DOE Award Number: DE-FE0011403

Project Title:
Distributed Sensor Coordination for Advanced Energy Systems

Final Report
April 29, 2015

Reporting Period: September 13, 2013 - March 12, 2015

Principal Investigator:

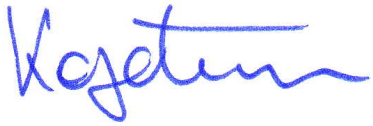
Dr. Kagan Tumer
School of Mechanical, Industrial and Manufacturing Engineering
Oregon State University

Email: kagan.tumer@oregonstate.edu
Voice: (541) 737 9899

Address of Submitting Organization:
Oregon State University
Office of Post Award Administration
Corvallis, OR 97339-1086

Prepared for:
U.S. Department of Energy
National Energy Technology Laboratory

PI Electronic Signature:



Disclaimer:

Contents

1 Abstract

Motivation: The ability to collect key system level information is critical to the safe, efficient and reliable operation of advanced power systems. Recent advances in sensor technology have enabled some level of decision making directly at the sensor level. However, coordinating large numbers of sensors, particularly heterogeneous sensors, to achieve system level objectives such as predicting plant efficiency, reducing downtime or predicting outages requires sophisticated coordination algorithms. Indeed, a critical issue in such systems is how to ensure the interaction of a large number of heterogenous system components do not interfere with one another and lead to undesirable behavior.

Objectives and Contributions: The long-term objective of this work is to provide sensor deployment, coordination and networking algorithms for large numbers of sensors to ensure the safe, reliable, and robust operation of advanced energy systems. Our two specific objectives are to:

1. Derive sensor performance metrics for heterogeneous sensor networks.
2. Demonstrate effectiveness, scalability and reconfigurability of heterogeneous sensor network in advanced power systems.

The key technical contribution of this work is to push the coordination step to the design of the objective functions of the sensors, allowing networks of heterogeneous sensors to be controlled. By ensuring that the control and coordination is not specific to particular sensor hardware, this approach enables the design and operation of large heterogeneous sensor networks. In addition to the coordination coordination mechanism, this approach allows the system to be reconfigured in response to changing needs (e.g., sudden external events requiring new responses) or changing sensor network characteristics (e.g., sudden changes to plant condition).

Impact: The impact of this work extends to a large class of problems relevant to the National Energy Technology Laboratory including sensor placement, heterogeneous sensor coordination, and sensor network control in advanced power systems. Each application has specific needs, but they all share the one crucial underlying problem: how to ensure that the interactions of a large number of heterogenous agents lead to coordinated system behavior. This proposal describes a new paradigm that addresses that very issue in a systematic way.

Key Results and Findings: All milestones have been completed. Our results demonstrate that by properly shaping agent objective functions, we can develop large (up to 10,000 devices) heterogeneous sensor networks with key desirable properties. The first milestone shows that properly choosing agent-specific objective functions increases system performance by up to 99.9% compared to global evaluations. The second milestone shows evolutionary algorithms learn excellent sensor network coordination policies prior to network deployment, and these policies can be refined online once the network is deployed. The third milestone shows the resulting sensor networks networks are extremely robust to sensor noise, where networks with up to 25% sensor noise are capable of providing measurements with errors on the order of 10^{-3}. The fourth milestone shows the resulting sensor networks are extremely robust to sensor failure, with 25% of the sensors in the system failing resulting in no significant performance losses after system reconfiguration.

2 Technical Background

The following sections provide relevant background material, including the Defect Combination Problem, the Rankine Cycle Defect Combination Problem, cooperative coevolutionary algorithms, muliagent reinforcement learning, and difference evaluation functions.

2.1 The Defect Combination Problem (DCP)

Many real world sensing applications require large sets of disparate sensing devices to coordinate their actions in order to collectively optimize their network attenuation, coverage areas, and sensing schedules [10, 14, 17]. In this work, a set of up to 10,000 sensing devices must coordinate their sensing schedules in order to optimize their aggregated attenuation within a sensor networkr. This wok focuses on the Defect Combination Problem (DCP) domain introduced in [2]. This problem assumes that there exists a set of imperfect sensors **X** which have constant attenuations due to manufacturing defects or imperfections. Each of the sensors x_i has an associated attenuation a_i (which can be positive or negative) in its reading, such that if it is taking a measurement of A (actual value) it measures $A + a_i$ where a_i is the device's individual error. The problem then becomes how to best choose a subset of the **X** sensors that minimizes the aggregated attenuation of the combined readings:

$$G = \frac{\left| \sum_{i=1}^{N} n_i a_i \right|}{\sum_{i=1}^{N} n_i} \tag{1}$$

where G is the aggregated attenuation of the combined sensor readings, a_i is the attenuation of a particular sensor i, N is the number of sensors, and $n_i \in \{0, 1\}$ based upon whether the sensor chooses to be "on" or "off".

This is an NP-complete optimization problem [2, 16] and simply choosing the single sensor with the best attenuation is an inadequate solution, as is choosing the best K sensors ($1 \leq K \leq N$). To illustrate this, consider the case where there are 6 sensing devices whose attenuations are $a_1 = -0.19$, $a_2 = 0.54$, $a_3 = 0.1$, $a_4 = -0.14$, $a_5 = -0.05$, and $a_6 = 0.21$. Choosing only the best sensor a_5 would yield an aggregated attenuation of $|0.05|$, while choosing sensors a_3, a_4, and a_5 will yield an aggregated attenuation of $|0.03|$, which is better than the single best sensing device a_5 alone. This is still not the optimal solution in this 6 sensor case however, as combining sensors a_1 and a_6 results in an aggregated attenuation of $|0.01|$. In this problem, individual sensors acting independently without coordinating their actions can drastically decrease the system performance. Consider the case where sensors a_1 and a_6 are turned on in conjunction with sensor a_2, the aggregated attenuation jumps to from $|0.01|$ to $|0.18|$. Finding good solutions requires a great deal of coordination between sensors, as any one sensor can heavily impact the system performance.

2.2 Heterogeneous Sensor Network

We develop a sensor network in a well known power generation system: a vapor power Rankine cycle [6, 12]. The Rankine cycle is one of the simplest thermodynamic power-producing cycles, and serves as a testbed to demonstrate the effectiveness of our approach. It is important to note

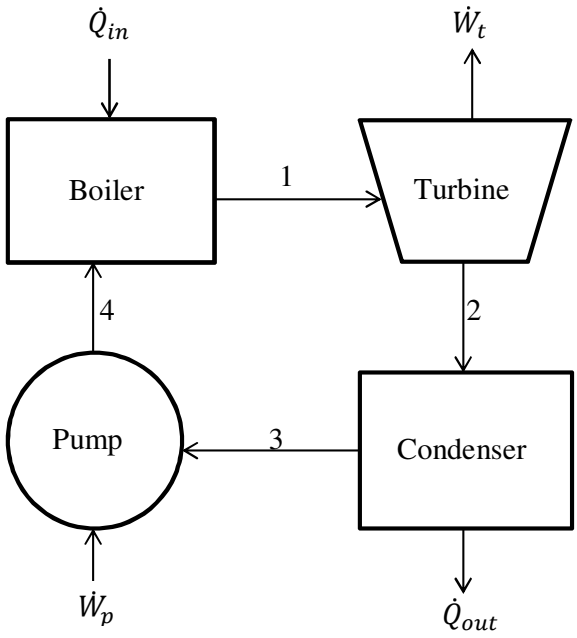


Figure 1: A vapor power Rankine cycle. The working fluid travels through a boiler, turbine, condenser, and pump in succession. The work output of the turbine is used to generate electricity.

that although the Rankine cycle is not complex enough to demonstrate effectiveness of control algorithms, it is adequate to demonstrate the effectiveness of our sensor network training algorithm. Further, the Rankine cycle represents a hardware-based power plant simulation, as we deal with all the typical components (e.g. boiler, turbine, condenser, pump) that we see in steam cycle power plants. In our approach, the model is treated as a black box, and only the output of the model is used to train sensors operating within that model. In a Rankine cycle, the working fluid passes through a boiler and becomes saturated vapor. Next, the fluid goes through the turbine, which results in an energy output which is used to produce electricity. The fluid then passes through a condenser and becomes a saturated liquid. Finally, the fluid passes through a pump and returns to the boiler, completing the cycle. The Rankine cycle is shown in Figure 1. For the purposes of this analysis, we make the following assumptions:

A1. Each component of the cycle is considered to be a control volume.
A2. All processes of the working fluid are internally reversible.
A3. The turbine and pump operate adiabatically.
A4. Kinetic and potential energy effects are negligible.
A5. Saturated vapor enters the turbine. Condensate exits the condenser as a saturated liquid.
A6. The working fluid is water.

