Intelligent Building Evacuation: From Modeling Systems to Behaviors

Mahyar T. Moghaddam, Henry Muccini, and Julie Dugdale

Abstract Disaster risk management requires new approaches and mechanisms to improve citizens' safety in disasters. The Internet of Things (IoT) is among the technologies that could enhance awareness by providing real-time information. When an emergency happens, building occupants need to be evacuated to safe areas in the shortest possible time. Optimization algorithms could receive humans' mobility data from IoT resources and calculate the best route to follow. The algorithm we present in this chapter formulates and solves a linearized, time-indexed flow problem on a network that represents feasible movements of people at a suitable frequency. We evaluate the performance of the IoT system, including the algorithm, to confirm compliance with real-time use. While the optimization method gives a best case scenario, it does not reflect actual human behavior in evacuation. Humans may stay calm and follow our IoT system's instructions, but they may also have different characteristics and contexts or experience panic attacks, or emotional and social attachment. Thus, we recreate our scenarios with agent-based social simulations, which model occupants as computational agents in an artificial society. The simulations give insights towards a more efficient IoT infrastructure design. We apply our approach to a real location with actual data to prove its feasibility.

Keywords Emergency management · Internet of things · Internet of behaviors · Built environments · Performance · Software architecture · Agent-based

M. T. Moghaddam (✉)
University of Southern Denmark, Odense, Denmark
e-mail: mtmo@mmmi.sdu.dk

H. Muccini
University of L'Aquila, L'Aquila, Italy
e-mail: henry.muccini@univaq.it

J. Dugdale
University Grenoble Alps, Grenoble, France
e-mail: julie.dugdale@imag.fr

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

H. J. Scholl et al. (eds.), *Disaster Management and Information Technology*,
Public Administration and Information Technology 40,
https://doi.org/10.1007/978-3-031-20939-0_7

modeling · Human behavior modeling · Social attachment · Simulation · Optimization · Network flow

Introduction

The aggressive and unpredictable nature of hazards requires designing dynamic evacuation plans. Equipping buildings with the Internet of Things (IoT) resources can provide real-time awareness about, e.g., dangerous areas, congestions, and obstacles. IoT infrastructures are mainly composed of sensing, computation, actuating, and network facilities distributed over physical spaces (Muccini & Moghaddam, 2018; Muccini et al., 2018). The way those components are related and combined is specified by software architectures. In the emergency management context, IoT could adopt logic and rules to facilitate occupants' safety by tracking them, detecting bottlenecks, and updating safe evacuation paths.

Essential questions are as follows: (i) How can evacuation be facilitated by showing occupants the quickest path towards safe areas? (ii) How should IoT infrastructures be designed to be able to tackle the contextual and internal changes? (iii) How can dynamic (and sometimes irrational) human behavior be analyzed and considered in designing IoT infrastructures?

A building can be modeled as a network of *nodes* (corresponding to the building's space, organized into suitable square cells) and *arcs* (representing passages between adjacent cells). Such a model can be combinatorial since it could decompose both space (building plan) and time dimension into finite elements: unit cells and time slots. We previously proposed a network flow algorithm (Arbib et al., 2018, 2019b,a) that acts as the decision-maker of IoT-based evacuation infrastructures reacting to environmental events. IoT cameras continuously monitor the cells' occupancy and the flow among them. Collected data are used to create a second acyclic digraph, indexed on time, which models all the feasible transitions between adjacent cells at any given time slot and given the current occupancy status of each cell. Minimizing the total evacuation time then corresponds to solving a mathematical program that, in the final refinement, has the form of a linear optimization problem.

In addition to minimizing evacuation time, IoT architects should establish mechanisms to reduce the system response time by self-adaptive software architectures. Minimizing the response time of IoT-based emergency management systems is critical since endangered people need to receive the routing guide as quickly as possible. In self-adaptive systems, the position of computation could be dynamically changed if a quality issue is perceived. In other words, running the evacuation algorithm on a local server can enhance the performance in specific conditions. However, in a different situation, it may be more efficient to run it on the cloud. An adaptation manager typically performs the adaptation control that comprises the application logic and supervises the managed system (Moghaddam et al., 2021, 2020). We use queuing networks (Arbib et al., 2019a) to test the performance of our IoT system.

Considering human behavior is another crucial factor. In the context of socio-technical IoT (Dugdale et al., 2020) and Internet of Behaviors (IoB) (Moghaddam et al., 2022; Alipour et al., 2021, 2020), humans are immersed in the system, and their behaviors impact the system's quality and functionality. Our vision is that individual (goals, intentions, context, etc.) and social (collective social behaviors, social links, collaborations, etc.) dimensions of software systems are essential elements that must be considered when designing architectures for IoT applications. We propose an agent-based modeling (ABM) approach to model humans and their individual and social behaviors. In this way, we put humans, their context, goals, and safety at the heart of IoT system design while at the same time considering the software quality.

This chapter presents the following contributions:

- A self-adaptive IoT system that adopts an optimization algorithm and tackles the contextual and internal risks
- An ABM that models and simulates human behavior in disaster risk situations
- Applying our dynamic emergency evacuation approach to a real case, an exhibition venue in the Alan Turing Building at the University of L'Aquila, Italy

The chapter is structured as follows. Relevant literature is discussed in Section "Related Work". Section "An Intelligent Infrastructure for Evacuation" proposes the IoT infrastructure and its software architecture. The optimization algorithm for quick evacuation is presented in Section "A Flow Model for Quick Evacuation", and human behavior modeling is defined and developed in Section "Human Behavior Modeling in Evacuation". The application of the model to a real exhibition venue is presented in Section "Application". Lessons learned are given in Section "Lessons Learned", and conclusions are finally drawn in Section "Conclusion".

Related Work

Information technologies are receiving increasing attention in the emergency management domain (Huggins & Prasanna, 2020; Luna & Pennock, 2018). However, the use of IoT for evacuation planning is not widely explored. Some studies (Saini et al., 2022) distribute the processing over different computation layers to form a hybrid architecture (Muccini & Moghaddam, 2018). In those architectures, IoT resources capture the contextual data, and the fog layer analyzes the data and detects an emergency. The cloud layer then facilitates an evacuation algorithm to compute the safe and fast routes. Some studies focus on networking in cloud-based systems (Chung & Park, 2016) to obtain rapid and smooth responses to disasters. In that case, building local wireless disaster information network systems connected to each other delivers information about disaster situations. Some studies (Franchi et al., 2019) focus on fifth-generation (5G) mobile networks to facilitate IoT-based disaster management systems. 5G provides high reliability and performance when the IoT hardware, network infrastructure, and software platforms are natively integrated.

