metrics to evaluate our algorithm's performance.

A widely used metric in the literature is called the *hypervolume* [Zitzler et al., 2003]. This metric evaluates the learned coverage set by computing its volume with respect to a fixed reference point. The reference point is taken as a lower bound on the achievable returns so that the volumes are always positive. By definition, the solutions contained in the Pareto front dominate all other possible solutions. Thus, no other solution can further increase the volume under the Pareto front. This means that the hypervolume is the highest for the Pareto front. One drawback of the hypervolume is that it can be difficult to interpret. For example, when working in high-dimensional objective-spaces, adding or removing a single point can drastically change hypervolume values, especially if the point lies close to an extremum of said space.

To counterpoise these shortcomings, we evaluate our work using an additional metric called the ε-indicator I_ε [Zitzler et al., 2003], which measures how close a coverage set is to the Pareto front $\mathcal{F}$. Intuitively, I_ε shows that any solution of $\mathcal{F}$ is *at most* ε better with respect to each objective o than the closest solution of the evaluated coverage set:

$$I_\varepsilon = \inf_{\varepsilon \in \mathbb{R}} \{\forall\ \mathbf{V}^{\pi} \in \mathcal{F},\ \exists\ \mathbf{V}^{\pi'} \in \hat{\Pi} :\ ||\mathbf{V}^{\pi} - \mathbf{V}^{\pi'}||_\infty \leq \varepsilon\}, \tag{3}$$

where $||.||_\infty$, the L-infinity norm, returns the magnitude of the largest entry of a vector.

The main disadvantage of this metric is that we need the true Pareto front to compute it, which is unknown for our MOMDP (see Sec. 3). To still gain insights from our learned policies, we approximate the true Pareto front using the non-dominated policies across all runs.

We note that the ε-indicator metric is quite pessimistic, as it measures worst-case performance [Zintgraf et al., 2015], i.e., it will still report low performance as long as a single point of the Pareto front is incorrectly modeled, even if all

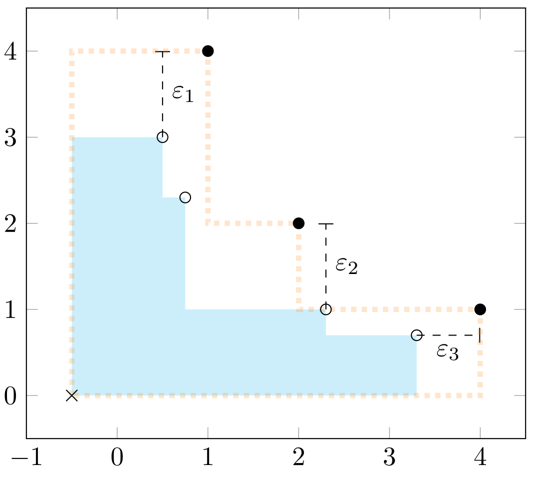

Figure 1: Example of a Pareto front (black dots) and a coverage set (white dots) in a 2-objective environment. The hypervolume metric (in light blue) measures the volume of all dominated solutions with respect to some reference point (cross). The reference point is taken as a lower bound on the achievable returns, which is why it can differ from the origin. The ε metrics first compute the maximum distance between each point in the Pareto front and its closest point in the coverage set (ε_i). We can then take their maximum value to compute the I_ε metric, or their mean value to obtain the $I_{\varepsilon-mean}$ metric of the coverage set.

the other points are covered. As such, we also use the $I_{\varepsilon-mean}$ [Reymond et al., 2022] which measures the *average* ε distance of the solutions in $\mathcal{F}$ with respect to the evaluated coverage set.

Fig. 1 shows a visual representation of the hypervolume and ε metrics in two dimensions.

3 COVID-19 model and MOBelCov MOMDP

3.1 Stochastic compartment model for SARS-CoV-2

To evaluate non-pharmaceutical interventions, we consider the compartmental model presented by Abrams et al., that was used to investigate exit strategies in Belgium after the first epidemic wave of SARS-CoV-2 [Abrams et al., 2021]. This model concerns a discrete-time stochastic model, that considers an age-structured population. The model generalises a standard SEIR model[1], extended to capture the different stages of disease spread and history that are associated with SARS-CoV-2 (i.e., pre-symptomatic, asymptomatic, symptomatic with mild symptoms and symptomatic with severe symptoms) and to represent the stages associated with severe disease, i.e., hospitalisation, admission to the intensive care unit (ICU) and death.

A visual representation of the model is depicted in Fig 2. It shows the different compartments, as well as the flow rates at which individuals move from one compartment to another.

These flow rates are defined by a set of ordinary differential equations, which are outlined as follows:

[1]A standard SEIR model divides the population into four different states, i.e., susceptible, exposed, infectious and recovered individuals.

$$
\begin{aligned}
\frac{d\mathbf{S}(t)}{dt} &= -\lambda(t)(\mathbf{S})(t),\\
\frac{d\mathbf{E}(t)}{dt} &= \lambda(t)(\mathbf{S})(t) - \gamma\mathbf{E}(t),\\
\frac{d\mathbf{I}^{presym}(t)}{dt} &= \gamma\mathbf{E}(t) - \theta\mathbf{I}^{presym}(t),\\
\frac{d\mathbf{I}^{asym}(t)}{dt} &= \theta p\mathbf{I}^{presym}(t) - \delta_1\mathbf{I}^{asym}(t),\\
\frac{d\mathbf{I}^{mild}(t)}{dt} &= \theta(1-p)\mathbf{I}^{presym}(t) - \{\psi + \omega_2\}\mathbf{I}^{mild}(t),\\
\frac{d\mathbf{I}^{sev}(t)}{dt} &= \psi\mathbf{I}^{mild}(t) - \omega\mathbf{I}^{sev}(t),\\
\frac{d\mathbf{I}^{hosp}(t)}{dt} &= \phi_1\omega\mathbf{I}^{sev}(t) - \{\delta_3 + \tau_1\}\mathbf{I}^{hosp}(t),\\
\frac{d\mathbf{I}^{icu}(t)}{dt} &= (1-\phi_1)\omega\mathbf{I}^{sev}(t) - \{\delta_4 + \tau_2\}\mathbf{I}^{icu}(t),\\
\frac{d\mathbf{D}(t)}{dt} &= \tau_1\mathbf{I}^{hosp}(t) - \tau_2\mathbf{I}^{icu}(t),\\
\frac{d\mathbf{R}(t)}{dt} &= \delta_1\mathbf{I}^{asym}(t) + \delta_2\mathbf{I}^{mild}(t) + \delta_3\mathbf{I}^{hosp}(t) + \delta_4\mathbf{I}^{icu}(t)
\end{aligned}
$$

where, for example, $\mathbf{S} = (S_1(t), S_2(t), ..., S_k(t))^T$ is the vector representing the susceptible members of the population of each age group k at time t.

In this set of ordinary differential equations, each state variable represents a vector over all age groups for a particular compartment at time t. Infection dynamics are governed by an age-specific force of infection λ:

$$\lambda(k,t) = \sum_{k'=1}^{K} \beta(k,k')I_{k'}(t), \tag{4}$$

where k is the respective age group of a total of K age groups, and $\beta(k,k')$ is the time-invariant transmission rate that encodes the average per capita rate at which an infectious individual in age group k makes an effective contact with a susceptible individual in age group k', per unit of time.

As we consider an age-structured population, we consider this extended SEIR structure for $K = 10$ age groups, i.e., $[0-10), [10-20), [20-30), [30-40), [40-50), [50-60), [60-70), [70-80), [80-90), [90, 100+)$. Contacts of the different age-groups, which impact the propagation rate of the epidemic, are modeled using social contact matrices. We define a social contact matrix for 6 different social environments: $C_{\text{home}}, C_{\text{work}}, C_{\text{transport}}, C_{\text{school}}, C_{\text{leisure}}, C_{\text{other}}$, for the home, work, transport, school, leisure, other environments respectively. The social contact matrix across all social environments is defined as:

$$C = C_{\text{home}} + C_{\text{work}} + C_{\text{transport}} + C_{\text{school}} + C_{\text{leisure}} + C_{\text{other}} \tag{5}$$

Under the social contact hypothesis [Wallinga et al., 2006], we have that:

$$\beta(k,k') = q \cdot C(k,k'), \tag{6}$$

where q is a proportionality factor.

Following Abrams et al. [2021], we rely on distinct social contact matrices for symptomatic and asymptomatic individuals, respectively C_s and C_a. Therefore, it is possible to define the transmission rates for both symptomatic and asymptomatic individuals as follows:

$$\boldsymbol{\beta}_s(k,k) = q_s \cdot C_s(k,k'), \tag{7}$$

and

$$\boldsymbol{\beta}_a(k,k') = q_a \cdot C_a(k,k'). \tag{8}$$

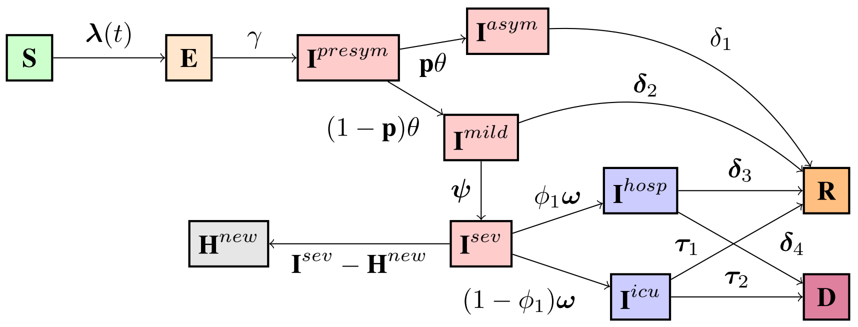

Figure 2: Schematic diagram of the compartmental model for SARS-CoV-2 presented by Abrams et al. [2021] which is used to derive the MOMDP.

Moreover, the age-dependent force of infection can be defined as follows:

$$\boldsymbol{\lambda}(t) = \boldsymbol{\beta}_a \times \{\mathbf{I}^{presym}(t) + \mathbf{I}^{asym}(t)\} + \boldsymbol{\beta}_s \times \{\mathbf{I}^{mild}(t) + \mathbf{I}^{sev}(t)\}, \tag{9}$$

where $\boldsymbol{\lambda}(t) = (\lambda(1,t), \lambda(1,t), ..., \lambda(K,t))$. For all further information about the different compartments and parameters please refer to the work of Abrams et al. [2021].

By formulating the set of differential equations defined above, as a chain-binomial, we can obtain stochastic trajectories from this model [Bailey, 1975]. A chain-binomial model assumes a stochastic model where infected individuals are generated by some underlying probability distribution. For the stochastic model we consider a time interval $(t, t+h]$, where h is defined as the length between two consecutive time points. Similar to Abrams et al. [2021], in this work we set $h = \frac{1}{24}$.

TODO: epi friends, please check Time- and age-dependent behavioral changes introduce substantial uncertainty regarding the outcome of an outbreak. To evaluate intervention strategies that modulate such behavioral changes, the use of stochastic epidemiological models is warranted [Abrams et al., 2021]. Moreover, the effect of stochasticity is most pronounced when studying the initial growth of an epidemic [Britton and Lindenstrand, 2009], be it during its emergence, or when implementing deconfinement strategies.

TODO: This is why we use the binomial model in our analysis.

3.2 Interventions strategies

To model different types of interventions, we follow the contact reduction scheme presented by Abrams et al. [2021]. Firstly, to consider distinct exit scenarios, we modulate the social contact matrices to reflect a contact reduction in a particular age group.

We consider a contact reduction function that imposes a proportional reduction of work (including transport) p_w, school p_s and leisure p_l contacts, which is implemented as a linear combination of social contact matrices:

$$\hat{C}(p_w, p_s, p_l) = C_{\text{home}} + p_w(C_{\text{work}} + C_{\text{transport}}) + p_s C_{\text{school}} + p_l(C_{\text{leisure}} + C_{\text{other}}) \tag{10}$$

We denote $\hat{C}_t$ the social contact matrix at timestep t, resulting from the reduction function $\hat{C}$. Secondly, we assume that compliance to the interventions is gradual and model this using a logistic compliance function (see details in Sec. A.2).

3.3 The MOBelCov Environment

In order to apply multi-objective reinforcement learning, we construct the MOBelCov MOMDP based on the epidemiological model introduced in Sec. 3.1 and graphically depicted in Fig. 2. Moreover, we consider a finite-horizon setting where we simulate the compartment model for a fixed number of weeks.

Action-space: Our actions concern the installment of a social contact matrix with a particular reduction resulting from the reduction function $\hat{C}$ (see Sec. 3.2). To this end, we use the proportional reduction parameters p_w, p_s, p_l defined in Sec. 3.2. Thus, each $\mathbf{a} \in \mathcal{A}$ is a 3-dimensional continuous vector in $[0,1]^3$ (i.e., $\mathbf{a} = [p_w, p_s, p_l]$) which impacts the social contact matrix according to Eq. 10.

Transition function: The model defined by Abrams et al. [2021] utilises a model transition probability $M(\mathbf{s}'_m \mid \mathbf{s}_m, \hat{C}, p_w, p_s, p_l)$ (see Sec. A for details on M), where $\hat{C}(p_w, p_s, p_l)$ returns the currently installed social contact matrix, that progresses the epidemiological model in one timestep. We use this function as the transition function in MOBelCov.

In a classical MDP, executing an action a_t in any state s_t leads to a next state s_{t+1} according to the transition function $\mathcal{T}$. At every timestep t, the agent is free to choose the action to perform. In our case, this potentially results in a different restriction $[p_w, p_s, p_l]$ every week. However, we argue that in the context of mitigation policies, consistency is important and policies that impose changes too frequently will be hard to adhere to.

In order to learn realistic and consistent mitigation policies, we introduce a *budget* regarding the number of times a policy can change its actions until the terminal state of the MOMDP is reached. Concretely, when the action changes, i.e., if the social restriction proposed by the policy is different from the one that is currently in place, we reduce the budget by one. We only allow action changes as long as there is budget left. We note that we can mimic a no-limit budget setting by choosing a budget that corresponds to the horizon of the environment.

Finally, for each timestep t, our transition function $\mathcal{T}$ uses the model transition probability M to simulate the model for one week, using $\hat{C}_t$ obtained from $\mathbf{a}_t$.

State-space: The state of the MOMDP concerns a 3-tuple. The first element, $\mathbf{s}_m$, directly corresponds to the aggregation of the state variables in the epidemiological model, i.e., a tuple,

$$\langle S_k, E_k, I_k^{presym}, I_k^{asym}, I_k^{mild}, I_k^{sev}, I_k^{hosp}, I_k^{icu}, H_k^{new}, D_k, R_k \rangle, \tag{11}$$

for each age group $k \in \{1, \ldots, K\}$, where S encodes the members of the population who are susceptible to infection and E encodes the members of the population who have been exposed to COVID-19. Moreover, I^{presym}, I^{asym}, I^{mild}, I^{hosp}, I^{icu} denote the members of the population infected with COVID-19 and are, respectively, presymptomatic, asymptomatic, mildly symptomatic, hospitalised, or in the ICU. Finally, H_k^{new} represents the number of newly hospitalised individuals in age group k.

We parameterize the epidemiological model using the mean of the posteriors as specified by Abrams et al. [2021] (details in Sec. A.1).

The second element of the tuple consists of the social contact matrix $\hat{C}_t$ that is currently in place. The reason to incorporate it in the state-space is two-fold. First, Abrams et al. [2021] define a compliance function, simulating the time people need to get used to the new rules set in place. As such, during the simulated week, there is a gradual shift from the current $\hat{C}_t$ to the new social contact matrix, $\hat{C}_{t+1}$. As such, we need to maintain the current $\hat{C}_t$, to maintain a Markovian environment.

Secondly, we require the current $\hat{C}_t$ to determine whether the action changes the social restrictions in place, and thus consume part of the budget.

The third element of the tuple concerns the budget $\boldsymbol{b}$. We incorporate a distinct budget per action-dimension, so p_w, p_s and p_l each have their own budget, resulting in a vector $\boldsymbol{b} = [b_w, b_s, b_l]$. As such, it is possible that, at timestep t, the budget for one of the proportional reductions is reduced but not the others.

Therefore, we define a state in MOBelCov as follows:

$$\mathbf{s} = \mathbf{s}_m \cup \hat{C} \cup b \tag{12}$$

Reward function: We define a vectorial reward function which considers multiple objectives: attack rate (i.e., infections and hospitalisations) and the social burden imposed by the interventions on the population.

The attack rate in terms of infections is defined as the difference in susceptibles from the current state to the next state [Libin et al., 2021a]. Since this is a cost that needs to be minimized, we defined the corresponding reward function as the negative attack rate:

$$\mathcal{R}_{\text{ARI}}(\mathbf{s}, \mathbf{a}, \mathbf{s}') = -(\sum_{k=1}^{K} S_k(\mathbf{s}) - \sum_{k=1}^{K} S_k(\mathbf{s}')). \tag{13}$$

The reward function to reduce the attack rate in terms of hospitalisations is defined as the negative number of new hospitalizations:

$$\mathcal{R}_{\text{ARH}}(\mathbf{s}, \mathbf{a}, \mathbf{s}') = -\sum_{k=1}^{K} H_k^{\text{new}}(\mathbf{s}). \tag{14}$$

Finally, we use the missed contacts resulting from the intervention measures as a proxy for societal burden. To quantify missed contacts, we consider the original social contact matrix C and the installed social contact matrix $\hat{C}$ to compute the difference $\hat{C} - C$. The resulting difference matrix quantifies the average frequency of contacts missed. To determine missed contacts for the entire population, we apply the difference matrix to the population sizes of the respective age groups that are currently uninfected (i.e., susceptible and recovered individuals). Formally, we define the social burden reward function $\mathcal{R}_{\text{SB}}$, as follows:

$$\mathcal{R}_{\text{SB}}(\mathbf{s}, \mathbf{a}, \mathbf{s}') = \sum_{i=1}^{K}\sum_{j=1}^{K}(\hat{C} - C)_{ij} S_i(\mathbf{s}) S_j(\mathbf{s}) + \sum_{i=1}^{K}\sum_{j=1}^{K}(\hat{C} - C)_{ij} R_i(\mathbf{s}) R_j(\mathbf{s}), \tag{15}$$

where $S_k(\mathbf{s})$ represents the number of susceptible individuals in age group k in state $\mathbf{s}$ and R_k represents the number of recovered individuals in age group k in state $\mathbf{s}$. In Sec. 5, we optimize PCN on two different variants for the multi-objective reward function: $[\mathcal{R}_{\text{ARH}}, \mathcal{R}_{\text{SB}}]$ and $[\mathcal{R}_{\text{ARI}}, \mathcal{R}_{\text{SB}}]$, to study the impact of these distinct attack rate quantities.

4 Pareto Conditioned Networks

In multi-objective optimization, the set of optimal policies can grow exponentially in the number of objectives. Thus, recovering them all is a computationally expensive process and requires an exhaustive exploration of the complete state space. To address this problem, we use Pareto Conditioned Networks (PCN), a method that uses a single neural network to encompass all non-dominated policies [Reymond et al., 2022]. The key idea behind PCN is to use supervised learning techniques to improve the policy instead of resorting to temporal-difference learning.

One major challenge of using neural networks as function approximators in RL is that the target (e.g., the optimal action of the policy) is not known in advance, as opposed to classical supervised learning where the ground-truth target is provided. As the behavior of the agent improves over time, the action used as target can change, often leading to hard-to-tune and brittle learners [Mnih et al., 2015, Fu et al., 2019].

Instead of trying to continuously improve the policy by learning actions that should lead to the highest cumulative reward, PCN flips the problem, by learning actions that should lead to any *desired* cumulative reward (be it high or low). In this way, all past trajectories can be reused for supervision, since their returns are known, as well as the actions needed to reach said returns. We can thus train a policy that, conditioned on a desired return, provides the optimal action to reach said return. By leveraging the generalization properties of neural networks, we can accumulate incrementally better experience by conditioning on increasingly higher reward-goals

Instead of using a neural network that takes a state $\mathbf{s}$ as input, PCN uses a single neural network that takes a tuple $\langle \mathbf{s}, \hat{h}, \hat{\mathbf{R}} \rangle$ as input. $\hat{\mathbf{R}}$ represents the *desired return* of the decision maker, i.e. the return PCN should reach at the end of the episode. $\hat{h}$ denotes the *desired horizon* that expresses the number of timesteps that should be executed before reaching $\hat{\mathbf{R}}$. At execution time, both $\hat{h}$ and $\hat{\mathbf{R}}$ are chosen by the decision maker at the start of the episode. Then, at every timestep, the desired horizon is updated according to the perceived reward $\mathbf{r}_t$, $\hat{\mathbf{R}} \leftarrow \hat{\mathbf{R}} - \mathbf{r}_t$ and the desired horizon is decreased by one, $\hat{h} \leftarrow \hat{h} - 1$.

PCN's neural network has an output for each action $a_i \in \mathcal{A}$. Each output represents the confidence the network has that, by taking the corresponding action in $\mathbf{s}$, $\hat{\mathbf{R}}$ will be reached in $\hat{h}$ timesteps. We can draw an analogy with a classification problem where the network should learn to classify $\langle \mathbf{s}, \hat{h}, \hat{\mathbf{R}} \rangle$ to its corresponding label a_i.

Similar to classification, PCN requires a labeled dataset with training examples to learn a mapping from input to label. However, contrary to classification, the data present in the dataset is not fixed. PCN collects data from the trajectories experienced while exploring the environment. Thus, the dataset improves over time, as we collect increasingly better trajectories.

Concretely, the PCN algorithm implements an iterative process that repeatedly executes 3 main steps:

1. Use the current policy to interact with the environment and generate trajectories (see Sec. 4.2). Initially, use a random policy to fill the dataset with random trajectories.
2. Update the dataset to incorporate these new trajectories and prune the least interesting trajectories in terms of returns (see Sec. 4.1).
3. Update the policy by training it on the updated dataset (see Sec. 4.3).

4.1 Building and updating the dataset

Given a trajectory composed of T transitions $\langle s_0, a_0, \boldsymbol{r}_0, s_1\rangle, \ldots, \langle s_{T-1}, a_{T-1}, \boldsymbol{r}_{T-1}, s_T\rangle$ we can compute, for each transition at timestep t, the future discounted return $\boldsymbol{R}_t = \sum_{i=t}^{T} \gamma^i \boldsymbol{r}_i$ and the leftover horizon $h_t = T - t$. Since for this trajectory executing action a_t in state s_t resulted in return $\mathbf{R}_t$ in h_t timesteps, we add a datapoint with input $\langle s, \hat{h}, \hat{\mathbf{R}}\rangle = \langle s_t, h_t, \mathbf{R}_t\rangle$ and output $a = a_t$ to the dataset. In other words, when the observed return corresponds to the desired return in that state, then a_t is the optimal action to take.

Since we have a fix-sized dataset, adding this trajectory means we need to remove another one. Our aim is to keep trajectories that span different parts of the objective space, such that we can explore as many types of trade-offs as possible. However, to improve the trade-offs we currently have in our dataset, we need to only keep solutions that are close to our current estimate of the Pareto front. Thus, we assign a priority for each trajectory in our dataset that combines two metrics: the *crowding distance*, which measures the distance of a solution with its closest neighbors, and the l_2-*norm* with the closest non-dominated solution, which indicates how close the solution is from our current estimate of the Pareto front.

The dataset is then updated by only keeping the trajectories with the top-N priorities, where N is the size of the dataset.

4.2 Generating trajectories with the current policy

In the previous section, we explain how PCN updates the dataset given a trajectory. In this section, we explain how PCN produces said trajectory.

It is unrealistic to expect PCN to reliably produce trajectories with high-valued desired returns when it has only been trained on datapoints originating from random trajectories. Rather, we expect PCN to produce trajectories with returns in the range of the ones from the current training data. Therefore, if we obtain trajectories reaching high returns, PCN will be able to confidently return high-return policies.

PCN leverages the fact that, due to the generalization capabilities of neural networks, the policies obtained from the network will still be reliable even if the desired return is marginally higher than what is present in the training data. In fact, they will perform similar actions to those in the training data, but lead to a higher return. Thus, we incrementally condition the network on increasingly higher returns, to obtain trajectories that extend the boundaries of PCN's current coverage set.

More precisely, we randomly select a non-dominated return $\mathbf{R}_{nd}$ and its corresponding horizon $\hat{h}$ from the dataset. By randomly picking a non-dominated return from the entire coverage set we ensure equal chance of improvement to each part of the objective space. To generate trajectories that improve upon $\mathbf{R}_{nd}$, we randomly select an objective o and increase $R_{nd,o}$ by a sample from the uniform distribution $\mathcal{U}(0, \sigma_o)$, where σ_o is the standard deviation for the selected objective, using all non-dominated returns from the trajectories in the dataset. This then becomes our desired return $\hat{\boldsymbol{R}}$. By restricting the improvement to at most σ_o, $\hat{\mathbf{R}}$ stays in the range of possible achievable returns and, by only modifying one objective at a time, the changes to the network's input compared to the training data are kept at a minimum.

4.3 Training the network for continuous actions

PCN trains the network as a classification problem, where each class represents a different action. Transitions $x = \langle \mathbf{s}_t, h_t, \mathbf{R}_t\rangle, y = a_t$ are sampled from the dataset, and the ground-truth output y is compared with the predicted output $\hat{y} = \pi(\mathbf{s}_t, h_t, \mathbf{R}_t)$. The predictor (i.e., the policy) is then updated using the cross-entropy loss function [Shore and Johnson, 1980]:

$$H = -\sum_{a \in \mathcal{A}} y_a \log \pi(a|\mathbf{s}_t, h_t, \mathbf{R}_t) \tag{16}$$

where $y_a = 1$ if $a = a_t$ and $y_a = 0$ otherwise.