As seen in Figure 1, there are four distinct states in the Rankine cycle, each of which lies between two of the components. At each state, the working fluid has an enthalpy h, which is a thermodynamic value indicating the energy stored in the working fluid. The enthalpy of a fluid is a function of temperature and pressure. The system performance is related to the enthalpy h_s at each plant

state s by the following relations:

$$\frac{\dot{W}_t}{\dot{m}} = h_1 - h_2 \tag{2}$$

$$\frac{\dot{Q}_{out}}{\dot{m}} = h_2 - h_3 \tag{3}$$

$$\frac{\dot{W}_p}{\dot{m}} = h_4 - h_3 \tag{4}$$

$$\frac{\dot{Q}_{in}}{\dot{m}} = h_1 - h_4 \tag{5}$$

where $\dot{m}$ is the mass flow rate of the working fluid, $\dot{W}_t$ is the work output of the turbine, $\dot{Q}_{out}$ is the heat output of the condenser, $\dot{W}_p$ is the work input to the pump, and $\dot{Q}_{in}$ is the heat input to the boiler. In order to evaluate these relations, the enthalpy of the working fluid at each state must be determined by measuring the temperature and pressure at each state; this requires the development of a sensing policy.

We apply a modified version of the DCP to a Rankine cycle power plant. There is a set of motes X_s at each of the four plant states, where $s \in \{1, 2, 3, 4\}$ is the state of the power plant (see Figure 1). Each mote $x_{s,i} \in X_s$ has sensors capable of measuring temperature and pressure, the two parameters needed to determine the enthalpy of the working fluid. The sensors in mote $x_{s,i}$ have a mean temperature attenuation $t_{s,i}$, and a mean pressure attenuation $p_{s,i}$. Further, each sensor has an associated measurement noise defined by the Gaussian distribution, where σ_t and σ_p are the standard deviations for temperature and pressure attenuations respectively. Thus, the temperature and pressure attenuations of the sensors on each mote are given by the following normal distributions:

$$e_{T,s,i} = N(\sigma_t, t_{s,i}) \tag{6}$$

$$e_{P,s,i} = N(\sigma_p, p_{s,i}) \tag{7}$$

Each mote is considered to be an agent. First, an agent decides whether to be "on" or "off." If an agent decides to be "on," then it must decide if it will measure temperature, pressure, or both temperature and pressure. The goal of the agents is to collectively take actions which will minimize the aggregate error in temperature and pressure readings. The aggregate attenuation for temperature at a state s is defined as:

$$g_{T,s} = \frac{\sum_{i=1}^{N_s} n_{s,i} e_{T,s,i}}{\sum_{i=1}^{N_s} n_{s,i}} \tag{8}$$

where N_s is the number of motes in state s, and $n_{s,i} \in \{0, 1\}$ denotes whether mote $x_{s,i}$ is measuring temperature or not. Similarly, the aggregate attenuation for pressure at state s is defined as:

$$g_{P,s} = \frac{\sum_{i=1}^{N_s} n_{s,i} e_{P,s,i}}{\sum_{i=1}^{N_s} n_{s,i}} \tag{9}$$

where $n_{s,i} \in \{0, 1\}$ denotes whether mote $x_{s,i}$ is measuring pressure or not. From Equations 8 and 9, the measured values of temperature and pressure at state s are:

$$T_{s,sensed} = T_s + g_{T,s} \tag{10}$$

$$P_{s,sensed} = P_s + g_{P,s} \tag{11}$$

where T_s and P_s are the true temperature and pressure at state s, respectively. Equations 8 and 9 can not be used to provide feedback to learning agents, because they can not be calculated directly in real-world applications, because the attenuation of each sensor is not known. However, using the system model and knowledge of the control inputs, the enthalpy at each state may be analytically determined. Thus, the enthalpy found from the sensor readings may be compared with the true enthalpy (found with system model) to determine the accuracy of the sensor network.

The enthalpy of the working fluid is a thermodynamic property which quantifies the level of energy in the fluid. Enthalpy change in a fluid corresponds to the fluid either absorbing or expelling energy, and is used to determine power levels in a power cycle. In the Rankine cycle power plant, the control inputs are $\dot{Q}_{out}$, $\dot{W}_p$, $\dot{Q}_{in}$, and $\dot{m}$, and are known values. Thus, using Equations 2 through 5 in addition to the assumptions made about the Rankine cycle, the enthalpy values h_1 through h_4 may be directly determined. The enthalpy at each state is also estimated by the sensor network, where the estimation of enthalpy is defined by:

$$h_{s,sensed} = f(T_{s,sensed}, P_{s,sensed}) \tag{12}$$

where $f(T, S)$ is the enthalpy equation based on thermodynamic empirical data, and $T_{s,sensed}$ and $P_{s,sensed}$ are the sensed temperature and pressure values, as defined in Equations 10 and 11. The error in the enthalpy reading at a given state is thus:

$$h_{s,error} = |h_s - h_{s,sensed}| \tag{13}$$

where h_s is the true enthalpy of state s, found using the Rankine cycle model. The error in enthalpy gives an indication of the effectiveness of the sensors measuring temperature and pressure. The objective of the entire sensor network is to minimize the total attenuation of enthalpy measurements at each state, which is equivalent to maximizing:

$$G(z) = -\sum_{s=1}^{4} h_{s,error} \tag{14}$$

The key difference between this approach and the DCP is the fact that the objective function given by Equation 14 uses data which is readily available in order to judge sensor efficacy. Recall that the DCP objective function (Equation 1) includes individual sensor attenuations, which are extremely impractical to obtain, especially as the size of the sensor network grows. Thus, the modification we have made to the DCP allows for implementation in real-world applications.

2.3 Multi-Target Time-Extended DCP

We create an abstract model of a heterogeneous distributed sensor network known as the Multi-Target Time-Extended Defect Combination Problem (MTTEDCP), shown in Figure 2. The MTTEDCP consists of n_s sensors and n_t targets to be sensed. Each sensor and target has a location $p = (x, y)$, and each sensor i has a sensing radius r and a measurement error a_i. If a target j has a value of A_j and location p_j and is within a sensor i's measurement radius, the measurement returned by the sensor is:

$$m_{i,j} = A_j + a_i + \delta|p_i - p_j| \tag{15}$$

where δ is a noise factor, and $|p_i - p_j|$ is the Cartesian distance from the sensor to the target. Thus, in addition to sensor error, a measurement becomes more inaccurate the farther away the sensor is from the target. The error of a sensor is given by the term:

