Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

Elsevier hereby grants permission to make all its COVID-19-related research that is available on the COVID-19 resource centre - including this research content - immediately available in PubMed Central and other publicly funded repositories, such as the WHO COVID database with rights for unrestricted research re-use and analyses in any form or by any means with acknowledgement of the original source. These permissions are granted for free by Elsevier for as long as the COVID-19 resource centre remains active.
An investigation of safe and near-optimal strategies for prevention of Covid-19 exposure using stochastic hybrid models and machine learning

Alexander Bilgram, Peter G. Jensen, Kenneth Y. Jørgensen, Kim G. Larsen, Marius Mikučionis, Marco Muñiz *, Danny B. Poulsen, Peter Taankvist

Department of Computer Science, Aalborg University, Denmark

**Abstract**

In this work investigate the use of stochastic hybrid models, statistical model checking and machine learning to analyze, predict and control the rapid spreading of Covid-19. During the pandemic numerous studies using stochastic models have been produced. Most of these studies are used to predict the effect of some restrictions. In contrast, in this paper we focus on the synthesis of strategies which prevent Covid-19 spreading. The computed strategies provide valuable information which can be used by the authorities to design new and more specific restrictions. We consider two large case studies that develop in the Copenhagen area in Denmark. Our experiments show that the computed strategies significantly prevent Covid-19 spreading, and thus provide valuable information e.g. expected social distance to minimize Covid-19 spreading. On the technical side, we demonstrate the applicability of analytical methods for preventing the spreading of Covid-19 in large scenarios.

1. Introduction

The spread of Covid-19 had a tremendous impact on societies around the world: people were restricted from traveling, people were working from home, and wearing face masks was customary today – a few years ago face masks was a rare sight in most western societies – and shops were closed. All of these restrictions were implemented by government to limit the spread of Covid-19 but before such drastic measures were implemented their effect should be assessed. Classic techniques for this is modeling the evolution of a pandemic using compartmental models (using ordinary differential equations) [1,2] and making informed guesses on how different measures affect the parameters of the ordinary differential equation. However, those models lose behavior of individual agents and assume everyone interacts with everyone — which is clearly not true. In contrast, agent-based models, model each individual agent and models its precise movements and interactions. Computer science has a long tradition for making models of computer systems individual processes (its agents) and synthesizing control strategies for them. Our efforts to model cyber–physical systems have also pushed us in the direction of stochastic hybrid games. Thus the modeling language and analyses employed by our tools should be ideal for evaluating different pandemic control mechanisms.

The crisis caused by the spread of Covid-19 has pushed the community to develop numerous models to predict the spreading of Covid-19 or to evaluate the impact of given restrictions e.g. use of masks. However, not much work has been done in the field of automatic generation of restrictions which minimize Covid-19 spreading. For this reason, in this work we focus on the synthesis of control strategies which prevent Covid-19 spreading. The strategies we synthesize can be seen as restrictions or schedules which if implemented by persons would reduce Covid-19 spreading.

In this work, we consider both compartmental models and agent-based models. We present two case studies that develop in the Copenhagen area. (1) Our first case study implements the Danish Serum Institute restriction of one direction walking around the Lakes (Søerne) in Copenhagen. We model the behavior of thousands of persons walking around the Lakes using the tool for microscopic Simulation of Urban Mobility (SUMO) [3]. SUMO allows for microscopic simulation of pedestrians and to add additional simulation attributes such as probabilities for coughing, infection probabilities, wind conditions, etc. We abstract the SUMO simulation to an stochastic hybrid game, then we use the tool Uppaal Stratego to compute a near-optimal strategy which minimizes Covid-19 spreading. (2) Our second case study is based on the fluid compartmental epidemiological model from [4] that describe the itineraries of a family in Copenhagen with a total of 613,288 inhabitants. The family itineraries contain locations which have to be visited within a period of time. Each location have distinct Covid-19 probability rates for being exposed. Therefore, finding appropriate schedules for the itineraries can prevent Covid-19 exposure. For this case study we compute strategies which minimize Covid-19 exposure.

**Keywords:**
- Stochastic hybrid models
- Machine learning
- Strategy synthesis
- Statistical analysis
- Covid-19

---

The project was funded by Poul Due Jensens Foundation, Denmark grant number [2020-030].

* Corresponding author.

E-mail address: maniz@cs.aau.dk (M. Muñiz).

https://doi.org/10.1016/j.dajour.2022.100141

Received 5 September 2022; Received in revised form 8 November 2022; Accepted 8 November 2022

Available online 17 November 2022

2772-6622/© 2022 The Authors. Published by Elsevier Inc. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).
(near-optimal), strategies which guarantee that certain locations are periodically visited (safe), and strategies that guarantee visiting locations while preventing Covid-19 exposure (near-optimal and safe).

For both our case studies, we perform an extensive evaluation of the strategies that we compute. We run experiments involving thousands of simulations in hundreds of CPUs, such computation took several days. Our results show that implementing the computed strategies can greatly prevent Covid-19 spreading. In addition, our experiments confirm that it is possible to compute strategies effectively for such real world based case studies involving thousands of agents. In particular for the Lakes case study we are able to compute strategies in real-time, for this we use techniques from [5] to periodically compute online strategies online for a short horizon.