While the original PCN algorithm is designed for MOMDPs with discrete actions-spaces, the problem we tackle (see Sec. 3) is defined in terms of a continuous action-space. We thus extend PCN to the continuous action-space setting. We change the output of the neural network such that there is a single output value for each dimension of the action-space. Since the actions should be bound in the domain of possible actions ($[0, 1]$ in the case of MOBelCov, see Sec. 3), we apply a tanh non-linearity function on each output. As such, we have a regression problem instead of a classification problem, as the labeled dataset now uses continuous labels $y = \mathbf{a}_t$ instead of categories. We thus use a Mean Squared Error (MSE) loss to update the policy:

$$MSE = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} (\hat{y}_a - y_a)^2 \tag{17}$$

Since learning the full set of Pareto-efficient policies Π^* requires that the policies $\pi^* \in \Pi^*$ are deterministic stationary policies [Roijers et al., 2013], we use the output $\hat{\mathbf{y}}$ as action at execution time. However, PCN improves its policy through exploration, by continuously updating its dataset with better trajectories. Thus, at training time, we use a stochastic policy by adding random noise to the action [Lillicrap et al., 2015]:

$$\mathbf{a}_t = \pi(\mathbf{s}_t, h_t, \mathbf{R}_t) + \eta s \text{ with } s \sim \mathcal{N}, \tag{18}$$

where $\mathcal{N}$ is the standard Normal distribution and η is a hyper-parameter defining the magnitude of noise to be added.

4.4 Coping with stochastic transitions

PCN trains its policy on a dataset that is collected by executing trajectories. It assumes that reenacting a transition from the dataset leads to the same episodic return. When the transition function $\mathcal{T}$ of the MOMDP is deterministic, the whole trajectory can be faithfully reenacted, which guarantees the same return. Combined with the fact that PCN's policy is deterministic at execution time, conditioning the policy on a target episodic return is equivalent to conditioning it on the V-value $\mathbf{V}$.

However, when $\mathcal{T}$ is stochastic this can no longer be guaranteed. To mitigate this, we add small random noise to $\mathbb{R}_t$ when performing gradient descent, which reduces the risk of overfitting [Zur et al., 2009]. Moreover, while the MOBelCov model is stochastic, the variation is entirely due to the sampling of the binomial distributions in the binomial-chain. While this variation accumulates over time, the time window we consider for each timestep (i.e., one week) is short enough that the accumulation stays limited. Thus, the possible next-states resulting from a state-action pair are similar to each other. This allows PCN to compensate if $\mathbf{r}_t = \boldsymbol{\mathcal{R}}(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$ is worse than expected.

Although we use a stochastic model to cope with the uncertainty of the outcome of the outbreak, it is possible to deterministically evaluate the set of ordinary differential equations that define the model. We assess the validity of our approach by executing PCN on this deterministic variant, and observing similar performance as with the MOBelCov model. We show the results in Sec. B.1.

5 Analysing COVID-19 deconfinement policies

Our goal is to use PCN to learn deconfinment strategies in the MOBelCov environment. We aim to learn policies that balance between the epidemiological objective of minimising the attack rate (i.e., $\mathcal{R}_{\text{ARH}}$ for hospitalisation and $\mathcal{R}_{\text{ARI}}$ for infection) and the social burden (i.e., $\mathcal{R}_{\text{SB}}$) experienced by the population population due to the implement mitigation measures. To this end, we consider two cases for the vectorial reward functions $[\mathcal{R}_{\text{ARH}}, \mathcal{R}_{\text{SB}}]$ and $[\mathcal{R}_{\text{ARI}}, \mathcal{R}_{\text{SB}}]$, to learn and analyse policies under different targets with respect to the considered attack rate.

To conduct this analysis, we apply our extension of PCN for continuous action-spaces on the MOBelCov model. As explained in Sec. 4.4, we extend PCN for environments with stochastic transitions.

Conform to Abrams et al. [2021], the simulation starts on the 1st of March 2020, by seeding a number of infections in the population. Two weeks later, on the 14th of March, the Belgian government initiated a full lockdown. This is implemented by fixing the actions p_w, p_s, p_l to $0.2, 0, 0.1$ respectively. This lockdown ended on the 4th of May 2020, at which point the government decided on a multi-phase exit strategy to incrementally reduce teleworking, reopen schools and allow leisure activities, such as the re-opening of bars and the cultural sector. It is from this day onward that PCN aims to learn policies for diverse exit strategies, compromising between the total number of daily hospitalizations and the total number of contacts lost as a proxy for social burden. The simulation lasts throughout the school holidays (from 01/07/2020 to 31/08/2020). Schools are closed during the school holidays, which is simulated by setting $p_s = 0$, regardless of the corresponding value outputted by the policy, i.e., during periods of school closure p_s is ignored.

We draw the analogy with the multi-phase exit strategy established by the Belgian government and the restriction on the number of action-changes imposed by the MOBelCov's budget $\boldsymbol{b}$. Indeed, on the 11th of May 2020, exactly one week after the end of the lockdown, stores and certain types of jobs were allowed to reopen, under strict conditions. This corresponds to altering p_w in our MOMDP. One week later, on 18th of May, primary and secondary schools reopened for limited sized class-groups, and the cultural sector reopened partially. This is equivalent to increasing p_s and p_l. Further changes of restrictions occured on the 8th of June, 1st, 9th and 25th of July. Thus, we argue that, with a limited budget, we can achieve realistic and implementable policies. In our experiments, we consider budgets of 2 to 5, as these

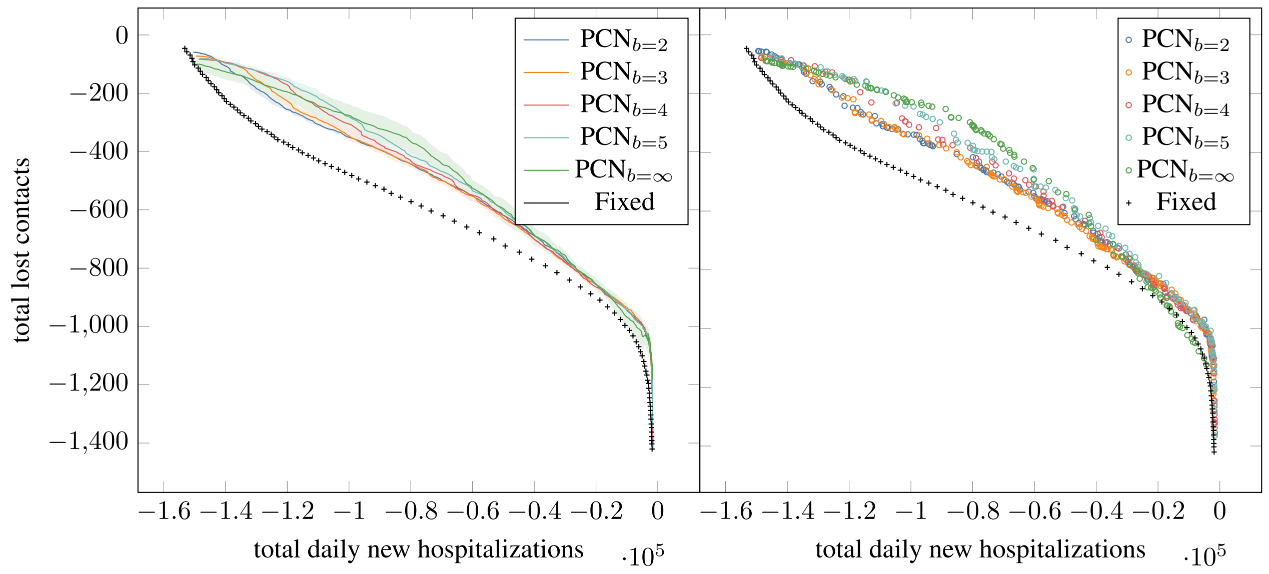

Figure 3: The Pareto front of policies discovered by PCN using MOBelCov, showing the different compromises between the number of hospitalizations and the number of lost contacts. On the right, we show, for each budget setting (colored, subscript indicates budget) the coverage set learned by the best performing run. On the left, we show an interpolated average of the coverage sets learned by the different runs, with the shaded regions corresponding to the standard deviation. For comparison, the basline is displayed on both plots (in black). As the budget increases, so does the size of the coverage set learnt by PCN. Changes are most noticeable in the less restrictive trade-offs in terms of social burden.

closely relate to the number of changes that occured until the end of the summer holidays of 2020. As an upper bound, we also consider a no-limit budget setting.

To evaluate the quality of the policies learned by PCN, we compare PCN to a baseline. This baseline consists of a set of 100 fixed policies, that iterate over all the possible social restriction levels, with values ranging between 0 and 1. Concretely, each policy uses a fixed proportional reduction $p_w = p_l = p_s = u, u \sim \mathcal{U}(0, 1)$ throughout the episode. In other words, the fixed policies directly operate in a fine-grained manner on the whole contact reduction function $\hat{C}$. This allows us to obtain a strong baseline for potential exit strategies over the objective space. We note that while such fixed policies are a feasible approach, they do not scale well in terms of action and objective spaces and they will not be able to provide an adaptive restriction level, which is what we aim to provide using PCN.

All experiments are averaged over 10 runs. The hyper-parameters and the neural network architecture can be found in Sec.s B.4 and Sec. B.5, respectively.

5.1 Learned coverage set

We learn a coverage set (see Fig. 3) that ranges from imposing minimal restrictions to enforcing many restrictions. In Fig. 3, we display on the right the coverage set of the best-performing run in terms of hypervolume, for each budget setting. On the left, we show an interpolated average of the coverage sets learned by the different runs.

Regardless of the imposed budget, we notice that the coverage sets discovered by PCN almost completely dominate the coverage set of the baseline, demonstrating that there are better alternatives to the fixed policies. This is most evident in the compromising policies, where one has to carefully choose when to remove social restrictions while at the same time minimizing the impact on daily new hospitalizations. In these scenarios, PCN learns policies that drastically reduce the total number of new hospitalizations (e.g., more than 20000) for the same social burden. We analyse the executions of such policies in Fig. 4 (middle plot), that shows a flattened hospitalization curve, with a gradual increase of social freedom during the school holidays such that the curve of the epidemic is flattened and gradually decreases over time.

Interestingly, we notice that the most restrictive policy (i.e. the one that prioritizes hospitalizations over social burden, see Fig. 4, bottom plot) still starts to gradually increase p_w and p_l from the end of July onward. This is because by then, the epidemic has mostly faded out, and it is safe to reduce social restrictions. The timing of this reduction is important

	$[\mathcal{R}_{\text{ARH}}, \mathcal{R}_{\text{SB}}]$			$[\mathcal{R}_{\text{ARI}}, \mathcal{R}_{\text{SB}}]$		
	Hypervolume	I_ε	$I_{\varepsilon-mean}$	Hypervolume	I_ε	$I_{\varepsilon-mean}$
PCN$_{b=2}$	158.370 ± 0.811	0.080 ± 0.011	0.033 ± 0.002	157.152 ± 1.023	0.087 ± 0.006	0.035 ± 0.002
PCN$_{b=3}$	158.721 ± 1.439	0.080 ± 0.012	0.032 ± 0.003	158.002 ± 2.081	0.084 ± 0.009	0.034 ± 0.005
PCN$_{b=4}$	160.642 ± 1.582	0.075 ± 0.007	0.028 ± 0.003	159.315 ± 2.601	0.088 ± 0.018	0.031 ± 0.006
PCN$_{b=5}$	163.104 ± 2.386	0.070 ± 0.023	0.022 ± 0.005	161.792 ± 2.464	0.075 ± 0.015	0.026 ± 0.005
Fixed	140.479 ± 0.000	0.139 ± 0.000	0.073 ± 0.000	140.479 ± 0.000	0.139 ± 0.000	0.073 ± 0.000
PCN	159.462 ± 7.713	0.264 ± 0.115	0.036 ± 0.020	159.852 ± 2.395	0.171 ± 0.093	0.032 ± 0.006

Table 1: Evaluation metrics for the coverage sets comparing hospitalizations with social burden. In general, an increase of budget results in a better coverage set. Training on infections (ARI) still provides a competitive coverage set in terms of hospitalizations. All PCN coverage sets outperform the baseline.

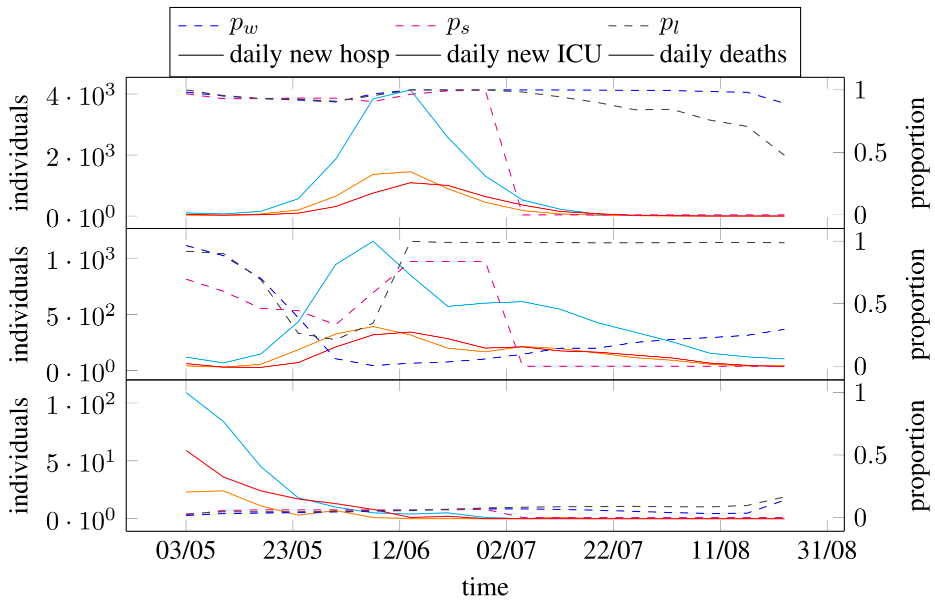

Figure 4: Selection of policies learned by PCN$_{b=5}$, from most restrictive in terms of social burden (top) to least restrictive (bottom). The x-axis represents the time, starting from the end of the lockdown, on the 4th of May, until the end of the school holidays, on the 1st of September. Since the lockdown is simulated before the start of the exit strategy, the start-state differs for each episode (i.e., the hospital already contains infected individuals). The left y-axis represents the number of individuals affected by the epidemic. The full-lined plots represent the number of individuals admitted into the hospital, ICU and deceased between the last timestep and the current one. The plot showing the newly deceased individuals closely relates to the ICU admissions. The right y-axis represents the proportional reduction in effect, with 1 meaning a business-as-usual policy, and 0 meaning a complete suppression of social contacts. The dotted-lined plots represent the proportional reductions for the work, leasure and school environments. We note that the school reduction automatically goes to 0 at the start of the school holidays.

as reducing restrictions too soon can lead to a new wave. PCN learns the impact of its decisions over time, and correctly infers the timing at which restrictions can be safely lifted.

Finally, the top plot shows that, without imposing social restrictions, the number of hospitalisations peaks on the 15th of June. By the beginning of July, the epidemic has spread out over the majority of the population TODO: herd immunity?, and the number of admissions at the hospital has been reduced to a fraction of the number of hospitalisations at the peak. Thus, without social restrictions, we do not take advantage of the natural decrease of social contacts due to school holidays, as the majority of the population has already been infected before the start of the holidays. TODO: mention the slow decrease of p_l during the summer? This is simply due to PCN not having found better

5.2 Impact of budgets on the coverage set

Fig. 3 demonstrates that the budget impacts the learnt coverage set. In general, an increase of budget is associated with an increasingly better coverage set, as policies learned using a higher budget dominate the ones learned with a lower budget. This is to be expected, as a higher budget give the agent more freedom to change its actions as the epidemic progresses. However, we also notice that this increase of coverage sets is not necessarily gradual.

Moreover, we observe that the difference is concentrated around the less restrictive policies in terms of social burden. We hypothesise that this region contains the most complex policies, as these try to maintain as much social freedom as possible, while containing the number of hospitalisations. In these cases, the timing of the actions greatly correlate with the timing and duration of the peak of the epidemic, and a higher budget allows for more fine-grained control about managing this timing. To confirm this, we select, for each budget setting, the solution where the difference in performance is most noticeable, corresponding to the solutions with a total number of hospitalisations around 80000 (which is in the middle of the range of possible hospitalisations, as can be seen in Fig. 3). We plot the execution of the corresponding policies in Fig. 5 and analyse their impact in terms of social burden. First, we observe that the lower-budget policies are unable to reduce the social restrictions past the peak of the epidemic. In contrast, the setting with no budget restrictions finely controls the restrictions as the epidemic progresses, completely removing restrictions by the end of the wave. Second, we note that the policy with a budget of 5 ressembles the execution of the one without restrictions. However, due to its budget, the policy is unable to progressively reduce the restrictions and instead resorts to a halfway compromise. Compared to this specific region of the coverage set, the difference in performance between different budget settings seems marginal around the extremas. At the extremas, the policies are less complex (e.g., business-as-usual, resulting in the same action executed throughout the episode) and are thus less impacted by the budget restrictions.

Finally, we observe that, while the extremas deliver similar trade-offs for any of the chosen budgets, these trade-offs differ for the setting without budget restrictions. Indeed, in this setting, PCN does not learn the most extreme policies with respect to restrictions, even though there are no constraints on the action-set. As explained in Sec. 4.2, PCN searches for increasingly better solutions using a stochastic policy. Thus, at every timestep, the action can change compared to the previous one. Continuously outputting the same action (e.g., no social restrictions) becomes a complicated task. In comparison, for the settings with a limited budget, the action stays the same as the previous one once the budget has been spent. As such, it is easier to learn the most extreme policies. Thus, in the specific case where we impose no budget, the freedom of action actually hinders PCN's search, for certain regions of the rweard-space.

5.3 R_{ARH} versus on R_{ARI}

Next, we assess the difference in coverage sets when optimizing on $\mathcal{R}_{\text{ARH}}$ versus $\mathcal{R}_{\text{ARI}}$. Although these reward functions have a different scale (there are more infected persons than hospitalised ones), our experiments show that infections and hospitalizations are correlated. This is expected, as during the initial phase of the epidemic, limited immunity was present in the population (i.e., limited natural immunity and no vaccines), which induces a tight coupling between infection and hospitalisation cases. This is confirmed in Table 1. In this table, we show the different performance metrics (hypervolume, I_ε, $I_{\varepsilon-mean}$) with respect to the objectives $[\mathcal{R}_{\text{ARH}}, \mathcal{R}_{\text{SB}}]$. The table is split in two parts. The left-side shows the different performance metrics, for PCN using $[\mathcal{R}_{\text{ARH}}, \mathcal{R}_{\text{SB}}]$ as optimization criterions. The right-side shows the same performance metrics, but with PCN using $[\mathcal{R}_{\text{ARI}}, \mathcal{R}_{\text{SB}}]$ as optimization criterions.

Even with the $\mathcal{R}_{\text{ARI}}$, the increased budget shows an increase in hypervolume in terms of hospitalisations. Moreover, those hypervolumes are close to the ones trained on $\mathcal{R}_{\text{ARH}}$, indicating that their coverage sets are similar. However, regardless of the imposed budget, the hypervolumes are slightly worse. This is to be expected, since those experiments are not directly optimising on $\mathcal{R}_{\text{ARH}}$. We make similar conclusions for I_ε. For budgets 2, 3 and 5, the difference between the worst-performing policy for the $\mathcal{R}_{\text{ARI}}$ variant and the $\mathcal{R}_{\text{ARH}}$ is less than 0.01, indicating less than 1% difference in return values between the two variants when comparing their worst-performing policy. As an exception, we notice that PCN without budget restrictions results in better performance across every metric for the $\mathcal{R}_{\text{ARI}}$ variant. Still, due to the high standard deviation of the no-budget, $\mathcal{R}_{\text{ARH}}$ setting, we do not believe this difference is meaningful. TODO: some kind of meaningful conclusion

5.4 Robustness of policy executions

The dataset of trajectories that PCN is trained on is pruned over time to keep only the most relevant trajectories. The returns of these trajectories are used in Fig. 3 to show the learned coverage set. Each of these returns can be used as desired return for policy execution. We now assess the robustness of the executed policies, by comparing the return obtained after executing the policy with the corresponding target return. For each run, we execute the policy 10 times

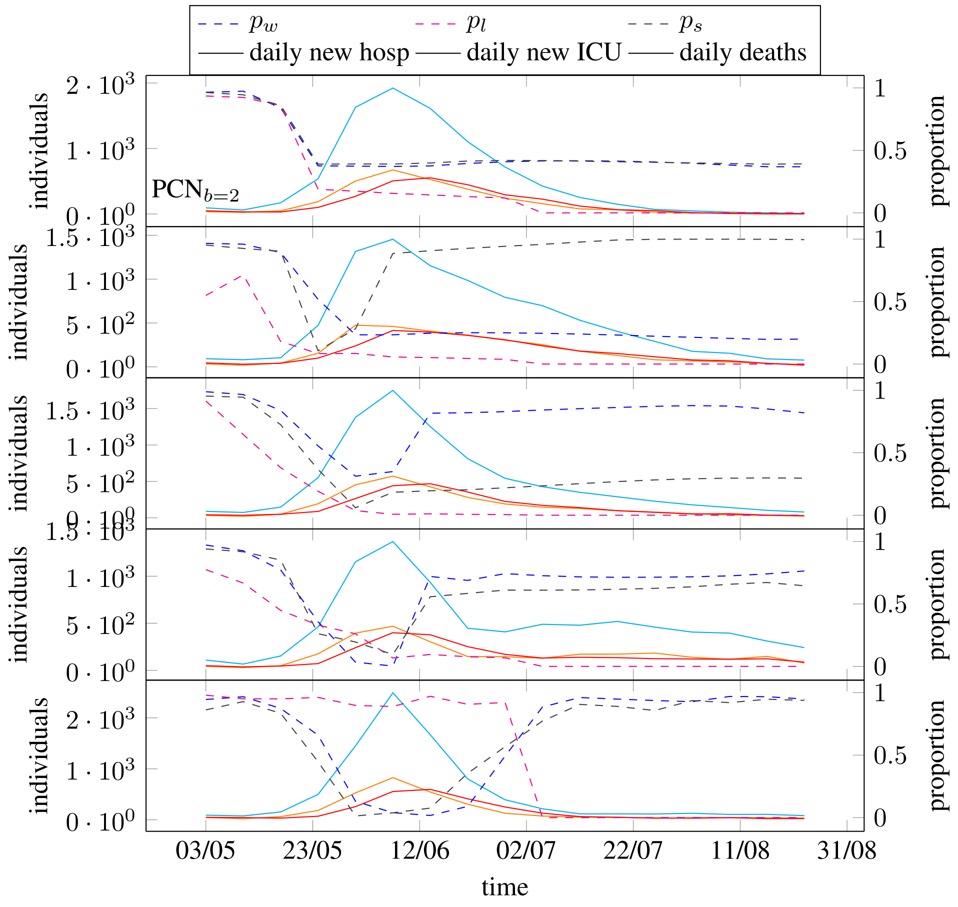

Figure 5: Execution of the policies attaining a number of hospitalisations around 80000, for different budgets. From top to bottom we display the policy executions with budget 2,3,4,5 and no-limit, respectively. We notice that the lower-budget policies are unable to reduce the social restrictions past the peak. The setting without budget restrictions finely controls the restrictions as the epidemic progresses, completely removing restrictions by the end of the wave. Finally, there is no consensus on which social environment to restrict most: certain policies provide similar restrictions for p_w and p_l, while others impose harsher restrictions on one social environment than the other.

	I_ε	$I_{\varepsilon-mean}$
PCN$_{b=2}$	0.047 ± 0.020	0.009 ± 0.004
PCN$_{b=3}$	0.048 ± 0.022	0.007 ± 0.002
PCN$_{b=4}$	0.064 ± 0.018	0.011 ± 0.003
PCN$_{b=5}$	0.058 ± 0.011	0.013 ± 0.003
PCN	0.035 ± 0.011	0.008 ± 0.004
Global average	0.050 ± 0.017	0.010 ± 0.004

Table 2: Comparing the difference in the desired return provided to PCN and the actual return PCN obtained when executing its policy. We see that, regardless of the setting, the learned policy faithfully receives a return similar to its desired return.

and compute the I_ε and $I_{\varepsilon-mean}$ metrics with respect to the coverage set learned during the run. We show that the executed policies reliably obtain returns that are similar to the desired return used to condition PCN.

Results are shown in Table 2. The I_ε indicators shows that, regardless of the budget, the decision maker will lose at worst a 0.050 normalized return in any of the objectives. On average, it will lose 0.010 normalized returns, i.e., on average, the return obtained by executing a policy will either result in an additional 1441 hospitalisations than expected, or result in 12 additional social contacts lost. Moreover, we emphasize that the learned coverage set contains the non-dominated returns encountered over the whole training procedure. Since the MOBelCov model is stochastic, for multiple executions of the same policy, the executions kept in the coverage sets are the ones for which the samples from the binomial-chain resulted in a better progression of the epidemic than average. Thus, we expect our policy-executions to be close to the target selected from the coverage set, but not on exactly target. Based on this analysis, we conclude that the policies trained by PCN are robust and produce returns as close as possible from their chosen target.

6 Related work

Reinforcement learning (RL) has been used in conjunction with epidemiological models to learn policies to limit the spread of diseases and predict the effects of possible mitigation strategies [Probert et al., 2019]. For example, RL has been used to model the mitigation of influenza [Das et al., 2008, Libin et al., 2021a, 2018].

RL and Deep RL have been used extensively as a decision making aid to reduce the spread of COVID-19. For example, to learn effective mitigation strategies [Ohi et al., 2020], to learn efficacy of lockdown and travel restrictions [Kwak et al., 2021] and to limit the influx of asymptomatic travellers [Bastani et al., 2021].

Multi-objective methods have also been deployed to learn optimal strategies to mitigate the spread of COVID-19. Wan et al. [Wan et al., 2021] implement a model-based multi-objective policy search method and demonstrate their method on COVID-19 data from China. Given that this method is model-based, a model of the transition function must be learned by sampling from the environment. The method proposed by Wan et al. [Wan et al., 2021] only considers a discrete action space which limits the application of their algorithm. Wan et al. [Wan et al., 2021] use linear weights to compute a set of Pareto optimal policies. However, methods which use linear weights can only learn policies on the convex-hull of the Pareto front [Vamplew et al., 2008], therefore the full Pareto front cannot be learned. We note that the method proposed by Kompella et al. [Kompella et al., 2020] considers multiple objectives. However, the objectives are combined using a weighted sum with hand-tuned weights which are determined by the authors. The weighted sum is applied by the reward function and a single objective RL method is used to learn a single optimal policy. In contrast to previous work, our approach makes no assumptions regarding the scalarisation function of the user and is able to discover Pareto fronts of arbitrary shape.

