AquaEIS: Middleware Support for Event Identification in Community Water Infrastructures

Qing Han, Sharad Mehrotra, Nalini Venkatasubramanian
Donald Bren School of Information and Computer Sciences, University of California, Irvine, USA
qhan3@uci.edu,sharad@ics.uci.edu,nalini@ics.uci.edu

Abstract

Real-time event identification is critical in complex distributed infrastructures, e.g., water systems, where failures are difficult to isolate. We present AquaEIS, an event-based middleware tailored to the problem of locating sources of failure (e.g., contamination) in community water infrastructures. The inherent complexity of underground hydraulic systems combined with aging infrastructure presents unique challenges. AquaEIS combines online learning techniques, model-driven simulators and data from limited sensing networks to intelligently guide human participants (e.g., staff) in identifying contaminant sources. The framework integrates the necessary abstractions with event processing methods into a workflow that iteratively selects and refines the set of potential failure points for human-driven *grab sampling*. The integrated platform utilizes Hidden Markov Model (HMM) based representations along with field reports for event inference; reinforcement learning (RL) methods have also shown promise for further refining event locations and reducing the cost of human engagement. Our approach is evaluated in real-world water systems under a range of distinct events. The results show that AquaEIS can significantly reduce the number of sampling cycles, while ensuring localization accuracy (detected 100% of the failure events as compared to a baseline that can only identify 38% of the events).

CCS Concepts • **Information systems** → *Online analytical processing*;

Keywords Event Identification, Event Processing, In-situ Sensing, Human Engagement, Complex System

ACM Reference Format:
Qing Han, Sharad Mehrotra, Nalini Venkatasubramanian. 2019. AquaEIS: Middleware Support for Event Identification in Community Water Infrastructures. In *20th International Middleware Conference (Middleware '19), December 8–13, 2019, Davis, CA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3361525.3361554

Middleware '19, December 8–13, 2019, Davis, CA, USA

ACM ISBN 978-1-4503-7009-7/19/12…$15.00
https://doi.org/10.1145/3361525.3361554

1 INTRODUCTION

With the surge in urban populations (68% projected by 2050) and the rapid development of the global economy, community-scale infrastructures and services have become increasingly complex and vulnerable to failures due to natural, technological and manmade events [1]. Quick and accurate identification of failures are critical to avoid escalation of problems (e.g., NYC blackout 2019, Beijing traffic breakdown 2012 and Flint water crisis 2014), which ensure the health, safety and well-being of citizens. Today, detection of anomalous events in these infrastructures is time consuming and often takes hours or days [31]. Dynamic behaviors arise due to the non-linearity and non-normality of underlying physical models; in many cases observation is limited due to the sparse number and resolution of sensing devices. In this paper, we address the unique challenges in rapidly identifying fault sources in distributed urban infrastructures, using community water systems as a driven use case.

Water distribution systems in communities today consist of large and diverse networks with many different components (pipes, junctions, pumps, valves, tanks and reservoirs). These infrastructures have been deployed over long periods (often decades, centuries); they encompass diverse geophysical regimes and are subject to extreme geologic conditions. As a result, they are inherently vulnerable to a variety of incidents [10]. Physical damage to water infrastructures (e.g., pipe breaks) can disrupt water service to communities; contamination events could result in economic and public health consequences and long-lasting psychological impacts. In recent years, water utilities have had increasing concerns about the possibility of harm due to accidental or intentional contamination of water systems [10]. Public awareness has also increased dramatically due to media coverage since 9/11 [5, 17]. In such scenarios, it is important to quickly identify contaminant sources to ensure the maximum effectiveness of intervention strategies and the minimization of exposure of populations at large to the contaminants.

Current status of water facilities and challenges: Today, cyber instrumenting of water systems is severely lacking and sparse at best; this is due to the fact that such buried

assets were deployed decades ago. Water meters are instrumented primarily for billing, SCADA systems, when available, are deployed at pump stations above ground, and water quality monitoring is mainly present at water storage and treatment plants. The network of hundreds or thousands of pipes is not actively monitored since devices and networks are expensive to install/deploy and maintain. Furthermore, many cities do not yet have the much needed data (e.g., water demand history, population distribution, and risk analysis) to compute an optimal sensor placement [4, 9]. Current sensing technologies also present limitations - while they can detect different types of chemical/microbial contaminants, they often yield binary (yes/no) indications of possible contaminant presence [10]. Lab examinations are required to obtain detailed information - the realistic response times could vary from 6 to 24 hours [23]. All of the above point to the need for improved real-time event identification frameworks for managing failure-prone water infrastructures of the future.

A deeper inspection of water infrastructures points to some inherent systemic complexities. Water systems are large-scale complex networks, where flow patterns are driven by time-varying water demands and distribution systems are looped, resulting in mixing and dilution of contaminants. Mathematically, this source identification problem is the canonical "inverse problem", where we aim to infer the root cause from a set of observations. Inverse problems are often ill-posed, where multiple (near-)solutions may exist [32]. Since source injections can originate at any point throughout the network at any time, the solution space can be very large (roughly number of network nodes times number of historical time steps considered), making it intractable to solve the problem uniquely. The choice of a solution is further complicated by the shortage of measurements as compared to source parameters. A pragmatic approach proposed by civil engineers argues that a unique solution is not needed, one can identify a set of locations for which a contaminant injection is possible [28]. Manual samplings (*grab samples*) can help further reduce the resulting set that is likely to be large due to the low density of in-situ sensing - the human engagement however involves cost and time [36]. This highlights the need for an integrated view of the state of cyber-physical-human systems (CPHS) to support cost-accuracy-time tradeoffs inherent in fault source identification.

Contributions of this paper:

• **Integrated platform for event identification in CPHS:** We propose and develop AquaEIS, a platform that supports a holistic approach to fault source identification in complex distributed settings. AquaEIS integrates multiple sensing modalities (devices, human-as-a-sensor), computing (ML, simulations) and domain knowledge to enable an observation, analysis and adaptation loop in the system. Though inspired by water system resilience, the proposed end-to-end solution can be applied to other "flow/traffic/current"-path-dependent systems. (Sec.2).

• **Event-driven profile generation:** Our approach abstracts the physical nature of phenomena spread (i.e., contaminant transport) into apriori profiles of anomalies represented using *impact matrices*, which are integrated with live readings from multiple data sources for event processing (Sec.2).

• **Online iterative event processing:** AquaEIS is intended to execute in an operational setting on-the-fly, where event processing learns from executions and triggers needed actions to reduce human efforts. At runtime, an iterative event processing strategy is executed in two steps: location inference and location refinement. For inference, we propose an HMM model with a pruning technique to quickly eliminate solutions that are inconsistent with physics (i.e., hydraulics) and measurement information (Sec.3); the refinement step implements a RL based approach used to determine optimal sampling locations such that it can help locate the source within a small number of sampling cycles (Sec.4).

• **Evaluation in real-world water systems:** We evaluate the AquaEIS approach using a complete ensemble of contamination events on two real-world water systems (Sec.5).

2 THE AQUAEIS APPROACH OVERVIEW

In this section, we first provide some background on fault identification in CPHS systems, and present our AquaEIS event identification system (EIS).

2.1 Background and Related Work

CPHS are becoming common in multiple domains (smart-grids, intelligent transportation, healthcare); however, gathering and processing data for operational purposes in such settings have cross-layer concerns at multiple levels of the system (devices, networks, data, compute etc.). This paper focuses on data-driven methods to isolate faults as applied to large scale distributed CPHS systems, e.g., water networks; key topics include modeling the underlying systems and their operations as well as enabling efficient data gathering, data analysis and event processing to support decision making. Integrating human agents in the sensing, analysis and actuation tasks is a viable path to deployment.

In large city-scale infrastructures, effective modeling and understanding of lifeline operations is key to enabling fast fault identification. Techniques for data and knowledge representations using graph-based techniques, ontologies and semantic web approaches have been used to model community data. Workflow processing systems (e.g., Ptolemy) and simulation techniques have been utilized to capture operational aspects and reason about the state of systems. Examples include efforts on transient signal processing for state estimation in power grids [12] and ML techniques for traffic prediction [29] and pipe leak detection [16]. Recently, approaches utilizing HMM [26], that captures the underlying causal dynamics of cascading failures in physical systems,

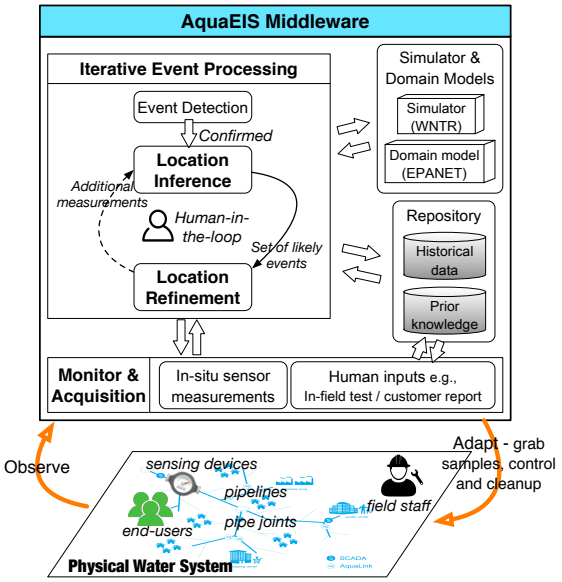


Figure 1. AquaEIS integrated event identification system.

have been successfully applied in multiple domains, e.g., social media [38], transportation [24] and power grids [18]. The area of community scale sensor placement [3, 35] has also gained traction in recent years for a range of fault localization [16, 36] and resilience in extreme conditions [15].