Literature rarely discusses the performance of IoT-based emergency management systems. In general, queuing networks (QNs) provide editors and environments for analyzing the performance of IoT systems (Arbib et al., 2019a). QNs are used as an analytic model for IoT systems (El Kafhali et al., 2018) to reduce the cost of computing resources while guaranteeing performance constraints. QNs also help to predict the system's response time (Huang et al., 2018) and estimate the minimum required processing resources to meet the service level agreement. QNs also model self-adaptive systems by separating the concerns in the environment from infrastructure events analysis (Moghaddam et al., 2020, 2021). For instance, Jung et al. (2008) takes advantage of layered QNs while considering the run-time quality of service (QoS) to automatically generate adaptation policies. Moghaddam et al. (2020) used QNs to model the IoT architectural adaptation and control mechanism. In such systems, functional control elements are in charge of environmental adaptation, and autonomic control elements handle the functional system's architectural adaptation.

Apart from the IoT infrastructure and its quality concerns, a suitable routing algorithm should enable quick evacuation. In the domain of evacuation routing, pioneering work was done by Choi et al. (1988) who modeled a building evacuation problem by dynamic flow maximization where arc capacities depend on flows in incident arcs. Although dating back to the 1980s and limited to a theoretical analysis, the paper provides a good starting point and deserves consideration in the light of the progress done in linear programming solution tools. Chen and Feng (2009) propose a flow control algorithm to compute evacuation paths according to the building plan and the total number of evacuees. The model aims at minimizing total evacuation time while assigning an optimal number of evacuees to each evacuation path. However, as network size increases, the associated problem can no longer be solved in real time. Some researchers (Schloter & Skutella, 2017) base evacuation planning on a transshipment problem, and some (Abdelghany et al., 2014) integrate genetic algorithms with a microscopic pedestrian simulation assignment model. One crucial issue addressed by the recent literature is the ability to find suitable solutions in a short time as required by a practical computational core of a real-time IoT system.

Guiding people based on an optimization algorithm could be desirable, but people may not follow the given guides. For the simulation of human behavior, agent-based social simulations (ABSS) are a good tool (Dugdale et al., 2019). In ABSS, an agent is defined as an autonomous software entity that can act upon and perceive its environment (Ferber & Weiss, 1999). When agents are put together, they form an artificial society, each perceiving, moving, performing actions, communicating, and transforming the local environment, much like human beings in real society. In ABSS, the agents typically represent humans or groups of humans interacting with the environment (Dugdale, 2013). An effective method used to model pedestrian movement in agent-based systems is the social force model (Helbing & Molnar, 1995; Beck et al., 2014). A belief-desire-intention (BDI) agent architecture (Rao et al., 1995) can be used to model the cognitive reasoning of individual human agents. Our simulation environment (PedSim Pedestrian Simulator, 2022) comprises *a 2D/3D map, humans, obstacles, points of interest,* and *IoT resources.* A person can have some points of interest, and they stop when they are sufficiently attracted to this point of interest. When the simulation starts, the simulator generates a routing graph from all obstacles on the map that is automatic and based on run-time situations. The agents find their planned and potential routes from the edges of the route graph.

An Intelligent Infrastructure for Evacuation

An IoT-based emergency evacuation system can collect human and disaster data to be analyzed for further actuation. For such an application, suitable architectures that are automatically adapted based on system and environment dynamics are required. In previous work (Moghaddam et al., 2021; Muccini & Moghaddam, 2018), we proposed three architectural patterns as shown in Fig. 1. The patterns are composed of an *IoT element* layer and one or several *control* layers. The control can be performed locally and/or centrally and remotely. It is here where a centralized cloud and distributed edge and fog can form the *hierarchical* pattern. Thus, the patterns (Moghaddam & Muccini, 2019) characterize *IoT* systems based on their levels of *distribution* and *collaboration* (Muccini & Moghaddam, 2018). Distribution specifies whether data analysis software ought to be deployed on a single node (*centralized*) or on several nodes (*distributed* and *hierarchical)* that are

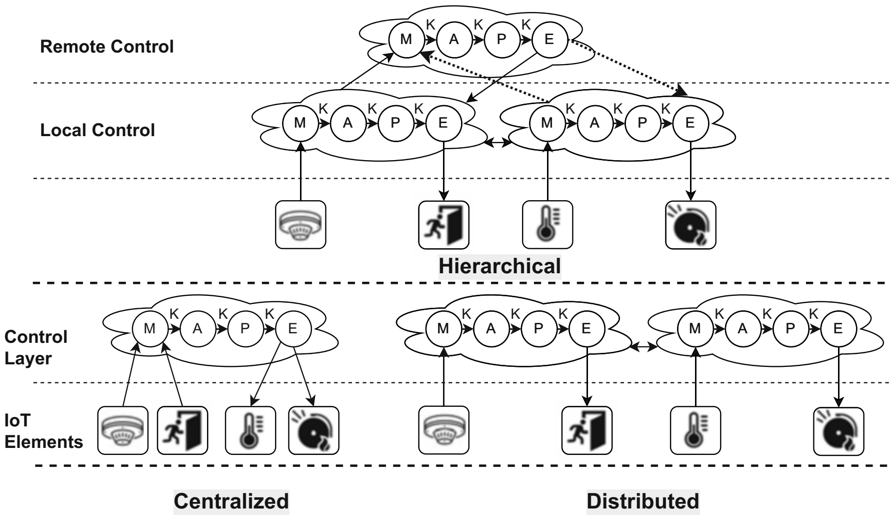

Fig. 1 *IoT* architectural patterns based on control components' composition. The *centralized* pattern comprises processing on a *central local* or *remote* controller. The *distributed* pattern includes the processing on *independent* or *collaborative* controllers. The *hierarchical* pattern contains *independent* or *hybrid* (i.e., with distributed collaborative) controllers

dispersed across the *IoT* system. Collaboration involves interaction among control components to satisfy the goals, requirements, and strategies. This collaboration may appear as a level of information sharing, coordinated analysis or planning, or synchronized execution (Muccini et al., 2018).