1.1. Related work

Statistical methods and tools are playing a major role in the study of Covid-19 developments. As such there exist numerous studies and tools. In [6] a decision framework to help software startups to increase their chance of success during the Covid-19 pandemic is proposed. The overall goal of that work is thus different than ours. However, their proposed framework which uses the Nominal Group Technique methodology, could profit from using the techniques we propose to synthesize strategies which could aid software start ups. The work in [7] studies a plan for minimizing the costs of distributing Covid-19 vaccines in the European Union (EU). The plan is then obtained by solving a mix integer program. While that work focuses on minimization of costs, our work focuses on minimization of Covid-19 spreading. The techniques that we present are complementary, because the vaccine distribution problem could be represented using stochastic hybrid models which will allow a more refined description of the vaccine distribution process. Then one could compute safe and near-optimal strategies for minimization of costs and ensuring guarantees on the distribution of Covid-19 vaccines. In [8] A multi criteria evaluation methodology for assessing the impact of Covid-19 in EU countries is presented. This work is orthogonal to our. However, we can consider their results to construct more accurate models, or to calibrate our models. The study in [9] considers the impacts of Covid-19 in the behavioral change of passengers. In particular it studies the attitudes of passengers towards preventive measures. The results from that work could be included in our work to refine our models. For example, the study measures the compliance of passengers with respect to social distancing. We could use their measures to calibrate our models. In [10] the authors evaluate the effect of restriction policies e.g. school closures in Australia. Similarly [11] uses agents to evaluate Covid-19 development in Iran. In [12] compartmental model is used to study the spread of Covid-19 in Wuhan. Other works focus on the development of dedicated tools for performing simulations e.g. Covasim [13]. The key difference between our work and studies similar to the ones mentioned above, is that we synthesize strategies where as the other approaches evaluate different scenarios. Our work builds directly on previous Covid-19 models from [4] and techniques for computing near-optimal strategies from [5]. In [14] a predictive analytics model for Covid-19 pandemic using artificial neural networks is presented. The work is focused on prediction where as in our work we focus on optimization. In our second case study we implement the SEIHHR (Susceptible Exposed Infected Hospitalized Recovered) epidemiological model. During the Covid-19 pandemic, the mathematical epidemiology community have produced a number of other relevant models. The work in [15] presents generalizations of the SIR model e.g. the SIR-PH model. In [16] the authors present the SEIHQRD model which adds the critically ill class whereas the work in [17] SEIAHR includes the asymptomatic class. Note that these works are complementary, since we could use them as the underlying epidemiological model.

Structure of the paper. The structure is as follows, In Section 2 we give an overview of the underlying models and tools which we use in our work. Section 3 we present case study related to computing near-optimal strategies for walks around the Lakes. In Section 3.1 we give a detailed description of the scenario. In Section 3.2 we describe the tools and preliminaries required for this case study. Sections Section 3.3 and Section 3.4 describe the models and the controllers that we use in this case study. Section 3.5 presents our experimental results for this case study. Our second case study involving the family itineraries in Copenhagen is presented in Section 4. Section 4.1 describes in detail the case study. Section 4.2 describes the tools and methodology used in this case study. Later Section 4.4 presents our experimental results for this case study. Finally Section 5 presents our conclusion.

2. Preliminaries

2.1. Computing Strategies with Uppaal

Uppaal [18,19] is a family of software tools used for the design, analysis and verification of complex systems with wide application in industrial scenarios. Uppaal Stratego [20] is a branch of Uppaal which can synthesize strategies for complex systems and has been successfully used in several industrial case studies [21–23], and in this work to minimize the spreading of Covid-19.

The models that we use in this work are large, complex, and optimized for scalability. Therefore, in order to introduce the models and functionality of Uppaal we present a small example. Consider Fig. 1 which is a Uppaal Stratego model implementing a Stochastic Hybrid Game [5,24] between a controller and the environment. A Strategy indicates the controller which actions to take in order to minimize a certain function. In the rest of this work we use the Greek letter $\sigma$ to refer to strategies.

The model represents an agent (the controller — solid arrows) which needs to commute from Home to Work. The agent can go by Bus or by Train. If the agent chooses to go by bus there is a probability weight of 5 that the bus is Full and a probability weight of 1 that the bus is almost Empty. Note that taking the bus takes up to 20 min indicated by the invariant $T=28$. Additionally, there is a risk for the agent to become infected. The expression $risk' = 4*\beta$ is the derivative of the variable risk with respect to time. Thus if the bus is full, the likelihood for the agent to become infected is 4 times higher than when the bus is empty. Similarly if the agent decides to take the Train there is probability weight of 2 for the agent to miss the train and to wait for $T=30$ time units. On the contrary the agent takes the train and reaches work in at most $T=15$ time units. Note that there is a risk for the agent to get infected with derivative $risk' = 4*\beta$.

There are a number of interesting questions that we would like to ask the model. For example, Would it be possible to go with certainty from Home to Work within 50 time units?. This could be answered by computing a safe strategy. Another question could be, Would taking the bus or the train minimize the risk of infections?. This could be answered by computing a near-optimal strategy. In fact approximated expected value using 1000 simulations for variable risk under a near-optimal strategy is $16.75 \pm 0.5$, in contrast the approximated expected value is $24.8 \pm 1.06$. Finally, it would be interesting to minimize the risk of infection while guaranteeing that the agent reaches work within a time bound. This can be answered by computing a safe and near-optimal strategy. In Section 3 we compute near optimal strategies, and in Section 4 we compute safe and near-optimal strategies to prevent Covid-19 spreading.

2.2. SUMO

SUMO [25] is an open source tool which allows to model and simulate traffic and pedestrian dynamics at a microscopic level. It network transformation, waiting time calculations, traffic light performance, etc. SUMO provides features for modeling a vast number of scenarios and
Fig. 1. Stochastic hybrid game representing an agent (the controller — solid arrows) which needs to commute from Home to Work in a given environment (the opponent — dashed arrows). A strategy can be used to minimize the risk of becoming infected.

Fig. 2. Rate diagram of the basic compartmental SEIHR model.

possibilities to inter-operate with other tools. There is also a wide active community which offer support.