Human-interactive systems have leveraged human capabilities to complex systems to make intelligent decisions for a while. For example, [39] uses crowd-sourced information for timely data delivery and [20] models and analyzes customer behavior for efficient energy management. ML approaches today are becoming available as generalized services and the role of ML such as RL for rapid analysis of complex events is invaluable. The basic idea of RL is to have a learning agent interacting over time with its environment to achieve a goal by capturing the most important aspects of the real problem [33] - this has emerged as a powerful tool for solving sequential decision-making problems. It is well known for its great success in online games - AlphaGo Zero [30], and recently has been successfully applied in natural language processing [11], software development [37], recommendation systems [34] and ridesharing order dispatching [21].

2.2 Enabling Event Identification in Water Systems

AquaEIS (Fig.1) is designed as a sensor-simulation-data integration platform and operated as a real-time event processing service. It composes ML (RL), modeling (HMM), and data collection methods (in-situ and grab-sampling) at different levels of granularity and latency for a comprehensive system. This is required since limited sensing devices only capture phenomena spread (e.g., contamination), not the fault source, in the networked system. Sensor data has inaccuracies; techniques to process the data have uncertainties. Additional sensing in the form of human operators can help reduce these uncertainties by strategic sampling at selected

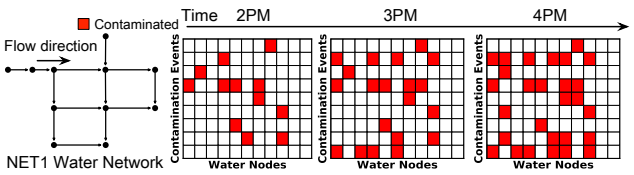


Figure 2. Illustration of impact matrices of NET1

points in the network, which involves cost and time. The first step to enable such a system is accurate modeling of features of the physical system and its operations. Water systems are usually described as a set of links and nodes, where links represent pipes, valves and pumps, and nodes represent reservoirs, tanks, joints and end-users (e.g., building). The contaminant spreads through the system based on the network topology and its flow patterns taking the system from one configuration (state) to another. We represent configurations of interest using an event-based architecture and design abstractions that can capture the state of the system.

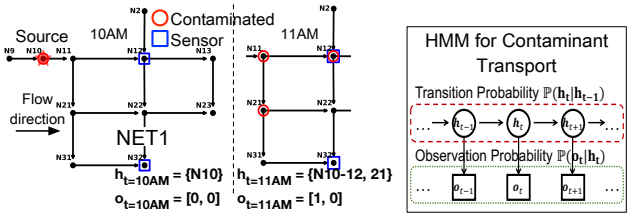
Event-driven profiles generation - In this step, our goal is to capture signatures/profiles of how contamination events manifest in the underlying network. Given a contaminant injected at a certain node (source), the associated time delays and impact on other nodes depend on the transient flow paths and rates, which are made explicit and thus available for use [28]. To incorporate this knowledge, we generate apriori profiles of anomalies using a commercial-grade water quality model - EPANET [27]. Specifically, we simulate a large ensemble of contamination events over a long enough period such that the length of the simulation is as long as the longest travel time from sources to other nodes. A contamination event is an injection of the contaminant at a particular node beginning at a particular time of day and we consider events that can occur at every node in the network starting at any time of a day. It is worth noting that the contaminant injected at same node but at a different time can propagate differently due to time-varying flows. We assume one event at a time, where the contaminant is injected continuously over time. This ensemble of simulations generates profiles of anomalies which we represent using a collection of impact matrices indexed by time The impact matrix is an explicit event-to-impact mapping, which contains contamination events as rows and all network nodes as columns. Entries are given a value of 1 to indicate that the event can contaminate the given node, and a value of 0 otherwise. For example in Fig.2, 9 events are simulated on NET1 water network (from US EPA) by introducing a contaminant at a random node at an arbitrary time. A matrix at 2PM captures contaminated nodes (columns) caused by these 9 events (rows) at 2PM. As time progresses, the matrix may or may not change depending on the flows and events. This collection of matrices can reflect the flow-path-dependent contaminant transport and will be integrated with live readings from in-situ sensors and human inputs for event identification.

Event monitoring and data acquisition - Often, a contamination event is witnessed and detected by in-situ sensing platforms, and the initial identification with sparse and limited measurements is likely to result in a fairly large set of possible sources. Given the inherent complexity of instrumenting underground systems; utilities in practice obtain additional measurements in the form of grab samples through human participants (e.g., field staff, community manager, customer), which can help in narrowing down the resulting set. Humans-as-sensors are indispensable in settings where sensing needs are uncertain and expensive. The advantage of manual grab sampling is its flexibility to provide spatial diversity at significantly reduced cost when compared with in-situ sensors, however, a grab sample at a certain location would only provide a single measurement in time as opposed to continuous monitoring from a sensor. Due to their characteristics, live sensor readings and human inputs are leveraged differently during the event processing.

Online iterative event processing - To enable a timely and reliable fault identification, event processing occurs in an iterative manner in two key steps: location inference and location refinement. When a contamination is detected, an approximate set of source locations is inferred using current available measurements. If the resulting set is large, optimal sampling locations are determined and human participants are invoked to gather additional information. The cycle of collecting grab samples and inferring source locations is continued until the true source of contamination is identified.

Location Inference - In the presence of an unknown contamination event, water quality varies in time subject to constraints imposed by the fluid mechanics. Live sensor readings capture quality changes over time, however, they are spatially sparse and only provide yes/no indications. To better infer the fault source from limited observations, a methodology should incorporate the knowledge of spatio-temporal variations of physical system dynamics. As such, this information allows to rule out the solutions that are inconsistent with the physical laws. The aggregation of information over time can help distinguish the events resulting in the same instantaneous readings. We therefore propose to formulate the problem of source identification as an HMM inference problem, i.e., inferring the unobserved/hidden water quality status based on time series of sensor readings. Human inputs as point measurements are used within a proposed pruning technique. It allows an efficient pruning of the search space by imposing the physically based "structural regularization" on the solution via the apriori generated profiles.

Location Refinement - Due to the limited measurements available at the early stages, the identified set of possible sources may be large or small depending on the sensor configuration and the contamination event. Additional efforts (using grab samples) may be required to reduce this uncertainty. Observe that, depending on the number of likely sources and maximum number of samples that can be taken



(a) Contaminant transport **(b)** HMM

Figure 3. (a) Example of contaminant transport on NET1. (b) Illustration of the HMM for contaminant transport.

at the same time, multiple sampling cycles may be required. Obviously, it is desirable to reduce the resulting set to a tractable number in as few cycles as possible. Therefore, location refinement is considered as a sequential decision problem, where the current selection should learn from previous decisions and consider its "long-term" influence on subsequent ones. However, it is non-trivial for an informed decision making in such a stochastic environment due to the dynamic flows and unknown events. We therefore formulate it using a RL framework, which offers an online mechanism to learn an optimal policy that selects sampling locations with more contributions in the "long run" [19, 33].

3 EVENT LOCATION INFERENCE

We now introduce the first step of event processing, i.e., location inference. Here we use measurements from both in-situ platforms (Sec.3.1) as well as human participants (Sec.3.2), along with pre-built profiles, to estimate contaminant sources. An in-situ sensor or a human input would yield a positive measurement (unsafe) if the contaminant concentration is above a certain threshold and a negative measurement (safe) otherwise.

3.1 Inferring using Live Sensor Readings

Due to the fact that a contaminant spreads through the water network based on the time-varying flows, the propagation behavior is considered as a structured stochastic process. We model it as an HMM and formulate the event identification as an HMM inference problem. We first apply the standard forward-backward algorithm and propose an approximate approach using particle filter that offers a viable alternative for improving the speed of estimation.

3.1.1 Modeling Contaminant Transport using HMM

We consider a water network consisting of V nodes denoted by $\mathcal{V} = \{1, ..., V\}$ and use $v \in \mathcal{V}$ to index node v. The water quality status of a network is represented by a vector $\mathbf{c} = (c_1, ..., c_V) \subseteq \{0, 1\}^V$, where $c_v = 1$ indicates that node v is contaminated. Let $\Gamma(\mathbf{c}) = \{v \in \mathcal{V} : c_v = 1\}$ denote a set of contaminated nodes for quality status $\mathbf{c}$, and $\mathbf{c}_t$ and $\Gamma_t = \Gamma(\mathbf{c}_t)$ denote the status at time t. Since sensors can only provide binary measurements, sensor reading at node v at

time t is denoted by $\boldsymbol{L}_{v,t} \in \{0,1\}$, where $\boldsymbol{L}_{v,t} = 1$ indicates contamination. Let $\mathcal{V}^o \subseteq \mathcal{V}$ denote the subset of nodes where sensors are deployed. The objective is then to infer about quality status Γ_t given observation $\boldsymbol{L}_{\mathcal{V}^o,t}$ over time. To locate the source, we want initial status $\Gamma_{t=0}$. It is worth noting that due to the sparse sensor network, many events may be noticed by same sensor at the early stages, adding to the difficulty of inferring the source. At time t, we model the water quality status Γ_t, which is not directly observed, as the hidden state $\mathbf{h}_t \equiv \Gamma_t$, and the sensor readings $\boldsymbol{L}_{\mathcal{V}^o,t}$ as the observed state $\mathbf{o}_t \equiv \boldsymbol{L}_{\mathcal{V}^o,t}$. For example in Fig.3a, a contamination event occurs at node N10 starting at 10AM on NET1 water network. As the time progresses, nodes may be contaminated ($\mathbf{h}_t$) and sensors may detect the contamination ($\mathbf{o}_t$). In Fig.3b, the evolution of $\mathbf{h}_t$ is formed as a Markov chain with state transition probability $\mathbb{P}(\mathbf{h}_t|\mathbf{h}_{t-1})$ and conditional on $\mathbf{h}_t$, the observation is an independent process following observation probability $\mathbb{P}(\mathbf{o}_t|\mathbf{h}_t)$.