7 Conclusion and discussion

Making decisions on how to mitigate epidemics has important ethical implications with respect to public health and societal burden. In this regard, it is crucial to approach this decision making from a balanced perspective, to which end we argue that multi-objective decision making is crucial. In this work, we establish a novel approach, i.e., an expert system, to study multi-faceted policies, and this approach shows great potential with respect to future epidemic control. We are aware of the ethical implications that expert systems have on the decision process and we make the disclaimer that all results based on, or derived from, the expert system that we propose should be carefully interpreted by experts in the field of public health, and in a much broader context of economics, well-being and education. We note that the work in this manuscript was conducted by a interdisciplinary consortium that includes computer scientists and scientists with a background public health, epidemiology and bio-statistics.

In this work, we focus on the clinical outcomes of the intervention strategies and use the reduced contacts as proxy for the quality of life lost. This could be extended into more formal economic evaluations by assessing the health and monetary benefits and costs of different interventions for various stakeholders. The COVID-19 pandemic demonstrated the broad impact of infectious diseases on sectors outside health care. This stresses the need to capture a societal and thus multi-objective perspective in the decision making process on public health and health care interventions. Our learned policies confirm this, showing that focusing solely on reducing the number of hospitalizations results in taking drastic measures – more than a thousand social interactions lost per person over the span of 4 months – that may have a long-lasting impact on the population.

TODO: Although we use an age-structured compartment model, with social contact matrices to model social interactions, this remains a model that evaluates the progression of the pandemic as an aggregated process over the population. Individual-based models could provide more detailed and localized policies, potentially further improving the quality of possible trade-offs, and would provide an interesting avenue for future work. However, due to the computational cost of simulating such models, and the number of interactions required by reinforcement learning in general, this remains a challenging problem.

TODO: Moreover, although we are able to interpret and analyse the obtained policies and their corresponding trade-offs, as we can plot the Pareto front for 2 objectives, this approach cannot be used for problems with more objectives. Defining interpretation and visualization tools for many-objective problems is necessary if we want to include additional objectives for our problem.

TODO: Additionally, PCN is able to cope with model stochasticity, because the stochasticity is limited. In settings where the stochasticity is more pronounced, e.g., designing policies to control the initial outbreak of an epidemic, further methodological extensions are warranted.

TODO: Finally, while we have focused on social burden as a trade-off for hospitalisations, our approach can be used for related problems, involving different compromises, such as balancing the number of lost schooldays with respect to new infections in schools [Torneri et al., 2021], contact tracing effort versus its effect [Willem et al., 2021], the impact of antivirals on the epidemic versus resistance to antivirals [Torneri et al., 2020], or the cost of universal testing versus its impact on the epidemic [Libin et al., 2021b].

To conclude, we show that multi-objective reinforcement learning can be used to learn a wide set of high-quality policies on real-world problems, providing the decision maker with insightful and diverse alternatives, and showing the impact of taking extreme measures. TODO: This can be done for realistic settings, where we budget the number of changes of social reductions, such that the policies are practical and can be deployed in real-life scenarios.

References

Pieter J. K. Libin, Arno Moonens, Timothy Verstraeten, Fabian Perez-Sanjines, Niel Hens, Philippe Lemey, and Ann Nowé. Deep reinforcement learning for large-scale epidemic control. In Yuxiao Dong, Georgiana Ifrim, Dunja Mladenić, Craig Saunders, and Sofie Van Hoecke, editors, *ECML*, pages 155–170, Cham, 2021a. Springer International Publishing. ISBN 978-3-030-67670-4.

Diederik M Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. *JAIR*, 48:67–113, 2013.

Conor F. Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M. Zintgraf, Richard Dazeley, Fredrik Heintz, Enda Howley, Athirai A. Irissappane, Patrick Mannion, Ann Nowé, Gabriel Ramos, Marcello Restelli, Peter Vamplew, and Diederik M. Roijers. A practical guide to multi-objective rl and planning, 2021.

Lander Willem, Steven Abrams, Pieter JK Libin, Pietro Coletti, Elise Kuylen, Oana Petrof, Signe Møgelmose, James Wambua, Sereina A Herzog, Christel Faes, et al. The impact of contact tracing and household bubbles on deconfinement strategies for covid-19. *Nature communications*, 12(1):1–9, 2021.

Steven Abrams, James Wambua, Eva Santermans, Lander Willem, Elise Kuylen, Pietro Coletti, Pieter Libin, Christel Faes, Oana Petrof, Sereina A Herzog, et al. Modelling the early phase of the belgian covid-19 epidemic using a stochastic compartmental model and studying its implied future trajectories. *Epidemics*, 35:100449, 2021.

Mathieu Reymond, Bargiacchi Eugenio, and Ann Nowè. Pareto conditioned networks. In *Proceedings of the 21st International Conference on AAMAS (2022)*, 2022.

Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M Fonseca, and Viviane Grunert Da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on evolutionary computation*, 7 (2):117–132, 2003.

Luisa M Zintgraf, Timon V Kanters, Diederik M Roijers, Frans Oliehoek, and Philipp Beau. Quality assessment of morl algorithms: A utility-based approach. In *Benelearn 2015: proceedings of the 24th annual ML conference of Belgium and the Netherlands*, 2015.

Jacco Wallinga, Peter Teunis, and Mirjam Kretzschmar. Using data on social contacts to estimate age-specific transmission parameters for respiratory-spread infectious agents. *American journal of epidemiology*, 164(10): 936–944, 2006.

Norman TJ Bailey. The mathematical theory of infectious diseases and its applications. In *The mathematical theory of infectious diseases and its applications*, pages 413–413. Charles Griffin & Company Ltd 5a Crendon Street, High Wycombe, Bucks HP13 6LE., 1975.

Tom Britton and David Lindenstrand. Epidemic modelling: aspects where stochasticity matters. *Mathematical biosciences*, 222(2):109–116, 2009.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Justin Fu, Aviral Kumar, Matthew Soh, and Sergey Levine. Diagnosing bottlenecks in deep q-learning algorithms. In *International Conference on Machine Learning*, pages 2021–2030. PMLR, 2019.

John Shore and Rodney Johnson. Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. *IEEE Transactions on information theory*, 26(1):26–37, 1980.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Richard M Zur, Yulei Jiang, Lorenzo L Pesce, and Karen Drukker. Noise injection for training artificial neural networks: A comparison with weight decay and early stopping. *Medical physics*, 36(10):4810–4818, 2009.

William JM Probert, Sandya Lakkur, Christopher J Fonnesbeck, Katriona Shea, Michael C Runge, Michael J Tildesley, and Matthew J Ferrari. Context matters: using reinforcement learning to develop human-readable, state-dependent outbreak response policies. *Philosophical Transactions of the Royal Society B*, 374(1776):20180277, 2019.

Tapas K Das, Alex A Savachkin, and Yiliang Zhu. A large-scale simulation model of pandemic influenza outbreaks for development of dynamic mitigation strategies. *Iie Transactions*, 40(9):893–905, 2008.

Pieter JK Libin, Timothy Verstraeten, Diederik M Roijers, Jelena Grujic, Kristof Theys, Philippe Lemey, and Ann Nowé. Bayesian best-arm identification for selecting influenza mitigation strategies. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 456–471. Springer, Cham, 2018.

Abu Quwsar Ohi, MF Mridha, Muhammad Mostafa Monowar, Md Hamid, et al. Exploring optimal control of epidemic spread using rl. *Scientific reports*, 10(1):1–19, 2020.

Gloria Hyunjung Kwak, Lowell Ling, and Pan Hui. Deep reinforcement learning approaches for global public health strategies for covid-19 pandemic. *PLOS ONE*, 16(5):1–15, 05 2021.

Hamsa Bastani, Kimon Drakopoulos, Vishal Gupta, Ioannis Vlachogiannis, Christos Hadjicristodoulou, Pagona Lagiou, Gkikas Magiorkinis, Dimitrios Paraskevis, and Sotirios Tsiodras. Efficient and targeted covid-19 border testing via rl. *Nature*, 599(7883):108–113, 2021.

Runzhe Wan, Xinyu Zhang, and Rui Song. Multi-objective model-based reinforcement learning for infectious disease control. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1634–1644, 2021.

Peter Vamplew, John Yearwood, Richard Dazeley, and Adam Berry. On the limitations of scalarisation for multi-objective reinforcement learning of pareto fronts. In *Australasian joint conference on artificial intelligence*, pages 372–378. Springer, 2008.

Varun Kompella, Roberto Capobianco, Stacy Jong, Jonathan Browne, Spencer Fox, Lauren Meyers, Peter Wurman, and Peter Stone. Reinforcement learning for optimization of covid-19 mitigation policies. *arXiv preprint arXiv:2010.10560*, 2020.

Andrea Torneri, Lander Willem, Vittoria Colizza, Cecile Kremer, Christelle Meuris, Gilles Darcis, Niel Hens, and Pieter JK Libin. Controlling sars-cov-2 in schools using repetitive testing strategies (preprint). 2021.

Andrea Torneri, Pieter Libin, Joris Vanderlocht, Anne-Mieke Vandamme, Johan Neyts, and Niel Hens. A prospect on the use of antiviral drugs to control local outbreaks of covid-19. *BMC medicine*, 18(1):1–9, 2020.

Pieter JK Libin, Lander Willem, Timothy Verstraeten, Andrea Torneri, Joris Vanderlocht, and Niel Hens. Assessing the feasibility and effectiveness of household-pooled universal testing to control covid-19 epidemics. *PLoS computational biology*, 17(3):e1008688, 2021b.

A Stochastic Compartmental Model

In this work we utilise the compartmental model proposed by Abrams et al. [2021] and extend the model to a multi-objective Markov decision process (MOMDP). Fig. 2 shows a visual depiction of the compartment model. The flows of the deterministic model are defined by a set of ordienary differential equations, outlined in Sec. 3.1.

Intervening in the spread of the virus by, for example, reducing social contacts or government interventions introduces uncertainty in the further course of the outbreak. Therefore, to understand how this uncertainty affect the spread of the disease we introduce a stochastic component model which can model the uncertainty generated by interventions in social contacts.

By formulating the set of differential equations defined in Sec. 3.1, as a chain-binomial, we can obtain stochastic trajectories from this model [Bailey, 1975]. A chain-binomial model assumes a stochastic model where infected individuals are generated by some underlying probability distribution. For the stochastic model we consider a time interval $(t, t+h]$, where h is defined as the length between two consecutive time points. Similar to Abrams et al. [2021], in this work we set $h = \frac{1}{24}$. Abrams et al. [2021] define the set of differential equations as a chain binomial as follows:

$$\begin{aligned}
\mathbf{S}_{t+h}(k) &= \mathbf{S}_t(k) - \mathbf{E}_{new,t+h}(k),\\
\mathbf{E}_{t+h}(k) &= \mathbf{E}_t(k) + \mathbf{E}_{new,t+h}(k) - \mathbf{I}^{presym}_{new,t+h}(k),\\
\mathbf{I}^{presym}_{t+h}(k) &= \mathbf{I}^{presym}_t(k) + \mathbf{I}^{presym}_{new,t+h}(k) - \mathbf{I}^{asym}_{new,t+h}(k) - \mathbf{I}^{mild}_{new,t+h}(k),\\
\mathbf{I}^{asym}_{t+h}(k) &= \mathbf{I}^{asym}_t(k) + \mathbf{I}^{asym}_{new,t+h}(k) - \mathbf{R}^{asym}_{new,t+h}(k),\\
\mathbf{I}^{mild}_{t+h}(k) &= \mathbf{I}^{mild}_t(k) + \mathbf{I}^{mild}_{new,t+h}(k) - \mathbf{I}^{sev}_{new,t+h}(k) - \mathbf{R}^{mild}_{new,t+h}(k),\\
\mathbf{I}^{sev}_{t+h}(k) &= \mathbf{I}^{sev}_t(k) + \mathbf{I}^{sev}_{new,t+h}(k) - \mathbf{I}^{hosp}_{new,t+h}(k) - \mathbf{I}^{icu}_{new,t+h}(k),\\
\mathbf{I}^{hosp}_{t+h}(k) &= \mathbf{I}^{hosp}_t(k) + \mathbf{I}^{hosp}_{new,t+h}(k) - \mathbf{D}^{hosp}_{new,t+h}(k) - \mathbf{R}^{hosp}_{new,t+h}(k),\\
\mathbf{I}^{icu}_{t+h}(k) &= \mathbf{I}^{icu}_t(k) + \mathbf{I}^{icu}_{new,t+h}(k) - \mathbf{D}^{icu}_{new,t+h}(k) - \mathbf{R}^{icu}_{new,t+h}(k),\\
\mathbf{D}_{t+h}(k) &= \mathbf{D}_t(k) + \mathbf{D}^{hosp}_{new,t+h}(k) + \mathbf{D}^{icu}_{new,t+h}(k),\\
\mathbf{R}_{t+h}(k) &= \mathbf{R}_t(k) + \mathbf{R}^{asym}_{new,t+h}(k) + \mathbf{R}^{mild}_{new,t+h}(k) + \mathbf{R}^{hosp}_{new,t+h}(k) + \mathbf{R}^{icu}_{new,t+h}(k)
\end{aligned}$$

where,

$$\begin{aligned}
\mathbf{E}_{new,t+h} &\sim \textit{Binomial}\left(\mathbf{S}_t(k), p^*_t(k) = 1 - \{1 - p^*_t(k)\}^{\mathbf{I}_t}\right),\\
p^*_t(k) &= 1 - exp\left[-h\sum_{k'=1}^{K} \beta_{asym}(k,k')\{\mathbf{I}^{asym}_t(k')\} + \beta_{sym}(k,k')\{\mathbf{I}^{mild}_t(k') + \mathbf{I}^{sev}_t(k')\}\right],\\
\mathbf{I}^{presym}_{new,t+h}(k) &\sim Binomial\left(\mathbf{I}^{presym}_t(k), 1 - exp(-hp(k)\theta)\right),\\
\mathbf{I}^{mild}_{new,t+h}(k) &\sim Binomial\left(\mathbf{I}^{presym}_t(k), 1 - exp\left[-h\{1-p(k)\}\theta\right]\right),\\
\mathbf{I}^{sev}_{new,t+h}(k) &\sim Binomial\left(\mathbf{I}^{mild}_t(k), 1 - exp\{-h\psi(k)\}\right),\\
\mathbf{I}^{hosp}_{new,t+h}(k) &\sim Binomial\left(\mathbf{I}^{sev}_t(k), 1 - exp\{-h\phi_1(k)\omega(k)\}\right),\\
\mathbf{I}^{icu}_{new,t+h}(k) &\sim Binomial\left(\mathbf{I}^{sev}_t(k), 1 - exp\left[-h\{1-\phi_1(k)\}\omega(k)\right]\right),\\
\mathbf{D}^{hosp}_{new,t+h}(k) &\sim Binomial\left(\mathbf{I}^{hosp}_t(k), 1 - exp\{-h\tau_1(k)\}\right),\\
\mathbf{D}^{icu}_{new,t+h}(k) &\sim Binomial\left(\mathbf{I}^{icu}_t(k), 1 - exp\{-h\tau_2(k)\}\right),\\
\mathbf{R}^{asym}_{new,t+h}(k) &\sim Binomial\left(\mathbf{I}^{asym}_t(k), 1 - exp\left(-h\delta_2(k)\right)\right),\\
\mathbf{R}^{hosp}_{new,t+h}(k) &\sim Binomial\left(\mathbf{I}^{hosp}_t(k), 1 - exp\{-h\delta_3(k)\}\right),\\
\mathbf{R}^{icu}_{new,t+h}(k) &\sim Binomial\left(\mathbf{I}^{icu}_t(k), 1 - exp\{-h\delta_4(k)\}\right).
\end{aligned}$$

Given MOBelCov also calculates new hospitalisations, $\mathbf{H}^{new}$, we define $\mathbf{H}^{new}$ for the stochastic compartmental model as follows:

$$\mathbf{H}^{new}_{t+h}(k) = \mathbf{H}^{new}_t(k) + \mathbf{I}^{hosp}_{new,t+h}(k).$$

For more details on this model and the chain-binomial representation of the differential equations, we refer the reader to the work of Abrams et al. [2021].

To create a version of MOBelCov with a stochastic transition function, M, we utilise the stochastic compartmental model outlined above. Given the transitions within the compartmental model are derived by an underlying probability distribution it is possible to utilise the stochastic compartmental model transitions for MOBelCovÂs previously outlined in Sec. **??** the contact matrix $\hat{C}$ applied the model state s_m progresses the model and returns a new model state s'_m. Given the underlying model dynamics are governed in a probabilistic manner, the model returns s'_m stochastically. Therefore, it is possible to use this process as a stochastic transition function, M, for MOBelCov.

A.1 A Note on Model Parameters

The model is parameterised using the mean of the posteriors as reported by Abrams et al. [2021].

The population size for each of the considered age groups was taken from the Belgian statistical agency STATBEL[2]. To initialise the model, we used the number of confirmed cases until 13 March 2020 [Abrams et al., 2021], as reported by the Belgian agency for public health Sciensano[3].

A.2 Modelling interventions

In order to model different types of interventions, we follow Abrams et al. [2021]. Firstly, to consider distinct exit scenarios, we alter the social contact matrices to reflect a contact reduction in a particular age group. Secondly, we assume that compliance to the interventions is gradual and model this using a logistic compliance function. We use the logistic compliance function in function of time t,

$$c(t, t_I) = \frac{\exp(\beta_0^* + \beta_1^*(t - t_I))}{1 + \exp(\beta_0^* + \beta_1^*(t - t_I))}, \tag{A.1}$$

where t_I indicates the time the intervention started. We initialise β_1^* to the value estimated in by Abrams et al. and choose $\beta_0^* = -5$, as an intercept to have $c(t) = 0$ for $t = 0$, in correspondence with Fig. F2 in the Supplementary Information of Abrams et al. [2021].

B Additional results

B.1 Comparison of coverage sets learned on ODE and Binomial models

The coverage sets displayed in Fig. 3 correspond to PCN trained on the MOBelCov model, which is stochastic. To show that PCN copes with the stochasticity of this model, we compare these coverage sets with the ones learned on the ODE model, which is deterministic.

Results are shown in Fig. B.1. First, in a similar fashion as Fig. 3, we show the interpolated average coverage set for different budget setting, with PCN trained on the ODE model. We observe similar trends as for the Binomial model. Next, for each budget setting, we compare the interpolated average coverage set of the Binomial setting with the ODE setting. We observe similar coverage set, regardless of the budget setting, indicating that PCN is able to cope with stochastic transitions.

B.2 Comparison of coverage sets learned on $\mathbf{R}_{\text{ARH}}$ and $\mathbf{R}_{\text{ARI}}$

In Sec. 5.3, we show that we can learn all trade-offs between social burden and hospitalisations, while using the attack rate over infections as reward function.

Results are shown in Fig. B.2. We observe that the learned coverage sets are similar, regardless of the budget setting. Still, the coverage set of when trained on $\mathbf{R}_{\text{ARI}}$ is systematically dominated by the one when trained on $\mathbf{R}_{\text{ARH}}$. TODO: Please check: is this actually true? While hospitalisations and infections are highly correlated, they differ in terms of age-groups. Older age-groups are more susceptible to be hospitalised after being infected, but they do not form the majority of the population. For trade-offs where infections and social burden need to be balanced, the proportional reductions target different social environments than for trade-offs balancing hospitalisations and social burden. For example, the work environment is majorly comprised of individuals with a more robust immune system, reducing the social contact in this environment greatly affects the number of infections, but has a lesser impact on the number of hospitalisations.

B.3 Policy executions

Depending on the budget, PCN learns a coverage set containing more than 150 different policies. To gain a better insight about their behavior, and how they differ from each other, we plot executions of each learned policy in Fig.s B.3-B.10. The plots are displayed from the least restrictive policy in terms of social burden to the most restrictive one. Since the MOBelCov model is stochastic, we show 10 executions of the same policy, on each plot.

[2]https://statbel.fgov.be/nl/themas/bevolking/structuur-van-de-bevolking#figures

[3]https://epistat.wiv-isp.be/covid/

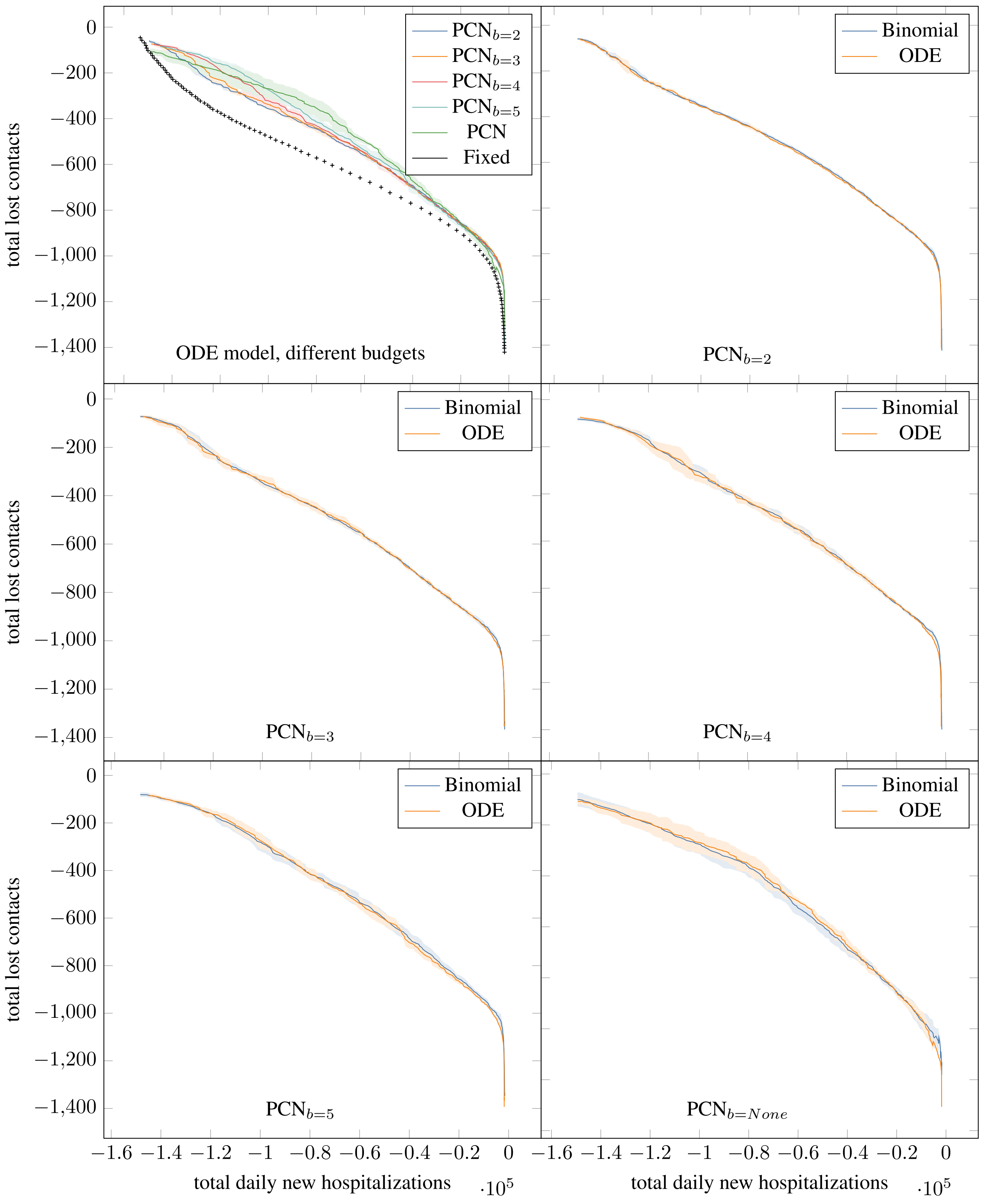

Figure B.1: Comparison of learned coverage sets when PCN interacts with the ODE (i.e., deterministic) variant of the compartment model. We observe that, regardless of the chosen budget, PCN learns a similar coverage set on both variants of the model, indicating that it is able to cope with the stochasticity present in the Binomial variant.

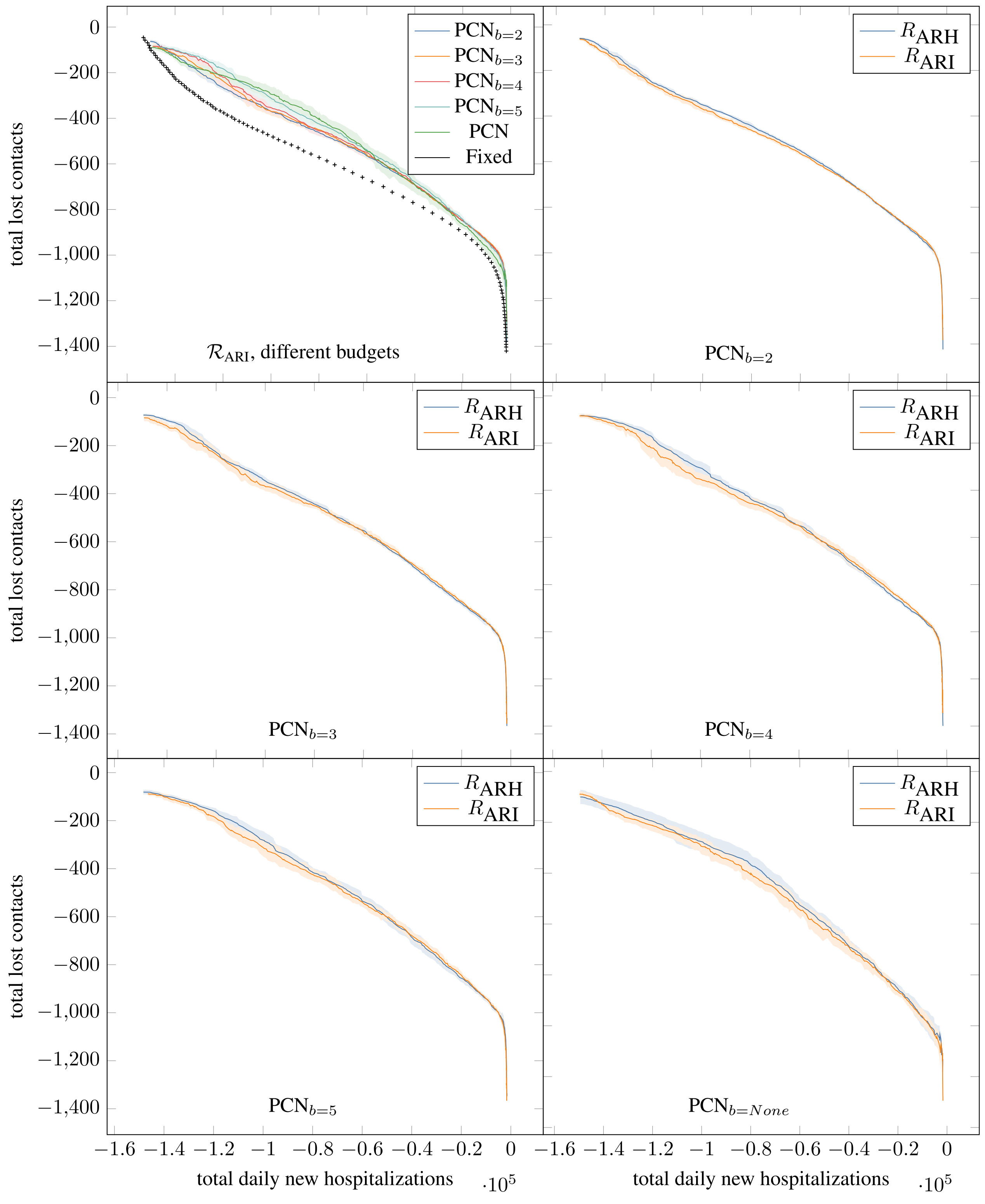

Figure B.2: Comparison of learned coverage sets when PCN learns using the attack rate of infections $\mathbf{R}_{\text{ARI}}$.

