Idea: A System for Efficient Failure Management in Smart IoT Environments*

Palanivel Kodeswaran
IBM Research
Bangalore, KA, India
Email: palankod@in.ibm.com

Ravi Kokku
IBM Research
Yorktown Heights, NY, USA
Email: rkokku@us.ibm.com

Sayandeep Sen
IBM Research
Bangalore, KA, India
Email: sayandes@in.ibm.com

Mudhakar Srivatsa
IBM Research
Yorktown Heights, NY, USA
Email: msrivats@us.ibm.com

ABSTRACT

IoT enabled smart environments are expected to proliferate significantly in the near future, particularly in the context of monitoring services for wellness living, patient healthcare and elderly care. Timely maintenance of failed sensors is of critical importance in such deployments to ensure minimal disruption to monitoring services. However, maintenance of large and geographically spread deployments can be a significant challenge. We present Idea that significantly increases the *time-before-repair* for a smart home deployment, thereby reducing the maintenance overhead. Specifically, our approach leverages the facts that (a) there is inherent sensor redundancy when combinations of sensors monitor *activities of daily living* (ADLs) in smart environments, and (b) the *impact* of each sensor failure depends on the activities being monitored and the functional redundancy afforded by rest of the heterogeneous sensors available for detecting the activities. Consequently, Idea identifies homes that need to be fixed based on expected degradation in ADL detection performance, and optimizes maintenance scheduling accordingly. We demonstrate that our approach leads to 3–40 times fewer maintenance personnel than a scheme in which failed sensors are fixed without considering their impact.

1. INTRODUCTION

There is a significant proliferation of *IoT* enabled smart home products ranging from home security, environmental controls such as temperature, light, humidity etc, to voice activated device controls. Of particular interest, are applications in the health care domain that enable remote monitoring of patients, particularly the elderly. Industry projections [1] show that elderly-care products are expected to grow to $10.3 billion in 2020 globally, with $2 billion for elderly safety monitoring in the US and European markets. A number of medical studies [2–6] show that continuously monitoring *Activities of Daily Living* or ADLs such as `Cooking`, `Eating`, `Showering`, `Taking Pills` etc. is of prime importance in geriatric care as they allow for the early detection of the onset of diseases such as Alzheimer's, Dementia, Mild Cognitive Impairment, etc. For instance, caregivers of patients with Alzheimer's disease rank tracking and identifying activities of daily living of the patient at the top of their list of needs for health assistance [2].

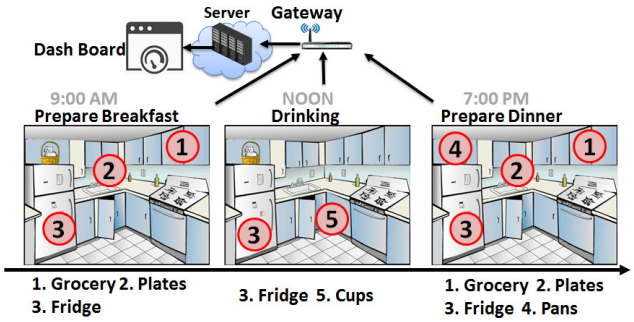


Figure 1: Sensors triggered for various activities.

Automated detection of an ADL is achieved by *learning patterns of sensors* that are *triggered in the smart home* when specific ADLs are performed. For example, in Figure 1, we depict the kitchen of a smart home that is instrumented with sensors to monitor the opening and closing of appliances such as microwave, fridge and multiple cabinets, etc. In the above home a `Drinking` ADL will be detected by monitoring the {`Cups_Cupboard`, `Fridge`} sensor events. For such residents, it is envisioned that a service provider will instrument a smart home and assist care givers by offering a service to remotely monitor the health of its residents by accurately detecting the ADLs being undertaken in the home.

Given the importance of health monitoring applications, it is critical to *continuously monitor and maintain functionality* of the IoT deployment [1]. Maintaining sensors by elderly residents is infeasible due to the miniature sizes as well as placement of these sen-

*Authors are listed in alphabetical order.

MobiSys'16, June 25-30, 2016, Singapore, Singapore

DOI: http://dx.doi.org/10.1145/2906388.2906406

[1]In this paper we consider activity detection for remote health/wellness monitoring as the motivating use case. However, the designed system can be used for other remote monitoring applications such as home security, home activity monitoring with baby-sitters, etc.

Label	Name	# User	Dur. (days)	# ADL (# Activity)	# Sensor (# Event)
kA	KasterenA [16]	1	25	16 (283)	14 (2006)
kB	KasterenB [16]	1	15	25 (172)	27 (22595)
kC	KasterenC [16]	1	19	27 (254)	23 (39861)
Ar	Aruba [17]	1	219	11 (6477)	39 (805268)
T1	Twor9-10 [17]	2	249	25 (3745)	100 (711421)
T3	Twor2009 [17]	2	57	14 (499)	100 (136504)
T2	TworSmr [17]	2	63	8 (1016)	100 (366075)
Mul	AdlNorm [17]	24	84	5 (120)	39 (6425)

Table 1: Datasets from IoT enabled Homes

sors [7]. Hence, we expect that the responsibility of maintaining the sensors, i.e. detecting and repairing failed sensors at the earliest to ensure that the monitoring applications continue to be effective will rest upon the service provider.