Given a water network, assuming there are N number of hidden states and M number of observed states, an HMM is characterized by the three elements: (a) state transition probability matrix - $A_{N\times N} = \{\mathbb{P}(\mathbf{h}_j|\mathbf{h}_i)\}$ for $1 \leqslant i,j \leqslant N$; (b) observation probability matrix - $B_{N\times M} = \{\mathbb{P}(\mathbf{o}_k|\mathbf{h}_j)\}$ for $1 \leqslant j \leqslant N$ and $1 \leqslant k \leqslant M$; (c) initial state distribution vector - π_0, which are represented as $\lambda = (A, B, \pi_0)$. Matrices A and B can be learnt from the apriori generated impact matrices by the Expectation Maximization algorithm [26]. Here the detail is omitted due to the page limitation. Given an observation sequence $o_{0:T} = (\mathbf{o}_0, \mathbf{o}_1 ..., \mathbf{o}_T)$ and a model λ, our goal is to find a probability distribution $\pi_t(\mathbf{h})$ to estimate the hidden state $\mathbf{h}_t$, where $\pi_t(\mathbf{h}) \equiv \mathbb{P}(\mathbf{h}_t = \mathbf{h}|o_{0:T}, \lambda)$ for $0 \leqslant t \leqslant T$ and T is the length of observation sequences. We use $\mathbb{P}(\mathbf{h}_t = \mathbf{h}|o_{0:T})$ for $\mathbb{P}(\mathbf{h}_t = \mathbf{h}|o_{0:T}, \lambda)$, since λ is fixed once a model is learnt. In particular, to locate the source, we want the state distribution at $t = 0$, i.e., $\pi_{t=0}$. Following the Bayes' rule and the conditional independence of $o_{t+1:T}$ and $o_{0:t}$ given $\mathbf{h}_t$, $\mathbb{P}(\mathbf{h}_t = \mathbf{h}|o_{0:T})$ can be written as

$$\begin{aligned}\pi_t(\mathbf{h}) \equiv \mathbb{P}(\mathbf{h}_t = \mathbf{h}|o_{0:T}) &= \mathbb{P}(\mathbf{h}_t = \mathbf{h}|o_{0:t}, o_{t+1:T}) \\ &\propto \mathbb{P}(o_{t+1:T}|\mathbf{h}_t = \mathbf{h})\mathbb{P}(\mathbf{h}_t = \mathbf{h}|o_{0:t})\end{aligned} \tag{1}$$

3.1.2 The Forward-Backward Algorithm

To solve (1), a short elaboration of standard forward-backward algorithm [25] is first given. This algorithm solves it by recursively computing a set of forward probabilities $\mathbb{P}(\mathbf{h}_t = \mathbf{h}|o_{0:t})$ and a set of backward probabilities $\mathbb{P}(o_{t+1:T}|\mathbf{h}_t = \mathbf{h})$. Given model $\lambda = (A_{N\times N}, B_{N\times M}, \pi_0)$ and a sequence of observations $o_{0:T}$, the forward messages $F_{T\times N}$ can be computed using:

$$F_{t,*} = F_{t-1,*} \cdot A \cdot \mathrm{diag}(\mathbf{o}_t) \qquad 1 \leqslant t \leqslant T \tag{2}$$

where $\mathrm{diag}(\mathbf{o}_t)$ is a diagonal matrix of vector $\mathbf{o}_t$, and $F_{t,i} \propto \mathbb{P}(\mathbf{h}_t = \mathbf{h}_i|o_{1:t})$ for $1 \leqslant i \leqslant N$ with $F_{0,*} = 1/N$. Similarly, the backward messages $R_{T\times N}$ are computed using:

$$R_{t,*} = A \cdot \mathrm{diag}(\mathbf{o}_t) \cdot R_{t+1,*} \qquad 0 \leqslant t \leqslant T-1 \tag{3}$$

where $R_{t,i} \propto \mathbb{P}(o_{t+1:T}|\mathbf{h}_t = \mathbf{h}_i)$ for $1 \leqslant i \leqslant N$ with $R_{T,*} = 1$. The algorithm runs in time $O(N^2T)$. However, we observe that the transition probability matrix A is sparse (Remark 1). This implies that substantial computation on zero entries can be reduced. We thus propose a particle filter based algorithm that exploits this specific structure of our HMM model and can reduce the computational complexity.

Remark 1 [Sparsity in transition matrix] *A feature of contaminant transport is the sparsity of transition probabilities, in the sense that only a small portion of nodes is contaminated in initial stages and with high probability, the new contaminated nodes $\Gamma_{t+1}\backslash\Gamma_t$ only occur on a small subset of nodes.*

3.1.3 A Particle Filter Based Approach

We present a particle filter based approximation, taking the advantage of Remark 1, to estimate efficiently the hidden water quality status. The basic idea is to represent $\mathbb{P}(\mathbf{h}_t|o_{0:T})$ by a set of random particles with associated weights and to estimate $\pi_t(\mathbf{h})$ in (1) based on them [2]. The number of particles P determines the computational complexity, which allows to control the running time by choosing an appropriate P.

Let $\{\mathbf{h}^i_{0:t}, w^i_t\}_{i=1}^P$ denote a set of particles that characterizes $\mathbb{P}(\mathbf{h}_t|o_{0:T})$ where $\{\mathbf{h}^i_{0:t}, i = 1, ..., P\}$ is a set of support points with weights $\{w^i_t, i = 1, ..., P\}$, and $\mathbf{h}_{0:t} = \{\mathbf{h}_\tau, \tau = 0, ..., t\}$ is a set of all states up to time t. The weights are normalized such that $\sum_i w^i_t = 1$. Then, $\pi_t(\mathbf{h})$ can be approximated as

$$\hat{\pi}_{0:t}(\mathbf{h}) = \sum_{i=1}^{P} w^i_t \delta(\mathbf{h}_{0:t} - \mathbf{h}^i_{0:t}) \tag{4}$$

where $\delta(x - x_0)$ is the Dirac delta function with the delta mass at x_0 and $\hat{\pi}_{0:t}(\mathbf{h}) = \{\hat{\pi}_\tau(\mathbf{h}) : \tau = 0, ..., t\}$. Algorithm 1 outlines the proposed particle filter based inference, where the two main steps at each time instant are: (a) sampling particles based on transition probabilities such that the probabilities can be reflected by the frequencies of particles, mathematically expressed as

$$\mathbf{h}^i_t \sim A_{\mathbf{h}^i_{t-1},*} = \mathbb{P}(\mathbf{h}_t|\mathbf{h}^i_{t-1}) \qquad 1 \leqslant i \leqslant P, \tag{5}$$

and (b) updating weights based on the instantaneous observations such that particles can be weighted based on the evidence, mathematically expressed as

$$w^i_t = w^i_{t-1} B_{\mathbf{h}^i_t, \mathbf{o}_t} = w^i_{t-1} \mathbb{P}(\mathbf{o}_t|\mathbf{h}^i_t) \qquad 1 \leqslant i \leqslant P. \tag{6}$$

In this manner, one can approximate $\hat{\pi}_{0:t}$ by augmenting existing particles that constitute an approximation to $\hat{\pi}_{0:t-1}$ with new observations $\mathbf{o}_t$. As the algorithm runs, some weights may become very small. To concentrate on those with large weights, we follow the rule of thumb to resample particles based on their weights $\{w^i_t\}_{i=1}^P$ when the ratio $1/\sum_{i=1}^P (w^i_t)^2$ falls below the threshold $P/2$ [13]. We use systematic resampling scheme, which takes $O(P)$ linear time and minimizes the potential variation [2]. The resampling will

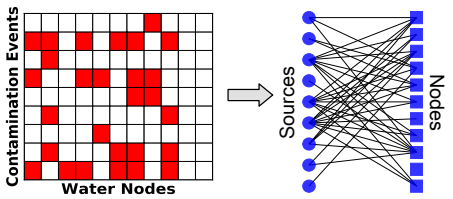


Figure 4. Impact matrix to source-to-node bipartite graph.

generate a new set of particles $\{\mathbf{h}_{0:t}^{i^*}, w_t^{i^*}\}_{i^*=1}^{P}$ and weights are reset to $w_t^{i^*} = 1/P$ since particles are i.i.d samples.