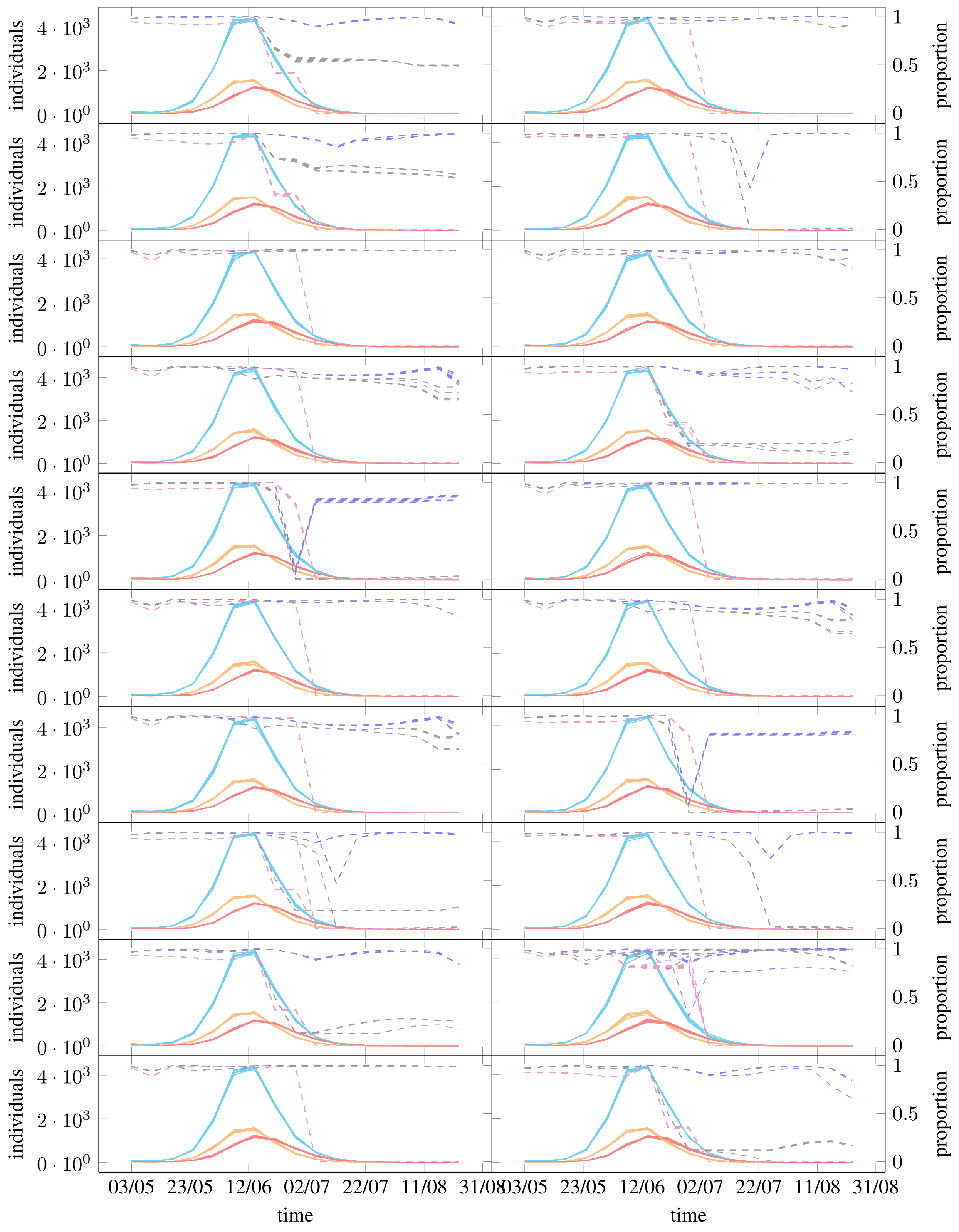

Figure B.3: Execution of policies 0 to 19, with a budget of 2.

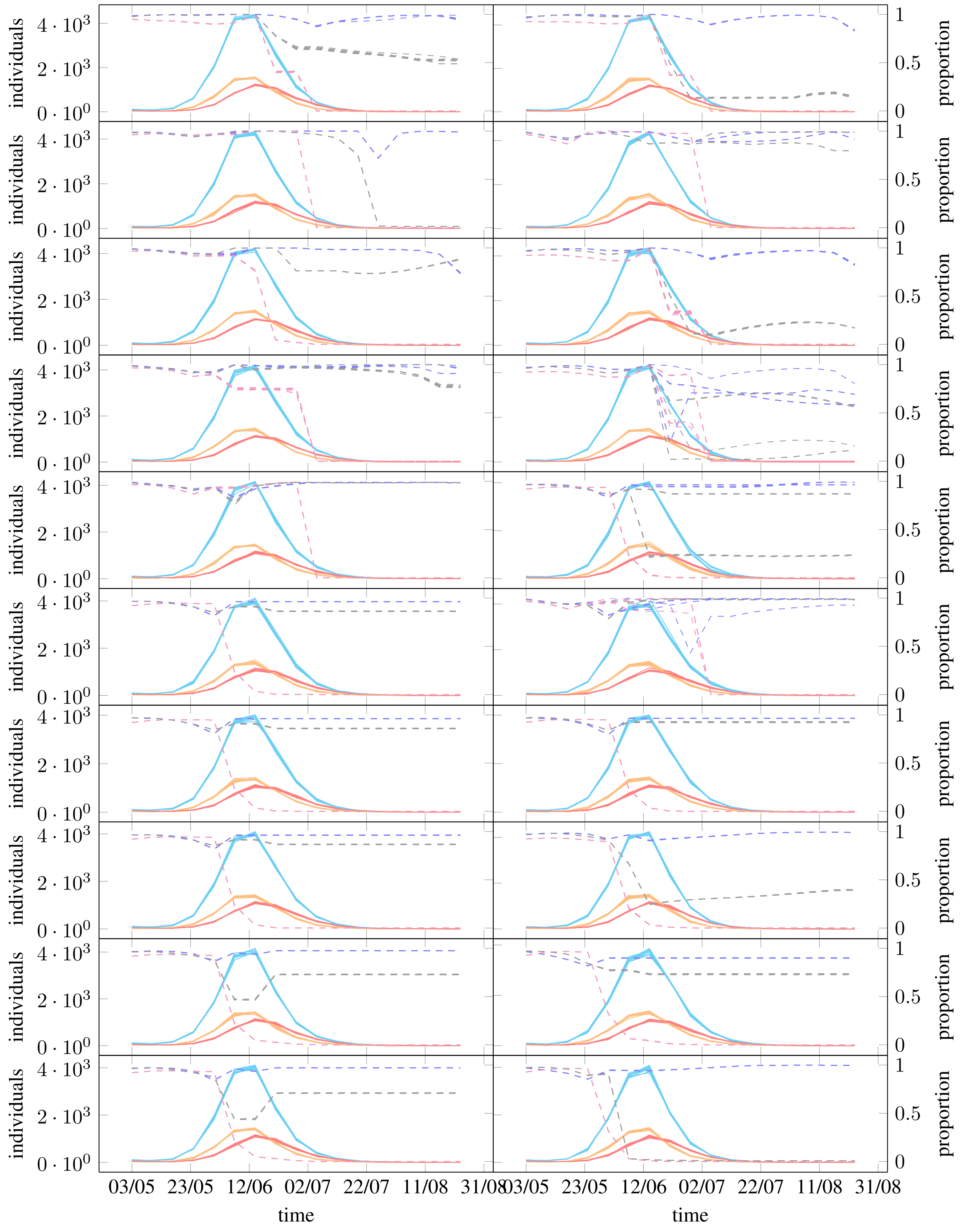

Figure B.4: Execution of policies 20 to 39, with a budget of 2.

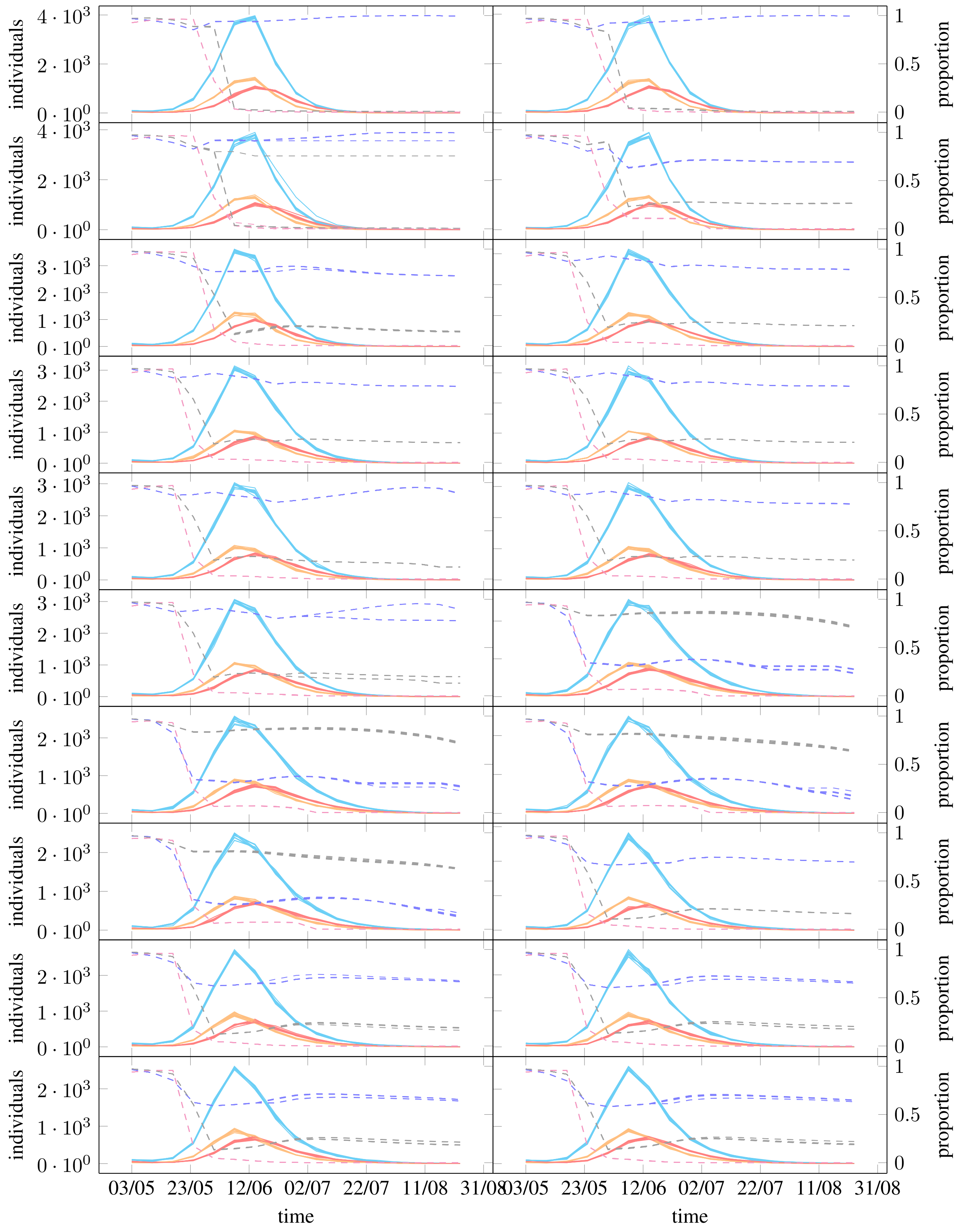

Figure B.5: Execution of policies 40 to 59, with a budget of 2.

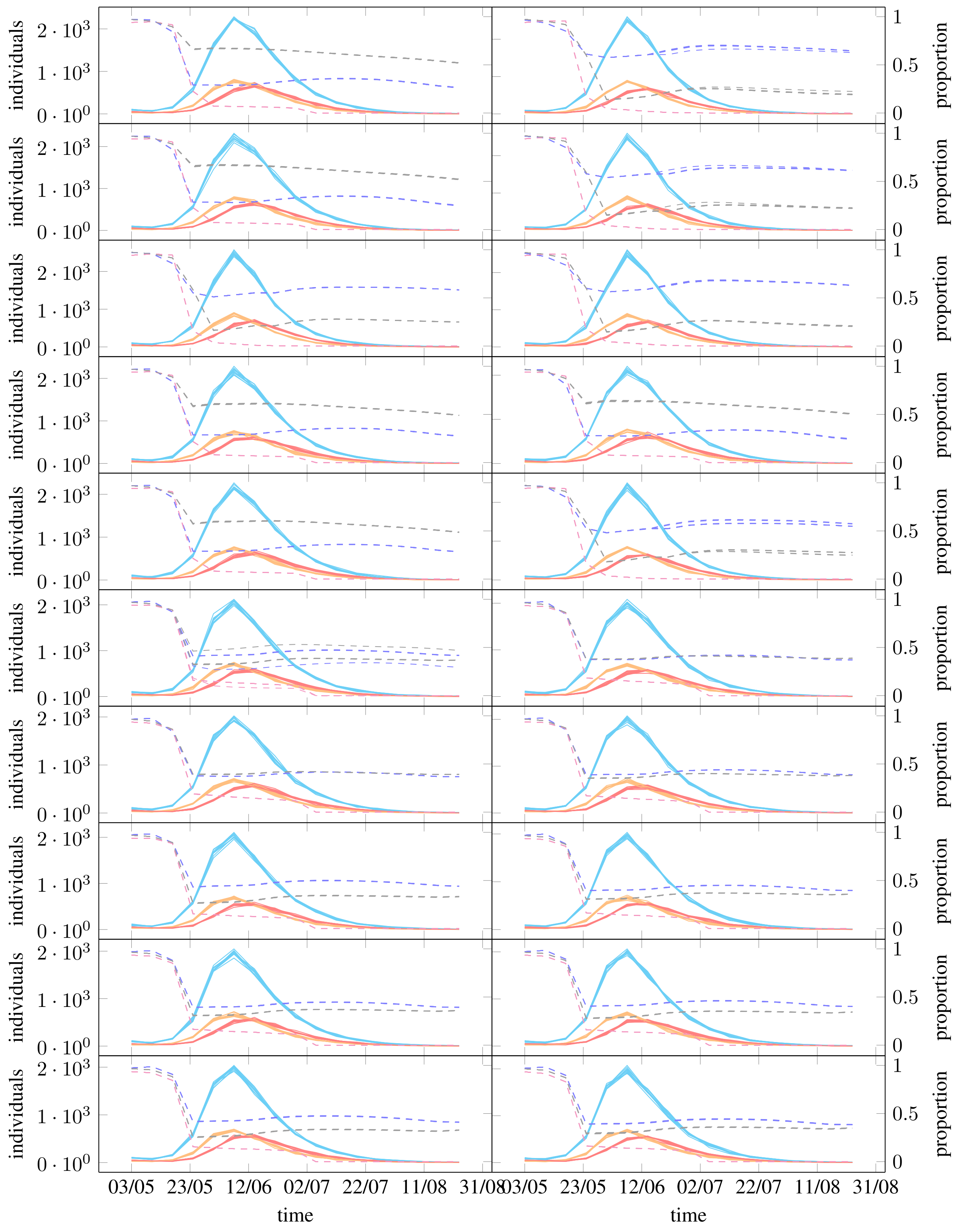

Figure B.6: Execution of policies 60 to 79, with a budget of 2.

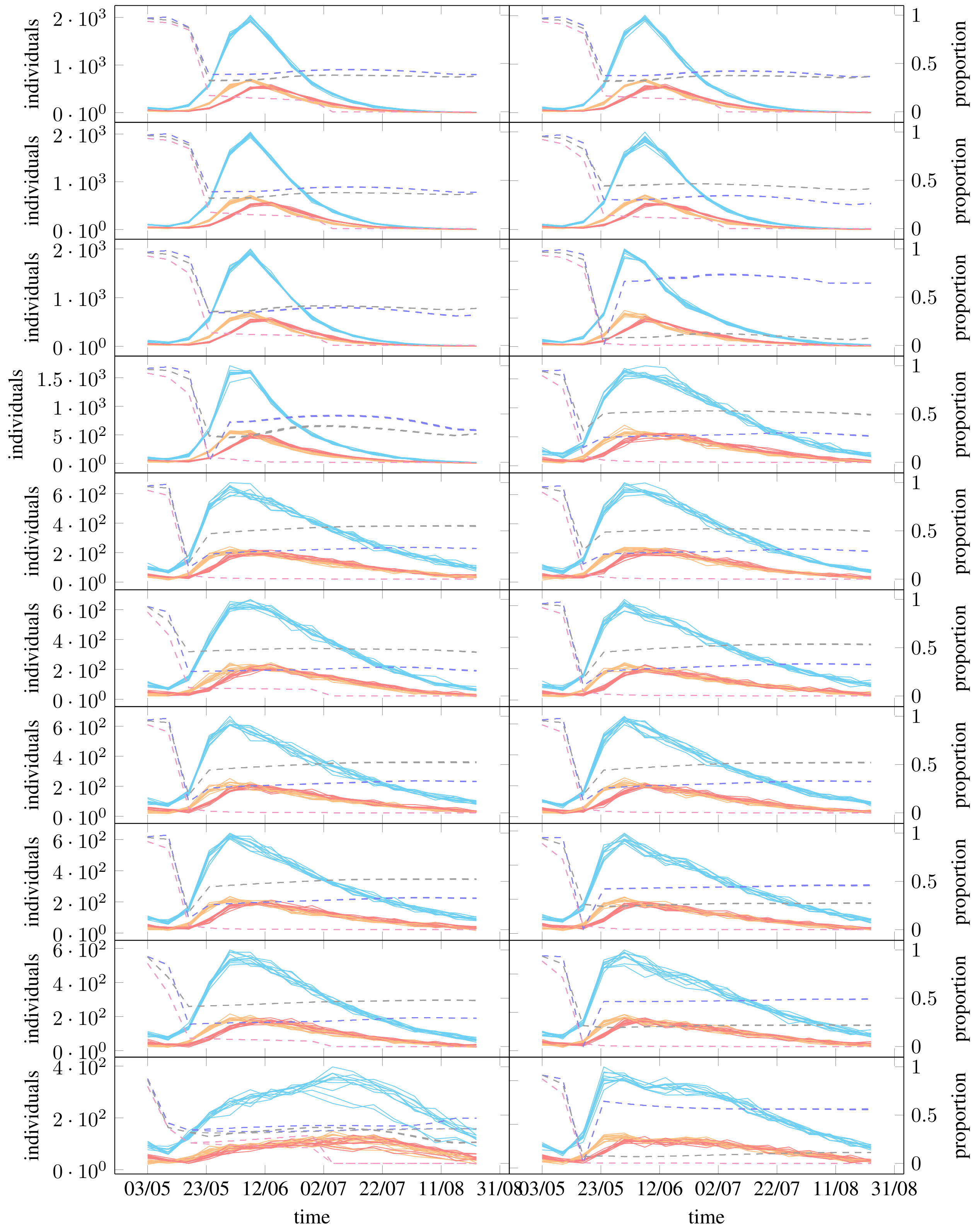

Figure B.7: Execution of policies 80 to 99, with a budget of 2.

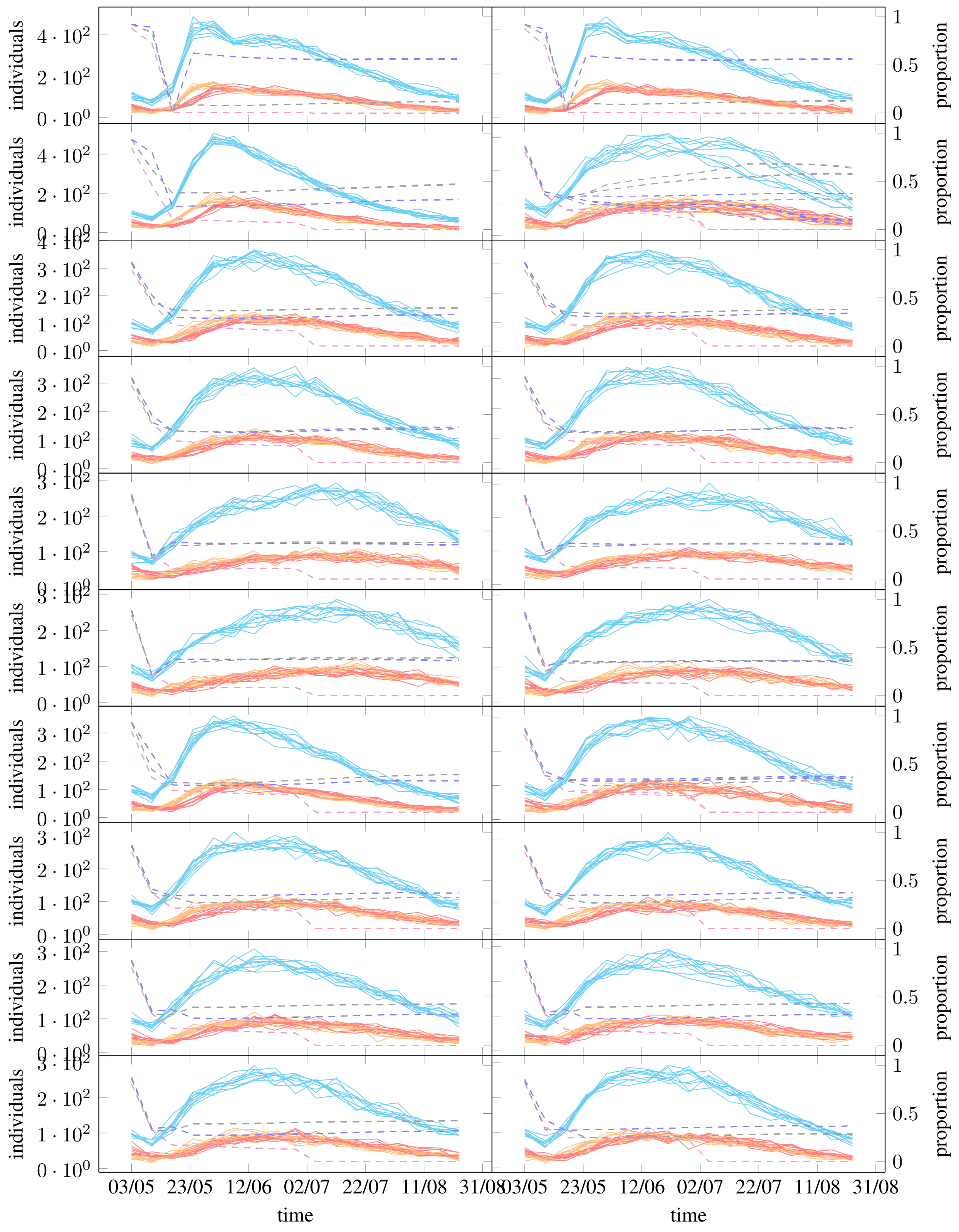

Figure B.8: Execution of policies 100 to 119, with a budget of 2.