In this work we mainly use the following SUMO components. Netconvert to import the Copenhagen Lakes from open street maps. Road networks which allow to model the relevant part of the map, roads lanes and intersections of the Lakes. Traci a software interface that gives access to objects in the running simulation, and enables communication with UpPaal Stratego.

2.3. SEIHR compartmental model

Compartmental models is a modeling technique applied in epidemiology. A simple compartmental model is the SIR [15] susceptible (S) being those that can affected by the disease, infectious (I) being those that have the disease and can infect others, recovered/removed (R). In both of our case studies we use a more refined compartmental model i.e. the SEIHR compartmental fluid model, where the population is divided in susceptible, exposed (E) being those that have the disease but not infectious, infectious, recovered/removed (R), and hospitalized (H). The rate diagram is given in Fig. 2. The arrow $E \xrightarrow{\alpha} I$ means a conversion from $E$ to $I$ with a rate $\alpha$ multiplied by the number of $E$ elements. Similarly for the other arrows. A major challenge consist in calibrating the model using appropriate rates. We use the rates as presented in previous work [4] which where provided by the Novo Nordisk Fonden project that fitted Danish data to a SEIHR model and estimated the parameters for Danish conditions.

3. Near-optimal strategies for walks around the lakes

In this case study we consider the one direction walking restriction in the Lakes (Søerne) in Copenhagen c.f. Fig. 3 The restriction was given by Statens Serum Institut (SSI¹). We construct an agent based model which considers the geographical location and the stochastic behavior of pedestrians. Our model can be seen as an stochastic hybrid game in which the controllable modes include the choice of allowing an agent to enter the lakes. Our goal is to compute a near-optimal strategy which minimizes the number of Covid-19 exposures by maintaining social distancing.

3.1. Scenario description

The Copenhagen lakes are a popular location for pedestrians in Copenhagen, and therefore also a potential hot spot for Covid-19 exposure. Recently, Copenhagen municipality introduced a one-way walk restriction around the lakes to help minimize Covid-19 exposure. In addition to this restriction we make the assumption that when a pedestrian wish to enter the Lakes for a walk she will request access to the Lakes, and she will not enter until the request has been granted. In reality, this assumption could be implemented by using pedestrian traffic lights at different points around the Lakes. Once a request has been issued, a controller is responsible to grant or deny access. For this scenario we implement several controllers described in Section 3.4.

3.2. Tools and methodology

To model the scenario as close to reality as possible we use the microscopic simulation tool SUMO [3,25] c.f. 2.2 where we consider a number of parameters such as, initial number of infected pedestrians, coughing probabilities, wind direction, different speeds for pedestrians, etc.

The architecture of our system is given in Fig. 4. The SUMO model can be seen as an Stochastic Hybrid Game where the controllable modes correspond to allowing an agent to enter the Lakes, and the uncontrollable modes correspond to the environment which includes the stochastic movement of the agents and their coughing probabilities. Ideally,

¹ https://www.ssi.dk
we would like to compute a strategy \( \sigma^H \) for a long horizon \( H \) (which could be a full day or week). Unfortunately, computing such a strategy is unfeasible given the that the number of choices grow exponentially with the horizon. To overcome this problem we apply the following methodology.

### 3.2.1. Online strategy synthesis

For this case study our goal is to compute a strategy (controller) \( \sigma^H \) to prevent Covid-19 infections for a long horizon \( H \). As the number of choices for the controller grows exponentially in the horizon, computing the strategy for a long horizon \( H \) is intractable. To overcome this problem we resort to the Online Strategy Synthesis \([5]\) methodology, where we periodically compute a near-optimal strategy \( \sigma^h \) for a short horizon \( h < H \). The strategy \( \sigma^h \) is less accurate than the strategy \( \sigma^H \) but it can be computed effectively and periodically. The methodology has successfully been applied to multiple case studies involving cyber-physical systems such as, intelligent traffic lights \([23]\), floor heating systems \([5]\), rerouting \([26]\) etc.

Fig. 4 depicts the architecture of our system. Whenever an agent requests to enter the Lakes, the status of the SUMO simulation is passed to a controller, the controller after some computation, returns a strategy \( \sigma^h \) which SUMO implements to continue with the simulation. For this case study we implement and compare several controllers, in particular we implement a UPPAAL STRATEGO controller.

### 3.3. Model description

The lakes can be seen from a birds-eye view in Fig. 3(a). The lakes are split into multiple smaller lakes with walking areas around and between all lakes. A SUMO recreation of the lakes can be seen in Fig. 3(b). All vehicular simulation has been omitted to simplify the scenario.

The full Copenhagen lakes are modeled in SUMO. However, simulating the complete network may include tens of thousands of pedestrians and cause computational problems. As such, as a proof of concept we focus only on the southernmost lake, the SUMO network recreation is seen in Fig. 3(c), in which we also see a large number of pedestrians walking around the lake.

We assume there to be some controller allowing or disallowing pedestrians to enter the network e.g. a pedestrian traffic light, such that any pedestrian (agent), who wants to enter the network sends a request to enter the network at some entry point to the controller. The controller then answers by either allowing the agent to enter or not. Furthermore, we assume some of the agents to be infected with Covid-19 and cough under a given distribution. If an infected agent coughs, given the wind conditions, the viral load is spread in a cone. If the cough overlaps with another agent the viral load of the agent is increased and thus its chance to be infected as well. This scenario can be seen in Fig. 5 in which the cough is modeled as a triangle from the agent.
3.4. Controllers

In our scenario, we have several entry points to the lakes. For every entry point, pedestrians make request to enter the Lakes. It is responsibility of the controller to allow or disallow an agent to enter. We make use of three different controllers, note that these controllers do not have access to SUMO variables such as, walking speeds, or wind speed. Algorithm 1 gives a high level idea of the interactions taking place within the SUMO model and the different controllers. Line 4 of Algorithm 1 uses one of the controllers described below to decide if an agent is allowed or not to enter.