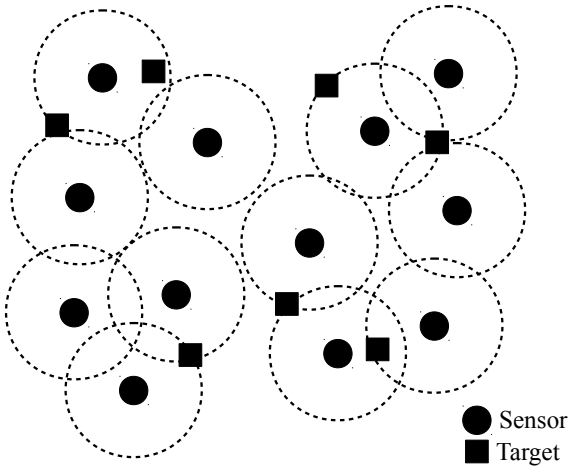


Figure 2: Visualization of the multi-target time-extended defect combination problem. Targets and sensors are distributed in a 2D environment, and each sensor has a limited observation radius (denoted by dashed circles).

$$e_{i,j} = a_i + \delta|p_i - p_j| \tag{16}$$

In the MTTEDCP, each sensor chooses what time-steps it will be active (measuring). When a sensor is inactive, it measures nothing (power conservation mode). When a sensor is active, it is measuring all sources within its observation radius. There are two objectives in the MTTEDCP. First, each source must be measured for every time step; this ensures that the system remains fully monitored continuously. Second, the average measurement error of each source is to be minimized. This minimization involves finding subsets of sensors to be activated such that their aggregate measurement errors is minimized. Thus, the MTTEDCP is a time-extended combinatorial optimization problem subject to a soft constraint of full source coverage. The system evaluation function of the MTTEDCP is given by:

$$G(z) = \sum_{\tau=1}^{\tau_{max}} \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} \frac{e_{i,j}}{f_{i,\tau}} \tag{17}$$

where τ is the time-step, τ_{max} is the max number of time-steps, i is the sensor index, j is the source index, and $f_{i,\tau}$ is an indicator function which is either 0 or 1 if sensor i is "off" or "on" at time-step τ. Solving the MTTEDCP involves choosing which sensors are on at which timesteps in order to minimize average measurement error. Multiple sensors measuring a target not only can improve measurement accuracy, but it increases the robustness of the system due to sensor redundancy. If only one sensor is measuring a target, failure of that sensor results in complete loss of potentially critical data. If multiple sensors are measuring a target, failure of a sensor results in only a change in accuracy of the target measurement, but no complete loss of data. Sensors each have different capabilities (heterogeneous sensors), as they each have different baseline measurement errors and noise levels, corresponding to varying qualities of sensors in the network.

2.4 Cooperative Coevolutionary Algorithms

Evolutionary algorithms (EAs) are a class of stochastic search algorithms which are inspired by biological evolution and contain three basic operators: solution generation, mutation, and selection [11]. These mechanisms are used on an initial set of candidate solutions (a population) to generate new solutions and retain solutions that show improvement. For multiagent settings, EAs are extended to cooperative coevolutionary algorithms, where multiple EAs operate in parallel and the fitness of each agent is a function of how a team of collaborating agents performs. A cooperative coevolutionary algorithm is shown in Algorithm 1.

Algorithm 1: Standard CCEA

```
Initialize N populations of k solutions ;
foreach Generation do
    foreach Population do
        produce k successor solutions ;
        mutate successor solutions
    end
    for i = 1 → 2k do
        randomly select one agent from each population ;
        add agents to team T_i ;
        simulate T_i in domain ;
        assign fitness to each agent in T_i
    end
    foreach Population do
        select k solutions using ε-greedy
    end
end
```

As agent fitness is a function of how a team of collaborating agents performs, fitness assignment is often context dependent and subjective. In order to provide optimal learning performance, fitness functions must be shaped; each agent must have a private objective to maximize which is aligned with the system objective and easy for the agent to change through its actions. Difference evaluation functions have these properties, and are discussed in Section 2.7.

2.5 Reinforcement Learning

Reinforcement learning involves learning how to map states to actions in order to maximize a numerical reward signal [15]. The learning agent is not told which actions to take in any given situation, but must discover optimal actions by testing different actions and finding which action yields the highest reward. In tasks requiring multiple time steps, the learning agent must consider future actions and rewards as well. For example, an action which yields a high immediate reward may lead to states with low rewards; in this case, the agent should select another action which leads to states having better rewards in the future. *Q-learning* allows for agents to learn actions which optimize the future sum of rewards, allowing for optimal policies to be learned. An agent learning with Q-learning keeps a value table (the Q-table) with values corresponding to each state action pair. When an agent takes an action in a particular state and receives a reward, the Q-table is updated according to:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right] \tag{18}$$

where $Q(s, a)$ is the Q-value corresponding to the state action pair (s, a), α is the learning rate, γ is the discount factor, and r_{t+1} is the reward received for taking action a_t in state s_t. Note that the value $Q(s_{t+1}, a)$ is the maximum expected value of the state the agent reaches upon taking the action a_t while in state s_t. This allows for the agent to optimize action selection to maximize the sum of future rewards, rather than simply maximizing immediate rewards. Given enough learning trials, Q-learning is theoretically guaranteed to find the optimal policy to maximize a system evaluation function [15].

2.6 Multiagent Reinforcement Learning

In the case of multiple agents simultaneously learning, each agent maintains a local Q-table and performs Q-learning independently. At each time step, each agent selects an action to execute. The joint action (set of actions executed by all agents) then determines system performance. The general multiagent reinforcement learning algorithm is shown in Algorithm 2. Ideally, agents learn to select actions which result in system wide coordination between agents, in order to optimize the system performance in a distributed manner.

Similar to CCEAs, it is critical that agent feedback in multiagent reinforcement learning be shaped to provide valuable information to learning agents. Agent rewards structures must ensure that agents learning to optimize their private reward functions are also improving overall system performance. One option is to provide each agent the overall system performance as a reward. In this case, each agent is learning to optimize the system reward. However, in large systems (many agents), agents have difficulties separating their impact on the system performance, as many agents are simultaneously affecting this value. Difference Evaluation Functions (Described in Section 2.7) may be used as reward functions in multiagent learning settings. These *Difference Rewards* have two key beneficial properties for multiagent systems. First, they are aligned with the system performance; this means agents maximizing their difference rewards will also improve overall system performance. Second, they are sensitive to agent actions; this means agents receive rewards with a favorable signal-to-noise ratio, and are able to easily determine the effectiveness of their actions.

2.7 Difference Evaluation Functions

In multiagent learning systems, it is critical that agents receive proper feedback regarding the effectiveness of their actions. A set of autonomous agents coordinating to achieve a system level objective results in difficulties in determining the impact of an individual agent on the system. For example, an agent may perform poorly, but the set of agents as a whole may perform well, resulting in high system performance. In this case, it is difficult to determine that the agent's actions were detrimental to the system.

In order to provide accurate agent specific feedback, we implement the *Difference Evaluation Function*, defined as [1, 3, 4]:

$$D_i(z) = G(z) - G(z_{-i} + c_i) \tag{19}$$

where $G(z)$ is the system evaluation function, and $G(z_{-i} + c_i)$ is the system evaluation function without the effects of agent i. The difference evaluation function approximates agent i's contribution to the system performance. Note that the second term in $D_i(z)$ is independent of agent i.

Algorithm 2: Standard Multiagent RL

```
Initialize value table for each agent ;
foreach Learning Episode do
    foreach Agent do
        Select action with ε-greedy selection ;
    end
    Determine system performance G(z) based on agents' actions ;
    foreach Agent do
        if Reward = G(z) then
            Q_i(a_t) ← (1 − α)Q_i(a_t) + G(z)
        end
        if Reward = D_i(z) then
            Q_i(a_t) ← (1 − α)Q_i(a_t) + D_i(z)
        end
    end
end
```

Thus, the difference evaluation function is *aligned* with the system evaluation function. An agent maximizing the value of $D_i(z)$ will also maximize the value of $G(z)$. Further, note that the difference evaluation function removes all portions of $G(z)$ which are independent of agent i. Thus, the signal-to-noise ratio of $D_i(z)$ is very favorable compared to that of $G(z)$, allowing for learning agents to more easily discern the impact of their actions on system performance. In addition to the theoretical properties of alignment and sensitivity, it has been theoretically demonstrated that difference evaluations improve the probability of agents selecting optimal actions in cases where the optimal joint action is deceptive [8].