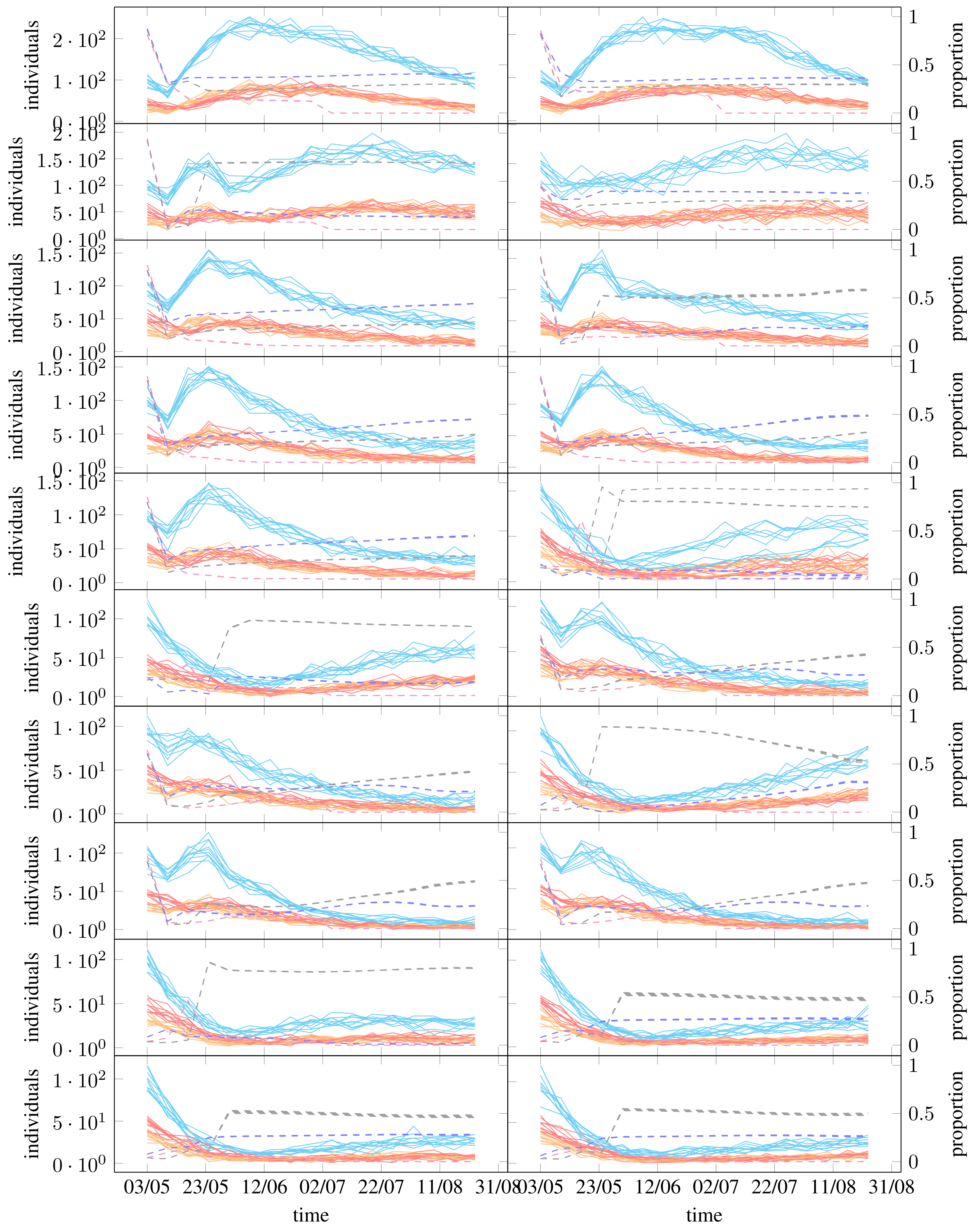

Figure B.9: Execution of policies 120 to 139, with a budget of 2.

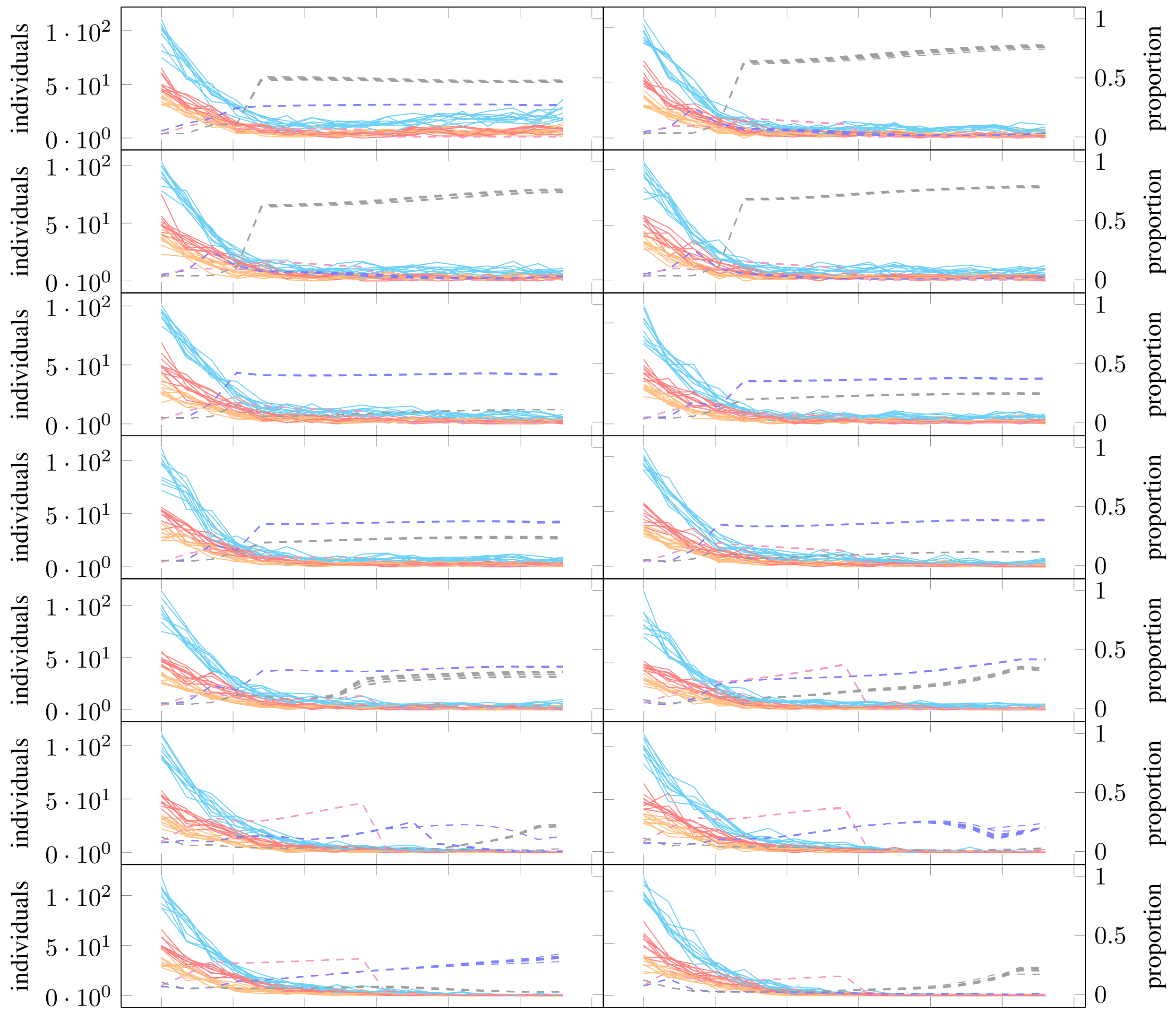

Figure B.10: Execution of policies 140 to 153, with a budget of 2.

B.4 Experiment parameters

We used the same hyper-parameters across all experiments. Each experiment resulted in 10 independent trials. Finally, we performed a grid-search over possible hyper-parameter values. All hyper-parameters used and their possible values explored during grid-search are displayed in Table B.1.

B.5 Neural network architecture

Next to the hyper-parameter search, we also performed a grid search over 4 different neural network architectures. All the architectures have the same structure. We use a compartment embedding sc_{emb}, a social contact matrix embedding sm_{emb} and a school-holidays embedding sh_{emb} that take as inputs the compartment, the previous p_w, p_s, p_l values (as they fully define the SCM $\hat{C}$) and a boolean flag for school holidays, respectively. All these embeddings have a same-sized embedding of 64, which are multiplied together to form the full state embedding. This state-embedding is used as input for another network, s_{emb}. Additionally, we use a common embedding c_{emb} for the concatenation of the desired return and horizon. Finally, the results of s_{emb} and c_{emb} are multiplied together, before passing through a fully connected network fc that has 3 outputs, one for p_w, p_s, p_l respectively.

All the architectures of the different components are displayed in Table B.2. The variant used in all experiments is `dense-big`.

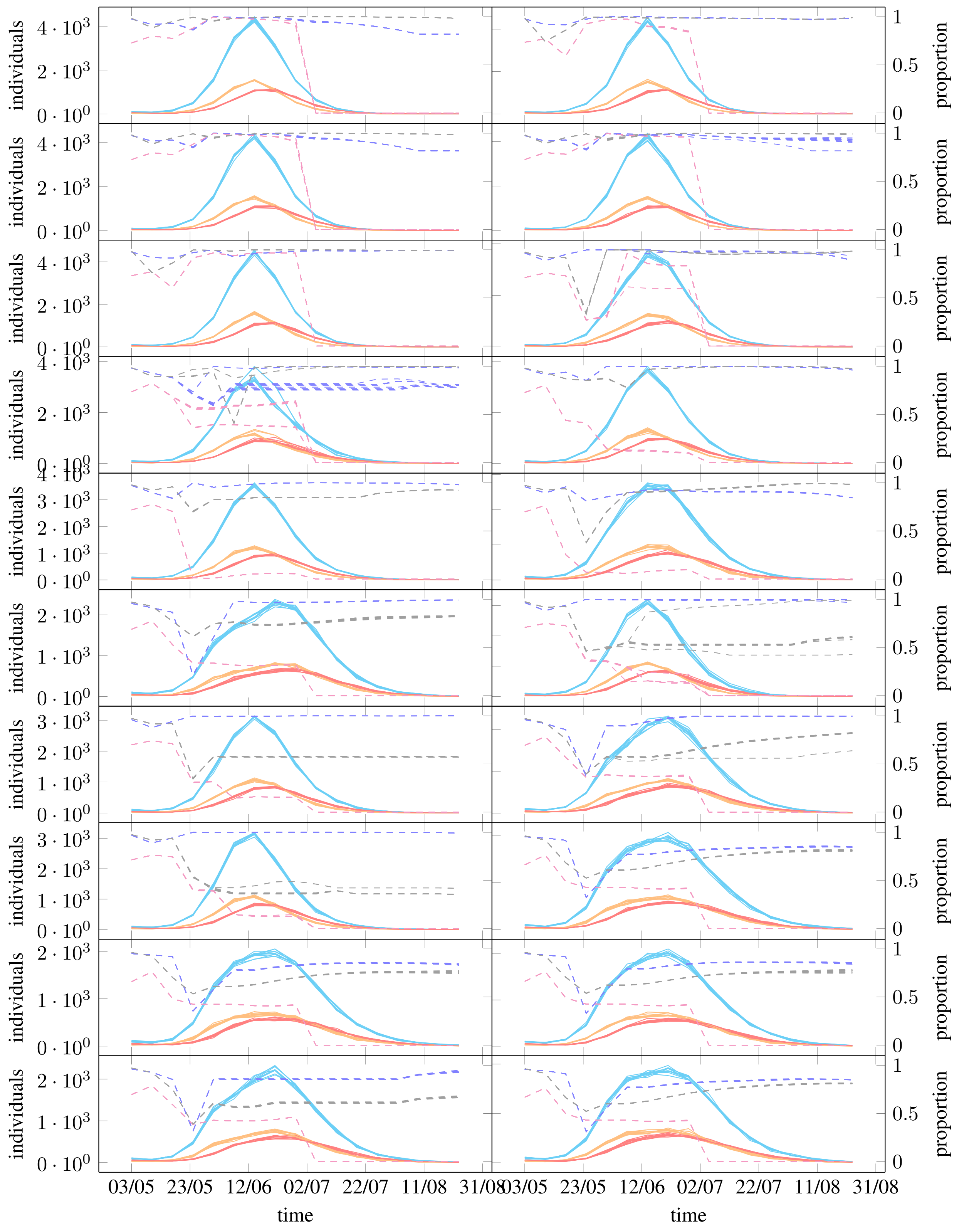

Figure B.11: Execution of policies 0 to 19, with a budget of 3.

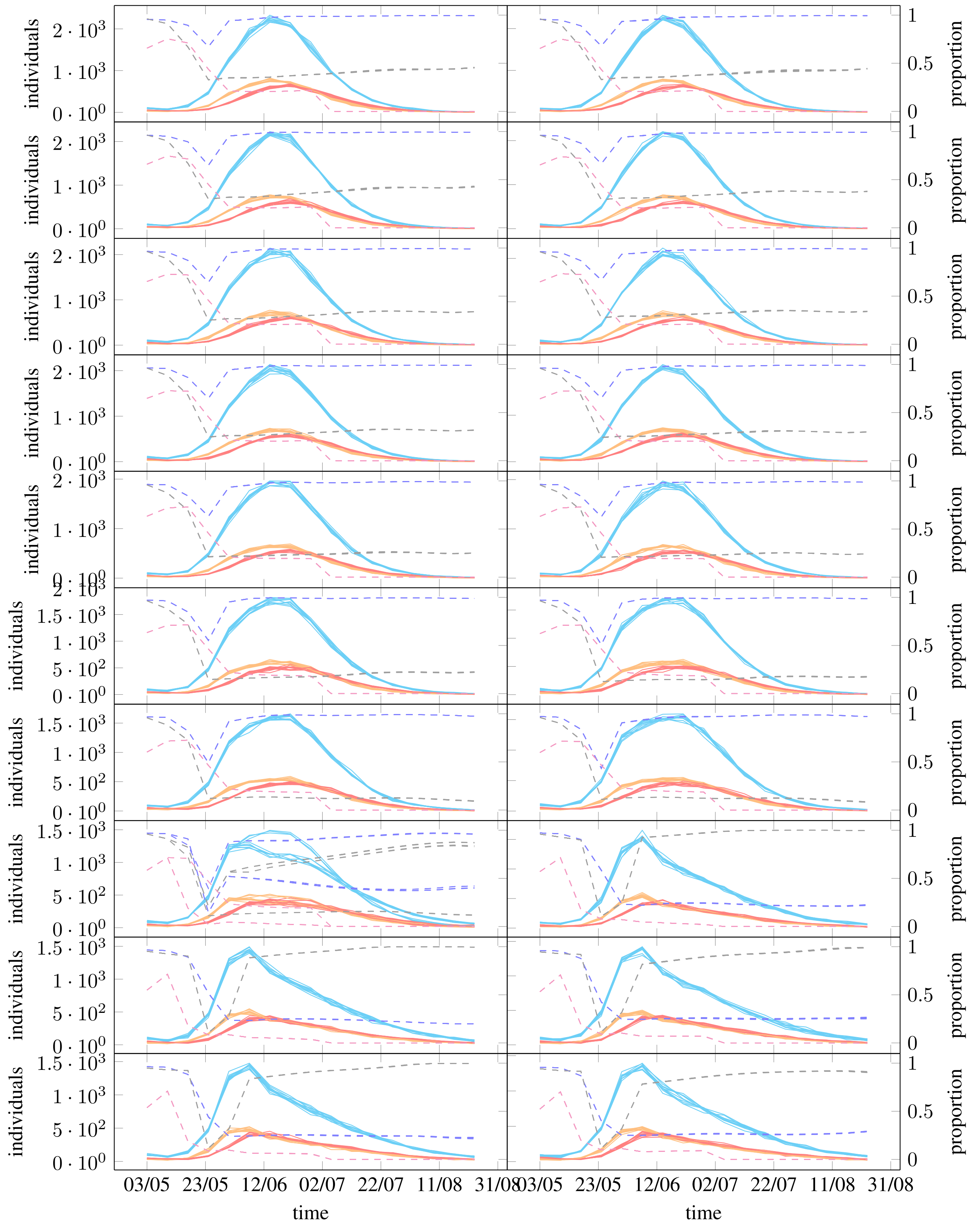

Figure B.12: Execution of policies 20 to 39, with a budget of 3.

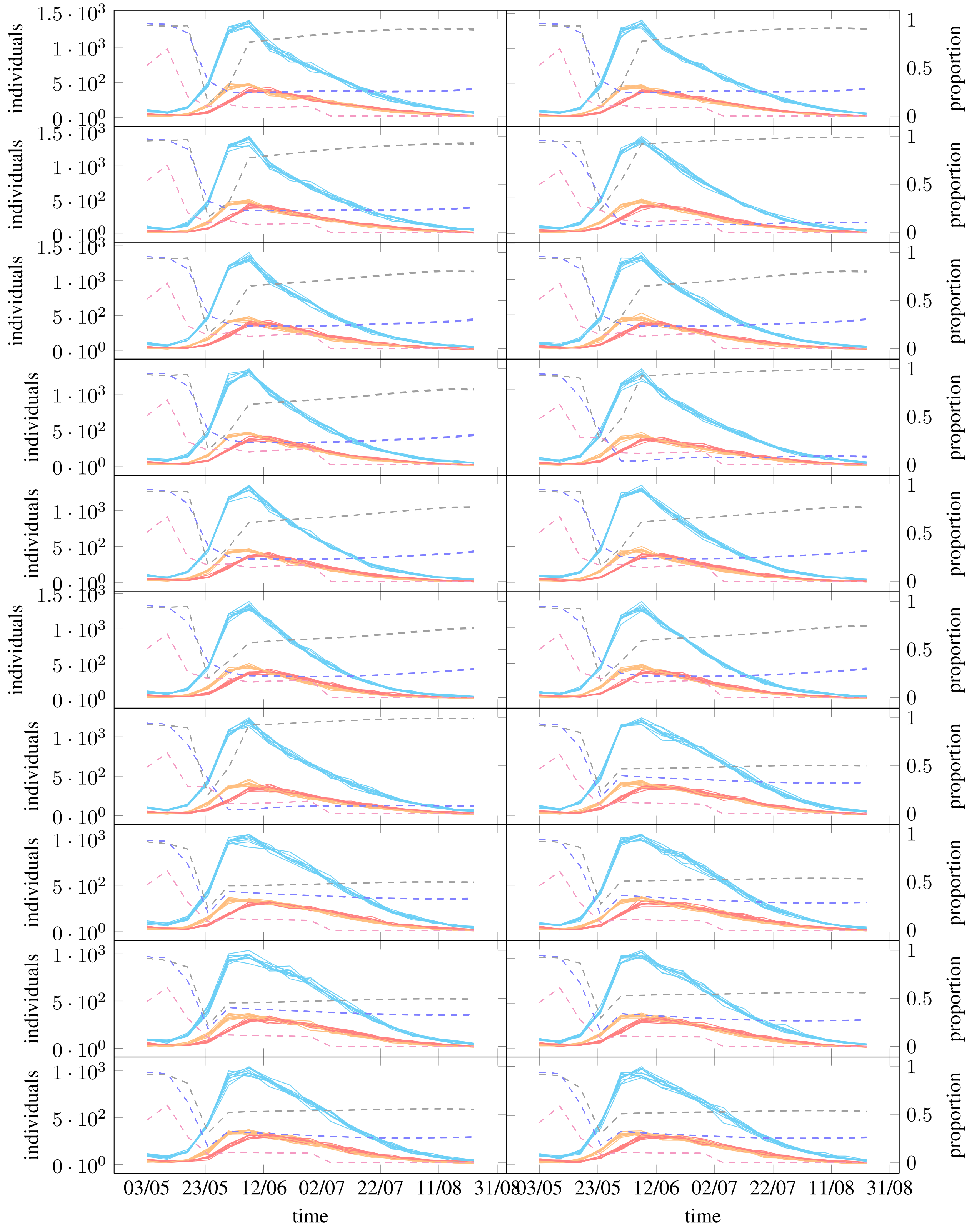

Figure B.13: Execution of policies 40 to 59, with a budget of 3.

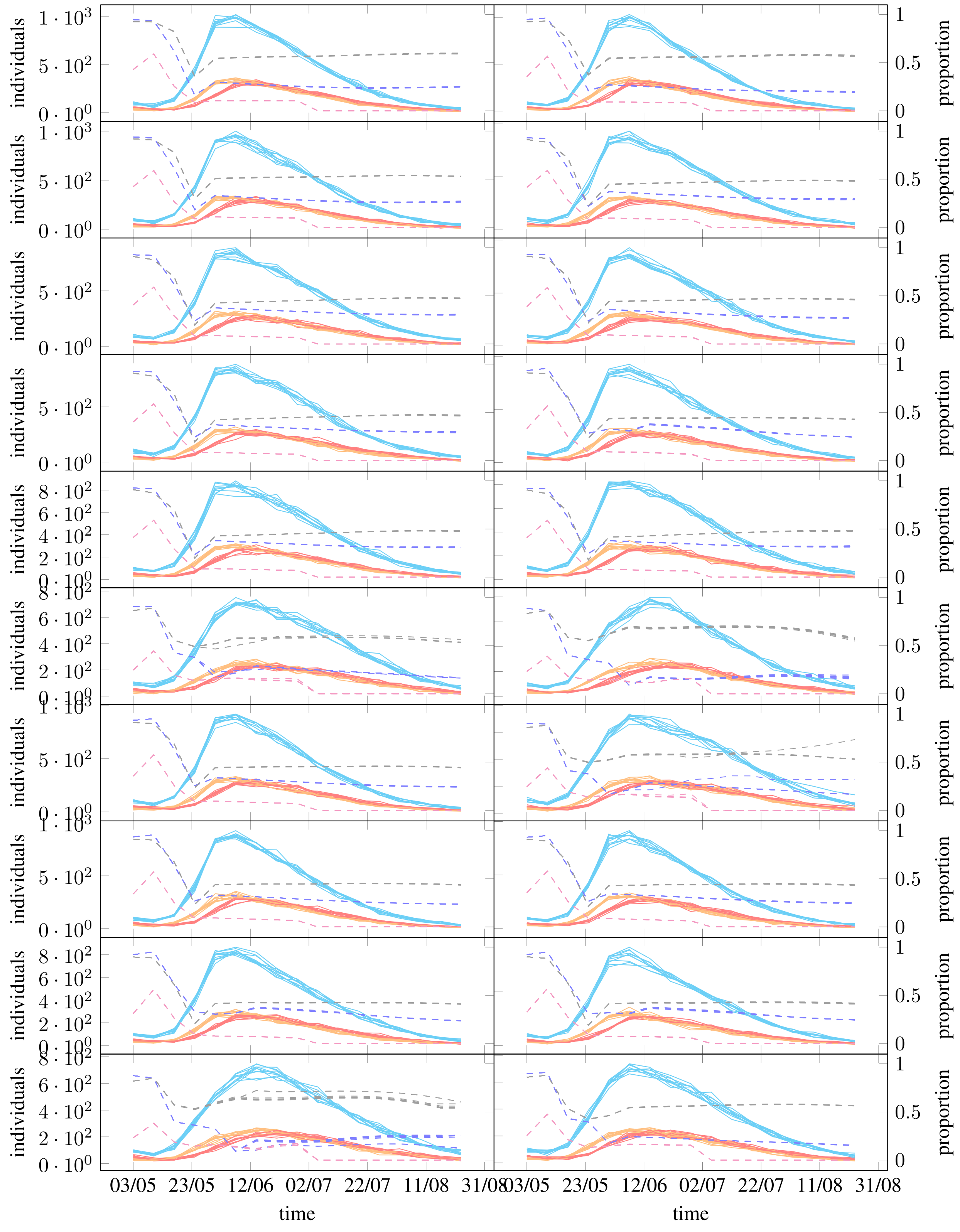

Figure B.14: Execution of policies 60 to 79, with a budget of 3.

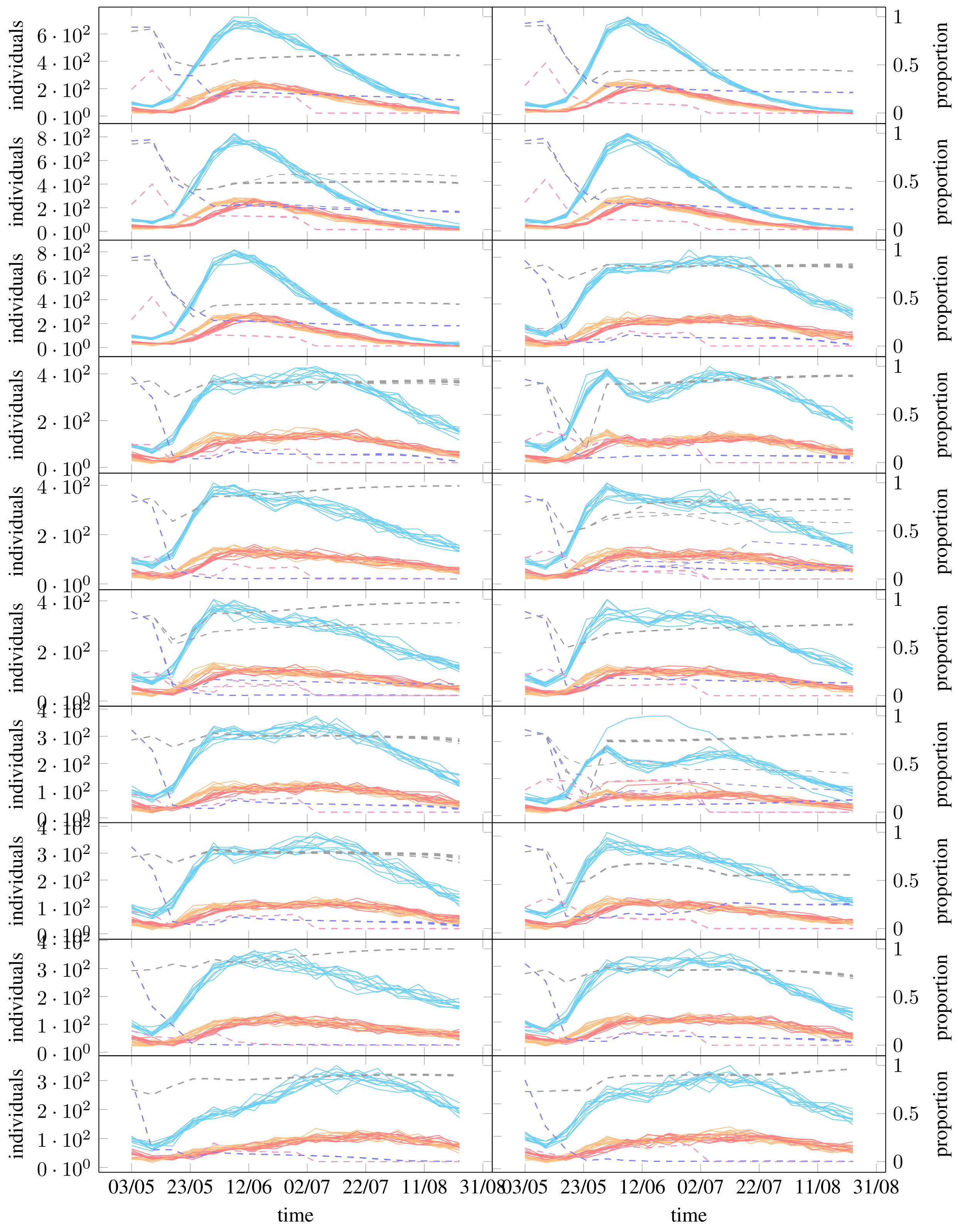

Figure B.15: Execution of policies 80 to 99, with a budget of 3.

