Reinforcement Learning Using Monte Carlo Policy Estimation for Disaster Mitigation

Mohammed Talat Khouj, Sarbjit Sarkaria, Cesar Lopez, Jose Marti

Chapter 11

REINFORCEMENT LEARNING USING MONTE CARLO POLICY ESTIMATION FOR DISASTER MITIGATION

Mohammed Talat Khouj, Sarbjit Sarkaria, Cesar Lopez and Jose Marti

Abstract Urban communities rely heavily on the system of interconnected critical infrastructures. The interdependencies in these complex systems give rise to vulnerabilities that must be considered in disaster mitigation planning. Only then will it be possible to address and mitigate major critical infrastructure disruptions in a timely manner.

This paper describes an intelligent decision making system that optimizes the allocation of resources following an infrastructure disruption. The novelty of the approach arises from the application of Monte Carlo estimation for policy evaluation in reinforcement learning to draw on experiential knowledge gained from a massive number of simulations. This method enables a learning agent to explore and exploit the available trajectories, which lead to an optimum goal in a reasonable amount of time. The specific goal of the case study described in this paper is to maximize the number of patients discharged from two hospitals in the aftermath of an infrastructure disruption by intelligently utilizing the available resources. The results demonstrate that a learning agent, through interactions with an environment of simulated catastrophic scenarios, is capable of making informed decisions in a timely manner.

Keywords: Disaster response, Monte Carlo estimation, decision assistance agent

1. Introduction

All of modern society, but in particular urban communities, rely heavily on the system of interconnected critical infrastructures. These systems are inherently complex in terms of interconnections and interdependencies. Thus, they are vulnerable to major disruptions that could cascade to other dependent systems with possible disastrous consequences. For example, the Indian Blackout of 2012 – the largest power blackout in history – caused massive disruptions to medical facilities, transportation systems, water treatment plants and other

interconnected infrastructures. It resulted in the loss of power to 600 million people, trapping miners, stranding railway passengers and plunging hospitals into darkness [6]. Such catastrophic incidents reveal the need for efficient planning and, more importantly, the need for careful decisions to be taken during the first few hours following an incident. The decisions are critical to successful mitigation, damage management, death prevention, injury, structural loss, control of financial costs and, ultimately, the overall resolution of the crisis [9].

This paper describes an intelligent decision making system that optimizes the allocation of resources following an infrastructure disruption and suggests how the resources may be utilized during disaster response. An underlying intelligent learning agent interacts continuously with a simulated environment and uses reinforcement learning (RL) to discover a policy that optimizes a long-term reward. The learning system employs Monte Carlo (MC) estimation for policy evaluation in reinforcement learning to gain experiential knowledge over a massive number of simulations using the interdependent critical infrastructure simulator (i2Sim). The approach enables the learning agent to explore and exploit the possible trajectories that lead to an optimum goal in a reasonable period of time.

2. Related Work

This section describes related work in the areas of disaster mitigation in interdependent critical infrastructures, agent-based modeling for disaster mitigation and disaster mitigation applications using reinforcement learning.

2.1 Disaster Mitigation

Critical infrastructures are characterized by complex interconnections and interdependencies. These systems are vulnerable to major disturbances that can cascade to other dependent systems, potentially leading to national disasters (Figure 1). Interdependencies between infrastructures are bi-directional relationships through which the state of one infrastructure is influenced by or correlated with the states of other infrastructures [15]. Thus, it is essential to address the resource allocation problem in the context of interdependent critical infrastructures for better mitigation planning.

The optimization of resource allocation in interconnected critical infrastructures is a topic that has been addressed extensively. For instance, Min, *et al.* [13] have presented an integrated system to model the physical and financial impacts attributed to critical infrastructure interdependencies. Their framework comprises a system dynamics model, functional model and a non-linear optimization model. The purpose of the system dynamics model is to analyze the interdependencies between individual infrastructure components. The functional model is used to define the data requirements and the information exchanged between the models. The non-linear model enables the determination of optimal values of the control variables. The purpose of the work is to enable officials to respond to potential disruptions in a timely and effective manner.

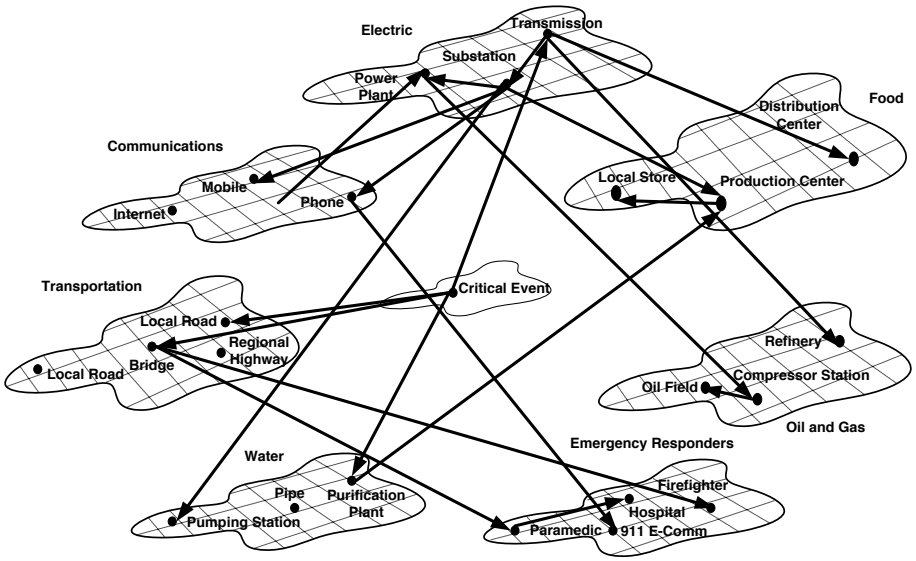

Figure 1. Interconnected critical infrastructures.