Self-adaptation is based on a *MAPE-K* (*monitor, analysis, plan, execute, and comprehensive knowledge*) approach. The *monitor* element aggregates and refines the data to be analyzed and updates the *knowledge* base of the control component. The *analyze* element interprets the monitored data based on the functional goals. The *plan* element builds actuation strategies, and the *execute* element processes the actuation strategies and prepares the type of message to be set to each set of actuators.

For the three patterns mentioned above, the computational component will thus become the central element that will provide evacuation recommendations while inputting situational awareness information. This central computational component has a mathematical logic that is proposed as an algorithm in the following section. In addition to running the algorithm, the presented architectures contain the mechanisms to determine the required architectural adaption based on the intended quality of service satisfaction level. The concept does not rely on any specific tool; thus, practical modeling solutions can be mapped within it. Section "Application" describes the steps taken to map the emergency handling system using this approach to improve its performance indices.

A Flow Model for Quick Evacuation

The following network construction basically follows Choi et al. (1988) and Arbib et al. (2018). The topology of the building to be evacuated is described by a graph $G = (V, A)$ that in Choi et al. (1988) is called a *static network*. Nodes of G correspond to the unit cells i obtained by embedding the building into a suitable grid that will be discussed in Section "Application". In general, cells may have different shapes or sizes: in our work, what is essential is that every cell can be traversed, in any direction, in a single time slot. Cell 0 conventionally represents the outside of the building or, in general, a safe place. Safe places can be disconnected areas, but as their capacity is assumed to be large enough to guarantee safety, we represent them all by a single cell (therefore, what we assume about cells traversing time does not apply to cell 0). The arcs of G correspond to passages between adjacent cells: the passage has full capacity if cells share a boundary that is not interrupted by walls; otherwise, it has a reduced capacity. With no loss of generality, arcs are directed. Let us denote:

T $= \{0, 1, \ldots, \tau\}$, set of unit time slots.

y_i^t = state of cell $i \in V$ at time $t \in T$, that is, the number of persons that occupy i at t: this number is a known model parameter for $t = 0$ (in particular, $y_0^0 = 0$) and a decision variable for $t > 0$.

n_i = capacity of cell i: it measures the maximum nominal amount of people that i can host at any time (in particular, $n_0 \geq \sum_i y_i^0$); this amount depends on cell shape and size; if cells can be assumed uniform, one can set $n_i = n$ for all $i \in V, i \neq 0$.

x_{ij}^t = how many persons move from cell i to an adjacent cell j in $(t, t+1]$: this gives the average speed at which the flow proceeds from i to j.

$c_{ij} = c_{ji}$ = capacity of the passage between cell i and cell j: this is the maximum amount of people that, independently of how many persons are in cell j, can traverse the passage in the time unit (independence from cell occupancy means neglecting system congestion: we will consider this issue later).

The flow model uses an acyclic digraph D with node set $V \times T$ and arc set

$$E = \{(i,t) \to (j, t+1) : ij \in A, t \in T\}$$

Referred to as *τ-time* or a *dynamic network* in Choi et al. (1988), D models all the feasible transitions (i.e., moves between adjacent cells) that can occur in the building in the time horizon T. Transitions are associated with the x-variables defined above, whereas y-variables define the occupancy of each room (and of the building) from time to time. The x- and y-variables are integers and subject to the following constraints:

$$y_j^t - y_j^{t-1} - \sum_{i:ij \in A} x_{ij}^{t-1} + \sum_{i:ji \in A} x_{ji}^{t-1} = 0 \qquad j \in V, t \in T, t > 0 \tag{1}$$

$$0 \leq x_{ij}^t + x_{ji}^t \leq c_{ij} \qquad t \in T, ij \in A \tag{2}$$

$$0 \leq y_i^t \leq n_i \qquad t \in T, i \in V \tag{3}$$

Equation (1) is just a flow conservation law: it expresses the occupancy of cell j at time t as the number y_j^{t-1} of persons present at time $t-1$, augmented by those that during interval $(t-1, t]$ move to j from another cell $i \neq j$ minus those that in the same interval leave cell j for another room $i \neq j$. Box constraints (2), (3) reflect the limited hosting capability of the elements of G.

Maximizing Outflow in a Given Time To model the relation between time and people outflow, we can try to maximize the number of persons evacuated from the building within τ:

$$\max \quad y_0^\tau \tag{4}$$

To find the minimum total evacuation time, we can solve a max flow problem for different τ, looking for the smallest value that yields a zero-valued optimal solution.

To reduce computation time, this optimal τ can be computed by logarithmic search. The method can thus provide the decision-maker with the Pareto frontier of the conflicting objectives $\min\{\tau\} and \max\{y_0^\tau\}$. Linearizing arc capacities, which is quite standard in applications, can be found in our previous work (Muccini et al., 2019). The presented optimization model could result in a quick evacuation. However, people do not always behave in an optimal way, and how this has been taken into account in the simulator is discussed below.

Human Behavior Modeling in Evacuation

The agent-based model for IoT socio-technical systems consists of four classes of agents, *humans, cyber elements, physical space, and IoT resources*, which all are part of the *environment* class. A class is, by definition, a template for an agent. When the model is implemented and the simulator is run, various agents within the same class but with potentially different attributes satisfy the social behavior and contextual heterogeneity. For instance, many human agents with the same attributes are created but with possibly different values for the attributes, thus creating a heterogeneous artificial society.

Environment agents represent the perimeter within which the agents interact with each other. In IoT, the environment could be an indoor or outdoor space containing humans, physical space (including IoT resources), and cyber elements.

Human agents represent occupants and are modeled using the a belief-desire-intention (BDI) architecture. A *belief* represents the agent's own knowledge of events and locations. A *desire* outlines the motivational state of an agent, activities that the agent would like to perform. An *intention* represents the deliberative state of an agent, i.e., a selected desire. Once an intention is chosen, the agent develops a plan to achieve that intention (goal). The agent's decision-making and dynamic path routing are influenced by the desire to avoid congestion and obstacles. Human agents have attributes such as movement speed, perceptive radius, vision, social force (personal and interpersonal radius), and social attachment. This is updated using real geospatial data obtained from the IoT infrastructure.

Physical space agents represent the topology of the space, such as obstacles, walls, doors, passageways, and installed devices. These agents have certain forces and characteristics, such as wall force, passageway, and door flow capacity. The physical space is divided into cells, each containing human agents, and the flow between those cells represents the occupants' mobility.