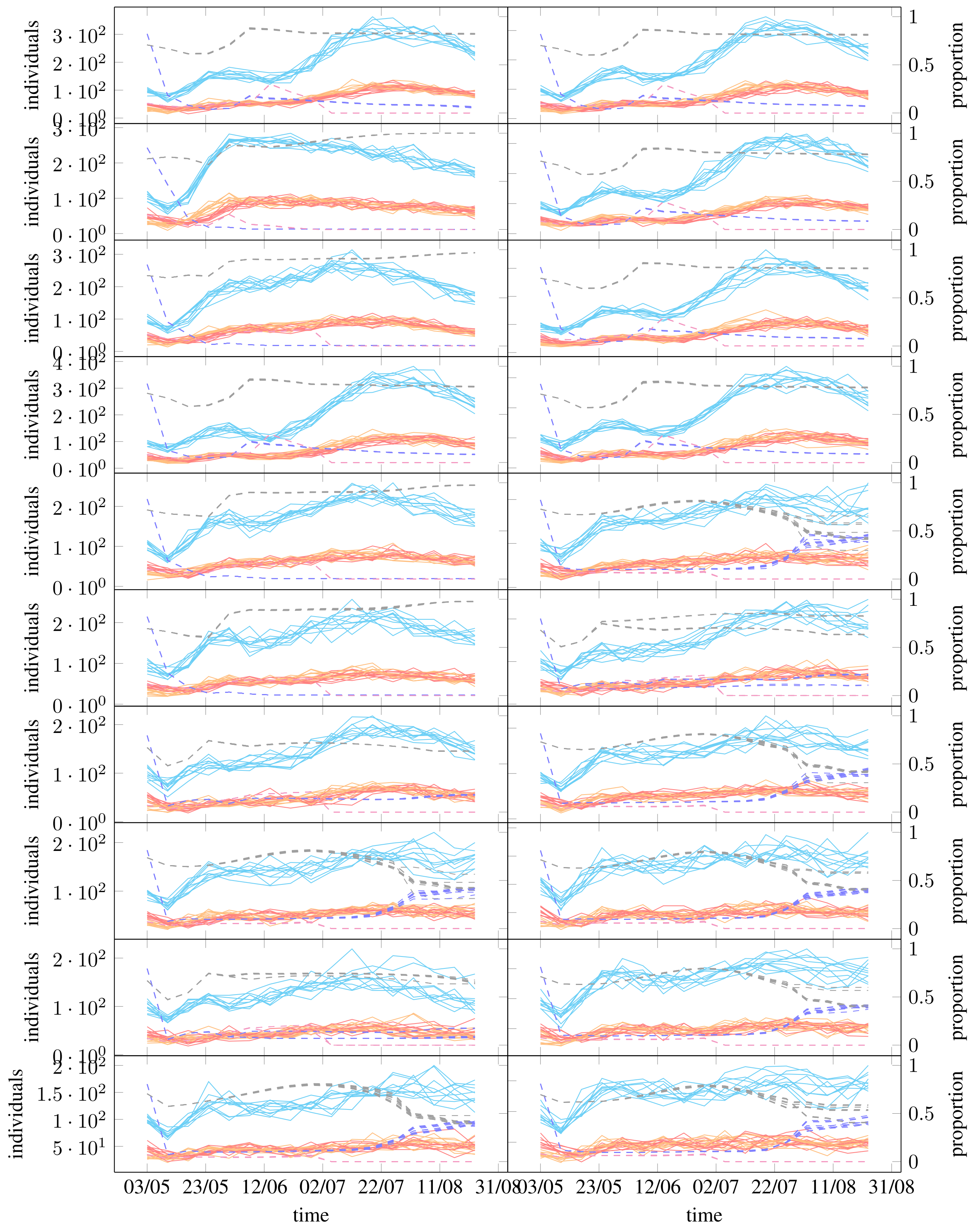

Figure B.16: Execution of policies 100 to 119, with a budget of 3.

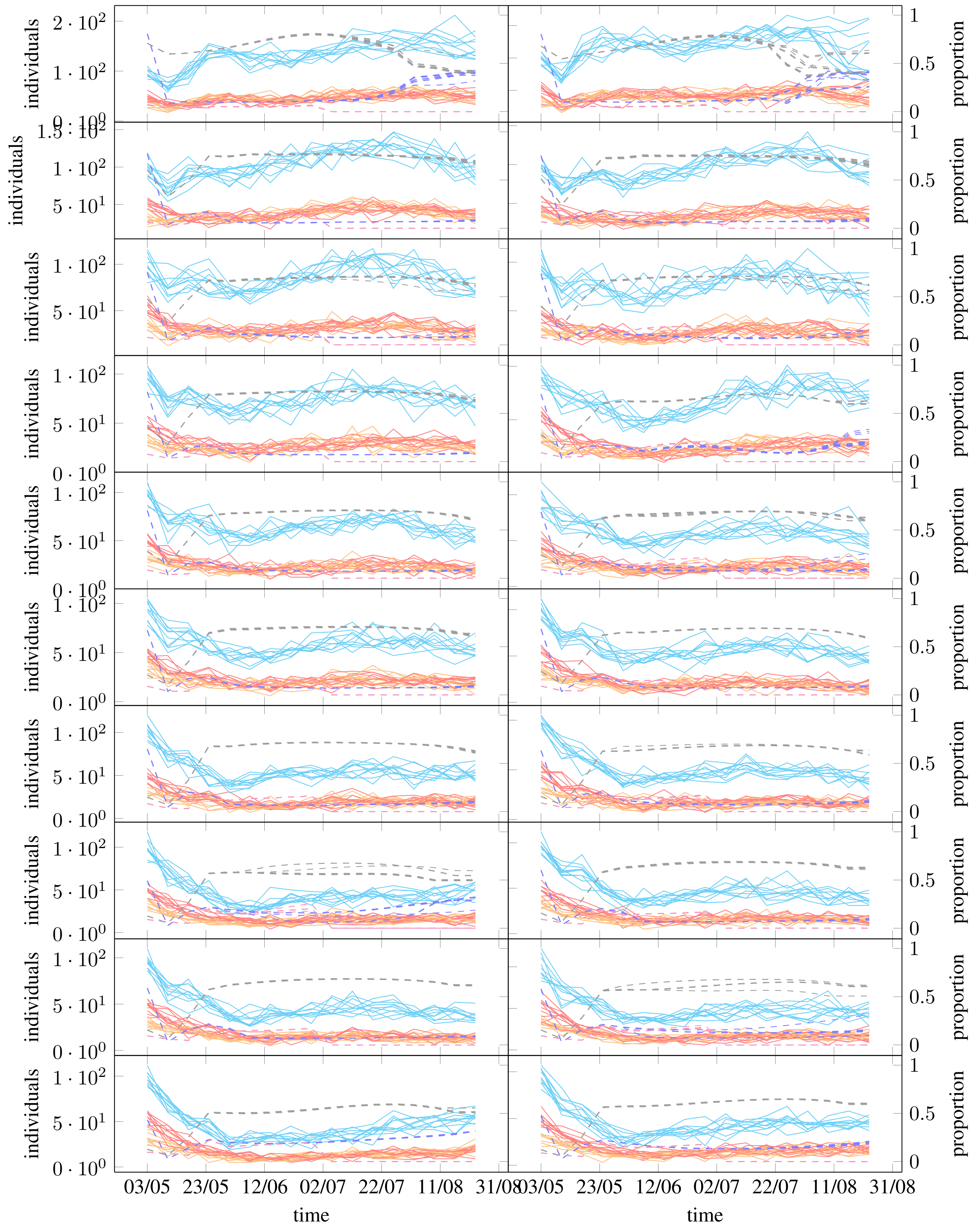

Figure B.17: Execution of policies 120 to 139, with a budget of 3.

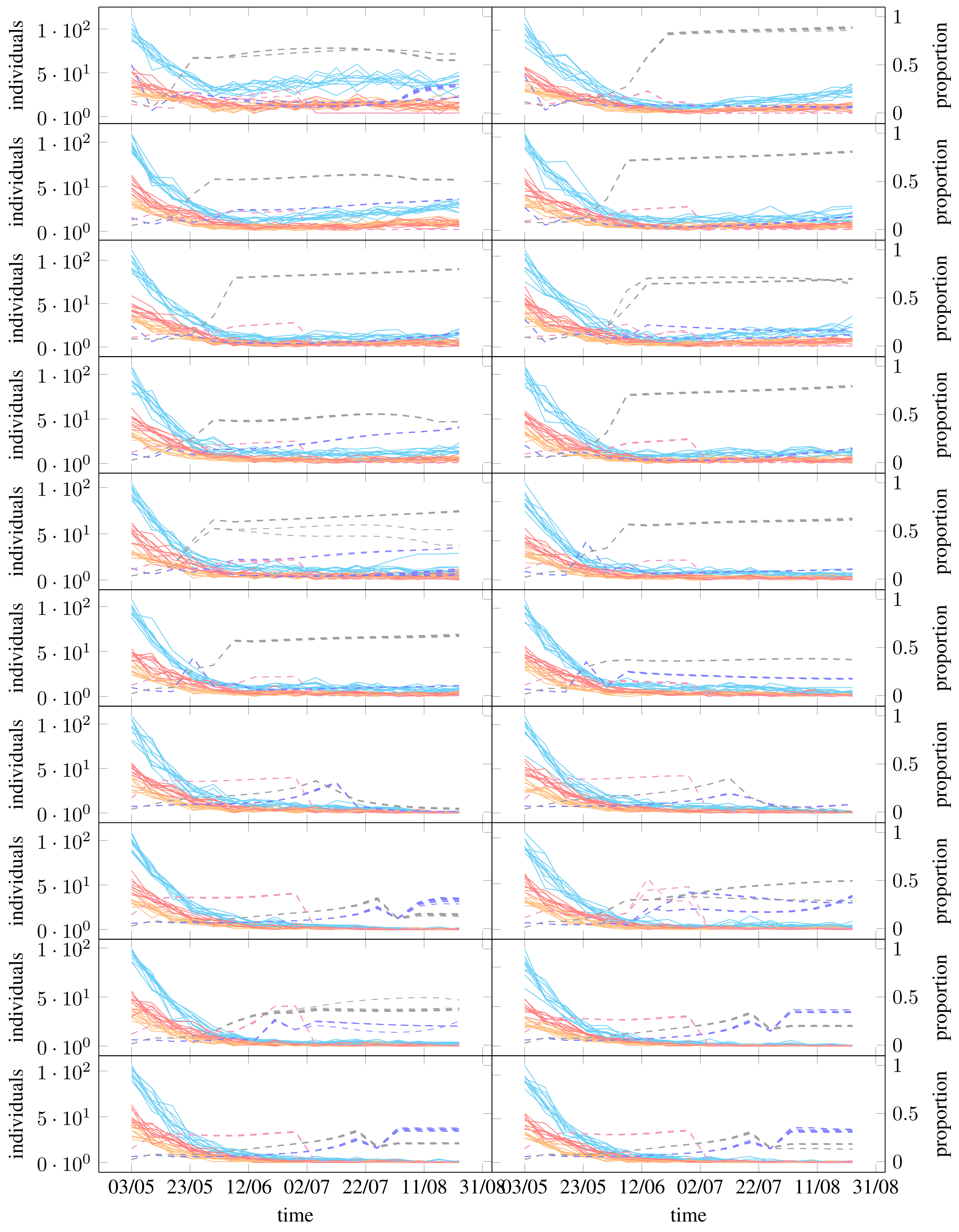

Figure B.18: Execution of policies 140 to 159, with a budget of 3.

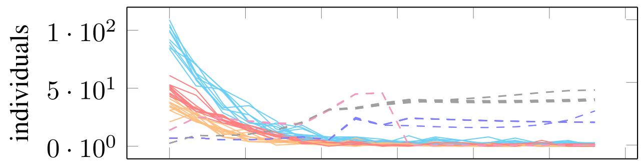

Figure B.19: Execution of policies 160 to 160, with a budget of 3.

hyper-parameter	value	grid-search
learning rate	0.001	
total training timesteps	300000	
batch size	256	$256, 1024$
model updates	50	
episodes between updates	10	
ER size (in episodes)	1000	$400, 500, 1000$
initial random episodes	200	$50, 200$
exploration noise	0.1	$0, 0.1, 0.2$
desired return noise	0.05	$0, 0.05, 0.1, 0.2$
reward scaling	$[10000, 100]$	

Table B.1: The different hyper-parameters used by our extension of PCN. The right-most column also shows, when applicable, the different values tried during grid-search.

variant	sc_{emb}	sm_{emb}	sh_{emb}	s_{emb}	c_{emb}	fc
conv1d-small	conv1d(10,20) relu conv1d(20,20) relu linear(100,64) sigmoid	linear(3,64) sigmoid	linear(1,64) sigmoid	linear(64,64) sigmoid	linear(3,64) sigmoid	linear(64,64) relu linear(64,3)
conv1d-big	conv1d(10,20) relu conv1d(20,20) relu linear(100,64) sigmoid	linear(3,64) relu linear(64,64) sigmoid	linear(1,64) relu linear(64,64) sigmoid	linear(64,64) relu	linear(3,64) sigmoid	linear(64,64) relu linear(64,3)
dense-small	linear(130,64) sigmoid	linear(3,64) sigmoid	linear(1,64) sigmoid	linear(64,64) sigmoid	linear(3,64) sigmoid	linear(64,64) relu linear(64,3)
dense-big	linear(130,64) relu linear(64,64) sigmoid	linear(3,64) relu linear(64,64) sigmoid	linear(1,64) relu linear(64,64) sigmoid	linear(64,64) relu	linear(3,64) sigmoid	linear(64,64) relu linear(64,3)

Table B.2: The 4 different neural network architectures explored for our experiments. All the displayed results use the `dense-big` variant.

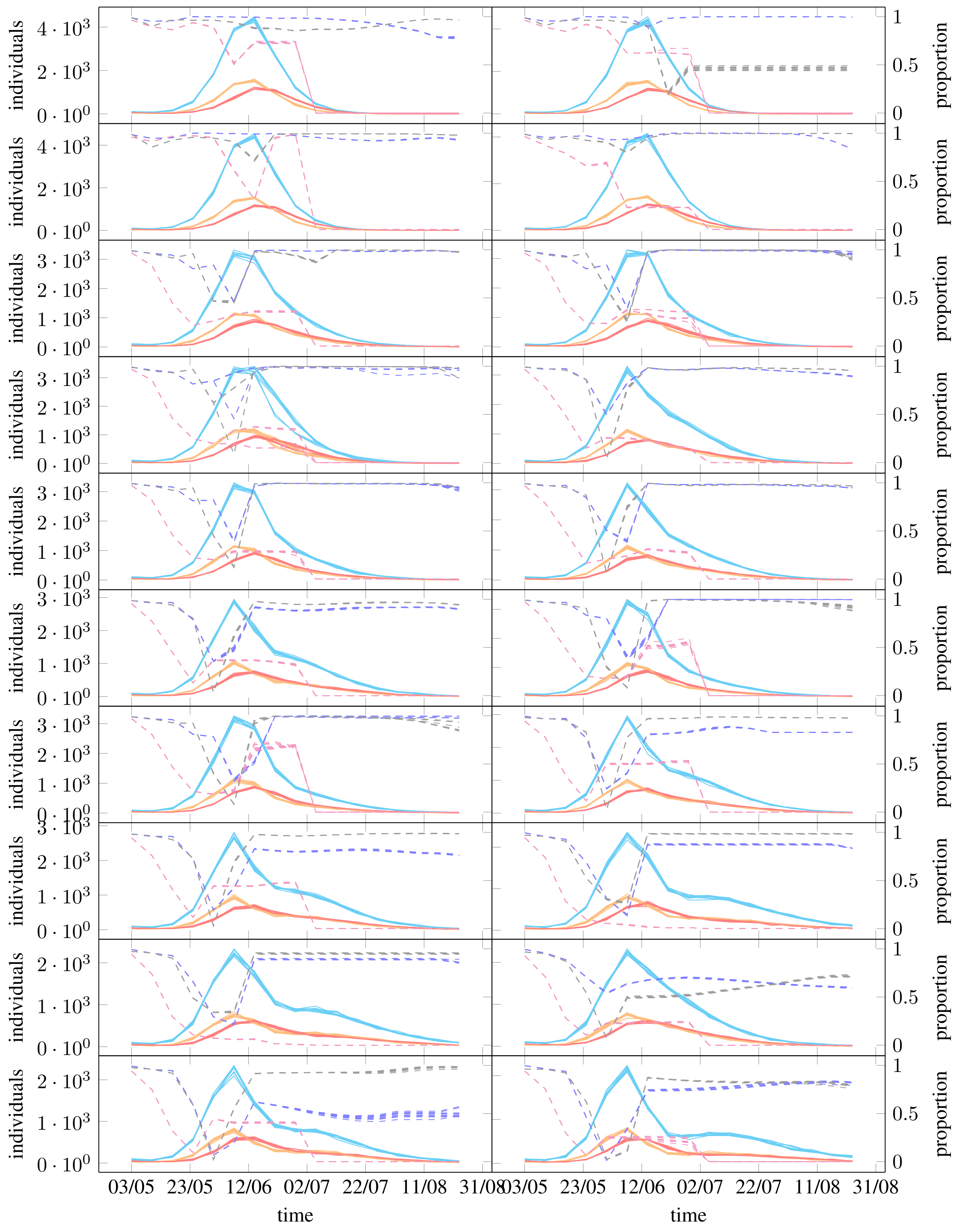

Figure B.20: Execution of policies 0 to 19, with a budget of 4.

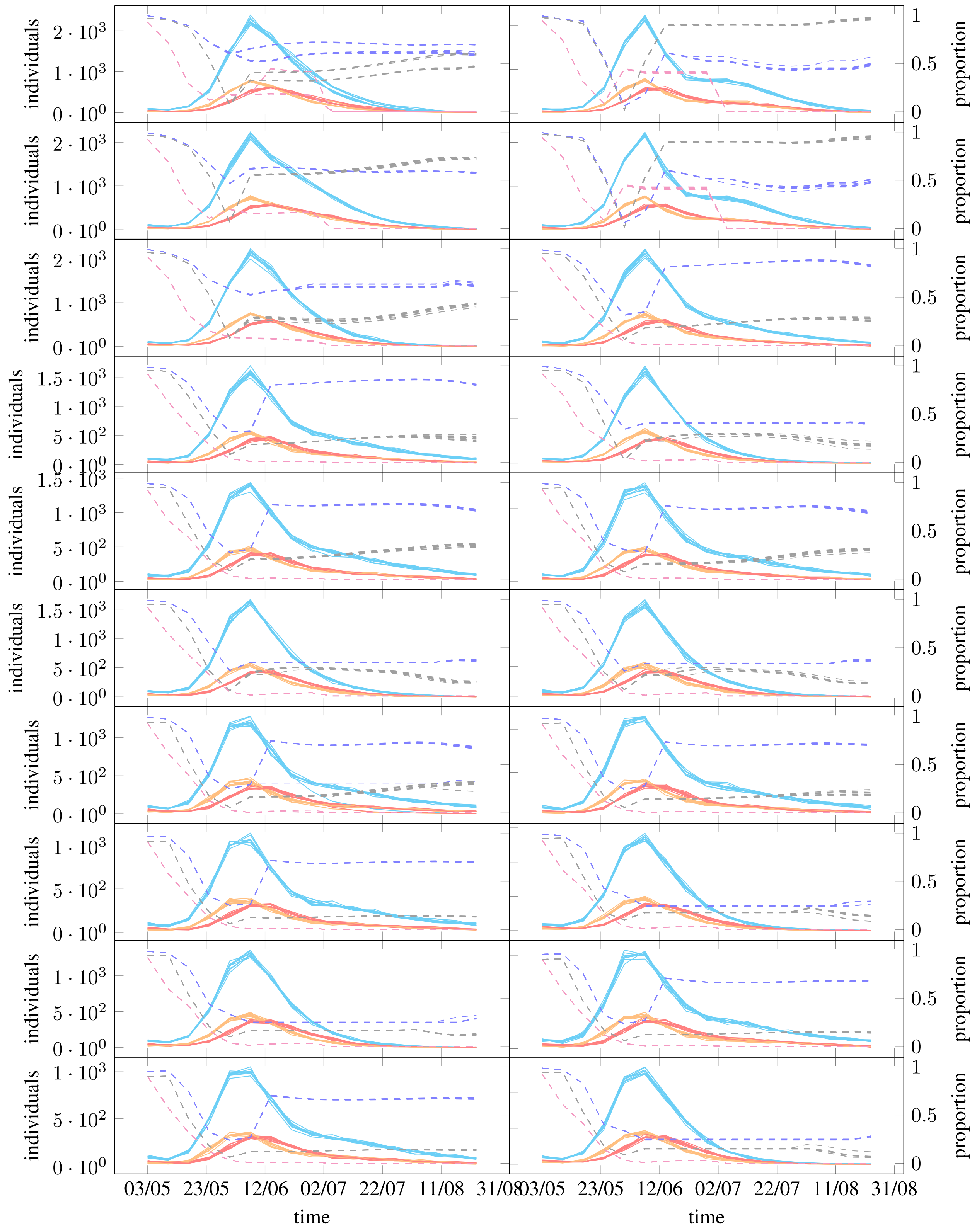

Figure B.21: Execution of policies 20 to 39, with a budget of 4.

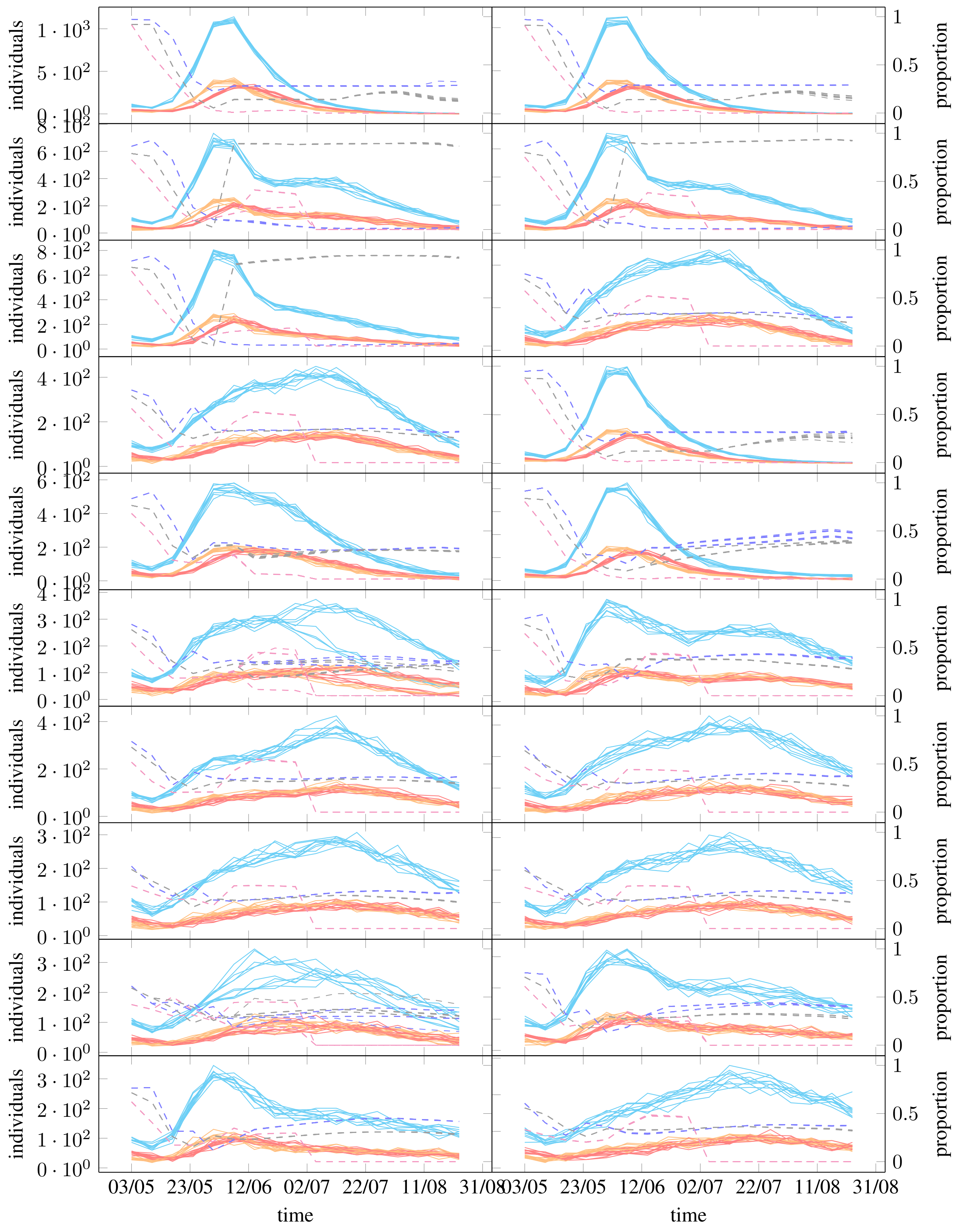

Figure B.22: Execution of policies 40 to 59, with a budget of 4.