3 Milestone and Cost Report

3.1 Cost Summary

Table 1: Cost Status

Quarter	**Actual Expense** (in $)	**Projected Expense** (in $)
Sep 13 - Dec 31 2013	38,789	61,000
Jan 1 - Mar 31 2014	67,511	45,000
Apr 1 - Jun 30 2014	57,040	46,000
Jul 1 - Sep 30 2014	48,647	50,000
Oct 1 - Dec 31 2014	45,705	54,000
Jan 1 - Mar 31 2014	42,299	43,992
Total	299,991	299,992

3.2 Milestone Completion

This research led to 6 publications [3, 4, 5, 8, 9, 13]. All milestones have been met. In this section, we provide a summary of each milestone. In the following four sections, we provide details on results supporting each of the milestones.

Milestone 1 (December 2013): Improve effectiveness of heterogeneous sensor networks by at least 10% using new objective functions and evolutionary algorithms for sensor networks ranging from 100 to 1000 devices
Status: This milestone is complete, and is supported by publications [3, 8, 9, 13]. See Section 4 for results.

Milestone 2 (June 2014): Improve effectiveness of heterogeneous sensor networks by at least 10% using new objective functions and using reinforcement learning for sensor networks ranging from 100 to 1000 devices
Status: This milestone is complete, and is supported by publications [4, 9]. See Section 5 for results.

Milestone 3 (December 2014): Improve system efficiency by at least 10% and scalability by at least 100% in advanced power system simulation, for sensor networks ranging from 100 to 1000 devices.
Status: This milestone is complete, and is supported by publications [5, 9]. See Section 6 for results.

Milestone 4 (March 2015): Improve system reconfigurability by at least 10% and scalability by at least 100% in advanced power system simulation for networks with up to 2000 devices. Reconfigurations will be tested with 10%, 25%, and 50% failures/changes in system conditions.
Status: This milestone is complete, and is supported by publications [5, 9]. See Section 7 for results.

4 Improving Sensor Network Efficiency with Evolutionary Algorithms (Milestone 1)

The first milestone of this work is to improve the effectiveness of heterogeneous sensor networks by at least 10% using new objective functions and evolutionary algorithms, in networks ranging from 100 to 1000 devices [7]. We conducted experiments with varying network sizes in the Rankine Cycle Defect Combination Problem, using the cooperative coevolutionary algorithm detailed in Section 2.4. We tested both the global and difference evaluation functions to assign fitness. The global evaluation function is the standard CCEA implementation, used for a baseline comparison. The difference evaluation function is analyzed to determine the performance improvements it can attain over global evaluation functions.

Three experiments involving training heterogenous sensor networks were conducted, with 100, 500, and 1000 sensing devices. The performance of the networks as a function of evolutionary time are shown in Figures 3-5.

As seen in Figures 3-5, the use of difference evaluation functions significantly reduces measurement error in the sensor network. In the 100 agent network, accuracy was improved by an

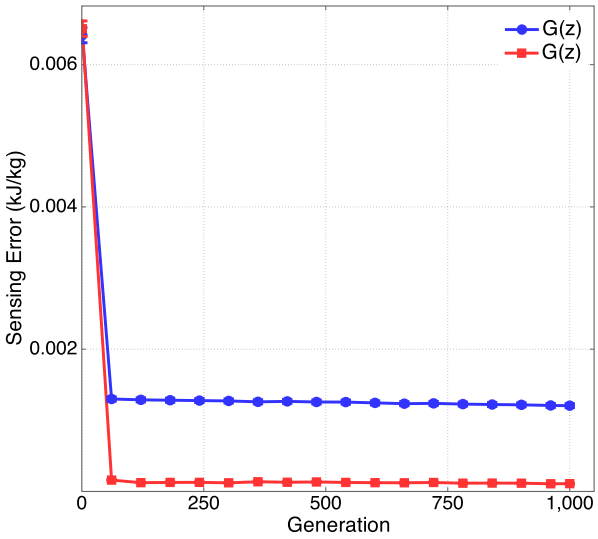


Figure 3: Heterogeneous sensor network performance with 100 agents. Use of the difference evaluation function results in 90.9% improvement in network accuracy.

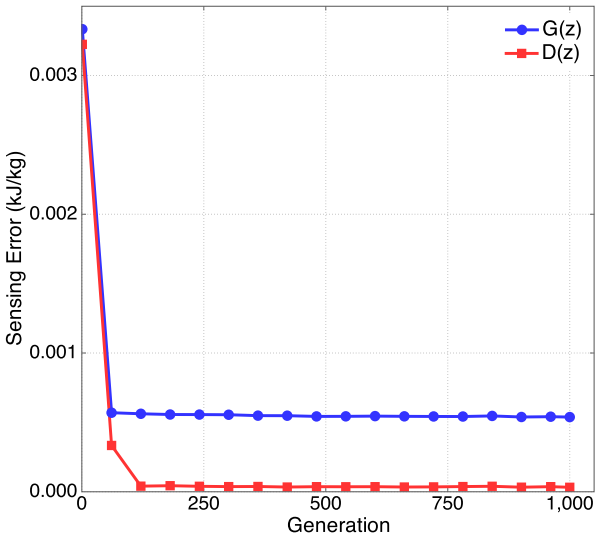


Figure 4: Heterogeneous sensor network performance with 500 agents. Use of the difference evaluation function results in 98.8% improvement in network accuracy.

average of 90.0%. In the 500 agent network, accuracy was improved by an average of 98.8%. In the 1000 agent network, accuracy was improved by an average of 99.9%. There are two interesting conclusions that may be drawn from this information. First, the use of new objective functions (difference evaluations) and evolutionary algorithms can significantly improve network accuracy by up to two orders of magnitude. Second, as the network grows in size and complexity, the benefits of using our algorithm increase; in other words, the more complex the network, the more beneficial learning with difference evaluations becomes. This can be attributed to the fact that agent-specific feedback becomes more valuable in larger systems, where any particular agent has a relatively smaller effect on system performance as a whole.

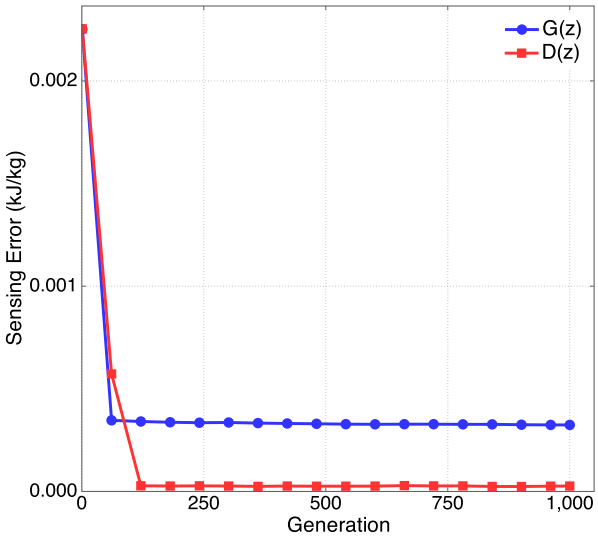


Figure 5: Heterogeneous sensor network performance with 1000 agents. Use of the difference evaluation function results in 99.9% improvement in network accuracy.

5 Improving Sensor Network Efficiency with Reinforcement Learning (Milestone 2)

The second milestone involves improving the effectiveness of a heterogeneous sensor network using new objective functions and reinforcement learning, in varying network sizes. Although the experiments in Section 4 demonstrate that evolutionary algorithms can improve system performance, this approach is limited to offline optimization; in other words, optimizing network performance with evolutionary algorithms requires optimizing the network prior to deployment (or during network downtime). However, reinforcement learning methods can be used to optimize a network which is currently in operation, as a small number of sensors can change their policies online and the effects of these changes can be used to determine if the agents should alter their individual policies.