3.4.1. Default controller

This controller models the usual behavior of pedestrians wishing to enter the lakes i.e. the controller allows agents issuing enter requests to enter immediately. This serves as a baseline for simulating the scenario with no external input.

3.4.2. Naive controller

This controller is used to model social distancing, an agent prior to enter the Lakes, looks a number of meters around her. If there is enough room, the controller allows the agent to enter. Different social distancing values will result in different controllers. The social distance used for our simulations is 5 meters.

3.4.3. Uppaal Stratego controller

The controller which implements the online strategy synthesis methodology explained in Section 3.2.1. The controller computes online near-optimal strategies for whether or not to allow an agent to enter the network c.f. Section 2.1.

The corresponding Uppaal Stratego model is given in Fig. 6. Note that the Uppaal Stratego model is an abstraction of the SUMO model, and will be used to perform reinforced learning. Following Algorithm 1, line 4 will create an instance of this model using the current observations from the SUMO model e.g. positions of the walking agents. Then Uppaal Stratego is going to be called to compute a strategy for each agent.

In the following we give a detailed description of the Uppaal Stratego model. The controller is initialized with the known information of the world such as pedestrian locations, current requests, etc. The initialization is performed by an external C++ library called by functions restart_world() and initvars(). The Uppaal Stratego functionality to load libraries is recent and allows easy implementation of tasks such as parsing files, etc. In this case the library parses a JSON file with thousands of lines with the observable state of the SUMO simulation. The observable states includes current positions of agents, but it does not include their speeds, or if they are infected.

After initialization, at location SelectAction the action of the agent is chosen as either entering the network or not entering the network. If the controller chooses not to enter a penalty is increased depending on how many agents are waiting to enter. In this way, the penalty can be used to force agents into the Lakes. However, if the penalty is too high, the number of Covid-19 exposures might increase.

At the next location there are three choices with probability weights 3, 6, 3. These choices are used to coarsely approximate the unknown walking speeds of the agents in SUMO.

The next transition set the chosen values for the agent in the C++ library. Finally the function do_sim_step calls the C++ library to simulate the choice for some time horizon and calculates the penalty of the choice. In this step we use the C++ library to compute the posterior of the abstract world where e.g. agents are represented as circles moving at the chosen speeds. Since we have hundreds of agents and the learning methods of Uppaal Stratego are going to call this function hundreds of times, this step needs to be implemented as efficient as possible. Therefore, having an optimized compiled library is quite convenient in comparison to an implementation in Uppaal Stratego which would be interpreted at running time.

Given this model, every time there is an agent request to enter the lakes, Uppaal Stratego executes hundreds of simulations to compute a near-optimal strategy which minimizes the variable penalty, and thereby the possible Covid-19 exposure.

Table 1 describes the main parameters that we use for computing SUMO simulations. The wind is a two dimensional vector which influences the length of the triangle which can infect other agents, the width of the triangle is given by the constant triangle degree. Every agent has a variable accumulating the viral load to which the agent has been exposed, if the viral load exceeds the constant infection threshold, the agent becomes infected. At every simulation step, agents have a cough probability. The SUMO model for pedestrians accepts a number of parameters, we use the default parameters but we use only three different walking speeds.

### Table 1: Relevant parameters for simulations.

| Parameter          | Value                  |
|--------------------|------------------------|
| wind north         | 5 m/s                  |
| wind east          | 5 m/s                  |
| infection threshold| 0.3                    |
| cough probability  | 0.1                    |
| viral load cough   | 0.05                   |
| triangle degree    | 40°                    |
| walking speeds     | 0.7, 0.9, 1.2          |

![Fig. 5](image_url) SUMO simulation, infected agent coughing. A red cone induced by the wind direction is formed. An agent intersecting the cone increases its viral load.
### Table 2
Results København Lakes. Every row is computed using 30 simulations per controller. \( \bar{M}[\text{inf}] \) denotes the arithmetic mean of the new infected, similarly \( \bar{M}[\text{steps}] \) denotes the arithmetic mean of simulation steps.

| Scenario | Controller | Default | Stratego | Naive |
|----------|------------|---------|----------|-------|
|           | inf Pr.    | \( \bar{M}[\text{inf}] \) | \( \bar{M}[\text{steps}] \) | \( \bar{M}[\text{inf}] \) | \( \bar{M}[\text{steps}] \) |
| 100       | 0.01       | 1.70    | 1872.67  | 1.53  | 3017.70  | 1.73  | 1890.40 |
|           | 0.05       | 11.27   | 1897.07  | 7.30  | 3035.00  | 10.43 | 1887.60 |
|           | 0.10       | 26.23   | 1903.83  | 15.77 | 3084.43  | 25.13 | 1907.27 |
| 1000      | 0.01       | 1.17    | 2531.10  | 6.93  | 3318.53  | 1.87  | 2536.90 |
|           | 0.05       | 9.93    | 2518.73  | 7.00  | 3317.23  | 6.93  | 2543.60 |
|           | 0.10       | 17.90   | 2494.10  | 12.67 | 3371.20  | 19.23 | 2559.90 |
| 2000      | 0.01       | 1.27    | 3473.20  | 1.07  | 3626.33  | 1.87  | 3449.47 |
|           | 0.05       | 7.40    | 3493.43  | 6.97  | 3637.80  | 7.60  | 3393.87 |
|           | 0.10       | 16.80   | 3333.90  | 16.13 | 3699.30  | 14.40 | 3464.63 |
| 600       | 100        | 0.01    | 45.23    | 2028.40| 11.90   | 6947.53| 29.77  | 2584.83 |
|           | 0.05       | 259.73  | 2043.23  | 72.03 | 3637.80  | 7.60  | 3393.87 |
|           | 0.10       | 429.73  | 2007.40  | 149.03| 3371.20  | 19.23 | 2559.90 |
| 1100      | 100        | 0.01    | 29.70    | 2748.83| 12.57   | 7281.10| 25.17  | 3056.17 |
|           | 0.05       | 172.67  | 2818.53  | 63.40 | 7320.37  | 6.93  | 2543.60 |
|           | 0.10       | 331.50  | 2804.80  | 137.57| 7405.07  | 19.23 | 2559.90 |
| 1600      | 100        | 0.01    | 21.13    | 3701.17| 13.73   | 7821.50| 14.40  | 3699.47 |
|           | 0.05       | 113.93  | 3649.43  | 60.13 | 7687.73  | 7.60  | 3393.87 |
|           | 0.10       | 238.57  | 3729.67  | 126.37| 7814.73  | 19.23 | 2559.90 |
| 2100      | 100        | 0.01    | 175.10   | 2079.20| 30.31   | 8981.42| 69.27  | 2584.83 |
|           | 0.05       | 771.03  | 2087.33  | 169.57| 8874.30  | 467.27| 3331.30 |
|           | 0.10       | 977.80  | 2083.73  | 364.80| 8892.97  | 366.63| 3841.23 |