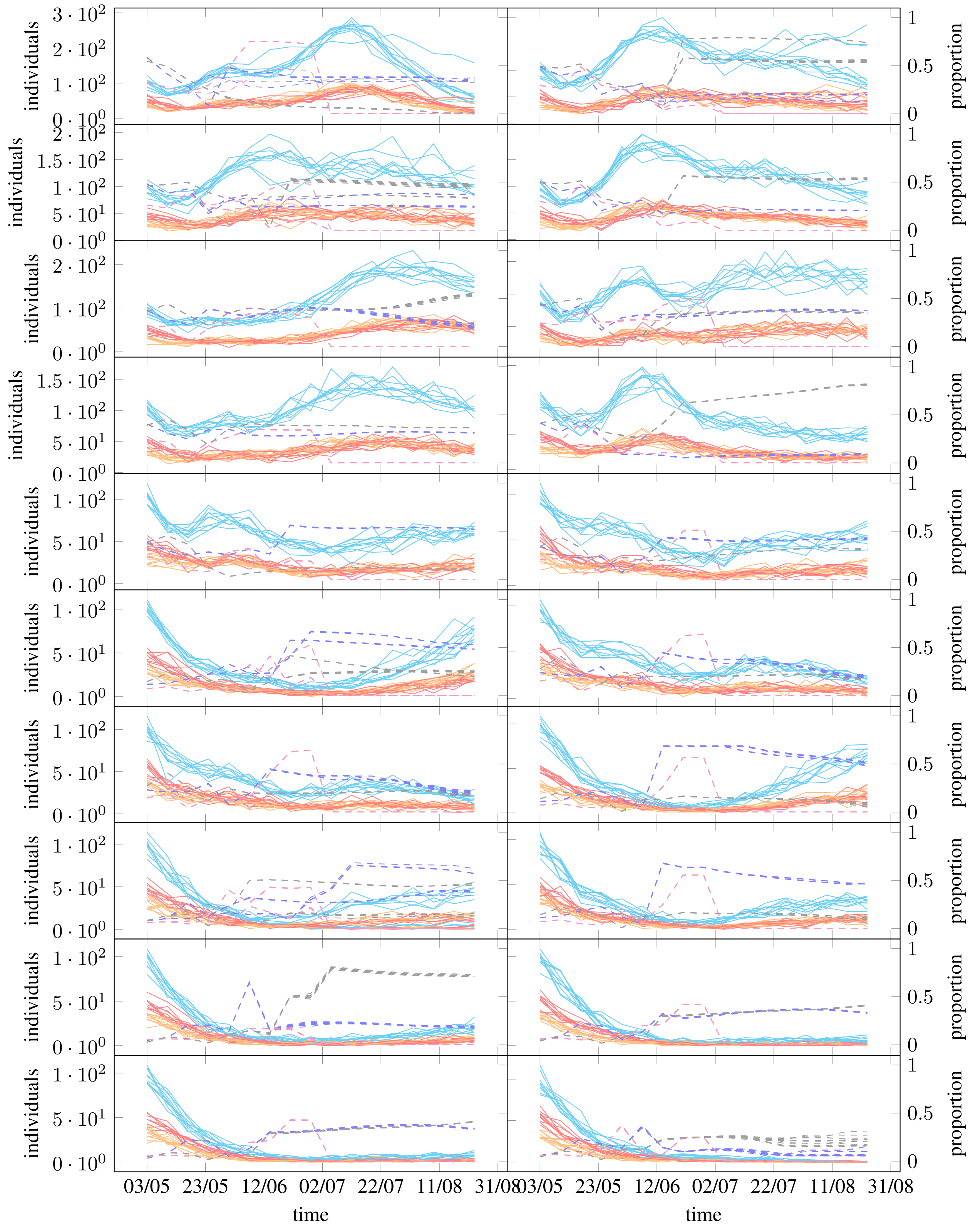

Figure B.23: Execution of policies 60 to 79, with a budget of 4.

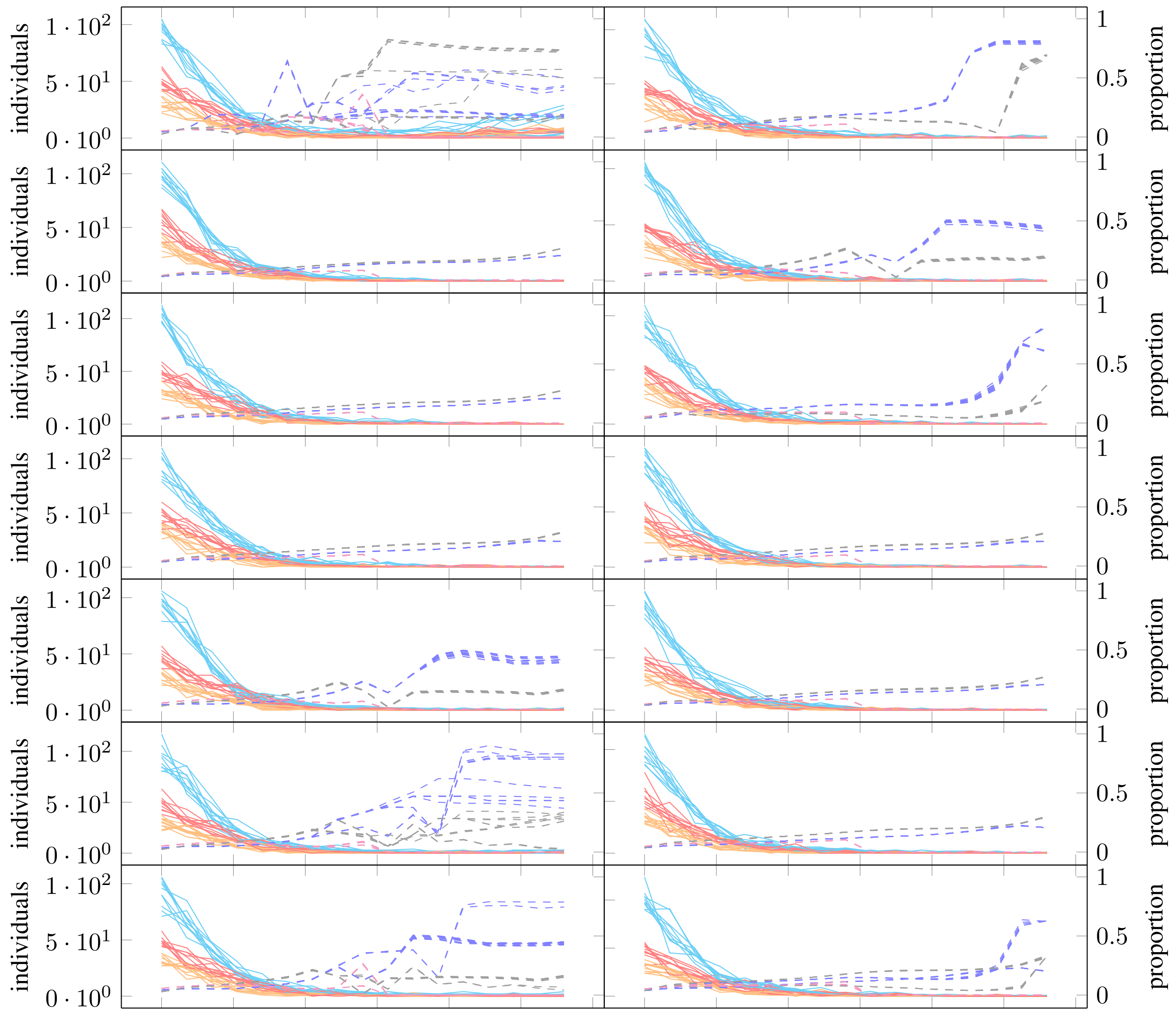

Figure B.24: Execution of policies 80 to 93, with a budget of 4.

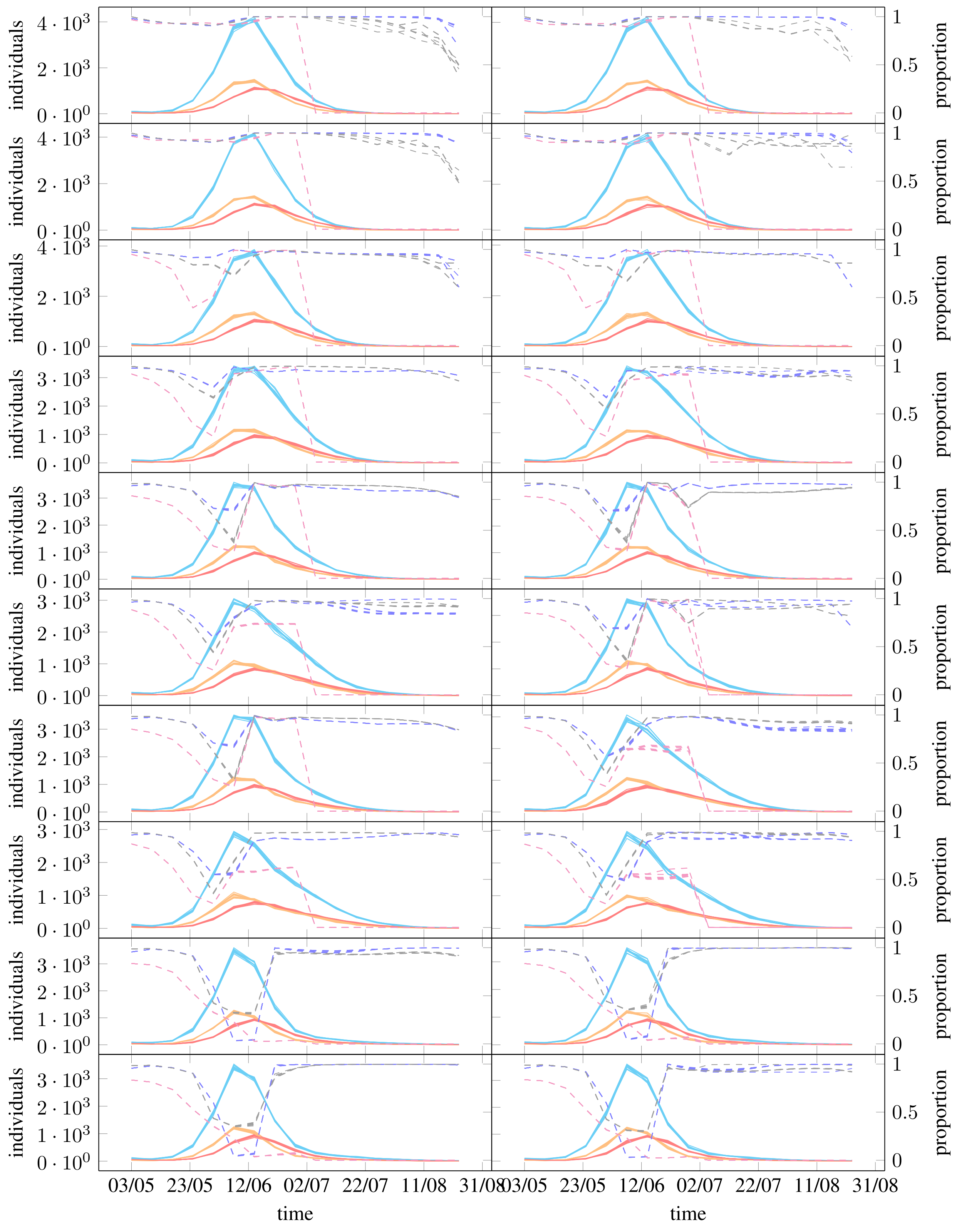

Figure B.25: Execution of policies 0 to 19, with a budget of 5.

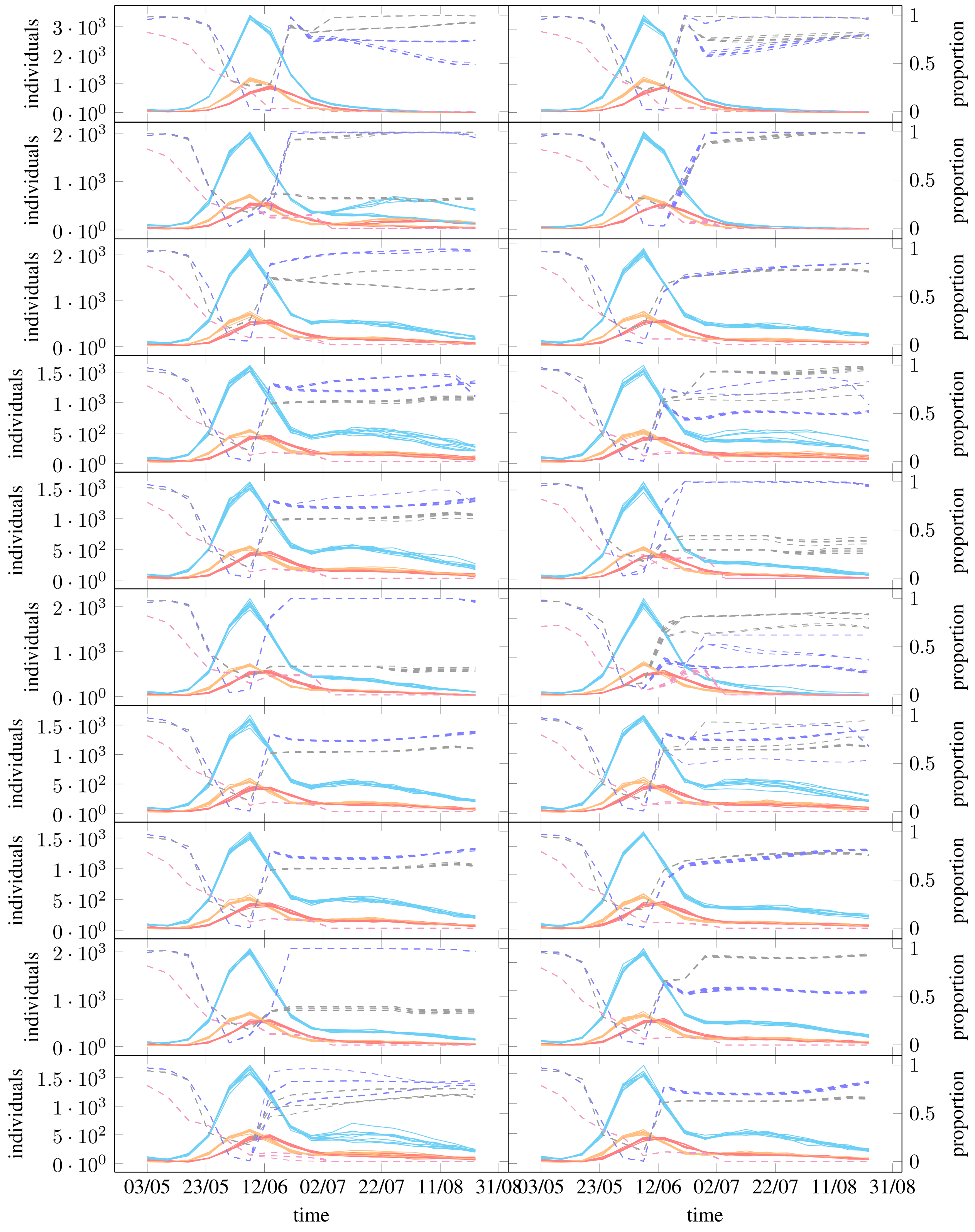

Figure B.26: Execution of policies 20 to 39, with a budget of 5.

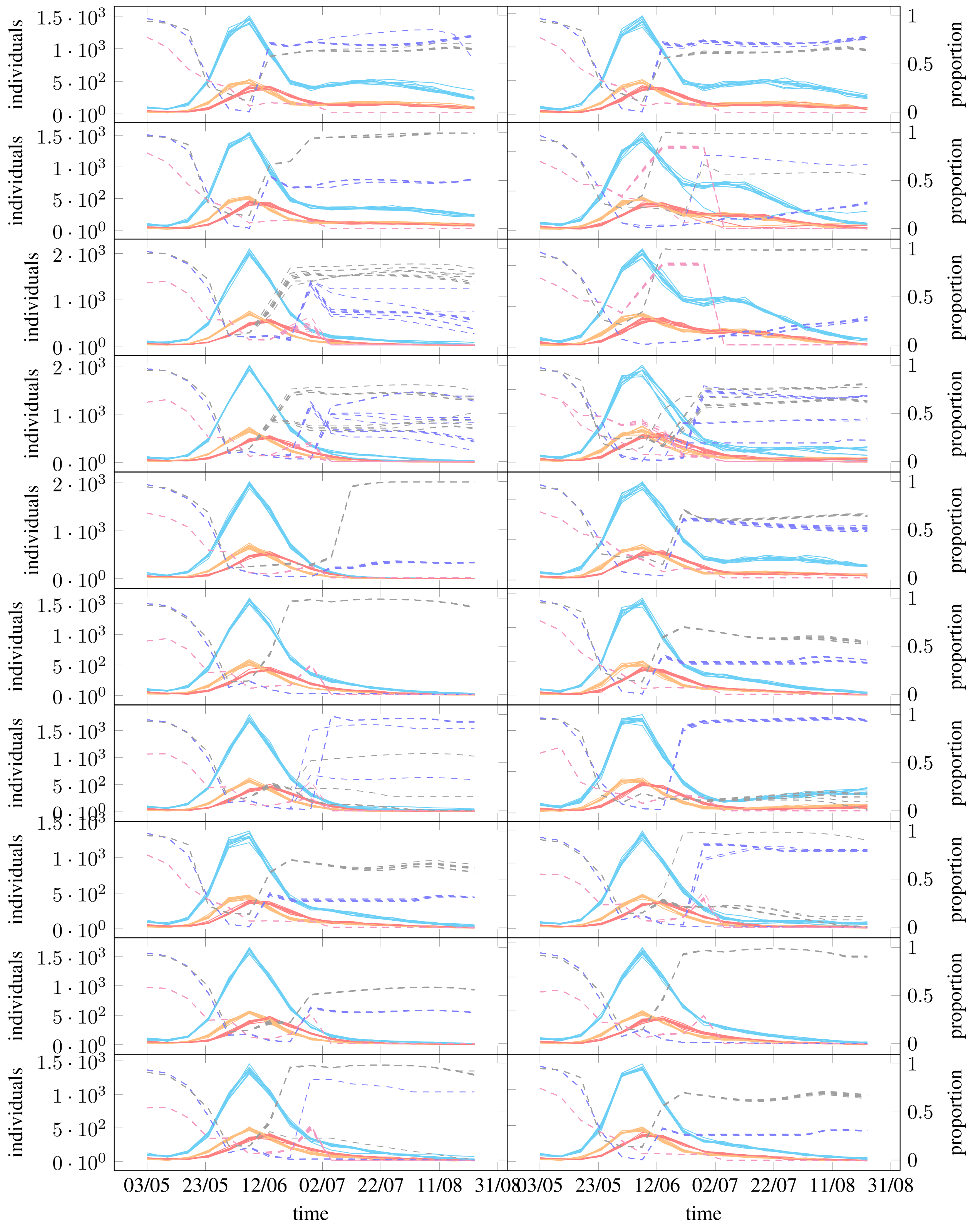

Figure B.27: Execution of policies 40 to 59, with a budget of 5.

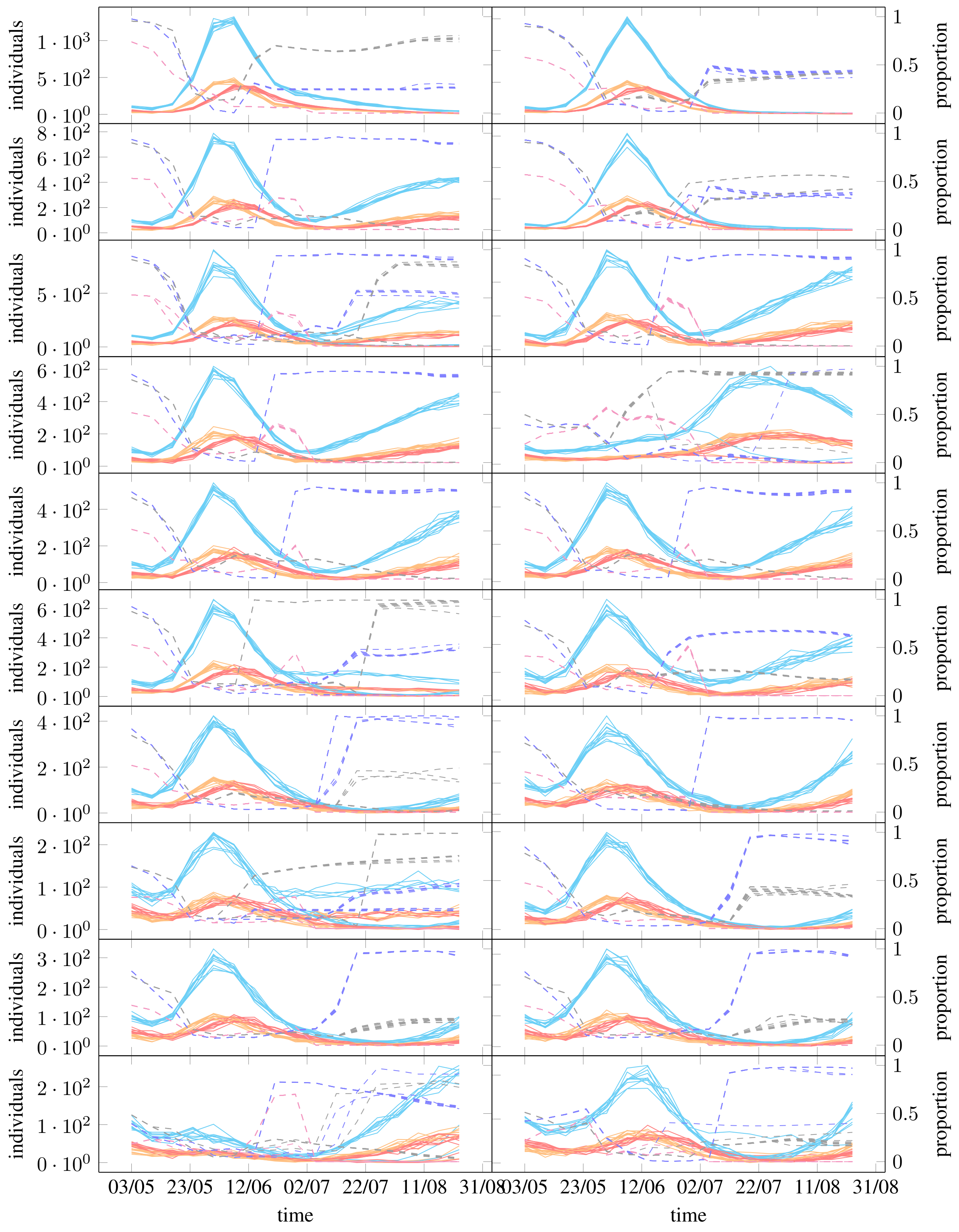

Figure B.28: Execution of policies 60 to 79, with a budget of 5.

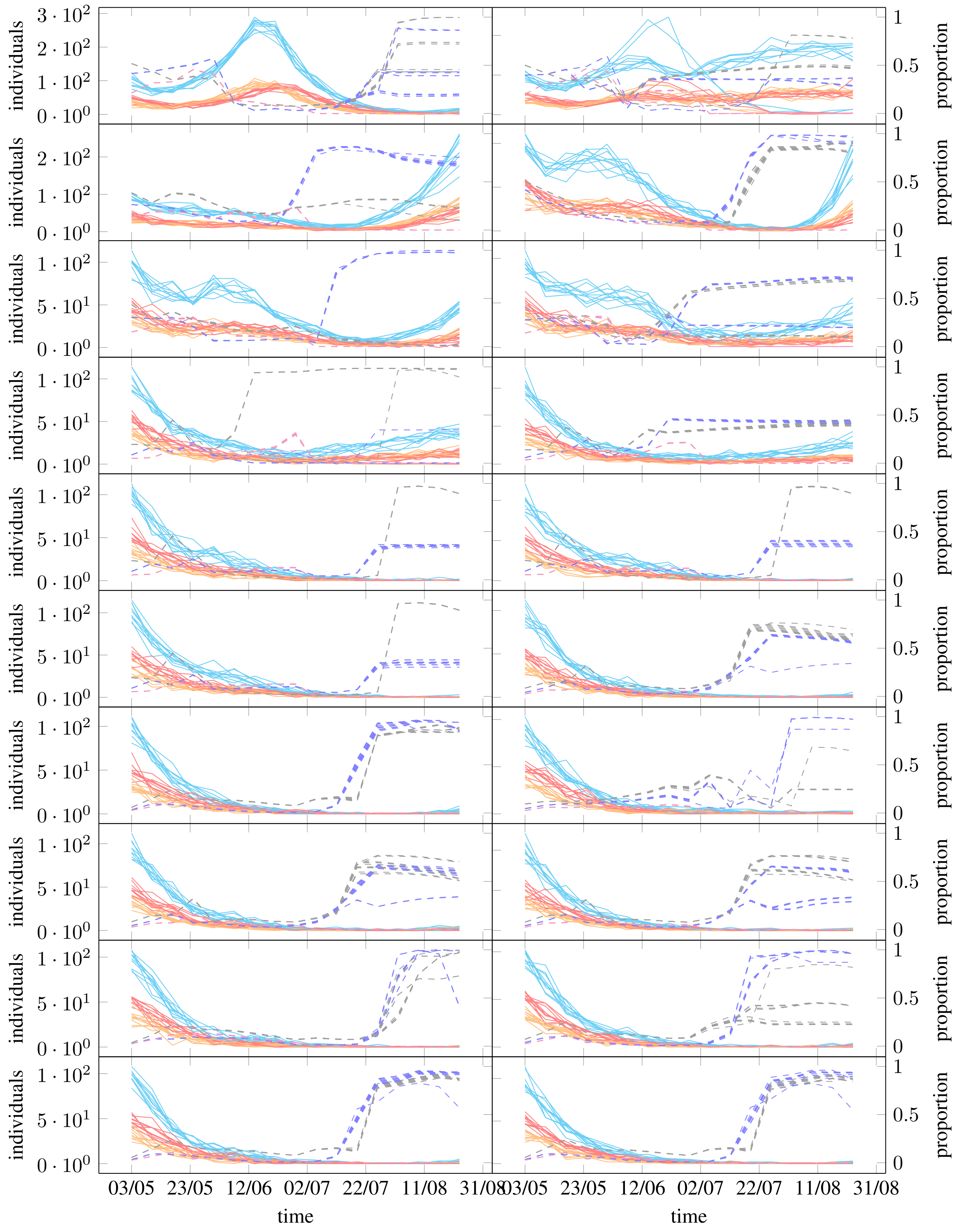

Figure B.29: Execution of policies 80 to 99, with a budget of 5.