O'Reilly, *et al.* [14] have specified a system dynamics model that describes the interactions between interconnected critical infrastructures. They use the model to analyze the impact of a telecommunications infrastructure failure on emergency services. The important conclusion is that lost communications negatively impacts medical services and drastically increases treatment costs.

Similarly, Arboleda, *et al.* [2] have addressed the impact of failures of interdependent infrastructure components on the operation of healthcare facilities. The goal was to determine the unsatisfied demand of interconnected infrastructure systems and the resulting costs using a network flow model. Linear programming was used to assess the level of interdependency between a healthcare facility and the primary infrastructure systems linked to it.

In other work, Arboleda and colleagues [1] examined the internal operating capabilities of healthcare facilities in terms of the interactions between different service areas (emergency room, intensive care unit, operation room and wards). This was performed using a system dynamics simulation model. The goal was to assess the vulnerabilities of a healthcare facility during a disaster. The approach enabled the identification of policies to best mitigate the effects of a disruption.

Arboleda, *et al.* [3] have also integrated a network flow model and system dynamics model. This was done to simulate the impact of infrastructure system disruptions on the provision of healthcare services.

These studies and others make it clear that wise decisions to reallocate and utilize the available resources are vital when dealing with interconnected critical infrastructures. Informed decisions can potentially mitigate death and devas-

tation following natural or human-initiated catastrophes. The decisions must be made on the basis of sound knowledge and experience. In fact, the work described in this paper is motivated by the fact that decisions need to be carefully studied and pre-assessed before they are implemented. Moreover, they must be monitored and modified as the situation evolves. The next section discusses the application of agent-based models to address these issues.

2.2 Agent-Based Modeling

An agent-based model is a system of multiple autonomous decision making entities called agents. The agents are capable of sensing and interacting with each other within a modeled environment based on a set of predefined rules. The rules govern the behavior of the modeled agents and enable them to perform appropriate actions.

Agent-based modeling offers three key advantages in the context of real-world applications [5]. First, it can capture emergent behavior that results from the interaction of individual entities (agents). Second, it facilitates detailed system descriptions by modeling and simulating the behavior of interacting entities. Third, it provides great flexibility to tune the complexity of individual entities to scenarios of interest.

These advantages have encouraged the application of agent-based modeling approaches by the disaster response community, the objective being to enhance disaster mitigation efforts. Atanasiu and Leon [4] have developed a multi-agent system based risk assessment tool for seismic hazards. Their tool, which incorporates an adaptive knowledge base, is designed to help create a risk management plan for better earthquake safety and response. The approach simulates emergency response actions for a set of earthquake scenarios at different urban locations. The results, which are displayed using a geographic information system (GIS), helps improve the quality of decision making. The decisions are typically made post-event for restoration and recovery operations aimed at rehabilitating the damaged infrastructure.

Thapa, *et al.* [18] have proposed an agent-based model for patient information acquisition and real-time decision making during emergencies. The model seeks to promote timely diagnosis and treatment of high-risk patients during emergency situations. The approach engages reinforcement learning in conjunction with an embedded dynamic programming mechanism to evaluate and improve a system value function and its policy.

2.3 Reinforcement Learning

Applications of reinforcement learning in agent-based models have attracted the interest of the critical infrastructure research community. The machine learning technique enables an agent to gain experiential knowledge by interacting with a massive number of disaster scenarios. The trained agent is then able to assist in disaster mitigation.

Wiering and Dorigo [19] have developed an intelligent system that enables decision makers to mitigate the consequences of natural and human-initiated disasters (e.g., forest fires). Such disasters involve many interacting sub-processes that make it difficult for human experts to estimate costs. The system of Wiering and Dorigo uses reinforcement learning to learn the best policy or actions to be chosen in a variety of simulated disaster scenarios.

Su, *et al.* [16] have proposed a path selection algorithm for disaster response management. The algorithm is designed for search and rescue activities in dangerous and dynamic environments. The algorithm engages reinforcement learning to help disaster responders discover the fastest and shortest paths to targeted locations. To accomplish this, a learning agent interacts with a two-dimensional geographic grid model. After a number of trials, the agent learns how to avoid dangerous states and to navigate around inaccessible states.

3. Intelligent Decision Making

This section discusses how a reinforcement learning agent can be used for resource allocation in simulated interdependent critical infrastructures. The scenarios are modeled using i2Sim, a hybrid discrete-time simulator, which can handle vast numbers of interactions with the reinforcement learning agent. The simulated environment is based on an urban community similar to the Downtown Vancouver Model [8]. The model incorporates four electrical power substations (P1, P2, P3 and P4), a water pumping station (W) and infrastructure assets such as venues (V1 and V2) and hospitals (H1 and H2). The continued interactions enable the agent to learn, improve its performance and make optimal decisions.

3.1 i2Sim

i2Sim is a hybrid discrete-time simulator that combines agent-based modeling with input-output production models. The simulator can model and play out scenarios involving interdependent systems. i2Sim is designed as a real-time simulator that can also serve as a decision support tool while a disaster is actually occurring. The simulation capability of i2Sim enables decision makers to evaluate the predicted consequences of suggested actions before they are executed [10].

The dynamic aspects of an i2Sim model are implemented by the movement of tokens between i2Sim production cells (i.e., modeled infrastructures such as power stations) through designated channels (i.e., lifelines such as water pipes). In fact, i2Sim cells and channels correspond to discrete entities in the real world. Figure 2 presents an example i2Sim model.