IoT resources agents are a category of physical space agents that capture human agents' mobility behavior and give human agents instructions. The IoT physical resources include sensors, network facilities, processors, and actuators. IoT resources agents' behavior is in line with their proper functionality, their mobile or static position, and their coverage.

Cyber elements agents represent the software that is run on the IoT resources agents. They provide cyber twins for sensing, network, processing, and actuating and reflect the attributes of physical space agents. Their behavior is specified by their level of functionality and quality.

Application

Our proposed model has been applied to the evacuation of the Alan Turing building, in Italy, which is sometimes used for exhibitions. The building consists of 29 rooms, 4 main corridors, and 34 sets of IoT sensors and actuators. Room sizes vary greatly and, as a consequence, so does the average time for a person to cross them from door to door. The complex building structure, as well as data on people attendance collected during events, made this study case ideal for illustrating a general methodology for system sizing and development. We run various simulations to assess the application of our models on:

- Discovering the optimal evacuation time that results from crowd routing via ideal evacuation paths and comparing it with the evacuation time that derives from static shortest paths (Section "Algorithm Simulations")
- Evaluating the performance of the IoT infrastructure that runs the algorithm (Section "Algorithm Simulations")
- Providing guidelines about human behavior in an emergency (Section "Algorithm Simulations")

Algorithm Simulations

We split each room into unit cells, behaving like a (virtual) square room that can be traversed in a unit time slot. In practice, we embedded the building plan into a square grid as shown in Fig. 2. To decide the cell size, we looked at both the error (areas not covered by cells) introduced by room approximation and the number of nodes in the resulting graph G. The latter is in an inverse proportion of cell size; the former varies irregularly with cell size (for more details, see Arbib et al., 2019b). We considered square cells of 3×3 meters, which led to only 144 nodes (Fig. 3). The selected cell size reduces the largest error for all rooms and facilitates IoT camera monitoring.

Simulation The simulation code was written in the OPL language, and problems were solved by CPLEX version 12.8.0. All experiments were run on a Core i7 2.7 GHz computer with 16Gb of RAM under Windows 10 pro 64-bits. In all tests, we computed the minimum time required for 225 persons, randomly distributed in the building rooms, to reach a safe place. This data comes from an experiment performed in the building during the *researchers' night* event when the IoT system

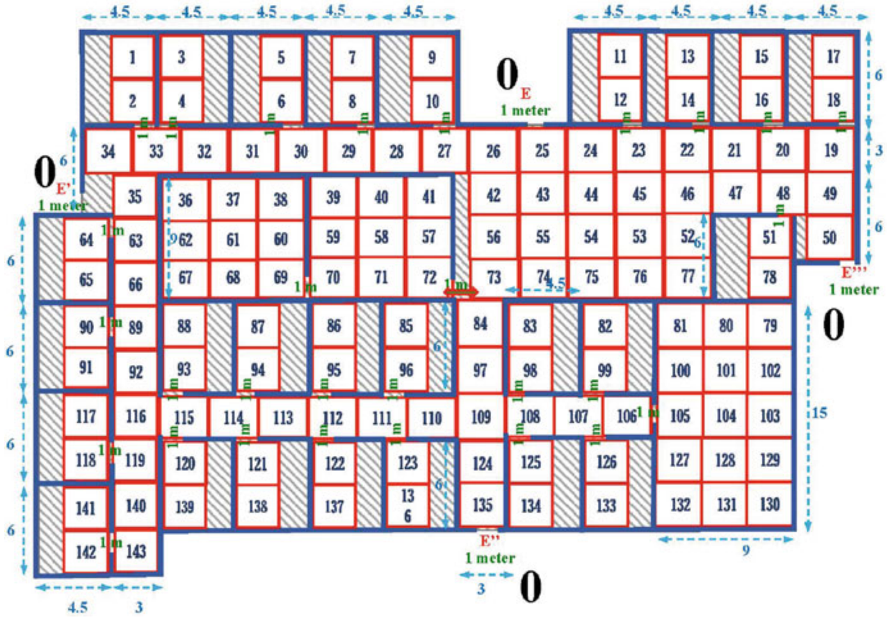

Fig. 2 Plan embedding the Alan Turing building into square grids with a low resolution: 3 × 3 cells. The area that is not covered by cells (error) is shown in gray

recorded the simultaneous presence of 225 people as a peak value. We solved problem (1–4) for $\tau = 1, 2, \ldots$ until a solution of value 225 was found.

To get a reliable model, some parameters such as walking velocity under various conditions, door entrance capacities, and room capacities were set to numbers that reflect reality. We set these model parameters based on empirical observations reported in the literature (Table 1).

Table 2 reports the number of evacuees at each τ and the computation time of each solution step. In terms of evacuation, everyone has reached a safe place in 47.5 seconds; on the other hand, computation requires 1.82 seconds in the worst case and is therefore totally compliant with real-time applications.

This simulation depicts an ideal situation in which human agents autonomously choose the best among all the available routes in the building. Of course, managing such an ideal evacuation is not easy and perhaps unpractical. As a general practice, evacuation is conducted through predetermined routes. Considering this fact, we suppose that the prescribed evacuation routes are the shortest paths from any cell to a safe place. To evaluate this situation, we find the subgraph of G formed by the shortest paths from any cell to 0 (as from a static evacuation map), construct its time-indexed network, and solve the problem (1)–(4) for increasing τ. Evacuating 225 individuals takes, of course, more time: 1 minute and 10 seconds. Thus, compared to