The proposed algorithm runs in $O(PT)$ time, where the computational complexity scales linearly with the number of particles. In addition, according to the law of large numbers theorem, the accuracy of the discrete approximation increases in the order of $O(1/\sqrt{P})$ [7]. By choosing the number of particles, one could have a tradeoff between the computational cost and the estimation accuracy.

Algorithm 1 The particle filter based inference

Input state transition matrix A, emission matrix B, initial state vector π_0, observations $\{\mathbf{o}_t : t = 0, 1, ...\}$
Output hidden state estimation $\hat{\pi}_\tau(\mathbf{h})$ for $0 \leqslant \tau \leqslant t$

Initialization at $t = 0$: draw particles $\{i, i = 1, ..., P\}$ from π_0: $\mathbf{h}_0^i \sim \pi_0$; set weights $w_0^i = P^{-1}$
for $t = 1, 2, ...$ **do**
  for $i = 1, 2, ..., P$ **do**
    Sample particle $\mathbf{h}_t^i$; update weight w_t^i via (5, 6)
  end for
  if $1/\sum_{i=1}^{P}(w_t^i)^2 < P/2$ **then**
    Resample $\{i^j = j\}_{i=1}^{P}, j \in \{1, ..., P\}$ using $\{w_t^i\}_{i=1}^{P}$
    Set $\mathbf{h}_{0:t}^{i^*} = \mathbf{h}_{0:t}^{i^j}$, $w_t^{i^*} = P^{-1}$ for $i^* = 1, ..., P$
  end if
end for
Compute $\hat{\pi}_{0:t}(\mathbf{h})$ using (4)

3.2 Inferring using Human Inputs

The fact that in-situ sensors are sparsely deployed and can only provide binary measurements leads to the issue of event detectability (Remark 2). Therefore, human involvement in data capture is valuable. Human-driven grab samples as single measurements in time are used to prune away inconsistent solutions. As hinted above in Sec.2.2, the flow-path-dependent hydraulics determines if response from a node is positive (unsafe), contaminant sources will be its upstream nodes, while if negative (safe), its upstream nodes can then be excluded.

Remark 2 [Detectability of in-situ sensors] *It is likely that multiple contamination events manifest same sensor readings over time, i.e., $o_{0:t}(\mathbf{h}) = o_{0:t}(\mathbf{h}')$ for $\mathbf{h} \neq \mathbf{h}'$, thus are not distinguishable using only in-situ sensors.*

To better illustrate the pruning process, we first build a bipartite graph for each impact matrix (Fig.4). The bipartite graph is a source-to-node map indicating whether or not a network node can be contaminated by an event occurring at a particular source node but starting at any historical time step. A bipartite graph at time t is $\mathcal{G}_t = (\mathcal{X}, \mathcal{Y}, \mathcal{E})$, where $\mathcal{X} \subseteq \mathcal{V}$ is a set of sources, $\mathcal{Y} \subseteq \mathcal{V}$ is a set of nodes and $\mathcal{E} \subseteq \mathcal{X} \times \mathcal{Y}$ is a set of edges connecting source $i \in \mathcal{X}$ and node $j \in \mathcal{Y}$, namely an event occurring at i can have an impact on j at time t. We refer an edge $e \in \mathcal{E}$ as (i, j) where $i \in \mathcal{X}$ and $j \in \mathcal{Y}$, and $\mathcal{G}$ as $\mathcal{G}_t$ in general. Let $U(j)$ define the upstream reachability set of nodes $j \in \mathcal{Y}$, that is $U(j) = \{i : (i, j) \in \mathcal{E}, i \in \mathcal{X}\}$. Then, we can use the response from node j, which is either positive or negative, to determine if its upstream nodes $U(j)$ are possible sources (positive) or must be excluded (negative). Algorithm 2 describes this pruning operation, which is straightforward. Given a set of nodes with positive response PR and a set with negative response NR, graph $\mathcal{G}$ can be pruned by taking the intersection of sources that are upstream reachable from PR and removing those that are upstream reachable from NR.

Algorithm 2 The pruning process

Input a source-to-node map $\mathcal{G} = (\mathcal{X}, \mathcal{Y}, \mathcal{E})$, nodes with positive response PR and negative response NR.
Output a pruned map $\mathcal{G}' = (\mathcal{X}', \mathcal{Y}', \mathcal{E}')$.

Initialization $\mathcal{X}' = \emptyset$
Update $\mathcal{X}'$ using positive response $\mathcal{X}' = \bigcap_{j \in PR} U(j)$
Update $\mathcal{X}'$ using negative response $\mathcal{X}' \setminus = \bigcup_{j \in NR} U(j)$
Generate $\mathcal{G}'$ by pruning $\mathcal{G}$ using $\mathcal{X}'$

4 EVENT LOCATION REFINEMENT

After event inference is performed following the detection of a contamination (often by in-situ sensors), it is likely that the identified set of potential sources is fairly large due to the limited measurements. Additional efforts in the form of manual grab samples will be required to help refine the identified set, such that countermeasures can be executed effectively. This motivates us to design an intelligent strategy to provide optimal decisions on sampling locations. We first define the location selection of grab samples as a sequential decision problem and motivate a reinforcement learning (RL) framework for solving it. We discuss two challenges while applying RL and finally propose a RL based approach integrated with an entropy-constraint filter.

4.1 The Location Selection Problem

Recall that Sec.3.2 introduced the source-to-node graph with an edge indicating that a contaminant injected at a particular node can have an impact on a certain node. Given a set of sources $\mathcal{X}'$ identified by location inference at time t, a source-to-node map is $\mathcal{G}_t = (\mathcal{X}', \mathcal{Y}, \mathcal{E})$. The objective is to select

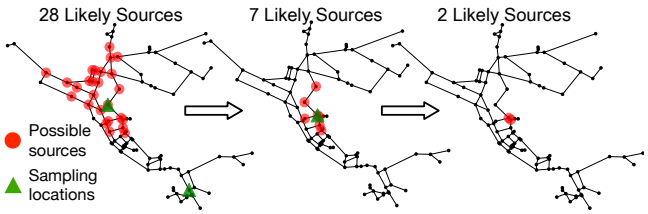


Figure 5. Example of sequential decisions on grab-sample locations with maximum 2 samples allowed at a time.

location(s) $\{j, j \in \mathcal{Y}\}$ based on $\mathcal{G}_t$ over time to reduce $\mathcal{X}'$ in as few sampling cycles as possible, subject to the maximum number of samples that can be taken at the same time. Figure 5 shows an example of 2 sampling cycles with maximum 2 grab samples allowed at a time, where given a set of 28 likely sources, 2 grab samples are taken which reduces the set to 7, followed by another round of sampling with 1 grab sample, and finally 2 possible sources are located.

Intuition behind RL approach - One could consider that the optimal sampling location in each cycle is a set of nodes with the maximum entropy on graph $\mathcal{G}$. It is possible to quickly narrow down the search space by querying nodes with highest uncertainty (maximal information) [6]. However, entropy itself for this problem suffers from two drawbacks: (a) we observe that due to the physical nature of contaminant transport, multiple (collections of) nodes can have same maximum entropy, but result in different source locations. This variance raises the question how to select node(s) from those with maximum entropy which can identify the true source. (b) More importantly, reducing uncertainty greedily is not completely aligned with our objective to seek an optimal *sequence of locations over time*. To address above issues, we propose to apply the RL framework, which is learning to optimize not only the immediate decision but also the next and, through that, all subsequent decisions [33].

4.2 Modeling Location Selection using RL

Figure 6 represents a comprehensive view of adapting RL framework into our problem of selecting sampling locations, where the learner and decision maker together is called *agent*, and the thing it interacts with, comprising everything outside the agent, is called *response environment*. The agent selects grab-sample locations and the response environment responds to it based on the abstracted source-to-node graph. The environment gives rise to numerical rewards that the agent seeks to maximize over time through its choice of sampling locations. In this manner, the agent is able to sense the state of response environment to some extent, takes actions that affect the state and meanwhile has a goal relating to it. The internal components of Fig.6 will be explained following.

To frame the problem of learning from executions to achieve a goal, we first define four key components:

State: At time step t, the state of response environment $s_t \in \mathcal{S}$ is defined as the source-to-node graph $s_t \equiv \mathcal{G}_t = (\mathcal{X}, \mathcal{Y}, \mathcal{E})$, where $\mathcal{S}$ is the state space. Specifically, a state s contains a set of sources $\mathcal{X}$ and a set of possible impacted nodes $\mathcal{Y}$ that are considered as potential grab-sample locations. The initial state s_0 is determined by the estimate from the location inference step.

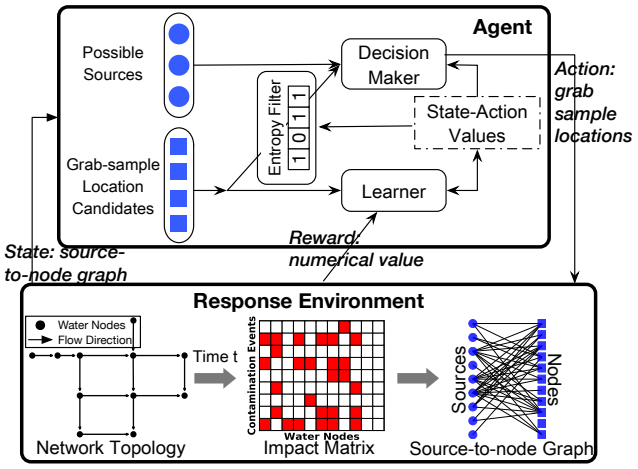


Figure 6. The comprehensive view of RL based approach with an entropy filter for sequential location selections.