We now analyze training a heterogeneous sensor network in the Rankine Cycle Defect Combination Problem using multiagent reinforcement learning. The following sections detail reinforcement learning, multiagent reinforcement learning, the algorithm used for training the heterogeneous sensor network, and the empirical results for heterogeneous sensor networks ranging from 100 to 1000 devices.

5.1 Heterogeneous Sensor Network Reinforcement Learning Algorithm

We implement multiagent reinforcement learning in a heterogeneous sensor network in the Rankine Cycle Defect Combination Problem. This domain is stateless, so the agents use an action-value update (simplified form of Q-learning when no state transitions are present), defined as:

$$Q(a_t) \leftarrow (1-\alpha)Q(a_t) + \alpha r_t \tag{20}$$

where r_t is the reward received for taking action a_t, $Q(a_t)$ is the recorded value of taking action a_t, and $\alpha \in [0, 1]$ is the learning rate. Note that higher values of α lead agents to placing more

emphasis on the most recent rewards. The multiagent learning algorithm is detailed in Algorithm 2. Agent actions are selected using ϵ-greedy selection; this means that agents select the best known actions with probability $1 - \epsilon$, and select a random action with probability ϵ.

5.2 Performance of Trained Network

We trained heterogenous sensor networks ranging from 100 to 1000 agents using the learning algorithm detailed in Algorithm 2. For each experiment, action-value tables were initialized by drawing values from a Gaussian distribution with zero mean and unit variance. ϵ was set to 0.1 for action selection. The learning rate α was set to 0.1. Both the system reward and difference rewards were tested as reward signals. Each experiment was conducted for 1000 runs to ensure statistical significance. The results for these experiments are shown in Figures 6-8. Error bars show error in the mean (σ/N), where $N = 1000$ is the number of statistical trials. Note that error bars are often obscured by plot markers and cannot be seen.

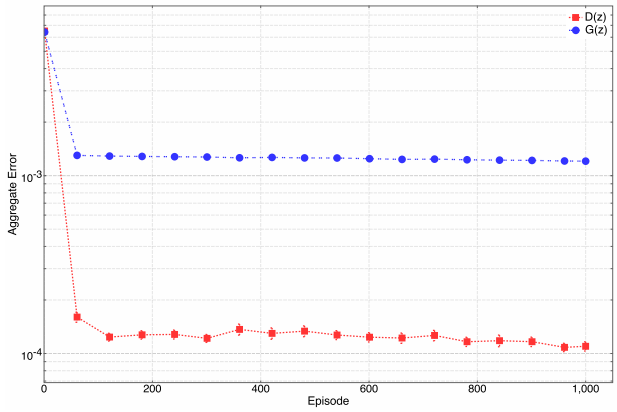


Figure 6: Heterogeneous sensor network performance with 100 agents. Use of the difference evaluation function results in 87% improvement in network accuracy.

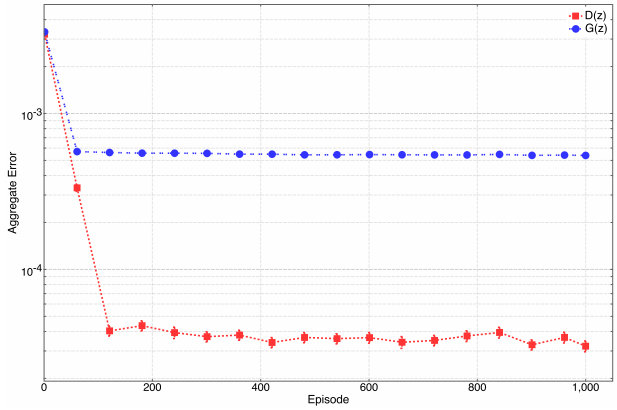


Figure 7: Heterogeneous sensor network performance with 500 agents. Use of the difference evaluation function results in 94% improvement in network accuracy.

As seen in Figures 6-8, use of difference rewards improve system performance by at least 87% over using the system reward to provide feedback. Measurement error is always below

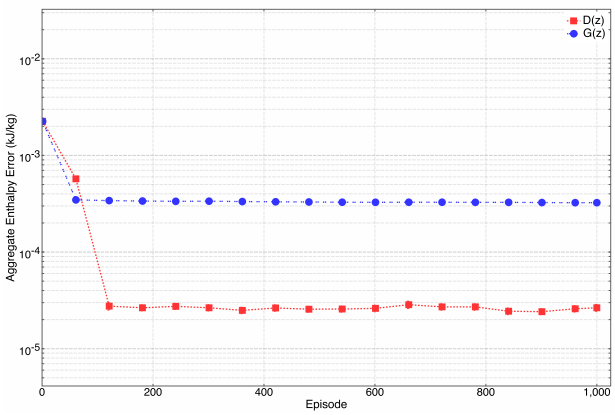


Figure 8: Heterogeneous sensor network performance with 1000 agents. Use of the difference evaluation function results in 92% improvement in network accuracy.

10^{-4} kJ/kg, regardless of sensor network size. This error is extremely small, and allows for plant controllers to have accurate state information in order to make informed control decisions.

5.3 Performance of Network in MTTEDCP

The first experiment in the MTTEDCP involves a sensor network of 500 devices is tested to monitor 50 sources over 25 timesteps, where each sensor must choose 5 timesteps to be active for. The sensors and sources are all assigned positions randomly in a 250 by 250 unit world, using a uniform random distribution. Each sensor has a sensing radius of 5-15 units (randomly assigned from uniform random distribution), and measurement error increases in magnitude by 0.1 for every distance unit away from a source a sensor is, with the sign (positive or negative) of the error being randomly assigned with equal probability. In this experiment, sensors have no baseline measurement error; all measurement error comes from proximity to the sources. Although unrealistic, these experiments give us an ideal performance baseline to test against for the other experiments.

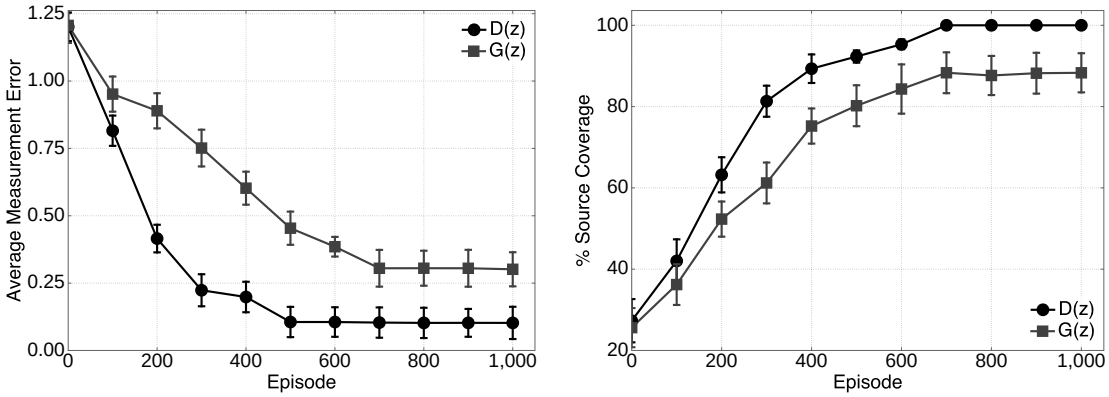


Figure 9: **Standard sensor network results. Sensor network average source error as a function of learning episode (left), and source coverage percentage as a function of learning episode (right). Difference rewards result in lower converged measurement error (**0.102 **vs.** 0.301**) and higher source coverage (100% vs. 88%) compared to global rewards.**

Figure 9 shows the results for this experiment. Difference rewards result in lower converged sensor error than global rewards (0.102 vs. 0.301). More significantly, difference rewards result in full source coverage over all time-steps, whereas global rewards only result in 88% coverage of sources. This means that if using global rewards to develop control policies, there exist sources which are not measured continuously, which is often unacceptable in a sensor network, particularly if source monitoring is a critical task. Difference rewards provide superior performance because they allow learning agents to determine their specific impact on the system, and adjust their actions accordingly to minimize measurement error and maximize source coverage.