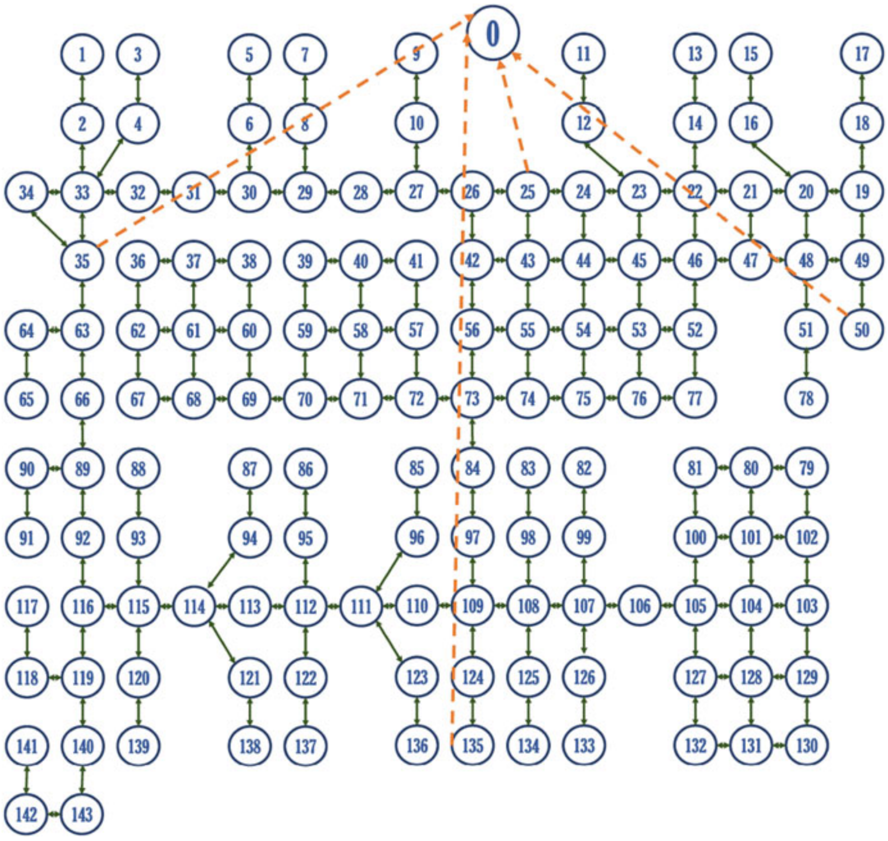

Fig. 3 Network associated with the plan of Fig. 2

Table 1 Evacuation model parameters

Model parameter	Assigned value
Walking velocity	1.2 m/s (Ye et al., 2008a)
Door capacity	1.2 p/m/s (Daamen & Hoogendoorn, 2012)
Cell capacity	1.25 p/m^2 (Matthews, 2015)

the Netflow model we propose, the shortest routes increase, in this case, the optimal evacuation time by 47% (Fig. 4).

Comparing Netflow and the shortest path, there are similar flows for 15 seconds. After that, the shortest paths approach experiences congestion, and evacuation slows down.

Table 2 Evacuation and computation time for 3 × 3 cells with time slots of 2.5 seconds

Tau	Evacuees	CPU time	Tau	Evacuees	CPU time
1	12	0.65 sec	11	132	0.96 sec
2	24	0.50 sec	12	144	1.11 sec
3	36	0.54 sec	13	156	1.18 sec
4	48	0.63 sec	14	168	1.30 sec
5	60	0.65 sec	15	180	1.52 sec
6	72	0.80 sec	16	192	1.40 sec
7	84	0.77 sec	17	204	1.52 sec
8	96	0.89 sec	18	216	1.78 sec
9	108	0.87 sec	19	225	1.82 sec
10	120	1.20 sec			

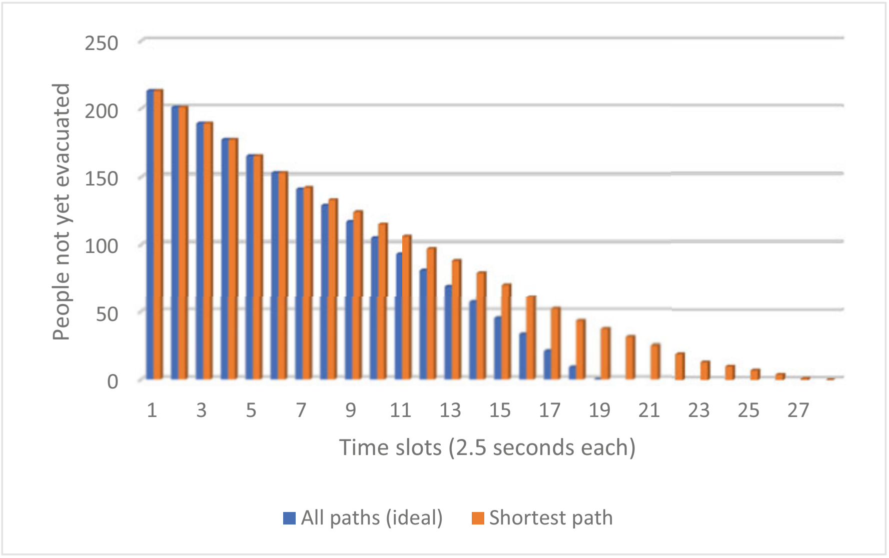

Fig. 4 Ideal vs. shortest paths evacuation

Software Architecture Simulations

In the suggested IoT-based environment for emergency response, CCTV cameras detect people's position and movement and feed them into the running algorithm. The algorithm decides on the actuation set based on the situation. As shown in Fig. 5, additional sets of sensors can be embedded for emergency detection to further enable controllers to decide about normal or critical mode and activate a particular set of actuators. In normal situations, the system shows a 2D representation of the monitored space and the crowd's position and movement. In this mode, the optimal flow algorithm is periodically run to estimate the minimum evacuation time required under current conditions. If an emergency happens, in addition to