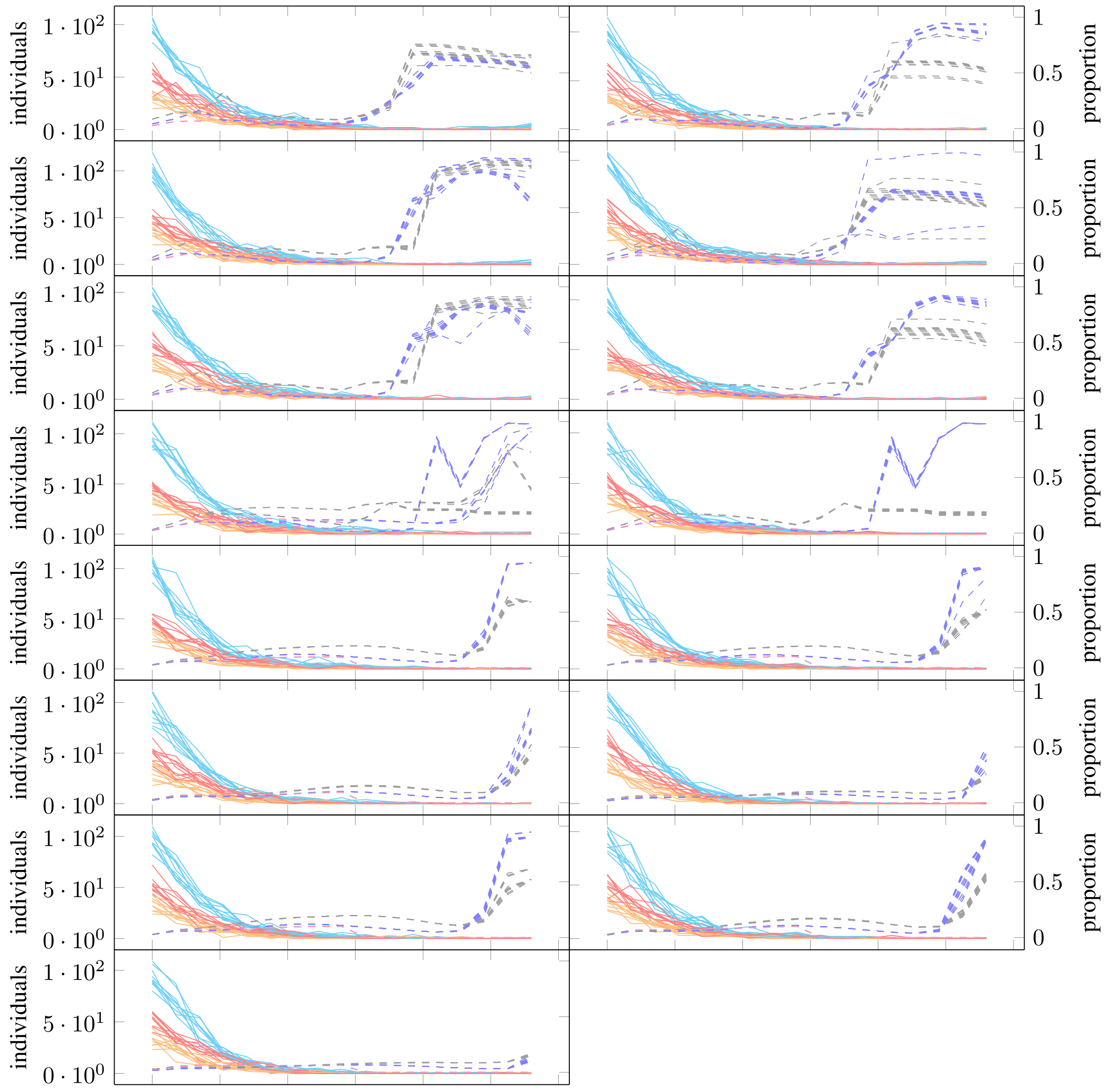

Figure B.30: Execution of policies 100 to 114, with a budget of 5.

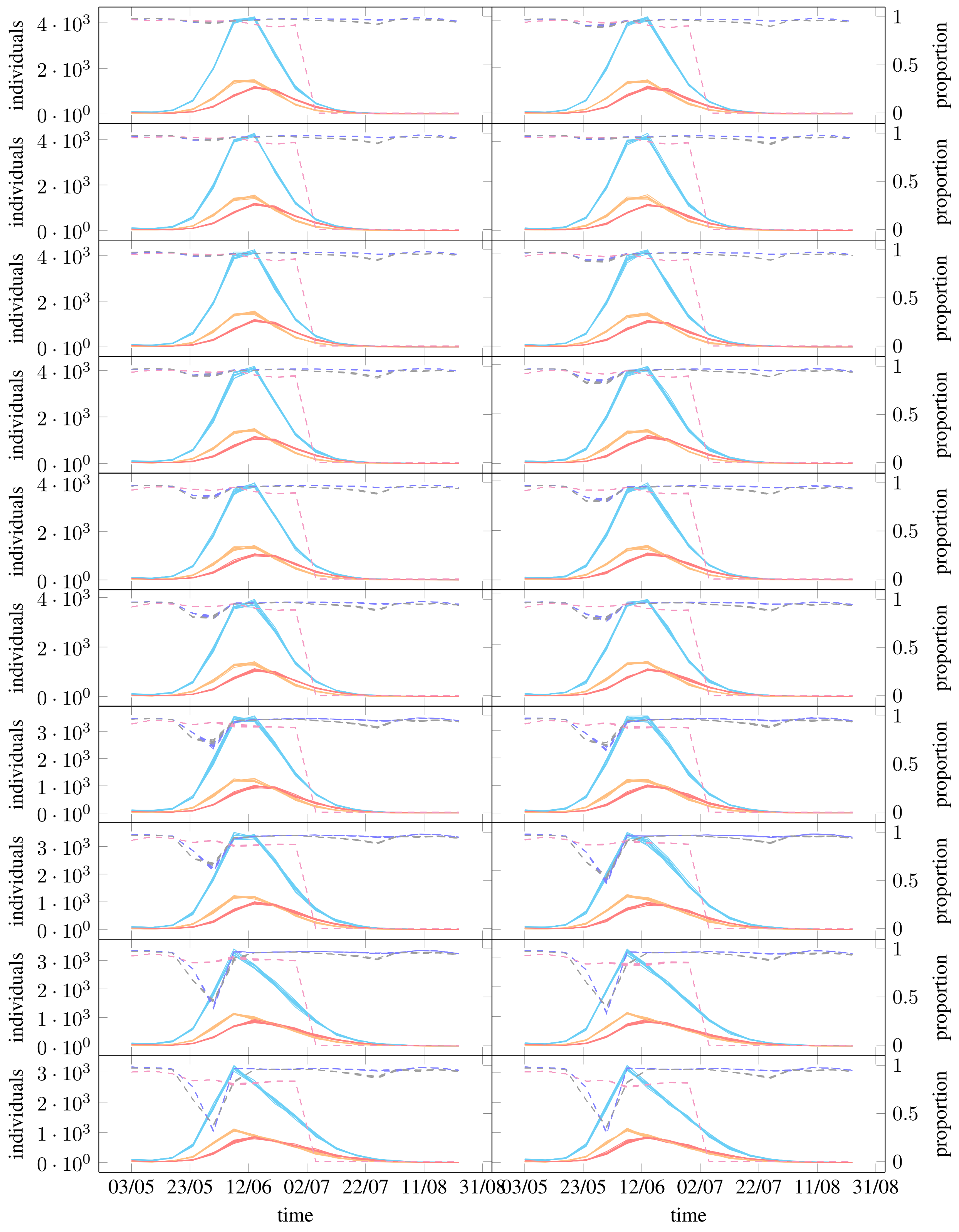

Figure B.31: Execution of policies 0 to 19, with no budget restriction ($b = \infty$).

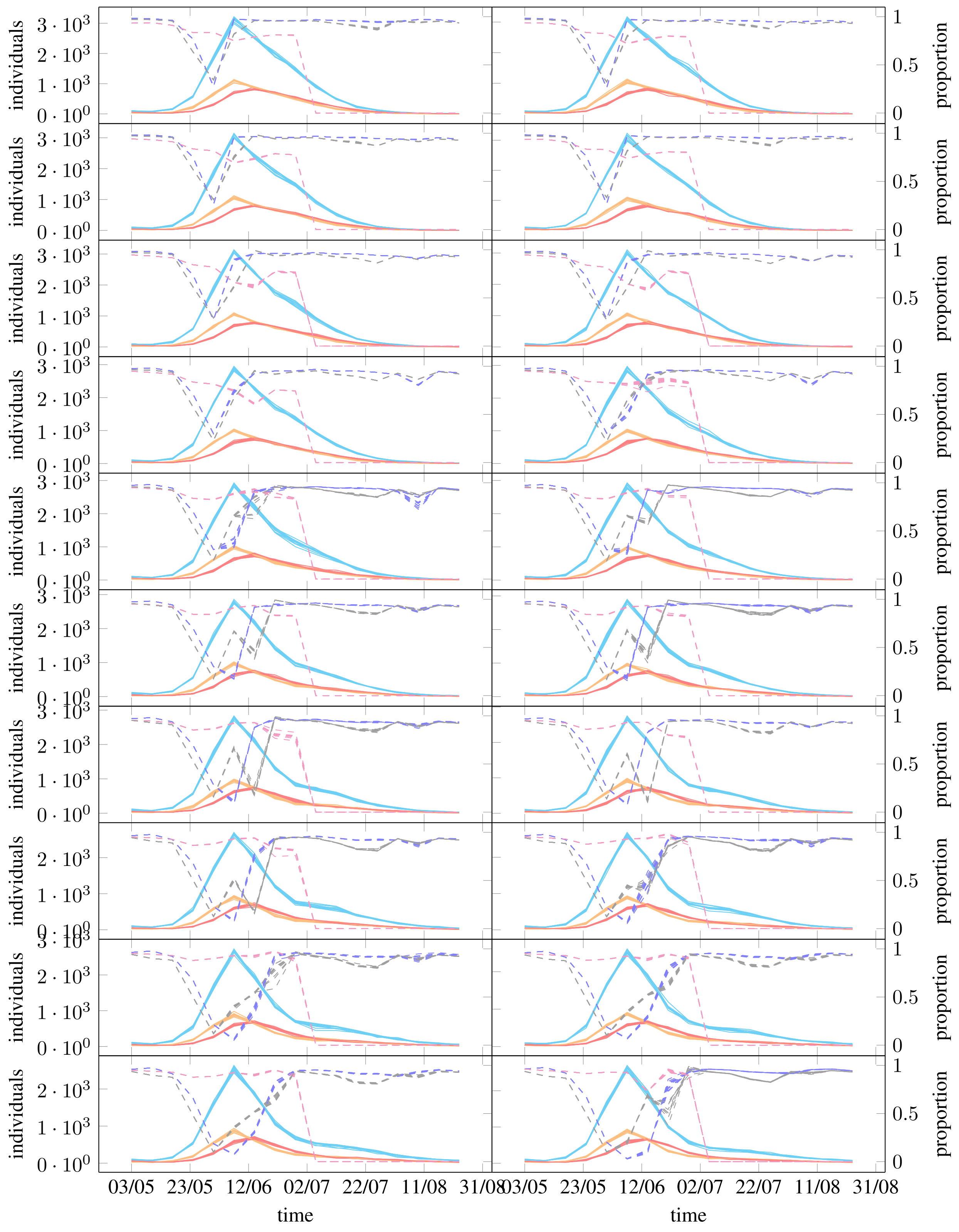

Figure B.32: Execution of policies 20 to 39, with no budget restriction ($b = \infty$).

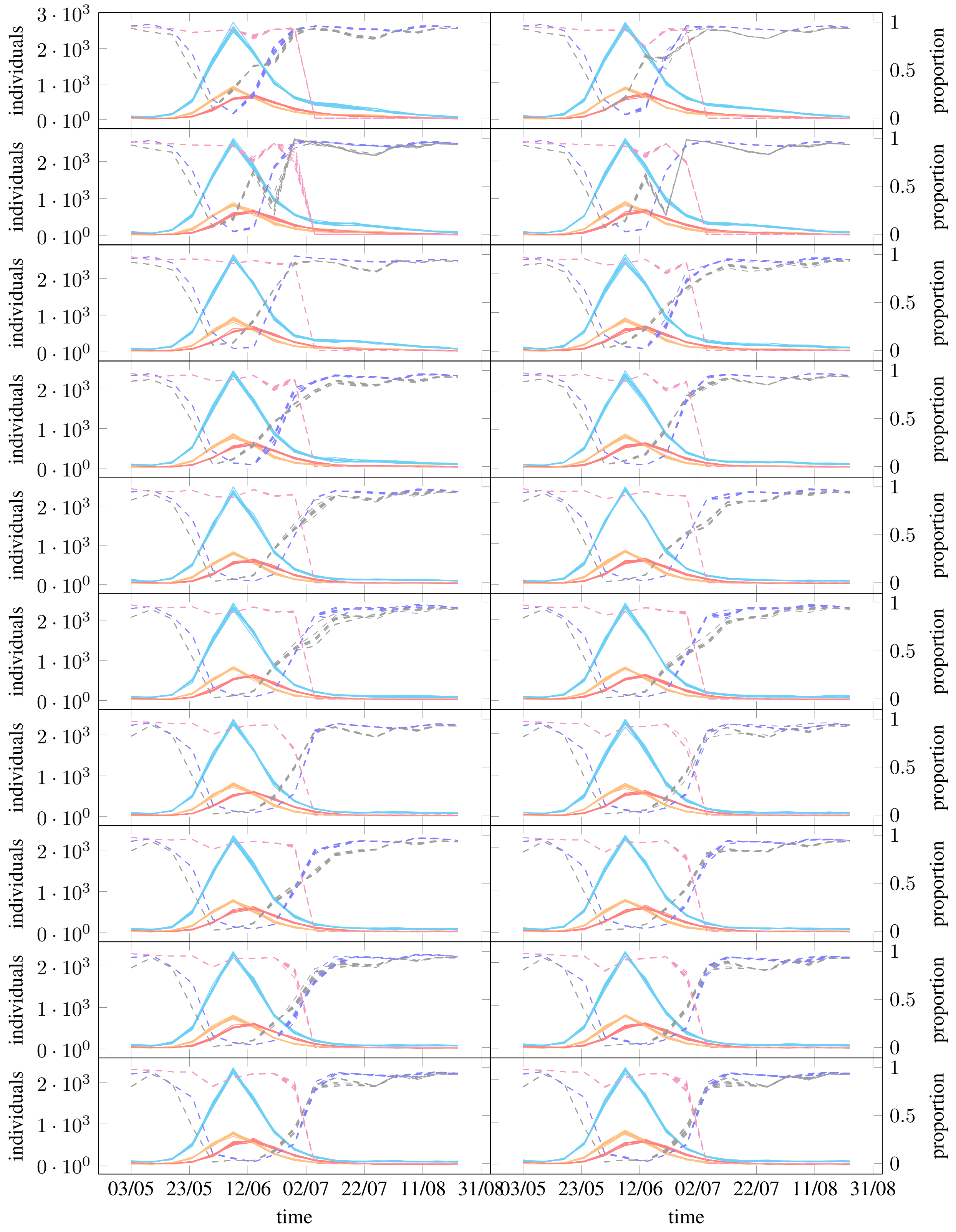

Figure B.33: Execution of policies 40 to 59, with no budget restriction ($b = \infty$).

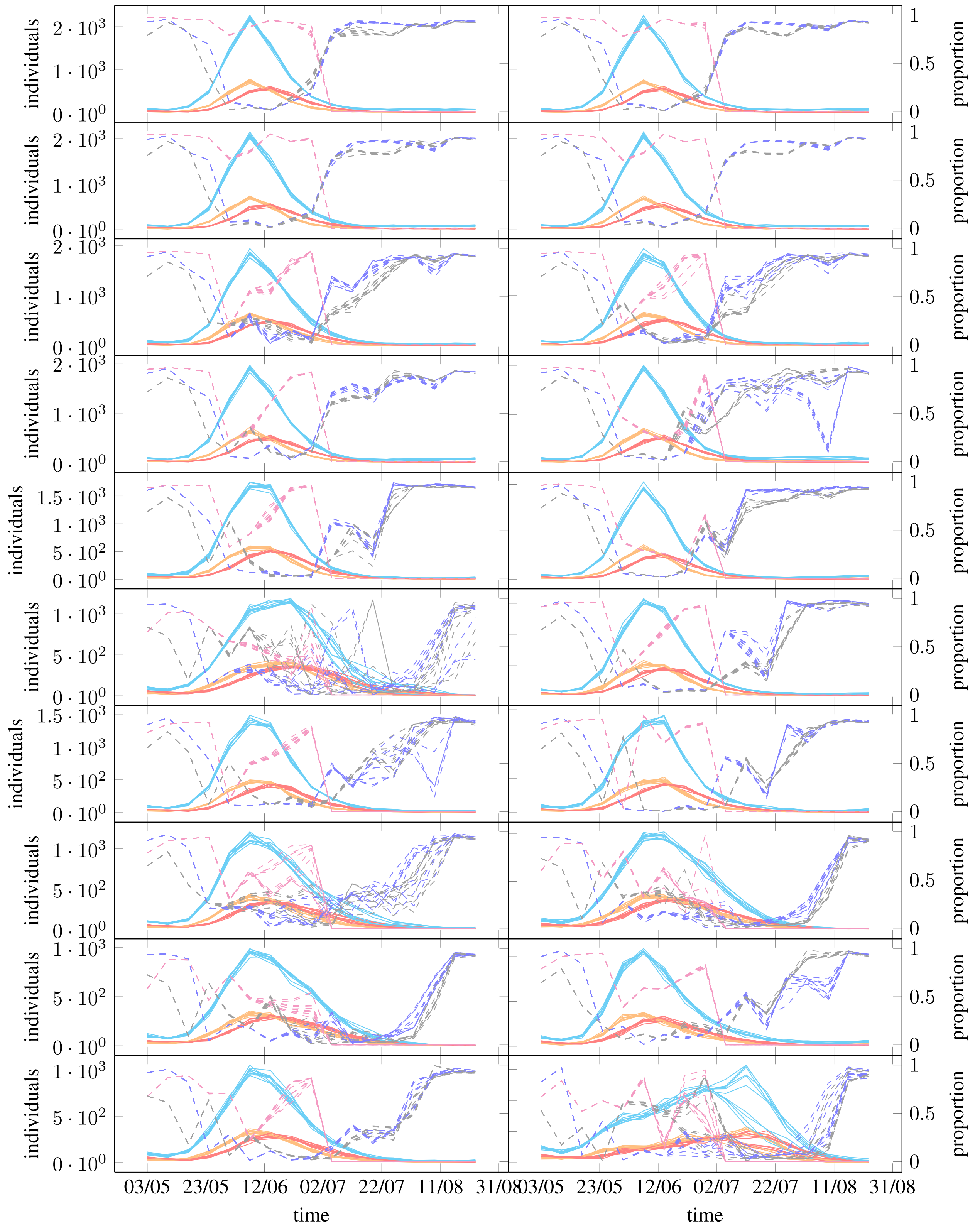

Figure B.34: Execution of policies 60 to 79, with no budget restriction ($b = \infty$).

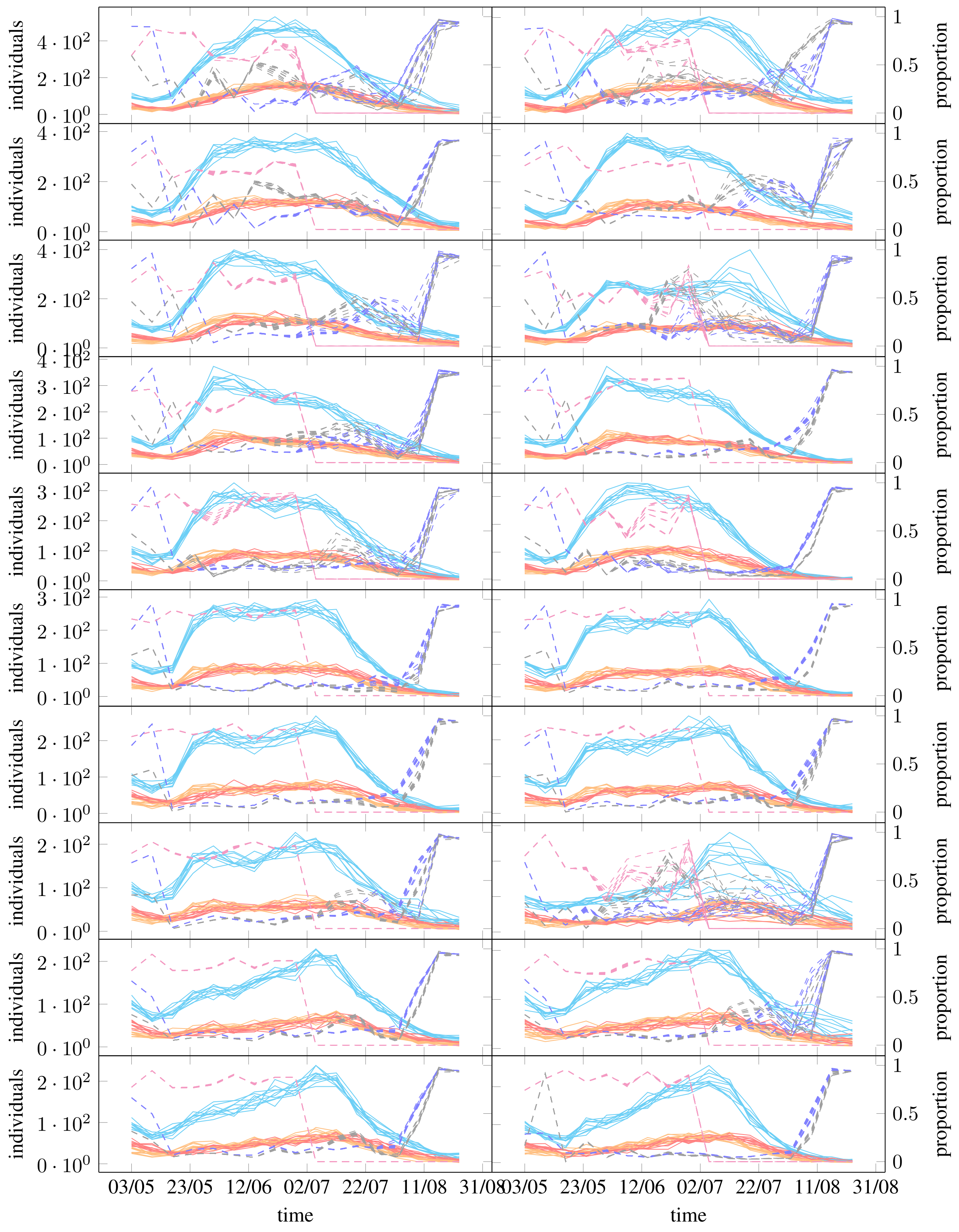

Figure B.35: Execution of policies 80 to 99, with no budget restriction ($b = \infty$).

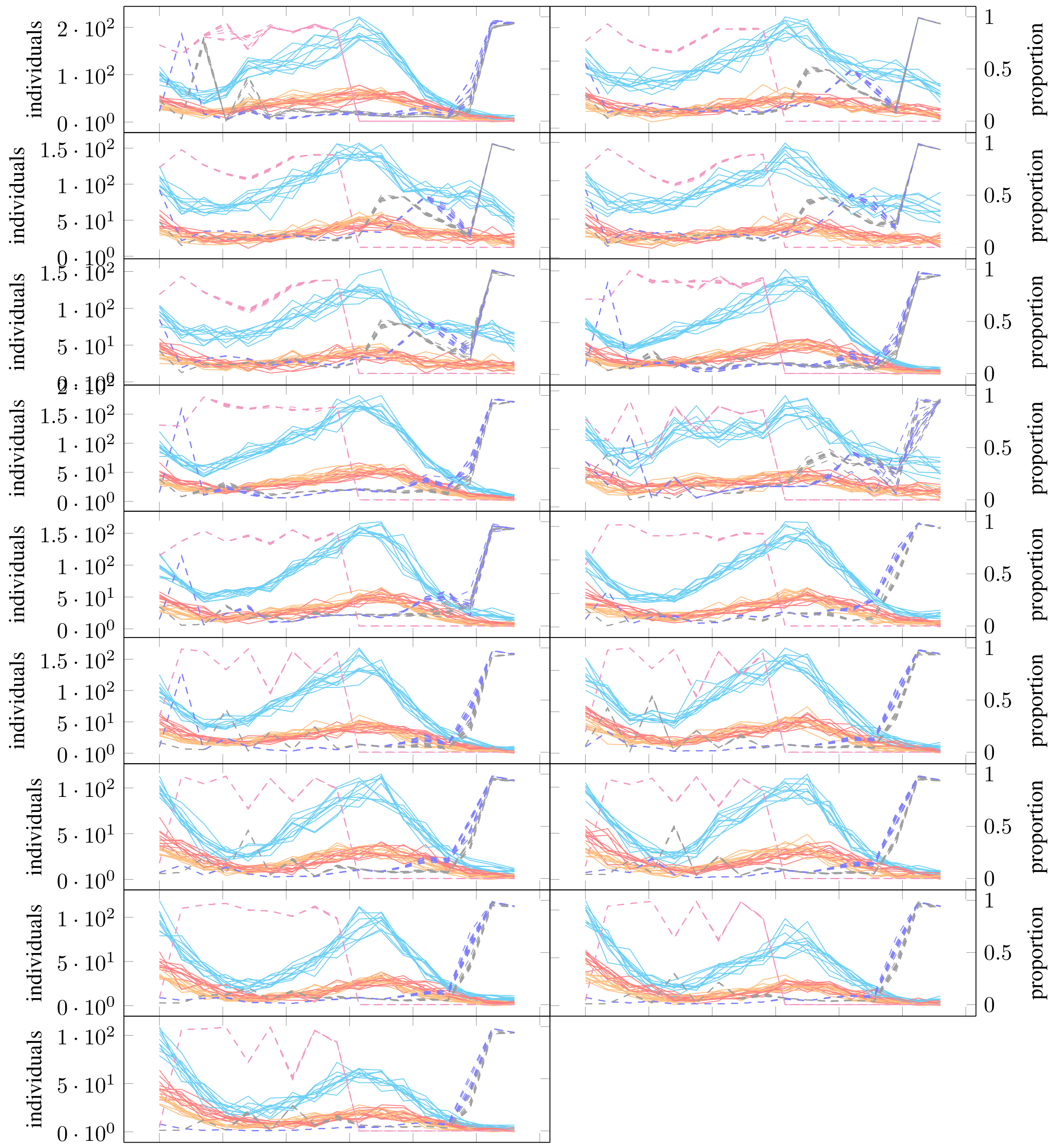

Figure B.36: Execution of policies 100 to 116, with no budget restriction ($b = \infty$).