3.5. Experiments

For our experiments we conduct 30 simulations for each scenario for every controller. Every simulation is run using the same seed for each controller, meaning each controller is given the exact same simulation configuration. A scenario consists of a number of persons that wish to enter the network in a given time \( \text{spawn} \) where every person is infected by a given probability \( \text{inf Pr.} \). We conduct an experiment for each combination of persons in 100, 600, 1100, 1600 and 2100, the time \( \text{spawn} \) include 100, 1000 and 2000 seconds, and \( \text{inf Pr.} \) includes 0.01, 0.05 and 0.10.

The results can be seen in Table 2. The column Scenario corresponds to the combinations of persons, \( \text{spawn} \), and \( \text{inf Pr.} \). Since we run 30 simulations for each scenario, the column Controller shows for...
every controller in the column $M[\text{inf}]$ the arithmetic mean of the new infected variable, analogously the column $M[\text{steps}]$ is the arithmetic mean of the number of steps for the simulation.

We see that both for the Naive controller and the \textsc{Uppaal Stratego} controller the computed mean results in much less infected compared to the Default controller. We also see that when the scenarios become larger the \textsc{Uppaal Stratego} controller generally minimizes Covid-19 infections by $\sim 50\%$ compared the Naive controller and even more compared to the Default controller. However, the trade off seems to be the simulation steps expended on completing the simulations. That is in order to minimize the exposures \textsc{Uppaal Stratego} distributes pedestrians conservatively at the cost of time. As such, the waiting time (number of simulations steps) for entering the Copenhagen lakes using the \textsc{Uppaal Stratego} controller is greater than both the Naive and the Default controller. As described in Section 3.4.3, the \textsc{Uppaal Stratego} controller applies a penalty to force pedestrians to enter the Lakes. This penalty could be increased to reduce the waiting times at the cost of potentially more Covid-19 exposure.

To illustrate our results Fig. 7 selects the scenarios where we fix the initial infected probability inf$_{Pr}$ to 0.1, and time spawn to 100. Fig. 7(a) shows the boxplots for the number of new infected. We observe that the arithmetic mean of new infected is less for \textsc{Uppaal Stratego} than for the other controllers. In contrast Fig. 7(b) shows the number of simulation steps needed until the last agent has walked around the lakes. This shows that objectives new infected and number of simulation steps (agent waiting time) are conflicting objectives. For instance we could increase the social distancing parameter of the Naive controller to 10 m, which we expect will reduce the expected number of infected but increase the simulation steps. Similarly we can modify the penalty function that \textsc{Uppaal Stratego} optimize. A next step could consider in finding Pareto optimal solutions.

![Diagram](https://via.placeholder.com/150)

**Fig. 6.** \textsc{Uppaal Stratego} controller.

4.1. Scenario description

We give a brief description of the scenario. The family consist of three members, a mother, a father, and a son. The family lives in Copenhagen with about 613,288 inhabitants, the members of the family spend time at work, school and occasionally at leisure activities. The following locations in Copenhagen are of interest for the family, the city of Copenhagen, Parken, ITU, Maersk, Vesterbro Ny School and Nyhavn. Each location have their own SEIHR ODE-based model c.f. Section 2.3. Each location has a population-size as well as a specific transition-rate for flow between susceptible (S) and exposed (E) reflecting the differences in being exposed at various locations.

Every family member have associated leisure activities, in particular Father is fanatic about FCK (FC København) and has season tickets for home matches at Parken (capacity of 38,000 spectators). In order to preserve the happiness of the father it is imperative that

Happy : Father visits Parken a number of times in a given period.

In the rest of this section we will use happy to refer to the above safety property.

4.2. Tools and methodology

As in the previous case study, the central formalism we use is a stochastic hybrid game $H$ between a controller and the environment. In this game, a strategy $\sigma$ indicates the controller which actions to take in order to minimize or maximize a given function. We use $H[\sigma]$ to describe the resulting stochastic hybrid game where the choices of the controller are restricted by $\sigma$.