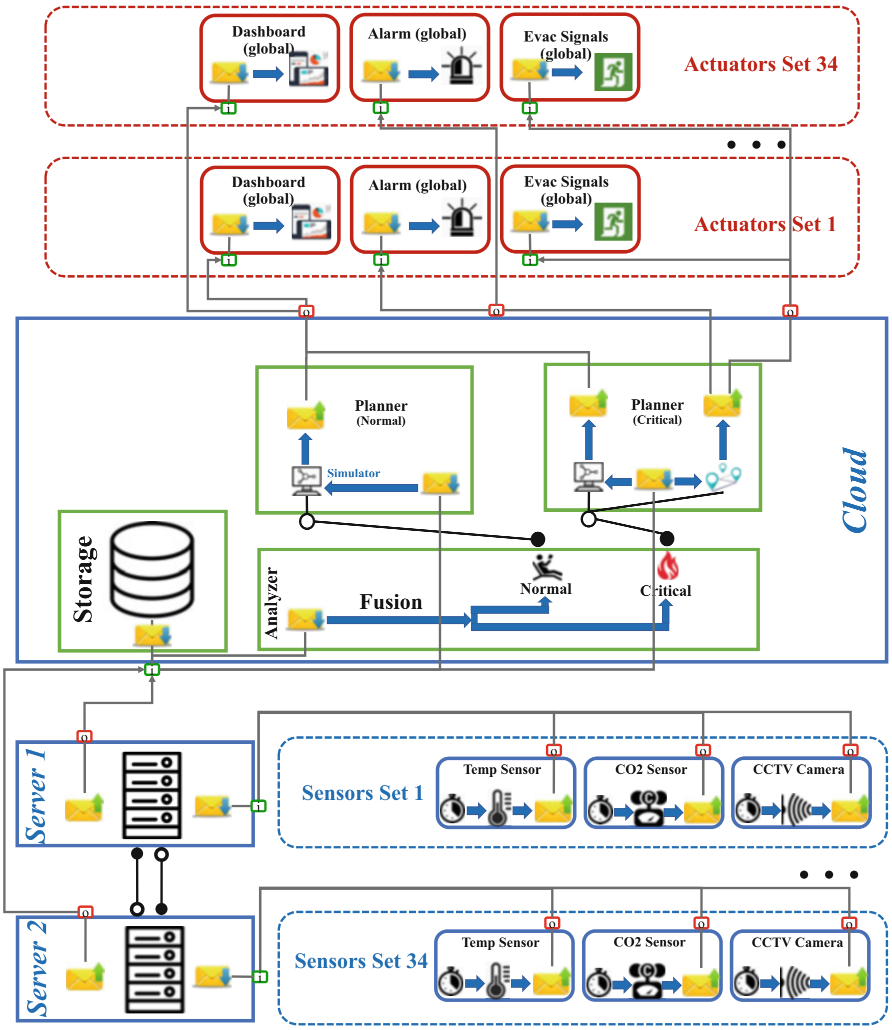

Fig. 5 Software architecture of the IoT-based emergency management system

the dashboard, alarm actuators are activated, and evacuation signs in each area show the best evacuation routes based on the network model described above. Specifying the position of computation (i.e., servers, cloud, or a mix of them) determines the architectural patterns at run time (see Section "An Intelligent Infrastructure for Evacuation", Fig. 1).

In our proposed infrastructure, in addition to the computational delay of the processing components, the sensors take some time to detect people's positions, transmit these data, and display the best evacuation routes. Reducing these delays to a minimum improves the system's functionality since people can follow the

Table 3 Response time for different modes and configurations (seconds)

Pattern	Response time (normal)	Response time (critical)
Centralized	1.2	2.65
Distributed	0.55	1.05
Hierarchical	0.95	1.5

given instructions more quickly, and more individuals will be in a better evacuation position at the subsequent monitoring time-spot. Reducing the delays mentioned above is a function of software architectural patterns, which can be improved by adequately relating the IoT components to one another. Using a probabilistic routing strategy, we modeled the same system with different architectural configurations with QNs. We used JMT (Casale et al., 2009) to model and simulate the QNs. For more information about QN models and service times used for the response time analysis, please see Arbib et al. (2019a).

In our QNs simulation, we assess the response time of the three architectural patterns (presented in Section "An Intelligent Infrastructure for Evacuation") in normal and critical situations. The response time (delay) that is analyzed is the mean time spent from starting the sampling to the time that actuation ends. As shown in Table 3, experimental results show that the distributed pattern minimizes system response time for the normal mode (0.55 seconds). Still, it should be adapted to a hierarchical pattern (1.5 seconds) when an emergency occurs. While the response time associated with the hierarchical pattern is 43% more than distributed (still better than the centralized pattern with a delay equal to 2.65 seconds), our routing algorithm must be run on a single processor.

Human Behavior Simulations

In our agent-based model of the Alan Turing Building, we set the simulation parameters either by using gathered real data or according to the literature. With a real population of 225 occupants in the building, we apply the following parameters:

- *Walking speed*—ranges from 0.7 to 1.2 m/s (Wagnild & Wall-Scheffler, 2013; Tolea et al., 2010).
- *Social force*—0.2 m was used an individual agent's radius by using the *biacromial diameter* in Patil et al. (2017). Thus, agents maintain a certain distance from each other and do not cross through each other. In addition, this eases setting the maximum number of agents per cell, room, and passage flow.
- *Wall force*—0.1 m is the wall force, which means that agents cannot get closer to 0.1m from a wall. The result is that agents do not cling to walls or pass through obstacles.
- Door flow capacity— 1.2 p/m/s, Ye et al. (2008b).
- Cell capacity— 1.25 p/m^2, Daamen and Hoogendoorn (2012).

We use the ***belief-desire-intention*** agent architecture:

- *Belief*—All agents believe that a disaster is happening and that they must seek safety.
- *Desire*—All agents desire or *goal* to reach an exit.
- *Intention*—Since the agents *perceive* their surroundings, they try to find the shortest and/or optimal paths to get to the exit (based on the algorithms).

While the agents have specific points of interest, they could change their target based on some contextual situations such as visible congestion and the intention of their friends or relatives. We set the target switch and abandon time, based on the importance of the agent's point of interest. We ran all the experiments on a *Corei7 2.7 GHz* computer with *16 Gb* of RAM memory under Windows 10 pro *64-bits*. For these simulations, we obtain information regarding the number of visitors who fall under the coverage area of various sensors, the route each agent took, the variation of their velocity, acceleration, stops index, the behavior of each agent (such as obstacle collision avoidance, anticipatory and passive collision avoidance), movement state (e.g., moving and looking), and visiting satisfaction. We also assess the number of visitors who visited a room, their access points, the location of collisions, and any queue length.

We used the historical data to assign different characteristics to human agents regarding age, gender, origin, and physical condition. In fact, each human agent has a profile including vision, maximum speed, and target force that are impacted by their characteristics. We observed various challenges regarding congestion, physical bottlenecks, and grouping. To increase realism, we incorporate social attachment. Congestion is common in emergencies; thus, the agents can form herds, which affect their decisions and movement towards exits. Speed of movement is essential, e.g., a man will walk more slowly to match the speed of a woman (Tolea et al., 2010), and groups of individuals typically move more slowly than a single person (Sarmady et al., 2009). The slow movement of interacting groups can consequently affect evacuation efficiency (Qiu & Hu, 2010). To model such behaviors, groups move at a slower speed than individuals. While these solutions are assessed in the simulation environment, they could provide situational awareness for the stakeholders (managers, operators, and visitors) and show possible movement patterns.