In i2Sim, each production cell performs a function. A function relates the outputs to a number of possible operating states – physical modes (PM) and resource modes (RM) – of the system. At every operating point along an event timeline, the i2Sim description corresponds to a system of discrete time equations expressed as a transportation matrix (Figure 3). The transportation

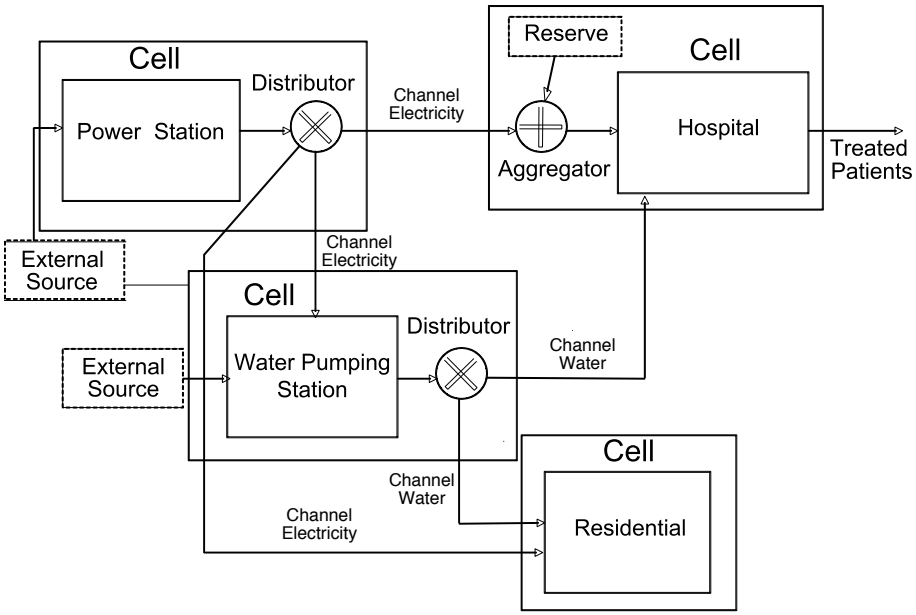

Figure 2. Example i2Sim model [11].

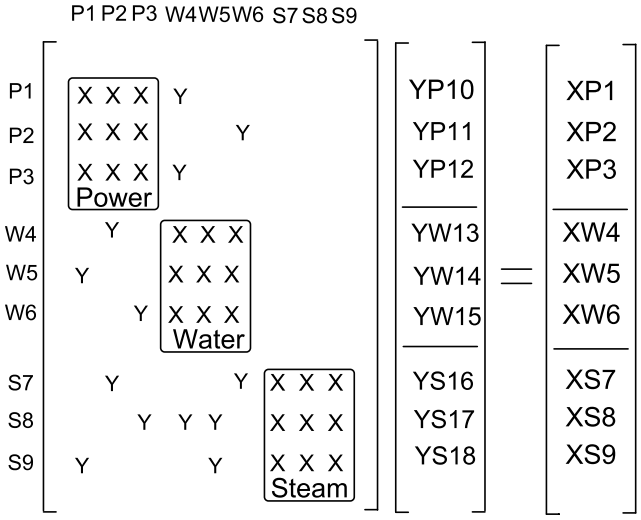

Figure 3. Transportation matrix showing infrastructure interdependencies [11].

matrix shows the interdependencies between the simulated quantities. In particular, the matrix in Figure 3 relates input quantities (XP1, XP2, ..., XS8, XS9) that arrive at the cells with the quantities that are produced as outputs of other cells (YP10, YP11, ..., YS17, YS18). These outputs can be distributed

(via distributors) or aggregated (via aggregators) before being supplied to other cells. For instance, a water pumping station depends on water and electricity that are supplied by other cells (water supply and electrical power station). In row W5 of the transportation matrix, XW5 (water arriving at cell 5) comprises water that outputs from cells YW13, YW14 and YW15 (through X coefficients (internal links)) and power that outputs from cell YP10 (through Y coefficients (interdependent links)). The pumped water is distributed to a number of interconnected critical infrastructures [11].

3.2 Reinforcement Learning

Reinforcement learning [17] is a machine learning technique based on interactions between an agent and its environment. These interactions enable a reinforcement learning agent to maximize a time-delayed goal in the presence of uncertainty. Reinforcement learning occurs through the accumulation of experience, with the goal of finding actions that yield the greatest long-term rewards.

The actions taken in a given situation are determined by a policy realized by an action-value function. In general, reinforcement learning provides three ways of learning the policy: (i) dynamic programming; (ii) Monte Carlo estimation; and (iii) temporal difference. Monte Carlo estimation and temporal difference are the favored methods because they are model free. We have chosen to employ Monte Carlo estimation because it is well suited to learning from episodic problems of the type encountered in the disaster mitigation domain. Experimental results involving similar work [8] reveal that convergence using step-by-step updates as prescribed by temporal difference learning take 2.6 times longer than episode-by-episode based updates as used in Monte Carlo estimation. The goal of the learning agent is to approximate the optimal action-value function leading to the best long-term reward that corresponds to the best trajectory. This recursive-learning algorithm uses incremental episode-by-episode back-ups to solve the well-known Bellman equation [17].