The \textsc{Uppaal} family provides a number of tools for computing strategies in different models. Fig. 8 [27] describes the workflow on the different models and strategies that we compute. First the case study is modeled as a stochastic hybrid game $H$ where the controllable actions correspond to the itinerary choices of an agent, and the uncontrollable actions correspond to the environment. The detailed stochastic hybrid game $H$ can be abstracted to a timed game [28] $\zeta$ where probability distributions have been abstracted to their support and non-linear constraints have been over-approximated. By using \textsc{Uppaal Tiga} on the timed game $\zeta$ one can compute the most-permissive strategy $\sigma$ with respect to a property $\varphi$. Note that executions of the game $H[\sigma]$ are guaranteed to satisfy the property $\varphi$. However $\sigma$ is not optimal and can be further refined. \textsc{Uppaal Stratego} will use machine learning and statistical methods to produce the safe and near-optimal strategy $\sigma^\ast$. The executions in the game $H[\sigma^\ast]$ are guaranteed to satisfy the property $\varphi$ while minimizing a certain cost function. Finally, note that is possible to directly compute a near-optimal strategy which will not provide any
guarantees. In the following we describe the strategies that we compute for this concrete case study.

Minimize A first approach consist on synthesizing a strategy which minimizes the Covid-19 exposure of Father during his daily activities. For this we can use Uppaal Stratego to synthesize a strategy $\sigma_{\text{min}}$ such that behaviors restricted to the strategy i.e. $H|\sigma_{\text{min}}$ minimize Covid-19 exposure. Note that behaviors in $H|\sigma_{\text{min}}$ may violate safety property Happy, i.e. behaviors which are contra-productive to the well-being of Father.

Guarantee Let us consider the case when we would like to ensure the property Happy i.e. Father visits Parken a number of times in a given period. For this the model $H$ is abstracted to a timed game $G$ where Uppaal Tiga computes a permissive strategy $\sigma$ which guarantee property Happy. Note that $H|\sigma$ include behaviors where the father is heavily exposed to Covid-19.

Guarantee and Minimize Ideally we would like to preserve the happiness level of Father while minimizing its Covid-19 exposure. Towards this goal the stochastic hybrid game under the permissive strategy $H|\sigma$ can be further restricted to behaviors which minimize Covid-19 exposure. To obtain this we use Uppaal Stratego to produce a safe and near-optimal strategy $\sigma'$. Note that all behaviors in $H|\sigma'$ satisfy Happy and attempt to minimize the risk of Covid-19 exposure for Father.

### 4.3. Model description

With the purpose of illustrating how strategy synthesis techniques and the Uppaal family of tools has been used to model, analyze and synthesize a variety of scenarios relevant for Covid-19 spreading, we transform of the model presented in [4] Section 4 to a stochastic hybrid game. Our transformation consist in syntactically declaring specific edges as controllable and the rest as uncontrollable. To illustrate this consider the itinerary of Father presented in Fig. 9(a) where solid edges correspond to controllable actions and dashed edges are uncontrollable actions.

The itinerary describes the whereabouts of Father on daily basis. He is at Home from 23:00 at night to 07:00 in the morning. In order to go to work ITU, he commutes between 07:00 to 08:00 through a part of Copenhagen Copen1. He stays at work from 08:00 to 16:00. After work at location ITU the Father has three controllable choices (solid edges), he can go to location NyHavn via Copen4, he can go to location Parken via Copen2, or he can go directly Home via Copen3. Note that for Father to return Home from locations Parken and NyHavn, Father has to visit extra locations in Copenhagen city area which increase Covid-19 exposure.

The model in [4] has an agent-based SEIHR model to describe the health status of each family member with transition-rates for flow between susceptible (S) and exposed (E). We accumulate the same transition-rates in the observable variable risk, e.g. the ODE $\text{risk} = \beta \cdot \text{risk}_p$ accumulates the transition-rate $\beta$ (c.f. Fig. 2) from susceptible (S) and exposed (E) while at being at location Parken. We use the observable variable risk to guide the strategy synthesis process. Note that there is a one to one correspondence between $\beta$ and risk. Therefore, minimizing risk results in minimizing the probability of Father becoming exposed. In addition, we need to keep track of the number of visits from Father to Parken, for this every time location Parken is visited a counter visit_parken is increased.

Finally, Fig. 9 describes a monitor for the property Happy. At the end of the period the monitor checks if the location Parken has been visited at least $N$ times, if this is not the case the monitor goes to location Error indicating a violation to the property Happy.

### 4.4. Experiments

A stochastic hybrid game is a rich mathematical formalism which enables the design, analysis, and verification of complex systems. Computing strategies for such systems is computationally hard and difficult
to scale. In this section we demonstrate that it is possible to compute such strategies for large real world scenarios. And that our resulting strategies can have a positive effect on preventing Covid-19 spreading.

The property Happy expects a number of visits from Father to location Parken in a given period of time (horizon). For our experiments we consider different number of visits and different periods these can be observed in columns “Desired Visits” and “Period” in Table 3. For every scenario we compute the expected value for the variable risk under the given controller. We consider the following strategies: Random, Minimize, Guarantee, and Guarantee and Minimize. Where strategy Random uses a uniform distribution for the controllable choices at location ITU in the itinerary of Father, and the other strategies are as described in Section 4.2.

The expected risk computed under the strategy Random serve a baseline to compare with the other strategies. We observe that for all scenarios strategy Minimize reduces the risk of exposure of Father however it provides no hard guarantees. The strategy Guarantee has in several scenarios the highest risk of exposure for the Father, however this strategy guarantees the property Happy. Finally, the strategy Guarantee and Minimize which guarantees property Happy has higher risk than strategy Minimize but is less risky than strategies Random and Guarantee. The results suggest that the best itinerary for the father is the one induced by the strategy Guarantee and Minimize.