In the next step, we evaluated the optimization algorithms under a realistic situation regarding human behaviors. We considered groups of 3 to 7 agents. This gives an interesting scenario as congestion at exits becomes more pronounced. In this simulation, agents' walking speed varies between 0.7 and 1.2 m/s since the speed of movement depends on other group members and the velocity of the slowest person (Wagnild & Wall-Scheffler, 2013). As shown in Fig. 6, we observed that evacuating 225 agents takes 1 minute and 57.5 seconds with shortest path algorithm and 1 minute and 47.5 seconds with Netflow.

We understood that grouping and attachment slow down evacuation, compared to optimization only. Evacuation time increases with the number of agents because

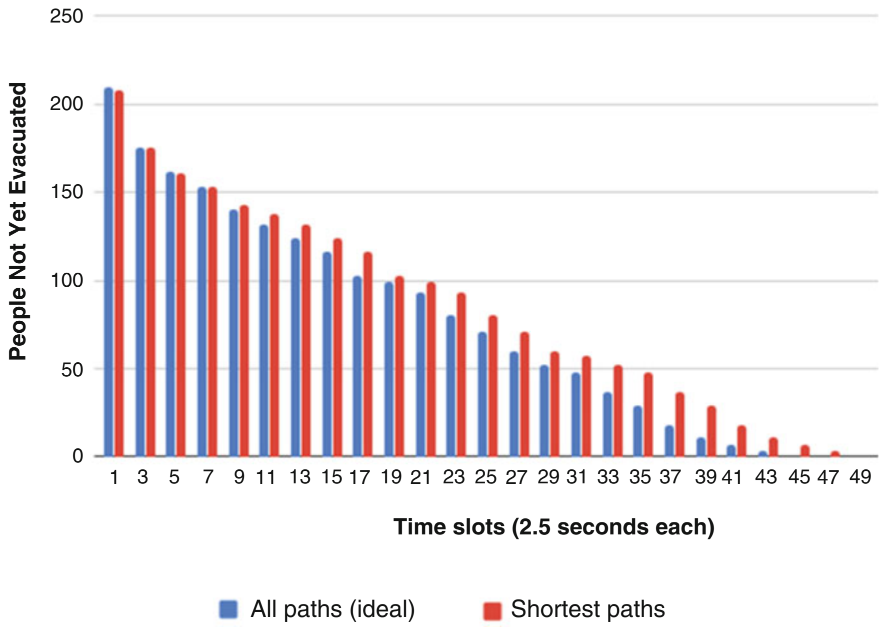

Fig. 6 Netflow vs. shortest path evacuation considering grouping and attachment with 225 agents

socially attached agents will not leave the building without their friends, family, or acquaintances.

Lessons Learned

The discussion of our approach and the results of our evaluation indicate how IoT architects could design an emergency management infrastructure by considering the architecture qualities, algorithm, and human behavior. More specifically, we learned that:

- While the design of a solid software architecture is crucial, its adaptation based on run-time environmental and internal situations could enhance quality.
- The core optimization algorithm for IoT-based emergency evacuation should consider the dynamic flow of people and congestion in order to perform efficiently.
- While an optimization approach is valid, considering realistic human behaviors in emergencies could benefit a solid IoT architectures design.

- Social links and preferences should be seriously considered and modeled within emergency handling systems since they can impact the functionality and quality of the system.
- Simulations enhance situational awareness of occupants, managers, and practitioners and improve their preparedness in case of disasters (Dugdale et al., 2021).

Conclusion

In this chapter, we presented an intelligent IoT infrastructure to handle emergencies. The system gets data from sensors and uses an optimization algorithm to lead people to safe areas as quickly as possible. The response time of the system was also assessed for various architectural patterns. The consideration of human behaviors in such risky situations was addressed by an agent-based modeling approach that gives insights towards a human-oriented design and adaptation of the infrastructure. In future work, the authors will consider more quality attributes such as fault tolerance, availability, and energy efficiency. Moreover, more empirical studies will be performed to get real data to input into simulations. We will explore more the links between human behavior and system behavior in the context of IoT-based emergency management.

Acknowledgments We kindly thank our colleagues Prof. Claudio Arbib and Prof. Paul Davidsson who contributed to the previous version of this chapter. This work is supported by the Innovation Fund Denmark for the project DIREC (9142-00001B) and the VASARI PON R&I 2014–2020 and FSC project.

References

Abdelghany, A., Abdelghany, K., Mahmassani, H., & Alhalabi, W. (2014). Modeling framework for optimal evacuation of large-scale crowded pedestrian facilities. *European Journal of Operational Research, 237*(3), 1105–1118.

Alipour, M., Dupuy-Chessa, S., & Céret, E. (2021). An emotion-oriented problem space for ui adaptation: From a literature review to a conceptual framework. In *2021 9th International Conference on Affective Computing and Intelligent Interaction (ACII)* (pp. 1–8). IEEE.

Alipour, M., Dupuy-Chessa, S., & Jongmans, E. (2020). Disaster mitigation using interface adaptation to emotions: a targeted literature review. In *10th International Conference on the Internet of Things Companion* (pp. 1–15).

Arbib, C., Muccini, H., & Moghaddam, M. T. (2018). Applying a network flow model to quick and safe evacuation of people from a building: A real case. *RSFF, 18*, 50–61.

Arbib, C., Arcelli, D., Dugdale, J., Moghaddam, M., & Muccini, H. (2019a). Real-time emergency response through performant iot architectures. In *International Conference on Information Systems for Crisis Response and Management (ISCRAM).*

Arbib, C., Moghaddam, M. T., & Muccini, H. (2019b) Iot flows: a network flow model application to building evacuation. In *A View of Operations Research Applications in Italy, 2018*, (pp. 115–131). Springer.

Beck, E., Dugdale, J., Van Truong, H., Adam, C., & Colbeau-Justin, L. (2014). Crisis mobility of pedestrians: from survey to modelling, lessons from lebanon and argentina. In *International Conference on Information Systems for Crisis Response and Management in Mediterranean Countries* (pp. 57–70). Springer.

Bertoli, M., C. G., & Serazzi, G. (2009). Jmt: performance engineering tools for system modeling. In *ACM SIGMETRICS Performance Evaluation Review* (pp. 10–15). ACM.

Chen, P.-H., & Feng, F. (2009). A fast flow control algorithm for real-time emergency evacuation in large indoor areas. *Fire Safety Journal, 44*(5), 732–740.

Choi, W., Hamacher, H. W., & Tufekci, S. (1988). Modeling of building evacuation problems by network flows with side constraints. *European Journal of Operational Research, 35*(1), 98–110.