The back-up formula is defined by the following equations for terminal and non-terminal states, respectively:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[R_I + \gamma R_T - Q(s,a)] \text{ (terminal)} \quad (1)$$

$$Q(s,a) \leftarrow Q(s,a) + \alpha[R_I + \gamma max_a Q(s',a') - Q(s,a)] \text{ (non-terminal)} \quad (2)$$

where $Q(s,a)$ is the action-value function of the current state-action pair; $Q(s',a')$ is the action-value function of the next state-action pair; α is the learning rate (i.e., extent to which the newly-required information overrides old information); R_I is the immediate reward; R_T is the terminal reward; and γ is the discount rate (i.e., influence that future rewards have on the learning process).

In Monte Carlo estimation, the back-up equation is used to recursively apply the terminal reward starting at the terminal state and back-stepping all the way

Table 1. Sample lookup table (s: state, a: action).

$(< s >, < a >)$	**Q(s,a)**
$(< PMXP4, RMYP4, PMXW, RMYW >, < DP4, DW >)$	–
–	–
–	–
–	–

to the start state. The estimate is computed by averaging the samples that are returned.

The action-value function can be implemented as a lookup table. The table associates a long-term predicted reward $Q(s, a)$ value with each state-action pair defined for the modeled system. The table represents the acquired experience of the reinforcement learning agent and is updated during the learning process.

Note that the simulated system presents the state of the modeled environment that is detected by the learning agent. In the example considered here, the state is defined using two critical infrastructures: Power Station 4 and the Water Pumping Station. The physical mode (PM) and the resource mode (RM) of Power Station 4 and the Water Pumping Station are specified as PMXP4 and RMYP4 for power, and PMXW and RMYW for water. The values of X and Y range from one to five. When X has a value of one, the modeled infrastructure has no physical damage; when X is equal to five, the modeled infrastructure has collapsed completely. Similarly, when Y has a value of one, all the required resources to maintain the minimum functionality of the modeled infrastructure are available; when Y is equal to five, the required resources are not available.

Table 1 presents a sample lookup table used by the learning agent. In the table, the state-action pairs are captured using the variables: PMXP4, the Power Station 4 physical mode (state); RMYP4, the Power Station 4 resource mode (state); PMXW, the Water Pumping Station physical mode (state); RMYW, the Water Pumping Station resource mode (state); DP4, the Power Station 4 distributor (action); and DW, the Water Pumping Station distributor (action).

3.3 RL-MC Based Learning

The primary contribution of this paper is the application of reinforcement learning with Monte Carlo estimation (RL-MC) to problems involving interconnected and interdependent critical infrastructures. In the RL-MC approach, the problem is formulated as follows: the operating mode (physical mode and resource mode) of each modeled infrastructure unit represents the state of the targeted system. The distribution ratio of the available resources (associated with every modeled critical infrastructure) represents the actions that the agent can perform at every visited state. Every state-action pair is represented by a utility function that estimates the probability of obtaining the long-term reward

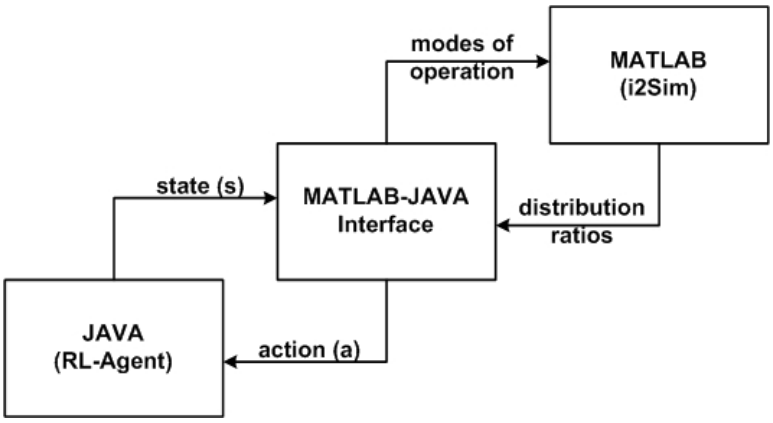

Figure 4. MATLAB-Java dependency diagram.

upon choosing action a in state s. The estimate is computed by averaging the sampled returns over the long term. (In the scenario considered in this paper, the return is the expected number of discharged patients from two hospitals, H1 and H2). Terminal (R_T) and immediate rewards (R_I) are applied in the RL-MC approach.

The learning system (RL-MC) is implemented as a Java program that communicates with the simulator (i2Sim), which is realized in MATLAB (Figure 4). Communications are established via a software interface designed to support data transfer between MATLAB and the Java program. The interface allows the states of the simulated system and actions from the learning agent to be exchanged [7]. From i2Sim, the agent recognizes the state of the simulated environment based on the physical operability (PM) and the resource availability (RM) of the modeled infrastructures. The state is identified by the operating conditions of two critical infrastructures, Power Station 4 and Water Pumping Station (PMXP4, RMYP4, PMXW, RMYW). Accordingly, the agent selects the best distribution ratios (actions) for the distributors associated with Power Station 4 and the Water Pumping Station (DP4 and DW). The chosen action uses the distribution of the monitored resources (power and water) that maximizes the total number of discharged patients from hospitals H1 and H2.

The choice of Monte Carlo estimation over the temporal difference and dynamic programming approaches is also motivated by the need to reduce intra-system communications in the architecture. In a reinforcement learning with temporal difference (RL-TD) approach, communications between the agent and i2Sim (Figure 4) introduce an overhead that is incurred at every time step [8]. In fact, a significant portion of the computational time is due to the MATLAB-Java communications interface alone. In the case of RL-MC, the communications overhead occurs only twice per episode, once at the beginning and once at the end. Consequently, the communication time is reduced by almost a factor of three, which is advantageous when modeling complex systems.