4.4.1. Experimental evaluation

Table 3 describes the results of our experiments. In general our results are encouraging, to see this consider scenario in the last row with 8 desired visits and a period of 70 days. The sampled expected risk $E[risk]$ for the Random strategy in which the Father makes itinerary choices randomly and uniformly is 92.04. The computed Minimize strategy minimizes the risk to 70.73. Note that strategies Random and Minimize can allow executions in which the desired number of visits

\[ x \leq \text{period} \]

\[ x = 0 \]

\[ 
\begin{align*}
\text{cnt} &< N \\
\text{cnt} &> N \\
\text{cnt} &< N
\end{align*}
\]

\[ \text{Error} \]
are not guaranteed. On the contrary the strategy Guarantee guarantees visits with higher risk 93.14, even higher than the Random controller. Finally, the strategy Guarantee and Minimize yields the value 71.42 which is closed to the minimal observed value while guaranteeing the number of visits. The strategy Guarantee and Minimize contains the itineraries that will enable Father to perform his activities while minimizing the risk of becoming exposed.

All experimental evaluation was run on AMD Opteron 6376 Processors. We use a timeout of 2 days and for every row in Table 3 we use the limits of 28,40,60 80,100,120, and 140 GB of RAM respectively. We observe that as the period (horizon) increases, the computation time for the synthesis of strategies and the memory for storage of strategies increase significantly. Uppaal allows to compute expected values within a confidence interval, for this it uses a parameter $\epsilon$. The results from Table 3 are computed with $\epsilon = 0.05$. We have been able to compute results for the smaller instances with $\epsilon = 0.005$. However, bigger instances timed out.

5. Conclusion

Summary. The Covid-19 disease has a negative impact on individuals, society and economy. Statistical methods are being heavily used to predict or assess different Covid-19 measures. In contrast in this work we focus on the synthesis of strategies which minimizes Covid-19 spreading. The resulting strategies can be used to suggest or refine preventing measures.

To demonstrate the applicability of our methodology we have considered to large case studies. In the first case study we model a hundreds of pedestrians walking with different speeds around the Lakes in Copenhagen. Infected agents can cough which might infect other agents. Agents request to enter the Lakes for a walk, the task of the controller is to grant or deny access. We are able to use Uppaal Stratego to generate online strategies which greatly minimize Covid-19 spreading. In addition, we observe that there is a trade-off between minimizing Covid-19 infections and the time agents need to wait to enter the Lakes. In the second case study we study a family with itineraries in the Copenhagen area. The task of the controller is find schedules which minimize Covid-19 spreading, guarantee itinerary objectives, and both guarantee itinerary objectives while minimizing Covid-19 spreading.

We have used stochastic hybrid games to model both case studies. We use several branches of Uppaal to compute such strategies. We run thousands of simulations on hundreds of CPUs. Our experiments show positive results, they show that it is possible to compute near-optimal strategies almost in real time while safe and near-optimal strategies can still be computed in hours or days. More importantly the computed strategies can greatly help in preventing Covid-19 spreading.

Contributions. Automata theory based modeling, verification and synthesis has been widely used in domains such as control of real timed systems, software verification, etc. Therefore one of the contributions of this paper is that is shows the applicability of formal method techniques in the epidemiological context for large and realistic scenarios. Another contribution is that we show how techniques as described in this work have been used in collaboration with Statums Serums Institute in Denmark to evaluate the impact of given measures. Finally, another important contribution is that our resulting strategies do provide a method to minimize the spreading of Covid-19 and that our experiments show that it is computationally possible to compute such strategies.

Caveats, limits and future work. The main caveat of our approach is that of most model based techniques and it relies with the accuracy of our models and reality. We have used some parameters from the literature and we have performed regression techniques using historical data to calibrate our parameters. The limits of the methods presented in this work are given by the size of the scenario, the accuracy of the strategies, and how realistic the model is. In both of our case studies when we reach large instances, the computations can take days or go out of memory. Similarly, if we increase confidence levels. Future research include efficient modeling and reduction techniques for large stochastic models. We also investigating on techniques for model calibration or learning models using historical data.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] W.O. Kermack, A.G. McKendrick, Contributions to the mathematical theory of epidemics—I, Bull. Math. Biol. 53 (1) (1991) 33-55.
[2] W.O. Kermack, A.G. McKendrick, Contributions to the mathematical theory of epidemics—II. The problem of endemicity, Bull. Math. Biol. 53 (1) (1991) 57-87.
[3] D. Krajzewicz, J. Erdmann, M. Behrisch, L. Bieker, Recent development and applications of SUMO - Simulation of Urban MOBility, Int. J. Adv. Syst. Meas. 5 (3&4) (2012) 128-138.
[4] P.G. Jensen, K.Y. Jørgensen, D. Krajzewicz, D. Krajzewicz, A. Haji, K. G. Larsen, M. Mikučionis, M. Muñiz, D.B. Poulsen, Fluid model-checking in UPPAAL for Covid-19, in: T. Margaria, B. Steffen (Eds.), Leveraging Applications of Formal Methods, Verification and Validation: Verification Principles, Springer International Publishing, Cham, 2020, pp. 385-403.
[5] K.G. Larsen, M. Mikučionis, M. Muñiz, J. Srba, J.H. Taankvist, Online and compositional learning of controllers with application to floor heating, in: M. Chechik, J.-F. Raskin (Eds.), Tools and Algorithms for the Construction and Analysis of Systems, Springer Berlin Heidelberg, Berlin, Heidelberg, 2016, pp. 244-259.
[6] S. Parthasarathy, A decision framework for software startups to succeed in COVID-19 environment, Decis. Anal. J. 3 (2022) 100037.
[7] V. Khodaee, V. Kayvanfar, A. Haji, A humanitarian cold supply chain distribution model with equity consideration: The case of COVID-19 vaccine distribution in the European union, Decis. Anal. J. 4 (2022) 100125.
[8] P. Xidonas, R. Steuer, A multicriteria evaluation methodology for assessing the impact of COVID-19 in EU countries, Decis. Anal. J. 4 (2022) 100123.
[9] N. Aydin, A.O. Kuşakcı, M. Deveci, The impacts of COVID-19 on travel behavior and initial perception of public transport measures in Istanbul, Decis. Anal. J. 2 (2022) 100029.
[10] S.L. Chang, N. Harding, C. Zachreson, O.M. Cliff, M. Prokopenko, Modelling transmission and control of the COVID-19 pandemic in Australia, Nature Commun. 11 (1) (2020) 5710.
A. Bilgram, P.G. Jensen, K.Y. Jørgensen et al. Decision Analytics Journal 5 (2022) 100141