Action: On the basis of state s_t and maximum number of samples that can be taken at the same time θ, the agent takes an action $a_t \in \mathcal{A}(s_t)$, where a_t is a set of nodes from which grab samples are taken, $\mathcal{A}(s_t)$ is a set of action candidates (combinations of nodes in $\mathcal{Y}$ based on θ) and $\mathcal{A}$ is the action space.

Reward: One step later at $t+1$, in part as a consequence of action a_t, the agent receives a numerical reward $r_{t+1} \in \mathcal{R} \subset \mathbb{R}$ and finds itself in a new state s_{t+1}. $\mathcal{R}$ is the reward space - a subset of real numbers $\mathbb{R}$. Since the aim is to locate the source in as few sampling cycles as possible, we count a penalty (negative reward) of -1 for each sampling cycle and give a delayed reward until the terminal state. A terminal state is either a single source or a set of indistinguishable sources (i.e., have exactly same impact at this time step, in other words, form a complete bipartite graph). A delay reward is given below, assuming true source is TS and $s_t = \mathcal{G}_t = (\mathcal{X}, \mathcal{Y}, \mathcal{E})$:

$$r_t = \begin{cases} 1/|\mathcal{X}| & \text{if } TS \in \mathcal{X}, |\mathcal{X}| < \text{threshold} \\ -\sum_{i \in \mathcal{X}} \text{dist}(i, TS) & \text{otherwise} \end{cases} \quad (7)$$

where s_t is a terminal state, $\text{dist}(i, j)$ computes the shortest distance in terms of number of links (pipes) between nodes i and j and the threshold is to encourage the agent to reduce the set of likely sources as much as possible, ideally, a singleton with one source. A negative reward is given if the first condition of (7) is not satisfied and closer to the true source receives a higher reward. This is to take care of the situations where the contamination event can not be located in a reasonable number of cycles. Instead, nodes that are close to it will be considered. The reward function allows to reflect different criteria that users are interested in. Other

metrics such as population exposed to a contaminant and the length of contaminated pipe can also be included.

Policy: A policy defines the agent's way of behaving by mapping states to probabilities of selecting each action, i.e., a probability distribution over set $\mathcal{S} \times \mathcal{A}$. The policy function $p(a|s) \equiv \mathbb{P}(a_t = (x_1, ..., x_{\theta'})|s_t = \mathcal{G}_t)$ takes as input the current state and outputs the probability of actions. The policy is a stochastic rule by which agent can explore the action space and select appropriate actions as a function of states.

According to the definitions above, the agent's goal at t is to select an action so as to maximize the cumulative reward in the long run:

$$J_t \equiv \sum_{k=1}^{T} \gamma^{k-1} r_{t+k} \tag{8}$$

where T is the final step (reach to a terminal state) and $\gamma \in [0, 1]$ is called the discount rate. The concept of discounting is to control how strongly the agent takes future rewards into account. As γ approaches 1, the agent will strive for a long-term high reward. To train the agent to execute an action in a given state toward the goal, we define a state-action-value function for policy p as q^p, i.e., $q^p : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$. If the agent is following policy p, then $q^p(s, a)$ is the expected reward return starting from state s taking the action a:

$$q^p(s, a) \equiv \mathbb{E}_p[J_t|s_t = s, a_t = a]. \tag{9}$$

The objective is to find the optimal state-action values $q^p(\cdot)$ following policy p that maximize the expected return for all state-action pairs, which is defined as:

$$q_*(s, a) \equiv \max_p q^p(s, a) \quad \text{for all } s \in \mathcal{S} \text{ and } a \in \mathcal{A}(s). \tag{10}$$

4.3 An efficient RL-based Approach

We first discuss two observations (Remarks 3 and 4) when solving the optimization problem in (10) and propose an efficient online temporal difference (TD) based learning approach. Once the optimal state-action values are learnt, grab-sample locations can be selected based on it.

According to the Bellman optimality equation [8], for a specific state-action pair (s, a), (10) can be rewritten by substituting (8) and (9) as:

$$\begin{aligned} q_*(s, a) &= \max_p \mathbb{E}_p[J_t|s_t = s, a_t = a] \\ &= \max_p \mathbb{E}_p[r_{t+1} + \gamma J_{t+1}|s_t = s, a_t = a] \\ &= \mathbb{E}[r_{t+1} + \gamma \max_{a' \in \mathcal{A}(s_{t+1})} q_*(s_{t+1}, a')|s_t = s, a_t = a] \\ &= \sum_{s' \in \mathcal{S}, r \in \mathcal{R}} f(s', r|s, a)[r + \gamma \max_{a' \in \mathcal{A}(s')} q_*(s', a')] \end{aligned} \tag{11}$$

where $f(s', r|s, a) \equiv \mathrm{P}(s_{t+1} = s', r_{t+1} = r|s_t = s, a_t = a)$ is the probability of those values (s', r) occurring at step $t + 1$, given preceding (s, a). To solve (10), we instead need to find a solution of (11) for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$ in an efficient manner.

Remark 3 [Stochastic response environment] *The response environment is stochastic because of the time-varying flow patterns and the presence of unknown contamination events.*

This uncertainty makes it infeasible to obtain the probability distribution of $f(\cdot|\cdot)$ in an explicit form. However, it is possible to estimate $f(\cdot|\cdot)$ through sequences of states, actions and rewards from actual or simulated interactions with the response environment. This sequence, called an *episode* in RL, likes this: $s_0, a_0, r_1, s_1, a_1, r_2, ..., s_{T-1}, a_{T-1}, r_T, s_T$ (terminal state). With a certain amount of these episodes, we can average the reward returns for each state-action pair and estimate the distribution of $f(\cdot|\cdot)$.

Remark 4 [Large state and action space] *The state space $\mathcal{S}$ and action space $\mathcal{A}$ are combinatorial and can be enormous. Generally speaking, a water network with V nodes can have $|\mathcal{S}| = 2^V$ numbers of possible sources and given a state $s = \mathcal{G} = (\mathcal{X}, \mathcal{Y}, \mathcal{E})$, there are $|\mathcal{A}(s)| = \sum_{i=1}^{\theta} i^{|\mathcal{Y}|}$ numbers of candidates from which grab samples can be taken.*

The problem with large $\mathcal{S}$ and $\mathcal{A}$ is not just the memory needed for a large set of state-action values $q(\mathcal{S}, \mathcal{A})$, but the time and data needed to compute them correctly. Consider a sequence of T sampling cycles, it may need $|\mathcal{A}(s_0) \times \mathcal{A}(s_1) \times \cdots \times \mathcal{A}(s_T)|$ numbers of trial-and-error learning to encounter every state and make a good decision. It quickly makes the search space intractable for a real-world water network (usually with > 100 nodes). We therefore propose a TD based approach - n-step expected Sarsa (NESarsa) integrated with an entropy-constraint filter. The TD methods can learn the distribution of $f(\cdot|\cdot)$ directly from repeated random interactions between the agent and its response environment (resolve Remark 3). One advantage is that TD methods are naturally implemented in an online, fully incremental fashion [33]. The entropy-constraint filter can help significantly reduce the state/action space by eliminating the nodes with lower entropy (less uncertainty), such that the learning speed can be improved (resolve Remark 4).

Mathematically, to find an optimal solution of (11), NE-Sarsa learns and updates the state-action value q for policy p recursively along with the interactions using

$$q_{t+n}(s_t, a_t) = q_{t+n-1}(s_t, a_t) + \alpha(s_t, a_t)[J_{t:t+n} - q_{t+n-1}(s_t, a_t)] \tag{12}$$

for $0 \leq t < T$, where T is the last step of an episode, $n \geq 1$ is the number of steps and $\alpha(s, a) \in (0, 1]$ is the learning rate. We set $\alpha(s, a)$ the reciprocal of the number of occurrences of (s, a). As such, the agent can update $q(\cdot)$ values largely depending on new information at the beginning due to no prior knowledge and adjust it slowly as more information available. The subscripts on $J_{t:t+n}$ indicate a truncated expected return for t using rewards up until $t + n$, instead of the full return shown in (8),

$$J_{t:t+n} = \begin{cases} r_{t+1} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n E_{t+n-1}(s_{t+n}) & t + n < T \\ J_t & t + n \geq T \end{cases} \tag{13}$$

and $E_t(s)$ is the expected reward return of state s.

$$E_t(s) = \sum_{s \in \mathcal{A}(s)} p(a|s) q_t(s,a) \tag{14}$$

In order to find optimal $q(\cdot)$ values for $\{(s,a) : s \in \mathcal{S}, a \in \mathcal{A}\}$ using limited computational resources within a certain time, we propose an entropy-ϵ-greedy policy. Here we filter out nodes carrying less information (small entropy on $\mathcal{G}$) and focus on the exploration and exploitation of relatively unpredictable nodes. The ϵ-greedy ensures that we not only exploit an action that has maximal estimated q value (greedy action), but also explore alternative actions with probability ϵ (non-greedy actions). Therefore, every action a has a chance of being executed in each state s, i.e., $p(a|s) > 0$. As illustrated in Alg.3, given a state s and action candidates $\mathcal{A}(s)$, we create a new set of candidates $\mathcal{A}_{\mathrm{H}}$ by eliminating those whose entropy $\mathrm{H}(\cdot)$ is lower than a given threshold h_o. We then give the minimal probability $\epsilon/|\mathcal{A}_{\mathrm{H}}|$ of selection for all non-greedy actions and the remaining bulk of the probability to the greedy action. Note that $\sum_{a \in \mathcal{A}(s)} p(a|s) = 1$. Algorithm 4 summarizes the learning procedure of an agent through the interactions with its environment. The algorithm terminates when changes on q values are less than a very small number. It is worth noting that this learning converges very fast (often seconds) depending on $\mathcal{X}_0$. The output state-action values are then used for location selection: given a state s, an action a with maximum $q(s,a)$ value will be selected.