6 Efficiency and Scalability of Sensor Networks (Milestone 3)

The third milestone of this work involves improving system efficiency and scalability for sensor networks ranging from 100 to 1000 devices. Scalability of the sensor network is critical for two key reasons. First, adding more sensors to the sensor network results in increased robustness, as the failure of a single sensor will have a smaller impact on the system. Second, larger sets of sensors create the possibility of increasing the accuracy of the sensor network, as a larger number of sensor subsets allows for the existence of more accurate subsets. However, increasing the size of the network also increases the coordination complexity; it is critical that algorithms used to develop sensor-specific control policies are robust to system scaling.

The following sections detail new objective functions which improve the scalability and efficiency of the distributed sensor network, as well as showing scalability results corresponding to milestone 3.

6.1 Expected and Average Difference Evaluation Functions

In order to improve scalability of the distributed sensor network, we introduce the expected difference evaluation and the average difference evaluation, which are modifications of the standard difference evaluation function. Recall that difference evaluations are defined as:

$$D_i(z) = G(z) - G(z_{-i} + c_i) \tag{21}$$

For the expected and average difference evaluations, the counterfactual term in the difference evaluation function is computed as an expected or average value for the agent (rather than replacing the agent with a single default agent). We define the expected difference evaluation as:

$$ED_i(z) = G(z) - \sum_{k \in K} P_k G(z_{-i} + c_k) \tag{22}$$

where:

- K is the set of all actions an agent may take
- P_k is the probability that the agent selects action k
- $G(z_{-i} + c_k)$ is the system reward if agent i took action k

The average difference evaluation function is equivalent to the expected difference evaluation function, assuming a uniform probability distribution for action selection. The average difference evaluation function is defined as:

$$AD_i(z) = G(z) - \frac{1}{K} \sum_{k \in K} G(z_{-i} + c_k) \tag{23}$$

As the number of agents in the system grows, the average and expected difference evaluations can help improve system performance, because they provide more specific feedback about an agent's entire policy, rather than a single action selection. This feedback is incorporated into the expectation of the agent across all actions, as this value gives the overall effectiveness of the agent's entire learned policy.

6.2 Results

We incorporate both the average and expected difference evaluation functions into the Rankine Cycle DCP, varying the size of the system from 50 to 1000 agents. These experiments demonstrate the effectiveness and scalability of the sensor network. Figures 10, 11, and 12 show the learned performance for 50, 500, and 1000 agent sensor networks, respectively.

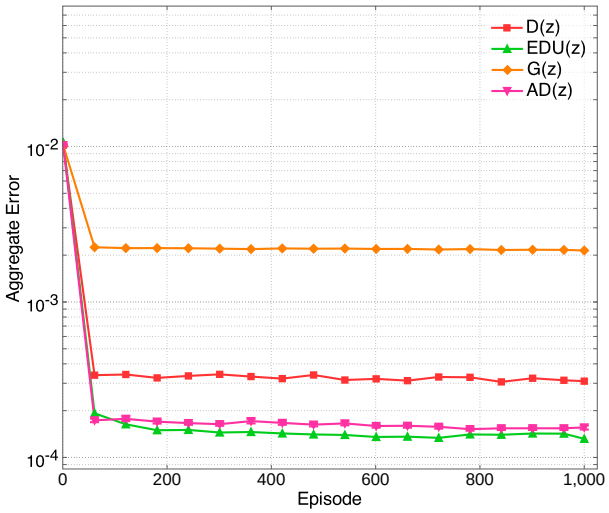


Figure 10: Sensor network performance with 50 agents. The expected and average difference evaluations provide an ever greater boost to system performance than the standard difference evaluation.

As seen in Figures 10-12, difference evaluations result in over an order of magnitude more accurate sensor measurements. Expected and average difference evaluations result in even more accurate measurements. Figure 13 shows the converged network measurement error as a function of the number of agents in the system. Regardless of network size, average and expected difference evaluations result in accuracy over an order of magnitude better than the global evaluation function.

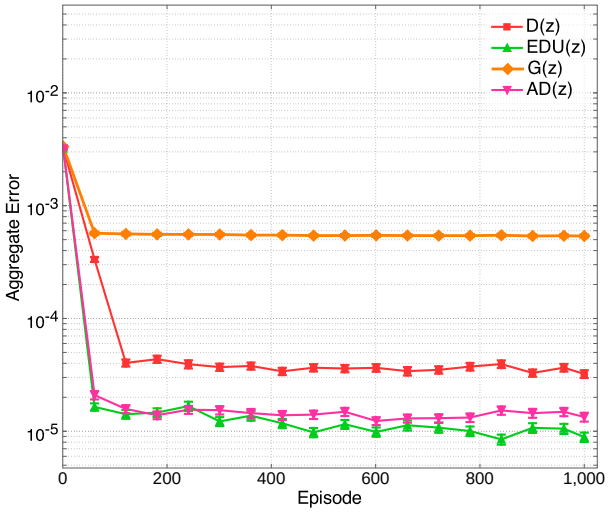


Figure 11: Sensor network performance with 500 agents. The expected and average difference evaluations provide an ever greater boost to system performance than the standard difference evaluation.

6.3 MTTEDCP Results

The next experiment involves a sensor network with noise and failures while varying the number of sensors from 100 to 10,000 in the MTTEDCP, in order to test the extreme scalability of the proposed algorithm. If there are N sensors, then there are $N/10$ sources and $N/20$ time-steps in each simulation. Each system size is tested for 500 statistical trials, and average data is reported in the results plots. The values reported are converged performance (after network experiences 10% failure and retrains) for each system size tested.

Figure 14 shows the results for this experiment. As the system size increases, both difference and global rewards result in lower average measurement error as well as improved coverage. This is because in the MTTEDCP, a larger system results in a greater number of sensor subsets which can be chosen from, allowing for superior solutions to be found. As the size of the system grows, difference rewards have increasingly better performance compared to global rewards; this is due to the increased coordination complexity in larger systems, which requires more advanced coordination techniques. When increasing system size from 500 to 10,000 agents, global rewards result in a 10.4% reduction in measurement error, while difference rewards result in a 55.8% reduction in measurement error. This is due to the fact that difference rewards are sensitive to the actions of an individual agent, and have a favorable signal to noise ratio.

Figures 10-14 demonstrate the effectiveness and scalability of the distributed sensor network when using expected and average difference evaluation functions, satisfying Milestone 3.

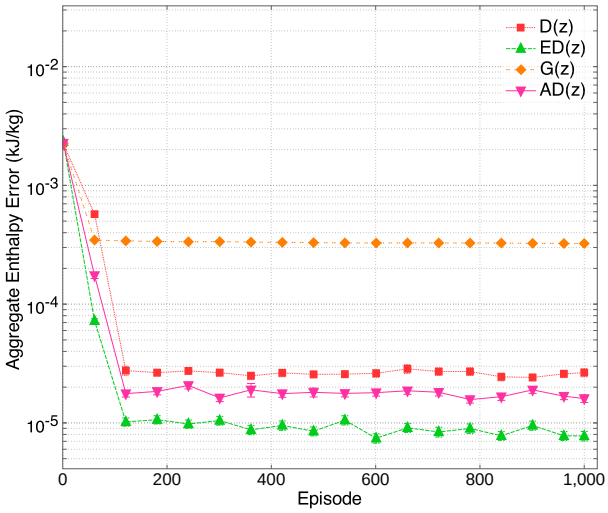


Figure 12: Sensor network performance with 1000 agents. The expected and average difference evaluations provide an ever greater boost to system performance than the standard difference evaluation.