Chung, K., & Park, R. C. (2016). P2p cloud network services for iot based disaster situations information. *Peer-to-Peer Networking and Applications, 9*(3), 566–577 (2016).

Daamen, W., & Hoogendoorn, S. (2012). Emergency door capacity: influence of door width, population composition and stress level. *Fire Technology, 48*(1), 55–71.

Dugdale, J. (2013). *Human behaviour modelling in complex socio-technical systems: an agent based approach.* PhD Thesis, Université Joseph-Fourier-Grenoble I.

Dugdale, J., Moghaddam, M. T., Muccini, H., & Narayanankutty, H. (2019). A combined netflow-driven and agent-based social modeling approach for building evacuation. In *International Conference on Principles and Practice of Multi-Agent Systems* (pp. 460–468). Springer.

Dugdale, J., Moghaddam, M. T., & Muccini, H. (2020). Human behaviour centered design: developing a software system for cultural heritage. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering in Society* (pp. 85–94).

Dugdale, J., Moghaddam, M. T., & Muccini, H. (2021). Iot4emergency: Internet of things for emergency management. *ACM SIGSOFT Software Engineering Notes, 46*(1), 33–36.

El Kafhali, S., Salah, K., & Alla, S. B. (2018). Performance evaluation of iot-fog-cloud deployment for healthcare services. In *2018 4th international conference on cloud computing technologies and applications (Cloudtech)* (pp. 1–6). IEEE.

Ferber, J., & Weiss, G. (1999). *Multi-agent systems: An introduction to distributed artificial intelligence* (Vol. 1). Addison-Wesley Reading.

Franchi, F., Marotta, A., Rinaldi, C., Graziosi, F., & D'Errico, L. (2019). Iot-based disaster management system on 5g urllc network. In *2019 International Conference on Information and Communication Technologies for Disaster Management (ICT-DM)* (pp. 1–4). IEEE.

Helbing, D., & Molnar, P. (1995). Social force model for pedestrian dynamics. *Physical Review E, 51*(5), 4282.

Huang, J., Li, S., Chen, Y., & Chen, J. (2018). Performance modelling and analysis for iot services. *International Journal of Web and Grid Services, 14*(2), 146–169.

Huggins, T. J., & Prasanna, R. (2020). Information technologies supporting emergency management controllers in new zealand. *Sustainability, 12*(9), 3716.

Jung, G., Joshi, K. R., Hiltunen, M. A., Schlichting, R. D., & Pu, C. (2008). Generating adaptation policies for multi-tier applications in consolidated server environments. In *2008 International Conference on Autonomic Computing* (pp. 23–32). IEEE.

Luna, S., & Pennock, M. J. (2018). Social media applications and emergency management: A literature review and research agenda. *International journal of disaster risk reduction, 28*, 565–577.

Matthews, D. (2015). *Special event production: The resources*. Routledge.

Moghaddam, M. T., & Muccini, H. (2019). Fault-tolerant IoT. In *International Workshop on Software Engineering for Resilient Systems* (pp. 67–84). Springer.

Moghaddam, M. T., Muccini, H., Dugdale, J., & Kjægaard, M. B. (2022). Designing internet of behaviors systems. In *2022 IEEE 19th International Conference on Software Architecture (ICSA)* (pp. 124–134). IEEE.

Moghaddam, M. T., Rutten, E., & Giraud, G. (2021). Hierarchical control for self-adaptive iot systems a constraint programming-based adaptation approach. In *HICSS 2022*.

Moghaddam, M. T., Rutten, E., Lalanda, P., & Giraud, G. (2020). IAS: an IoT architectural self-adaptation framework. In *European Conference on Software Architecture* (pp. 333–351). Springer.

Muccini, H., Arbib, C., Davidsson, P., & Tourchi Moghaddam, M. (2019). An iot software architecture for an evacuable building architecture. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*.

Muccini, H., & Moghaddam, M. T. (2018). Iot architectural styles. In: *European Conference on Software Architecture* (pp. 68–85). Springer.

Muccini, H., Spalazzese, R., Moghaddam, M. T., & Sharaf, M. (2018). Self-adaptive iot architectures: An emergency handling case study. In *Proceedings of the 12th European Conference on Software Architecture: Companion Proceedings* (pp. 1–6).

Patil, S. S., Jadhav, P. L., Dongare, S. S., & Deokar, R. B. (2017). Correlation of stature to arm span and biacromial shoulder width in young adults of western indian population. *International Journal of Education and Research in Health Sciences, 3*(2), 64–70.

PedSim Pedestrian Simulator (2022). https://www.pedsim.net/, Accessed February 10, 2022.

Qiu, F., & Hu, X. (2010). Modeling group structures in pedestrian crowd simulation. *Simulation Modelling Practice and Theory, 18*(2), 190–205.

Rao, A. S., Georgeff, M. P., et al. (1995). Bdi agents: from theory to practice. In *ICMAS* (Vol. 95, pp. 312–319).

Saini, K., Kalra, S., & Sood, S. K. (2022). Disaster emergency response framework for smart buildings. *Future Generation Computer Systems, 131*, 106–120.

Sarmady, S., Haron, F., & Talib, A. Z. H. (2009). Modeling groups of pedestrians in least effort crowd movements using cellular automata. In *2009 Third Asia International Conference on Modelling & Simulation* (pp. 520–525). IEEE.

Schloter, M., & Skutella, M. (2017). Fast and memory-efficient algorithms for evacuation problems. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms* (pp. 821–840). SIAM.

Tolea, M. I., Costa, P. T., Terracciano, A., Griswold, M., Simonsick, E. M., Najjar, S. S., Scuteri, A., Deiana, B., Orrù, M., Masala, M., et al. (2010). Sex-specific correlates of walking speed in a wide age-ranged population. *Journals of Gerontology Series B: Psychological Sciences and Social Sciences, 65*(2), 174–184.

Wagnild, J., & Wall-Scheffler, C. M. (2013). Energetic consequences of human sociality: Walking speed choices among friendly dyads. *PloS one, 8*(10), e76576.

Ye, J., Chen, X., Yang, C., & Wu, J. (2008a). Walking behavior and pedestrian flow characteristics for different types of walking facilities. *Transportation Research Record: Journal of the Transportation Research Board, 2048*, 43–51.

Ye, J., Chen, X., Yang, C., & Wu, J. (2008b). Walking behavior and pedestrian flow characteristics for different types of walking facilities. *Transportation Research Record, 2048*(1), 43–51.