Algorithm 3 The Entropy-ϵ-greedy policy

Input state-action values $q(\cdot)$, very small number $\epsilon > 0$, maximum number of samples at a time θ and state s.
Output policy $p(\mathcal{A}(s)|s)$

Initialize $p(a|s) = 0$ for $\forall a \in \mathcal{A}(s)$; $\mathcal{A}_{\mathrm{H}} = \emptyset$
for $a \in \mathcal{A}(s)$ **do**
    if $\mathrm{H}(a) > h_o$ **then** $\mathcal{A}_{\mathrm{H}} = \mathcal{A}_{\mathrm{H}} \cup \{a\}$ **end if**
end for
$a^* = \arg\max_{a \in \mathcal{A}_{\mathrm{H}}} q(s,a)$
for $a \in \mathcal{A}_{\mathrm{H}}$ **do**
    $p(a|s) = \epsilon/|\mathcal{A}_{\mathrm{H}}|$; $p(a|s)$ += $1 - \epsilon$ if $a = a^*$
end for

5 EXPERIMENTAL STUDY

To evaluate AquaEIS approach in practical settings, we implement a functional service for fault source identification that integrates and coordinates multiple sources and entities. This includes real data (from utilities), commercial-grade models (WNTR simulator and EPANET, TEVA-SPOT and CANARY domain models), and policies set by the US EPA in actual settings. We begin by describing the experimental setup and compare the proposed approaches with several existing baselines. Finally we evaluate the effectiveness of AquaEIS using a wide variety of contamination events.

Algorithm 4 The online NESarsa based learning

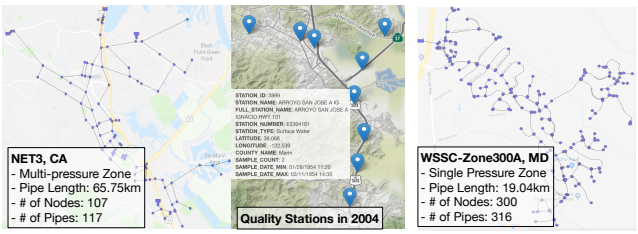
Input a set of sources $\mathcal{X}_0$, entropy-ϵ-greedy policy $p(\cdot|\cdot)$ (Alg.3) and number of steps n.
Output optimal state-action values $q(\cdot)$

Initialize $q(\cdot) = \mathrm{dict}()$
while stop criterion is not satisfied **do**
    Initialize state $s_0 = \mathcal{G}_0 = (\mathcal{X}_0, \mathcal{Y}, \mathcal{E})$; $T = \infty$
    Select action $a_0 \sim p(\mathcal{A}(s_0)|s_0)$
    for $t = 0, 1, 2, \ldots$ **do**
        if $t < T$ **then**
            Take action a_t; compute state s_{t+1} via Alg.2
            if s_{t+1} is terminal **then**
                Compute reward r_{t+1} via (7); $T = t + 1$
            else
                $r_{t+1} = -1$; select $a_{t+1} \sim p(\mathcal{A}(s_{t+1})|s_{t+1})$
            end if
        end if
        $\tau = t + 1 - n$
        if $\tau \geq 0$ **then** Update $q(s_\tau, a_\tau)$ via (12-14) **end if**
        if $\tau = T - 1$ **then** Break **end if**
    end for
end while

5.1 Experimental Setup

Water Network Model - Figure 7 shows two real-world water networks. NET3 (from US EPA) covers the Novato, CA water service area serving 78, 823 people operated by North Marian Water District (NMWD). WSSC-SUBNET (from WSSC) is a subnet of Prince George's County in Maryland serving 1, 137 people operated by Washington Suburban Sanitary Commission (WSSC). These two networks have different characteristics in terms of number of components, pipe properties (diameter, length, etc.), flow patterns and so forth. The average time of contaminant transport from one node to another is around 24min (NET3) or 15min (WSSC-SUBNET). The mean values can be interpreted in the following way: if a node is randomly selected as the contaminant source, one could expect that, depending on the sensor configuration, it needs at least this amount of time to have a new set of measurements. To discriminate the onset of either an anomalous event or natural variations in water quality, the event detection system CANARY needs to examine multiple outliers within a prescribed window [14]. We therefore set the detection window to 75min for NET3 and 45min for WSSC-SUBNET to allow for multiple readings. Once a contamination is confirmed by CANARY, collected data from past 75min (NET3) or 45min (WSSC-SUBNET) will be used to perform the event identification.

In-situ Sensor Configuration - In practice, most water systems are instrumented with a sparse set of sensors with new measurements every 15min. For example, according



(a) NET3 and its stations config in 2004 **(b)** WSSC-SUBNET

Figure 7. Two real-world water networks: (a) a multi-pressure zone of NMWD; (b) a single pressure zone of WSSC.

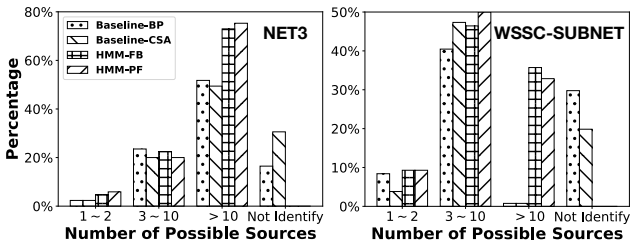


Figure 8. Comparison on distributions of number of identified sources by 2 baselines and 2 HMM based methods on NET3 and WSSC-SUBNET water networks.

to California Department Water Resources, in 2004 NET3 had only 9 quality sensors of 107 water nodes (Fig.7a). The water quality sensor is modeled with contaminant specific detection thresholds and only provides yes/no indications of contamination. In our experiments, the maximum number of sensors is set to 10% of the number of nodes in a network and they provide binary measurements. In this paper, sensors are assumed to perform with 100% accuracy (i.e., no failure) and their locations are generated by TEVA-SPOT.

Contamination Events - They are specified by the location at which the contaminant is introduced into the network and the time of its introduction. Given that it is difficult to predict a specific contamination incident, a large collection of distinct events are used to enhance the validity of results. Specifically, we consider the scenario in which a contaminant is injected over a 24-hour period starting at anytime of a day and every node in the network is considered a potential point of entry. This ensemble of events is generated using WNTR simulator with EPANET water quality model.

5.2 Evaluating the Location Inference Step

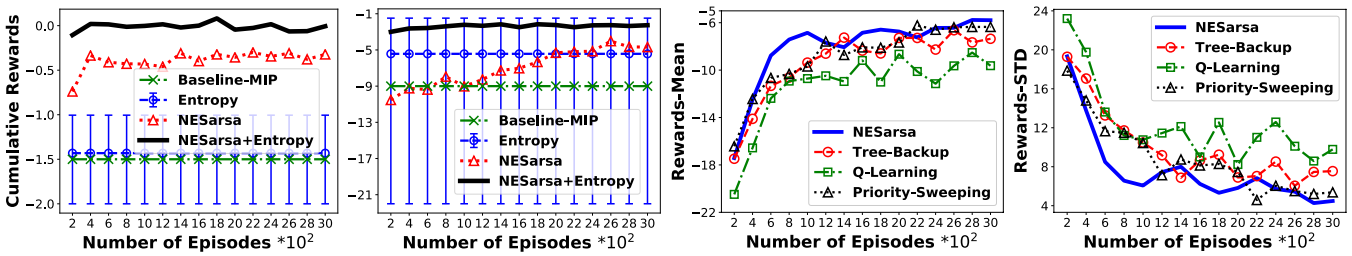
Methods to compare - We first validate our HMM formulation and show the effectiveness of the proposed particle filter based inference. Two existing baselines are compared: Bayesian probability formulation [22] and contaminant status algorithm (CSA) [28]. The Bayesian formulation calculates the probability of a node being the true source based on the deviation between simulated values and measurements, while CSA assigns a status to each node as either safe, unsafe or unknown by tracking the flows over time. We refer to these methods as Baseline-BP (Bayesian probability), Baseline-CSA, HMM-FB (forward-backward algorithm) and our proposed HMM-PF (particle filter based inference).

Performance metrics - Ideally, we want to identify a set of tractable number of sources, such that field staff can directly inspect the possible locations and confirm the true source. However, due to the detectability issue of in-situ sensors (Remark 2), a large set of sources ($PS = \{v : v \in \mathcal{V}\}$) is likely to be identified. This estimate, as the input into location refinement, should include the true source. Otherwise, it may misguide the further processing towards a wrong decision. Therefore, we evaluate the approach by checking if its resulting set PS includes the true source and the smaller the size $|PS|$, the more effective the approach.