3.4 RL-MC Algorithm

This section provides a technical description of how the RL-MC approach is realized within i2Sim.

In the RL-MC approach, the learning agent interacts with the modeled environment episodically. The agent follows a policy defined by the state-action value function. The agent attempts to learn an optimal policy. This sampling process terminates at a terminal state. At the terminal state, the estimation of the terminal state-action value function $Q(s, a)$ is determined based on the total return that is observed at the end of each episode using Equation (1). This process averages the observed total returns of the visited states in the trajectory. For non-terminal states, the estimation of $Q(s, a)$ occurs by backstepping Equation (2) to all state-action values in the sampled trajectory for each episode. In the limit, the learning agent successfully discovers the optimum trajectory.

The learning system implements the tabular form of the Q-learning algorithm using RL-MC. The lookup table is used to determine the action that is to be performed at the next state of the modeled environment. At any given state s, the learning agent performs an action a that delivers an adequate amount of resources (power and water) to the interconnected infrastructures. Note that 110 actions ($N_a = 110$) and 225 states ($N_s = 225$) are considered in modeled system. Each action a comprises instructions that specify the ratios of the five outputs of the P4 distributor (DP4) and the two outputs of the water distributor (DW). DP4 distributes power from Power Station 4 to five interconnected critical infrastructures: Hospital 1 (H1), Water Pumping Station (W), Venue 1 (V1), Power Station 2 (P2) and other interconnected infrastructures (Oth.). DW distributes the pumped water to the two modeled hospitals, H1 and H2.

The action vector a expresses the distribution ratios of the two distributors:

$$a = \begin{pmatrix} DP4 \rightarrow H1 \\ DP4 \rightarrow W \\ DP4 \rightarrow V1 \\ DP4 \rightarrow P2 \\ DP4 \rightarrow Oth. \\ DW \rightarrow H1 \\ DW \rightarrow H2 \end{pmatrix} \tag{3}$$

Given an initially-untrained lookup table, the RL-MC algorithm seeks to find the optimal action to perform in each state (optimum trajectory). If it is available, real-world experience could be used to initialize the lookup table as a starting estimate of the optimum schedule.

The environment state s is a vector that represents the operating conditions of the two modeled production cells (Power Station 4 and Water Pumping Station). The Power Station 4 and Water Pumping Station states are given by

PMXP4 and RMYP4 for power and PMXW and RMYW for water, respectively. This can be represented at any given time by:

$$s = \begin{pmatrix} PM1P4 & RM1P4 & PM1W & RM1W \\ PM1P4 & RM1P4 & PM1W & RM2W \\ \vdots & \vdots & \vdots & \vdots \\ PM1P4 & RM5P4 & PM5W & RM5W \\ PM2P4 & RM2P4 & PM1W & RM1W \\ PM2P4 & RM2P4 & PM1W & RM2W \\ \vdots & \vdots & \vdots & \vdots \\ PM2P4 & RM5P4 & PM5W & RM5W \\ \vdots & \vdots & \vdots & \vdots \\ PM3P4 & RM5P4 & PM5W & RM5W \\ \vdots & \vdots & \vdots & \vdots \\ PM4P4 & RM5P4 & PM5W & RM5W \\ \vdots & \vdots & \vdots & \vdots \\ PM5P4 & RM5P4 & PM5W & RM5W \end{pmatrix} \tag{4}$$

The first row in the equation above corresponds to state $s_1 =$ (PM1P4, RM1P4, PM1W, RM1W), where PM1P4 is the Physical Mode 1 of Power Station 4, RM1P4 is the Resource Mode 1 of Power Station 4, PM1W is the Physical Mode 1 of the Water Pumping Station and RM1W is the Resource Mode 1 of the Water Pumping Station.

The number of states N_s in the model (total number of rows in vector s) is given by:

$$N_s = Z^K = 15^2 = 225 \text{ states} \tag{5}$$

where Z is the number of resource modes available for each controlled production cell; and K is the number of controlled production cells.

The number of available actions N_a is given by:

$$N_a = DP4 \times DW = 10 \times 11 = 110 \text{ actions} \tag{6}$$

where $DP4$ is the distributor associated with Power Substation 4; and DW is the distributor associated with the Water Pumping Station.

The size of the lookup table L_S is given by:

$$L_S = N_s \times N_a = 225 \times 110 = 24,750 \text{ rows.} \tag{7}$$

The states and actions, as described above, suggest a theoretical maximum lookup table size of 24,750 elements.

As the simulation progresses, the history of actions and immediate rewards of each visited state (according to the policy) are accumulated. The immediate

reward R_I is applied at every time step by computing the difference in the discharged patients between the current and previous states:

$$R_I = (N_{H1} + N_{H2})_{current} - (N_{H1} + N_{H2})_{previous} \quad (8)$$

where N_{H1} is the number of discharged patients from Hospital 1; and N_{H2} is the number of discharged patients from Hospital 2. Note that the intermediate reward is a function of the number of patients discharged.

The terminal reward R_T is calculated and applied at the final time step only (terminal state) based on the total number of discharged patients from both hospitals:

$$R_T = N_{H1} + N_{H2}. \quad (9)$$

Each state-action value $Q(s, a)$ is updated according to Equation (1) for a terminal state or according to Equation (2) for a non-terminal state.

At the end of the episode, according to the RL-MC algorithm, this information and the terminal reward R_T are back-stepped through the sequence of state-action values performed during the episode.

4. Example Scenario

This section uses an example scenario to demonstrate the application of the learning agent to an urban community model simulated by i2Sim (Figure 5). The goal of the agent is to find the optimum trajectory that leads to the maximum outcome. The expectation is that this approach will converge quickly to the maximum number of discharged patients.