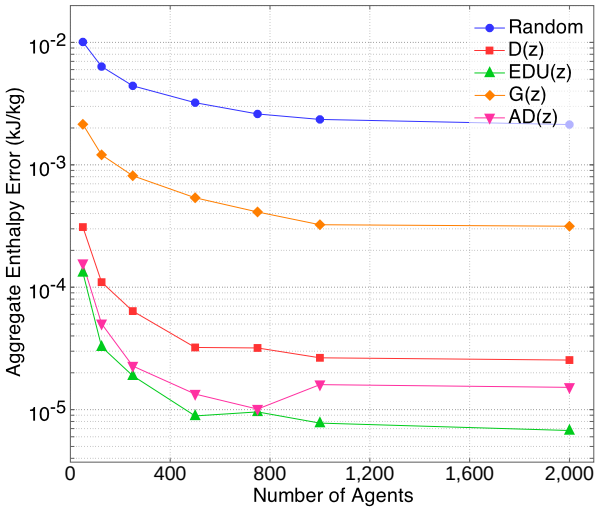


Figure 13: Sensor network performance: scaling between 50 and 2000 agents. Regardless of sensor network size, the expected and average difference evaluations result in the highest network performance.

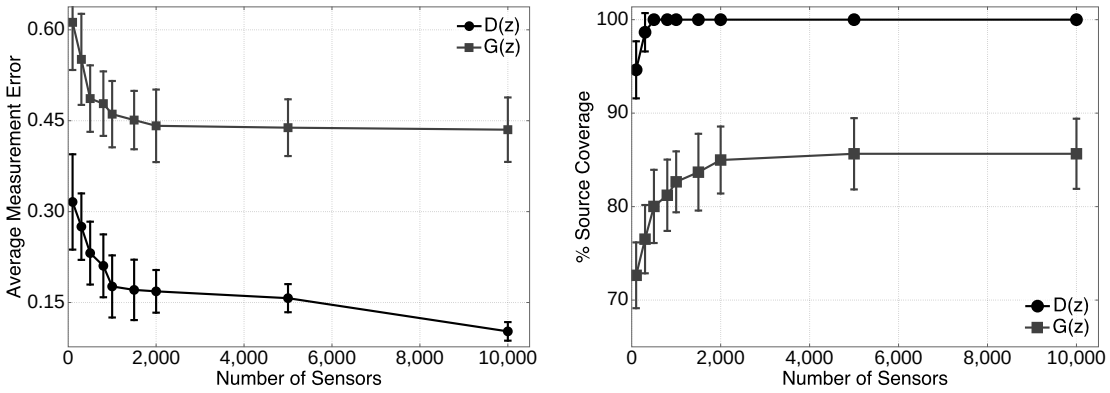


Figure 14: **Sensor network with noise and failure results: scaling. As the system size increases, both global and difference rewards result in improved measurement accuracy and source coverage. As the size of the system grows and coordination complexity increases, difference rewards perform increasingly better compared to global rewards.**

7 Sensor Network Reconfigurability (Milestone 4)

The fourth and final milestone of this work involves improving system reconfigurability and scalability in advanced power systems for networks with up to 10000 devices. In a sensor network with many (over 1000) devices operating in a harsh environment, device failure is inevitable. Thus, it is critical that sensor network control policies are develop which can quickly reconfigure to changing system conditions, to ensure that changes in system conditions do not lead to catastrophic failures.

7.1 Rankine Cycle DCP Results

For the sensor failure experiments, agents are trained as described in Section 2. After 1000 episodes, a portion of the sensors in the system fail (they are deactivated), and the remaining sensors must alter their policies to account for the changing system conditions. Figure 15 shows results for the case with 500 agents and 10% device failure.

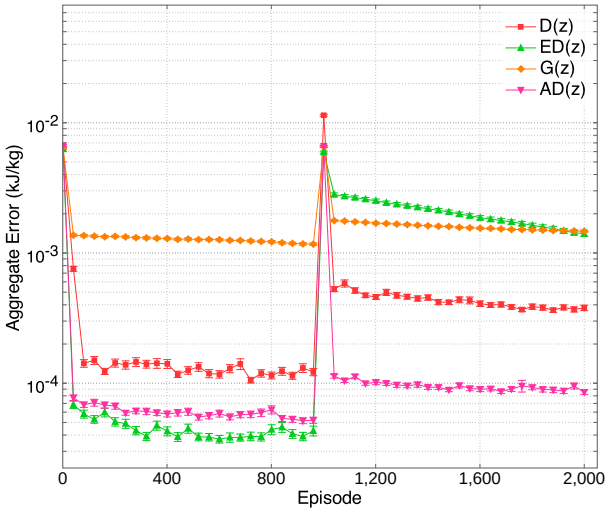


Figure 15: Sensor network performance: 500 agents, with 10% of the agents failing after 1000 episodes. Agents using average difference evaluations quickly recover from the failure, and maintain low error levels.

As seen in Figure 15, after 10% of the devices fail, agents using average difference evaluations quickly recover and learn a policy which still provides extremely low error levels, on the order of $10^{-4}kJ/kg$. Agents using expected difference evaluations recover slowly. This is because the learned probability distributions used for calculating expected difference evaluations are inaccurate after the failures in the system, and incorrectly bias agent rewards. Thus, although we concluded that average and expected difference evaluations provide similar converged performance, we find that average difference evaluations are more robust to device failure. Figure 16 shows results for the case with 2000 agents, 25% sensor noise, and 25% device failure.

As seen in Figure 16, the algorithm scales well even to 2000 agents. In contrast to the results in Figure 15, agents using expected difference evaluations do not see a large drop in performance compared to average difference evaluations after device failure. This is because the addition of sensor noise results in higher variance in the probability distribution used to compute the expected difference evaluation, improving performance after device failure. After 25% of the devices in the

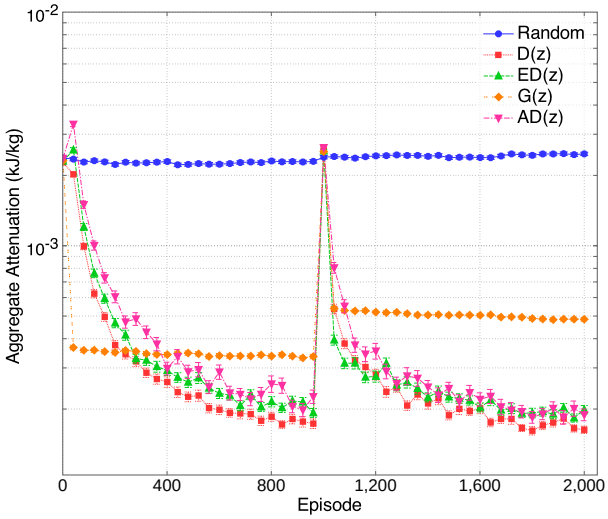


Figure 16: Sensor network performance: 2000 agents, with 25% sensor noise and 25% of the agents failing after 1000 episodes. Agents using average difference evaluations quickly recover from the failure, and maintain low error levels.

system fail and the system reconfigures, agents using any variant of difference evaluations regain almost all lost performance. However, agents using global evaluation functions have a permanent loss in performance, that is not regained after retraining. This experiment demonstrates that (i) difference evaluations can scale to extremely large network sizes, and (ii) difference evaluations result in highly reconfigurable networks in the fact of changing system conditions.