[11] N. Mahdizadeh Gharakhanlou, N. Hooshangi, Spatio-temporal simulation of the novel coronavirus (COVID-19) outbreak using the agent-based modeling approach (case study: Urmia, Iran), Inform. Med. Unlocked 20 (2020) 100403.

[12] F. Ndaïrou, I. Area, J.I. Nieto, D.F. Torres, Mathematical modeling of COVID-19 transmission dynamics with a case study of Wuhan, Chaos Solitons Fractals 135 (2020) 109846.

[13] C.C. Kerr, R.M. Stuart, D. Mistry, R.G. Abeywardana, K. Rosenfeld, G.R. Hart, R.C. Núñez, J.A. Cohen, P. Selvaraj, H. George, S. Kelly, C.S. Bennett, B. Wagner, S. Chang, A.P. Oron, E. Wenger, J. Panovsky-Griffiths, M. Fadulare, D.J. Klein, Covasim: an agent-based model of COVID-19 dynamics and interventions, MedRxiv (2021).

[14] Y. Kuvvetli, M. Deveci, T. Paksoy, H. Garg, A predictive analytics model for COVID-19 pandemic using artificial neural networks, Decis. Anal. J. 1 (2021) 100007.

[15] F. Avram, R. Adenane, D.I. Ketcheson, A review of matrix SIR arino epidemic models, Mathematics 9 (13) (2021).

[16] A. Ramos, M. Ferrández, M. Vela-Pérez, A. Kubik, B. Ivorra, A simple but complex enough SIR type model to be used with COVID-19 real data. Application to the case of Italy, Physica D 421 (2021) 132839.

[17] M. Prague, L. Wittkop, A. Collin, D. Dutartre, Q. Clairon, P. Moireau, R. Thiébaut, B.P. Hejblum, Multi-level modeling of early COVID-19 epidemic dynamics in french regions and estimation of the lockdown impact on infection rate, MedRxiv (2020).

[18] G. Behrmann, A. David, A tutorial on Uppaal, Lecture Notes in Comput. Sci. (2004) 200–236.

[19] A. David, K. Larsen, A tutorial on Uppaal, Lecture Notes in Comput. Sci. (2004) 200–236.

[20] A. David, K.G. Larsen, M. Mikučionis, J.H. Taankvist, On time with minimal expected cost! in: F. Cassez, J.-F. Raskin (Eds.), Automated Technology for Verification and Analysis 12th International Symposium, ATVA 2014, Sydney, NSW, Australia, November 3–7, 2014, Proceedings, Springer International Publishing, Cham, 2014, pp. 129–145.

[21] K.G. Larsen, M. Mikučionis, J.H. Taankvist, Safe and optimal adaptive cruise control, in: R. Meyer, A. Platzer, H. Wehrheim (Eds.), Correct System Design: Symposium in Honor of Ernst-RÜDiger Olderog on the Occasion of His 60th Birthday, Oldenburg, Germany, September 8–9, 2015, Proceedings, Springer International Publishing, Cham, 2015, pp. 260–277.

[22] K.G. Larsen, M. Mikučionis, M. Muizs, J. Srb, J.H. Taankvist, Online and compositional learning of controllers with application to floor heating, in: M. Chechik, J.-F. Rankin (Eds.), Tools and Algorithms for the Construction and Analysis of Systems: 22nd International Conference, TACAS 2016, Held As Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, the Netherlands, April 2–8, 2016, Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, 2016, pp. 244–259.

[23] A.B. Eriksen, C. Huang, J. Kildebogaard, H. Lahrmann, K.G. Larsen, M. Muniz, J.H. Taankvist, Uppaal stratego for intelligent traffic lights, in: Proceedings of the 12th ITS European Congress, Strasbourg, France, 19–22 June, 2017.

[24] A. David, K.G. Larsen, M. Mikučionis, M. Muizs, J.H. Taankvist, Uppaal stratego tutorial, Int. J. Softw. Tools Technol. Transf. 17 (4) (2015) 397–415.

[25] P.A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, E.W. ner, Microscopic traffic simulation using SUMO, in: The 21st IEEE International Conference on Intelligent Transportation Systems, IEEE, 2018.

[26] A. Bilgram, E. Ernstsen, P. Greve, H. Lahrmann, K.G. Larsen, M. Muizs, P. Taankvist, T. Pedersen, Online and proactive vehicle rerouting with uppaal stratego, Transp. Res. Rec. (2021) 03611981211000348.

[27] A. David, P.G. Jensen, K.G. Larsen, M. Mikučionis, J.H. Taankvist, Uppaal stratego, in: C. Baier, C. Tinelli (Eds.), Tools and Algorithms for the Construction and Analysis of Systems - 21st International Conference, TACAS 2015, Held As Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015. Proceedings, in: Lecture Notes in Computer Science, 9035, Springer, 2015, pp. 206–211.

[28] G. Behrmann, A. Cougnard, A. David, E. Fleury, K.G. Larsen, D. Lime, UPPAAL-tiga: Time for playing games! in: W. Damm, H. Hermanns (Eds.), Computer Aided Verification, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 121–125.