Figure 8 shows the distribution of number of possible sources identified by the four methods for an ensemble of $|\mathcal{V}| = 107$ scenarios on NET3 and $|\mathcal{V}| = 300$ scenarios on WSSC-SUBNET (one for each node in the network with an arbitrary injection time). The results show that 2 baseline methods are not able to identify all true sources, where Baseline-CSA fails to identify 31% scenarios on NET3 and Baseline-BP fails to identify 30% on WSSC-SUBNET. HMM-FB and HMM-PF have better, similar performance in the sense that their resulting sets include the true source for 100% scenarios, however they are not small enough for effective inspection. For example on NET3, around 70% scenarios result in more than 10 possible sources by both approaches. It also shows that the number of identified sources can vary considerably depending on the true event location. This confirms the necessarity of integrating additional information to refine the estimate. On average, the running time of HMM-PF is 3 times faster than HMM-FB (Remark 1), where we use $10,000$ and $30,000$ particles for NET3 and WSSC-SUBNET separately. Overall, HMM-PF yields a better performance, which will be used in the following experiments.

5.3 Evaluating the Location Refinement Step

Methods to compare - To validate the feasibility of RL based design, we compare our proposed NESarsa based approaches with an existing Mixed Integer Programming formulation [36] (Baseline-MIP) and a pure entropy approach (Entropy). Baseline-MIP selects sampling locations that maximize the total pair-wise distinguishability of possible sources, while Entropy selects a set of locations that has maximum entropy on aforementioned source-to-node graph with ties broken arbitrarily. To further show the effectiveness of our approach, we implemented 3 other RL methods including Tree-Backup, Q-Learning and Priority-Sweeping for comparison. Tree-Backup and Q-Learning are off-policy algorithms with different learning strategies, while NESarsa is an on-policy algorithm. The key difference is that off-policy methods instead use two separate policies for learning and making decisions to enhance the agent's exploratory behavior. Though more powerful and general, they are often of greater variance and



(a) $|PS| = 4, \theta = 1$ **(b)** $|PS| = 25, \theta = 2$ **(c)** Mean Value **(d)** Standard Deviation

Figure 9. Cumulative rewards versus number of episodes on NET3 for multiple algorithms. (a,b) Comparisons on average rewards with standard deviation error bars. (c,d) Comparisons on mean values and standard deviations of RL algorithms.

slower to converge [33]. In Priority-Sweeping, rather than directly learning the state-action values, it learns a model of the environment and uses this model to plan an action.

Performance metrics - Given an intractable number of possible sources, it is desirable to locate the true source in less number of sampling cycles. We use the cumulative reward defined in (8) for evaluation. Because it takes into account both the number of sampling cycles (measure of cost) and distance between the resulting set and the truth (measure of accuracy). The higher the reward, the better the result. Also, the speed of convergence reflects a measure of time.

Figures 9a/9b show the comparison of multiple methods on average cumulative rewards over 10 simulation runs versus 3000 episodes. For the readability of figures, average values of every 200 episodes are shown. Recall that an *episode* is a sequence of decisions and their rewards (Sec.4.3). Note that Baseline-MIP and Entropy methods do not depend on and thus do not change with the number of episodes.

In Fig.9a, 4 possible sources are identified and the maximum number of grab samples at a time is set to 1. Baseline-MIP achieves an average performance of Entropy that shows a large variance. As hinted in Sec.4.1, multiple (combinations of) nodes can have same maximum entropy due to the hydraulic behavior by nature, which can introduce the randomness in pure Entropy method. Two NESarsa(+Entropy) based approaches converge to higher rewards in 200 episodes, which may not be shown clearly in figures due to the scale of y-axis. Because they are able to learn the stochastic environment (Remark 3) with delayed rewards and have a global perspective on locations from which grab samples can enhance the "long-term" benefit. NESarsa+Entropy is a combination of NESarsa and Entropy-ϵ-greedy policy. In contrast, Baseline-MIP and Entropy suffer from the issue of sequential decisions due to their preferences of local optimal locations. They perform a single optimization analysis that may not contribute to the future decisions. NESarsa+Entropy achieves the best among others with a higher reward and fast convergence (i.e., better performance on cost-accuracy-time tradeoffs). It resolves the problem of entropy ambiguity through trial-and-error learning and overcomes the problem of exponentially large state/action space (Remark 4) by targeting informative nodes using Entropy-ϵ-greedy policy.

In Fig.9b, with 25 possible sources and 2 maximum number of samples, Entropy shows a mean value of -6 with considerable fluctuation, Baseline-MIP yields a stable but lower reward, while NESarsa(+Entropy) converge to the higher rewards (> -6). Compared with Fig.9a, their rewards all decrease, because the increasing number of possible sources results in a larger set of potential contaminated nodes, which requires more sampling cycles to locate a source. Due to the same reason, NESarsa based approaches need more explorations, thus more numbers of episodes, to converge.

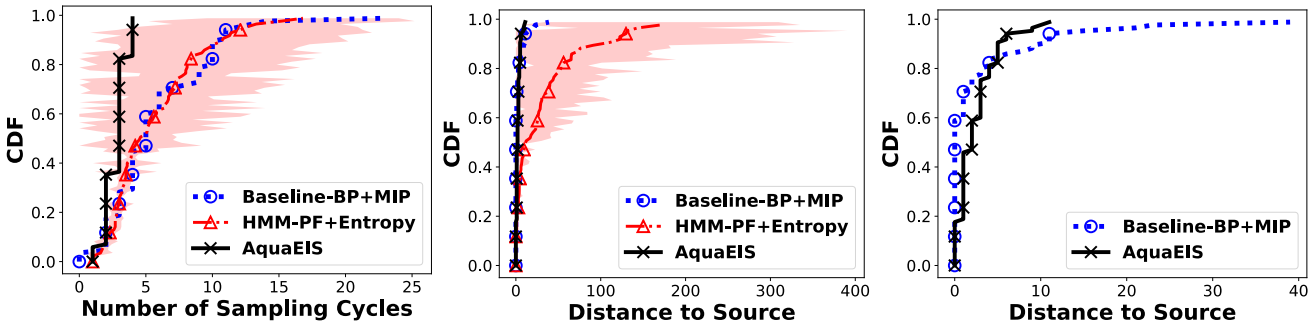
To further provide the insight into our proposed mechanism, Figures 9c/9d illustrate that NESarsa converges faster to a higher reward in comparison with the other 3 techniques. This is possibly because the uncertain environment and large state/action space make it hard for Tree-Backup and Q-Learning to explore using two policies and for Priority-Sweeping to learn an explicit model. It is worth noting that the fluctuation tails appearing in all methods are due to the exploration strategy (ϵ-greedy policy) and the performance will be stable once the state-action values are fixed.

5.4 Evaluation of the end-to-end AquaEIS Approach

Methods to compare - Viewed together, we now evaluate the performance of our AquaEIS approach for a complete ensemble of contamination events. We compare it with an existing work flow (a combination of Baseline-BP and Baseline-MIP (Baseline-BP+MIP)) implemented in Water Security Toolkit [10] and a combination of HMM-PF and pure Entropy (HMM-PF+Entropy) to further validate our design.

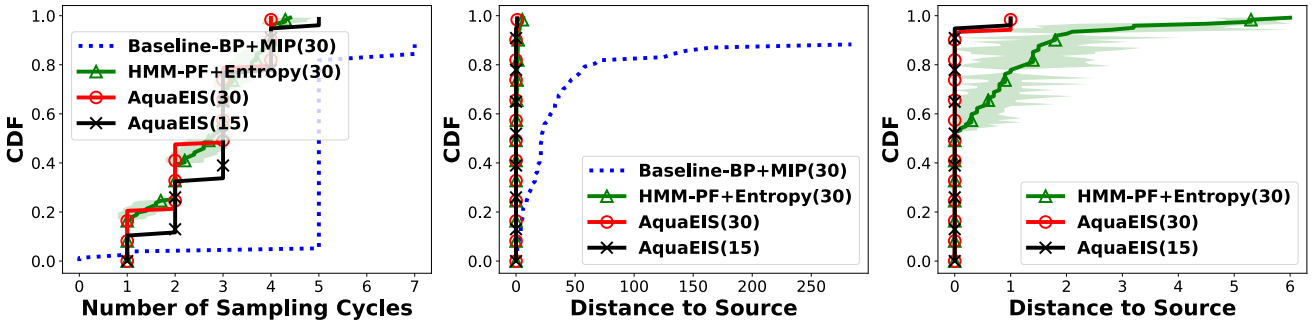
Performance metrics - An efficient event identification system needs to locate the source in a timely manner. We use the number of sampling cycles as a measure of cost and time, and the distance to the true source (shortest path in terms of number of links) as a measure of accuracy. Less number of cycles with a shorter distance indicates a better performance.

In Fig.10/11, each cumulative distribution function (CDF) is computed for a large ensemble of contamination scenarios. The scenario includes a single source at a random node in the network starting at an arbitrary time. Particularly, an ensemble of 300 and 1000 distinct scenarios with different configurations is analyzed on NET3 and WSSC-SUBNET separately. The plots in Fig.10 illustrate the statistical effect



(a) Number of sampling cycles **(b)** Distance to source **(c)** Zoom in of (b)

Figure 10. NET3 - Cumulative distribution function for (a) number of sampling cycles and (b,c) distance to the source.