4.1 Environment Description

The simulated urban community model consists of nine interdependent critical infrastructure cells. The modeled cells are connected to each other through channels (e.g., underground cables, water pipes and roads). The resources generated by different cells are aggregated or distributed to other interconnected cells by control elements called aggregators and distributors such as power aggregators and water distributors. A pre-defined scenario defines the capacity and the operating parameters (input variables) of the modeled entities. The information is obtained from public domain data or directly from facility managers.

Four power cells are incorporated in the electricity infrastructure: Power Station 1, Power Station 2, Power Station 3 and Power Station 4. The cells determine the amount of power distributed to the system that comes in from the high-voltage supply system. The stations are geographically separated. Each power substation supplies a specific amount of power to its interconnected critical infrastructures. For example, Power Station 4 supplies 586 MW to its connected infrastructures (Hospital 1 and Water Pumping Station).

Similarly, the Water Pumping Station provides water to the connected hospitals (Hospital 1 and Hospital 2). The Water Pumping Station obtains power

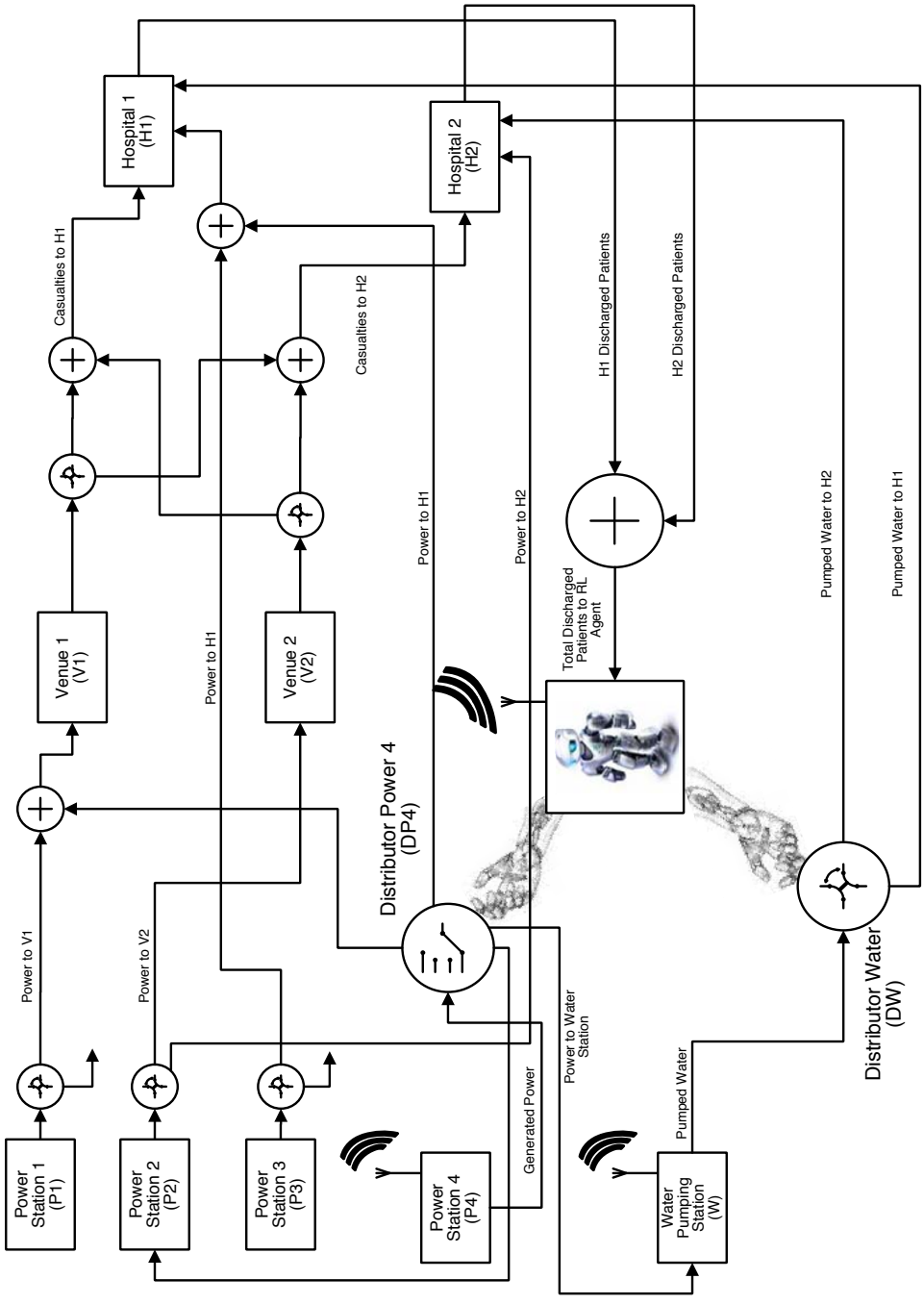

Figure 5. Schematic diagram of i2Sim and RL-MC.

from Power Station 4 and water from an external source. The output of the cell is high pressure pumped water that goes to a water distributor, which distributes the water via water channels (water pipes).

Venue 1 and Venue 2 are facilities that contain large numbers of people. Venue 2 is more than 65,000 m^2 in area and hosts up to 60,000 people. Venue 1 is slightly smaller at about 44,000 m^2 and hosts up to 20,000 people. It is assumed that both venues are hosting events and are fully occupied. Thus, the total population is 80,000.