7.2 MTTEDCP Results

The next experiment involves developing control policies for the MTTEDCP which are robust to sensor failures. In this case, sensors have a constant measurement error, drawn from a normal distribution with zero mean and a standard deviation of 0.01 to 0.1 (randomly drawn from uniform random distribution, to allow for different sensor capabilities/accuracy). Further, every time a sensor takes a measurement, noise drawn from a normal distribution with zero mean and a standard deviation of 0.025 is added to the constant measurement error. The distance-based error from the first experiment (measurement error magnitude increases by 0.1 for every distance unit away from the source the sensor is) is retained in this experiment.

After 1000 learning episodes, 10% of the sensors in the network are chosen at random and deactivated in order to simulate device failure. The remaining sensors are allowed to learn new policies for 1000 more learning episodes, in order to determine how well the agents can learn to recover from device failure. This experiment aims to determine how robust the multiagent system is to device failure, as well as determining how reconfigurable the system is under rapidly changing system conditions.

Figure 17 shows results for this experiment. As expected, difference rewards provide superior converged system performance compared to global rewards. After 10% network failure, difference rewards recover 94% of lost performance, while global rewards recover only 78% of lost performance. Further, after network reconfiguration, difference rewards still provide 100% source coverage, while global rewards source coverage drops from 83% to 80%. This experiment demonstrates that agents trained with difference rewards are more robust to agent failure, and are more

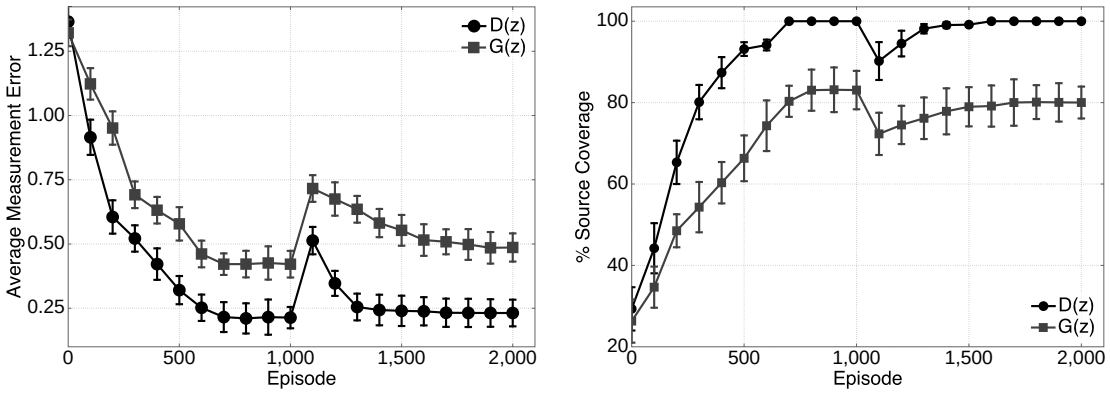


Figure 17: **Sensor network with noise and failures results. Sensor network average source error as a function of learning episode (left), and source coverage percentage as a function of learning episode (right). Difference rewards result in lower converged measurement error (**0.232 **vs.** 0.487**) and higher source coverage (100% vs. 80%) compared to global rewards. Difference rewards also recover more lost performance after failures than global rewards (94% vs 78%).**

reconfigurable in light of changing system conditions.

8 Conclusion

Reliably and accurately collecting system level information is critical for safe and reliable operation of advanced power systems. Recent technological advances allow for some level of decision making directly at the sensor levels. However, individual sensor control policies must allow coordinator behavior at the sensor level, so that the actions of all sensors lead to desirable system performance.

In this project, we ensure coordination of sensors in a large heterogeneous sensor network by addressing our two key objectives. First, we derive sensor performance metrics for heterogeneous sensor networks, such that individual sensors optimizing their specific feedback signals act to improve overall system performance. Second, using these performance metrics, we train heterogeneous sensor networks to be highly accurate, scalable, robust to sensor noise and failures, and highly reconfigurable when system conditions change dramatically.

Our results demonstrate that by properly shaping agent objective functions, we can develop large (up to 10,000 devices) heterogeneous sensor networks with key desirable properties. The first milestone shows that properly choosing agent-specific objective functions increases system performance by up to 99.9% compared to global evaluations. The second milestone shows evolutionary algorithms learn excellent sensor network coordination policies prior to network deployment, and these policies can be refined online once the network is deployed. The third milestone shows the resulting sensor networks networks are extremely robust to sensor noise, where networks with up to 25% sensor noise are capable of providing measurements with errors on the order of 10^{-3}. The fourth milestone shows the resulting sensor networks are extremely robust to sensor failure, with 25% of the sensors in the system failing resulting in no significant performance losses after system reconfiguration. Because of the massive scale of the coordination it allows, the impact of this research goes beyond sensing in a power plants. Indeed it is also applicable to a host of energy problems from power distribution, to managing smart grids, to designing micro-grids.

References

[1] A. Agogino and K. Tumer. Analyzing and visualizing multi-agent rewards in dynamic and stochastic domains. In *Journal of Autonomous Agents and Multi-Agent Systems*, pages 320–338, 2008.

[2] D. Challet and N. Johnson. Optimal combination of imperfect objects. *Physics Review Letters 89*, 2002.

[3] M. Colby. *Theoretical and Implementation Improvements for Difference Evaluation Functions*. PhD thesis, Oregon State University, Corvallis, OR, 2014.

[4] M. Colby, W. Curran, C. Rebhuhn, and K. Tumer. Approximating difference evaluations with local knowledge (extended abstract). In *In Proceedings of the Twelfth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 14)*. Paris, France, 2014.

[5] M. Colby, S. Kharaghani, C. HolmesParker, and K. Tumer. Counterfactual exploration for improving multiagent learning. In *In Proceedings of the Thirteenth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 15)*. Istanbul, Turkey, 2015.

[6] M. Colby and K. Tumer. Multiagent reinforcement learning in a distributed sensor network with indirect feedback. In *In Proceedings of the Twelfth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 13)*. Saint Paul, Minnesota, 2013.

[7] M. Colby and K. Tumer. Distributed sensor network control with evolutionary algorithms. In *In Proceedings of the 2014 International Society of Automation Power Industry Division Symposium*. Scottsdale, Arizona, 2014.

[8] M. Colby and K. Tumer. An evolutionary game theoretic analysis of difference evaluation functions. *In Proceedings of the Genetic and Evolutionary Computation Conference*, 2015.

[9] M. Colby and K. Tumer. Learning-based coordination of large heterogeneous distributed sensor networks. In *In Proceedings of the 2015 International Society of Automation Power Industry Division Symposium*. Kansas City, Missouri, 2015.

[10] A. Farinelli, A. Rogers, and N.R. Jennings. Maximising sensor network efficiency through agent-based coordination of sense/sleep schedules. *In Proceedings of the International Workshop on Energy in Wireless Sensor Networks*, 2008.

[11] D. Fogel. An introduction to simulated evolutionary optimization. In *IEEE Transactions on Neural Networks*. 1994.

[12] M. J. Moran and H. N. Shapiro. *Fundamentals of Engineering Thermodynamics*. John Wiley and Sons, Inc., fifth edition, 2004.

[13] A. Rahmattalabi, M. Colby, and K. Tumer. Evolving control policies for distributed sensor network coordination. In *In Proceedings of the 2015 International Society of Automation Power Industry Division Symposium*. Kansas City, Missouri, 2015.

[14] A. Rogers, A. Farinelli, and N. Jennings. Self-organising sensors for wide area surveillance using the max-sum algorithm. *Lecture Notes in Computer Science. Self-Organizing Architectures*, 2010.

[15] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Massachusetts, 1998.

[16] K. Tumer. Designing agent utilities for coordinated, scalable, and robust multiagent systems. In P. Scerri, R. Mailler, and R. Vincent, editors, *Challenges in the Coordination of Large Scale Multiagent Systems*. Springer, 2005.

[17] S. Williamson, E. Gerding, and N. Jennings. Reward shaping for valuing communications during multi-agent coordination. *In Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems*, 2009.