(a) Number of sampling cycles **(b)** Distance to source **(c)** Zoom in of (b)

Figure 11. WSSC-SUBNET - Cumulative distribution function for (a) number of sampling cycles and (b,c) distance to the source with 15 (5% of nodes) and 30 (10% of nodes) sensors.

of varying the scenario on the performance of approaches in identifying the source, where the maximum number of samples in each round is set to 2. In Fig.10a, AquaEIS is able to identify 100% scenarios in less than 5 sampling cycles, while other 2 methods can only identify 38% in average. HMM-PF+Entropy shows a large variance on those scenarios where more rounds of samplings are needed. With regard to the distance to the true source (Fig.10c), Baseline-BP+MIP locates the true source (distance = 0) for 60% scenarios, however, cannot identify a source (distance > 20) for 5%. AquaEIS can identify a node with distance < 10 for 100% and < 5 for 90%.

Fig.11 shows similar results on WSSC-SUBNET, where AquaEIS performs better with regard to number of sampling cycles and distance to the source. Here the maximum number of samples in each round is set to 1 because of the relatively smaller service area. Additionally, it shows the influence of the number of sensors. In particular, the CDFs are compared for the cases of 15 (5% of nodes) and 30 (10% of nodes) sensors. AquaEIS with less sensors (i.e., AquaEIS(15)) needs more rounds of samplings but can generate same performance on distance-to-source. The increasing number of human-driven grab samples causes the rise of the cost involved, while on the other hand preserves the accuracy as a complementary data source. AquaEIS(15) can exactly locate the true source for 95% scenarios with less than 5 sampling cycles. Furthermore, among all the validated scenarios, solutions using AquaEIS are obtained within a few seconds or minutes, making it a viable approach for real-time event identifications.

6 CONCLUDING REMARKS

In this paper, we presented the design and evaluation of AquaEIS, a middleware support for an efficient event identification in a complex distributed setting (i.e., water systems), which can optimize cost-accuracy-time tradeoffs. AquaEIS integrates and coordinates multiple sensing modalities (devices, human-as-a-sensor) and domain knowledge; algorithms for computation and messaging are implemented using custom modules (Remarks 1-4); recommendations for grab-sampling are generated in real-time. The validation studies presented are performed on an actual implementation with realistic data - our results indicate that we can narrow down sources of failures rapidly enough to allow for utility intervention. We are working towards a more comprehensive system release that includes dashboards for visualization/interaction, and extending the system to address device failures.

Acknowledgments

This work was supported by NSF Awards No. CNS-1143705, CNS-0958520 and CNS-1450768. We are grateful to Dan Hoffman and Lingyi Zhang at Maryland, and all members of DSM and ISG groups at UCI for providing input and datasets for this work.

References

[1] 2018. https://www.un.org/development/desa/en/news/population/2018-revision-of-world-urbanization-prospects.html. *United Nations* (2018).

[2] M Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. 2002. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Trans. on signal proc.* (2002).

[3] J. Berry, E. Boman, L. Riesen, W. E. Hart, C. A. Phillips, J. Watson, and U.S. EPA. 2012. TEVA-SPOT Toolkit and user's manual. *U.S. EPA* (2012).

[4] J Berry, H Lin, E Lauer, and C Phillips. 2006. Scheduling manual sampling for contamination detection in municipal water networks. In *ASCE Proc. Water Dist. Systems Analysis Symposium.*

[5] C. Cameron. 2002. Feds arrest Al Qaeda suspects with plans to poison water supplies. *FoxNews.com* (2002).

[6] Thomas M Cover and Joy A Thomas. 2012. *Elements of information theory.* John Wiley & Sons.

[7] Dan Crisan and Arnaud Doucet. 2002. A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on signal proc.* (2002).

[8] Avinash K Dixit, John JF Sherrerd, et al. 1990. *Optimization in economic theory.* Oxford University Press on Demand.

[9] U.S. EPA. 2004. Response protocol toolbox: planning for and responding to drinking water contamination threats and incidents. (2004).

[10] U.S. EPA. 2014. Water Security Toolkit User Manual. (2014).

[11] Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. Reinforcement learning for relation classification from noisy data. In *32 AAAI Conf. on Artificial Intelligence.*

[12] Georgios B Giannakis, Vassilis Kekatos, Nikolaos Gatsis, Seung-Jun Kim, Hao Zhu, and Bruce F Wollenberg. 2013. Monitoring and optimization for power grids: A signal processing perspective. *IEEE Signal Proc. Magazine* (2013).

[13] Neil Gordon, B Ristic, and S Arulampalam. 2004. Beyond the kalman filter: Particle filters for tracking applications. *Artech House, London* (2004).

[14] J. Hagar, R. Murray, T. Haxton, J. Hall, and S. McKenna. 2013. Using the CANARY event detection software to enhance security and improve water quality. *Proc. of Env & Water Resources Institute* (2013).

[15] Qing Han, Ronald Eguchi, Sharad Mehrotra, and Nalini Venkatasubramanian. 2018. Enabling state estimation for fault identification in water distribution systems under large disasters. In *IEEE 37th Symposium on Reliable Distributed Systems (SRDS).*

[16] Qing Han, Phu Nguyen, Ronald T Eguchi, Kuo-Lin Hsu, and Nalini Venkatasubramanian. 2017. Toward an integrated approach to localizing failures in community water networks. In *IEEE 37th International Conf. on Distributed Computing Systems (ICDCS).*

[17] M. Henneberger. 2002. A nation challenged: suspects; 4 arrested in plot against U.S. Embassy in Rome. *The New York Times* (2002).

[18] Qingqing Huang, Leilai Shao, and Na Li. 2016. Dynamic detection of transmission line outages using hidden Markov models. *IEEE Trans. on Power Systems* (2016).

[19] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. 1996. Reinforcement learning: A survey. *J. artificial intelligence research* (1996).

[20] Jungsuk Kwac, Chin-Woo Tan, Nicole Sintov, June Flora, and Ram Rajagopal. 2013. Utility customer segmentation based on smart meter data: Empirical study. In *IEEE conf. on smart grid communications.*

[21] Minne Li, Yan Jiao, Yaodong Yang, Zhichen Gong, Jun Wang, Chenxi Wang, Guobin Wu, Jieping Ye, et al. 2019. Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning. *WWW* (2019).

[22] Angelica V Mann, Sean A McKenna, William E Hart, and Carl D Laird. 2012. Real-time inversion in large-scale water networks using discrete measurements. *Computers & Chemical Eng.* (2012).

[23] R Murray, T Haxton, R Janke, WE Hart, J Berry, and C Phillips. 2010. Sensor network design for drinking water contamination warning systems. *U.S. EPA* (2010).

[24] Yan Qi and Sherif Ishak. 2014. A Hidden Markov Model for short term prediction of traffic conditions on freeways. *Transportation Research Part C: Emerging Technologies* (2014).

[25] Lawrence R Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. of IEEE* (1989).

[26] Lawrence R Rabiner and Biing-Hwang Juang. 1986. An introduction to hidden Markov models. *IEEE ASSP magazine* (1986).

[27] Lewis A Rossman. 1993. EPANET Water Quality Model. *U.S. EPA* (1993).

[28] De Sanctis, Annamaria E, Feng Shang, and James G Uber. 2010. Real-time identification of possible contamination sources using network backtracking methods. *J. Water Resources Planning Mgmt.* (2010).

[29] Luou Shen and Mohammed Hadi. 2010. *Freeway travel time prediction with dynamic neural networks.* Technical Report.

[30] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *Nature* (2017).

[31] Luigi Spezia, Mark J Brewer, and Christian Birkel. 2017. An anisotropic and inhomogeneous hidden Markov model for the classification of water quality spatio-temporal series on a national scale: The case of Scotland. *Environmetrics* (2017).

[32] Gilbert Strang. 1991. Inverse problems and derivatives of determinants. *Archive for Rational Mechanics and Analysis* (1991).

[33] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction.*

[34] Ryuichi Takanobu, Tao Zhuang, Minlie Huang, Jun Feng, Haihong Tang, and Bo Zheng. 2019. Aggregating e-commerce search results from heterogeneous sources via hierarchical reinforcement learning. *WWW* (2019).

[35] Praveen Venkateswaran, Mahima Agumbe Suresh, and Nalini Venkatasubramanian. 2019. Augmenting in-situ with mobile sensing for adaptive monitoring of water distribution Networks. (2019).

[36] Angelica Wong, James Young, William E Hart, Sean A McKenna, and Carl D Laird. 2010. Optimal determination of grad sample locations and source inversion in large-scale water distribution systems. *Water Dist. Sys. Analy.* (2010).

[37] Ziyu Yao, Jayavardhan Reddy Peddamail, and Huan Sun. 2019. CoaCor: code annotation for code retrieval with reinforcement learning. *WWW* (2019).

[38] Daniel Yue Zhang, Chao Zheng, Dong Wang, Doug Thain, Xin Mu, Greg Madey, and Chao Huang. 2017. Towards scalable and dynamic social sensing using a distributed computing framework.

[39] Qiuxi Zhu, Md Yusuf Sarwar Uddin, Nalini Venkatasubramanian, and Cheng-Hsin Hsu. 2018. Spatiotemporal scheduling for crowd augmented urban sensing. In *IEEE INFOCOM Conf. on Computer Communications.*