Two hospitals are modeled, Hospital 1 (main hospital) and Hospital 2 (alternative hospital). The input resources come from the four electrical power stations (electricity) and the water pumping station (water). Based on the availability of these resources, the rate of discharged patients for each hospital is known from historical data.

4.2 Scenario Description

The scenario was configured to reflect the damage caused by an earthquake. The simulated earthquake damaged Power Station 4. The physical structure of the power substation was not affected, but the resource availability RM was reduced due to a failure in one of the electrical feeders, RM2P4. Subsequently, as a result of the reduced electrical power, the water facility was not able to operate at full capacity and the power delivered to the venues and hospitals was also affected.

The earthquake produced casualties due to panic and chaos as people attempted to leave the venues. It was assumed that medical triage at the venues takes an average of 30 minutes per injured person. Upon completion of the assessment, emergency vehicles carried injured people to the emergency units at the hospitals for treatment. The travel time was assumed to be ten minutes.

At the emergency units, all the arriving patients were served on a first-come-first-serve basis. Thirty minutes was assumed to be required to stabilize each trauma patient. The ability of the hospital to function at full capacity was, of course, impacted due to the limited supply of power and water.

The goal of the learning agent was to experience this scenario and to suggest a way to mitigate the impact on the hospitals. This was accomplished by adjusting the distribution ratios of the power and water distributors intelligently, as discussed in the next section.

4.3 Simulation Results

The simulations involved 100 scenarios per test, where each scenario represented a ten-hour period following the disaster event. Upon starting a scenario, the physical operability that represents the damage to the cells and channels was set to model a disaster. Following this, no further changes were made with regard to the extent of the damage. However, the available resources of the associated infrastructures change as the scenario evolves. The lookup table was

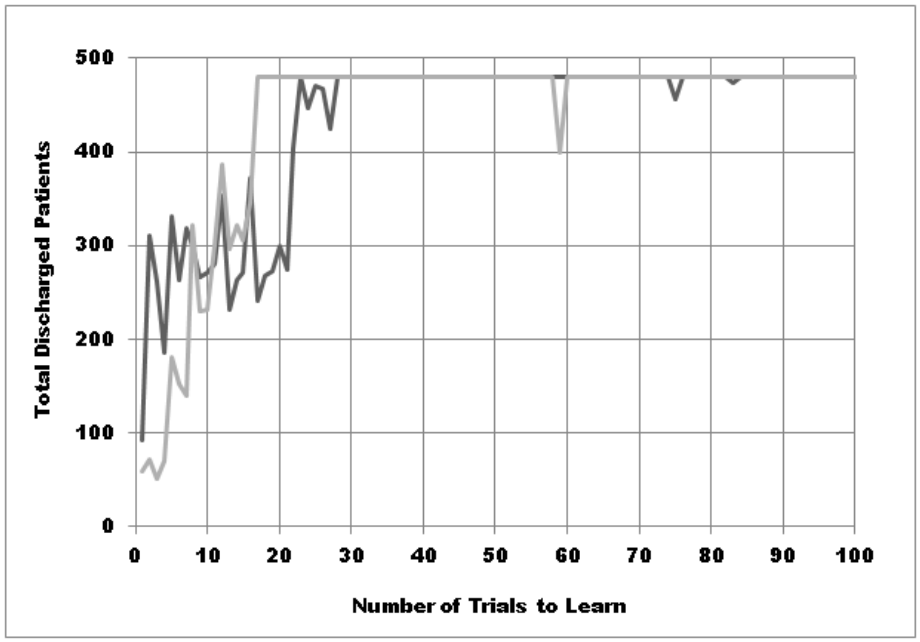

Figure 6. Agent learning behavior under RL-MC.

initialized randomly at the start of the first scenario and learning continued from one scenario to the next.

The model simulated a period of ten hours in five-minute increments. The statistics and system latencies used by the simulator were taken from an internal technical report [7]. The report helped guide the rates used in the simulation. For example, a crowd of 80,000 is expected to have up to 480 injuries.

Figure 6 shows the results for two consecutive sets of trials that were initialized independently (light and dark lines). In both cases, the convergence to an optimum solution occurred and all 480 patients were discharged from both emergency units during the lifetime of the simulation.

During the early phases of learning in both trials, the agent had little or no experience and was unable to maximize the number of discharged patients. However, this was not the case in the later runs, where the agent showed an ability to fully satisfy the demands of the modeled interconnected critical infrastructures by carefully balancing resources across all the infrastructure components.

In contrast, a naive decision maker might select a resource configuration that would only favor the hospitals, but this would be a sub-optimal solution. Instead, the actions taken by the trained agent were those that intelligently utilized the available limited resources (power and water) without exhausting them, which ultimately satisfied the sudden needs of all the interconnected critical infrastructures, including the venues and, most importantly, the hospitals.

The simulation of each scenario required about three minutes using a computer with an Intel Core i5 2.8 GHz CPU and 8 GB RAM. In total, 100 runs took about 600 minutes. This is important in real-world deployments because simulations should be faster than real time in order to assist emergency responders in making informed decisions as a situation unfolds.

4.4 System Deployment

We envisage that the intelligent agent would be deployed as part of a larger software system aimed at providing decision assistance during actual emergencies. The software system would incorporate an i2Sim simulator, a learning agent, a library of pre-configured scenarios and an interface through which a user would interact with the system.

The suggested usage flow would first require the user to identify the disaster taking place in terms of the affected infrastructures. The system would provide a list of scenarios from which the user would pick the best match (instead of defining and entering a new scenario from scratch). In addition to the pre-configured scenario, the system would make available a pre-trained agent for the scenario. Should the scenario match be satisfactory, the human emergency responder can look to the agent for suggested actions in the situation at hand. If the scenario does not match the actual disaster, the user would have to manually adjust the scenario to accurately reflect the real-world situation. The pre-trained agent for the closest-matching scenario could still be used as a starting point.

Using a pre-trained agent is the best option for reducing agent learning time; learning by a trained agent is much faster than when an agent starts with a blank slate. A second approach relies on human knowledge acquisition. Important components of the knowledge and experience of trained emergency responders would have to be identified and captured. The output of this activity would be used to initialize the agent to reduce its learning time.

5. Conclusions

The modeling and analysis framework presented in this paper is an innovative approach for studying the impact of natural or human-initiated disasters on critical infrastructures and optimally allocating the available resources during disaster response. The framework relies on i2Sim and reinforcement learning using Monte Carlo policy estimation (RL-MC). i2Sim permits the simulation of complex interconnected critical infrastructures while the RL-MC approach supports rapid learning based on experiential knowledge in order to provide intelligent advice on allocating limited resources. The experimental results reveal that decision makers can reduce the impact of disruptions by employing the look-ahead and optimization features provided by the framework. The loosely coupled nature of reinforcement learning also enables it to be applied to a variety of resource optimization scenarios.

Our future research will analyze the computational aspects of the learning system. A speed versus accuracy trade-off exists between approaches that use the conventional lookup table implementation of an action-value function and other approaches that use function approximation techniques.

Acknowledgement

This research was partially supported by the Ministry of Higher Education of the Kingdom of Saudi Arabia.

References

[1] C. Arboleda, D. Abraham and R. Lubitz, Simulation as a tool to assess the vulnerability of the operation of a health care facility, *Journal of Performance of Constructed Facilities*, vol. 21(4), pp. 302–312, 2007.

[2] C. Arboleda, D. Abraham, J. Richard and R. Lubitz, Impact of interdependencies between infrastructure systems in the operation of health care facilities during disaster events, *Proceedings of the Twenty-Third Joint International Conference on Computing and Decision Making in Civil and Building Engineering*, pp. 3020–3029, 2006.

[3] C. Arboleda, D. Abraham, J. Richard and R. Lubitz, Vulnerability assessment of health care facilities during disaster events, *Journal of Infrastructure Systems*, vol. 15(3), pp. 149–161, 2009.

[4] G. Atanasiu and F. Leon, Agent-based risk assessment and mitigation for urban public infrastructure, *Proceedings of the Sixth Congress on Forensic Engineering*, pp. 418–427, 2013.

[5] E. Bonabeau, Agent-based modeling: Methods and techniques for simulating human systems, *Proceedings of the National Academy of Sciences*, vol. 99(3), pp. 7280–7287, 2002.

[6] F. Daniel, India power cut hits millions, among world's worst outages, *Reuters*, July 31, 2012.

[7] M. Khouj and J. Marti, Modeling Critical Infrastructure Interdependencies in Support of the Security Operations for the Vancouver 2010 Olympics, Technical Report, Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, Canada, 2010.

[8] M. Khouj, S. Sarkaria and J. Marti, Decision assistance agent in real-time simulation, *International Journal of Critical Infrastructures*, vol. 10(2), pp. 151–173, 2014.

[9] K. Kowalski-Trakofler, C. Vaught and T. Scharf, Judgment and decision making under stress: An overview for emergency managers, *International Journal of Emergency Management*, vol. 1(3), pp. 278–289, 2003.

[10] J. Marti, J. Hollman, C. Ventura and J. Jatskevich, Dynamic recovery of critical infrastructures: Real-time temporal coordination, *International Journal of Critical Infrastructures*, vol. 4(1/2), pp. 17–31, 2008.

[11] J. Marti, C. Ventura, J. Hollman, K. Srivastava and H. Juarez-Garcia, i2Sim modeling and simulation framework for scenario development, training and real-time decision support of multiple interdependent critical infrastructures during large emergencies, presented at the *NATO RTO Symposium on How is Modeling and Simulation Meeting the Defense Challenges out to 2015*, 2008.

[12] J. Marti, E. Yanful and M. Ulieru, Disaster Response Network Enabled Platform, CANARIE Project Final Report, Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, Canada, 2012.

[13] H. Min, W. Beyeler, T. Brown, Y. Son and A. Jones, Toward modeling and simulation of critical infrastructure interdependencies, *IIE Transactions*, vol. 39(1), pp. 57–71, 2007.

[14] G. O'Reilly, H. Uzunalioglu, S. Conrard and W. Beyeler, Inter-infrastructure simulations across telecom, power and emergency services, *Proceedings of the Fifth International Workshop on the Design of Reliable Communication Networks*, 2005.

[15] S. Rinaldi, Modeling and simulating critical infrastructure and their interdependencies, *Proceedings of the Thirty-Seventh Annual Hawaii International Conference on System Sciences*, 2004.

[16] Z. Su, J. Jiang, C. Liang and G. Zhang, Path selection in disaster response management based on Q-learning, *International Journal of Automation and Computing*, vol. 8(1), pp. 100–106, 2011.

[17] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, Bradford/MIT Press, Cambridge, Massachusetts, 1998.

[18] D. Thapa, I. Jung and G. Wang, Agent based decision support system using reinforcement learning under emergency circumstances, *Proceedings of the First International Conference on Natural Computation*, pp. 888–892, 2005.

[19] M. Wiering and M. Dorigo, Learning to control forest fires, *Proceedings of the Twelfth International Symposium on Computer Science for Environmental Protection*, pp. 378–388, 1998.