On the other hand, from a service provider's perspective, the maintenance of large scale sensor deployments becomes a significant challenge due to the high labor costs involved in monitoring and repairing sensors that have a non-zero probability of failing across reasonably large number of geographically spread homes. Thus there is an inherent tradeoff between maintaining acceptable application QoS and minimizing operational costs for the service provider.

We note that the standard approach of deploying redundant sensors to minimize maintenance load explored in the context of homogeneous sensors [8–15] is not directly applicable in the setting described above. This is because smart home deployments combine information from multiple heterogeneous sensors such as contact, motion and temperature sensors to detect ADLs. Similarly, as explained in Section 2, deploying multiple redundant sensors is not sufficient as it only improves failure time logarithmically and faces operational challenges in smart home environments.

Our approach for mitigating the maintenance load leverages the observation that *a number of sensors are triggered when an ADL is performed by a given user, implying some sensors could be functionally redundant for the purpose of detecting the activity from sensor firings*. To elucidate, in Figure 1, and ADL detector will detect `Prepare_Dinner` ADL when at 7PM, events of {`Fridge`, `Freezer`, `Grocery`, `Plates`, `Pans`} sensors are observed.

Note that even if the `Grocery_Cupboard` sensor fails to trigger, the triggering of other sensors can still detect the occurrence of `Prepare_Dinner` with reasonable accuracy. Thus, one of the key aspects of this work is to quantify the functional redundancies present among heterogeneous sensors for the task of ADL detection.

To this end, we build a system for **Integrated ADL detection**, **estimation of functional redundancy** and **alerting for maintenance visit** (Idea). The Idea system tolerates a number of sensor failures by leveraging the functional redundancy among heterogeneous sensors, and minimizes the degradation in ADL detection performance in the presence of smart home sensor failures. The desired consequence of our design is the significant reduction in maintenance effort as no maintenance visits are required as long as the ADL detection accuracy is maintained above a certain (user specified) threshold.

Contributions

Idea builds on prior work on accurate ADL detection [3–6, 18–26] by explicitly quantifying the functional redundancies available among sensors in smart homes to enable robust detection in the presence of sensor failures. It also advances the nascent area of IoT deployment maintenance [27, 28] by presenting the first joint technique for achieving systematic quantification of functional redundancy among heterogeneous sensors and assessment of impact of sensor failure on ADL detection performance.

In summary, we make the following contributions.

- We present Idea– the first integrated ADL detector that extracts functional redundancy among sensors to provide robust ADL detection in the presence of sensor failures.
- We propose a novel method based on *rarity estimation* to leverage the extracted functional redundancies to identify sensor failures (Section 3.4).
- We propose a method that leverages Idea's rule based design to estimate the impact of sensor failures on ADL detection (Section 3.3).
- We design and implement a cloud based maintenance scheduling technique that significantly reduces (3-40 times) maintenance cost by increasing the time before repair for homes with failures.
- We present a thorough evaluation with publicly available datasets and realistic sensor failure models, and show that Idea's ADL detection method is more robust to failures than other approaches to ADL detection, in turn, leading to significant increase in the time to send repair mechanics to individual homes. Specifically, based on a representative home distribution in and around Manhattan, New York region, we show that Idea reduces the maintenance personnel overhead by a factor of 3-40 times, and increases average per-home intervisit times from 18 to 211 days, minimizing user annoyance (Section 4).

Road map

The rest of the paper is organized as follows. In Section 2 we benchmark the magnitude of sensor failures for large sensor deployments, thus motivating the need for Idea. We also describe the intuition leveraged by Idea to reduce maintenance overhead. In Section 3, we describe the various components of Idea. We evaluate Idea's performance in Section 4, describe related work in Section 5 and finally conclude in Section 6.

2. MOTIVATION

Our goal in this work is to design a solution for the efficient maintenance of IoT sensors in smart homes to ensure disruption free continuous remote monitoring of ADLs. We begin with a brief description of the eight ADL datasets analyzed as part of our work. The datasets listed in Table 1 consist of time stamped sensor events that are generated when the variety of contact, temperature and motion sensors are triggered when the user undertakes various ADLs such as sleeping, eating, preparing meals, taking medicines, personal hygiene etc. [2]. The eight datasets together capture the heterogeneity typically seen across different smart home deployments in terms of the type of sensors deployed as well as the number of sensors, ranging between 14 and 99.

To motivate the need for designing efficient solutions for the *IoT sensor maintenance problem*, we carry out a modeling study for a representative deployment of 1000 homes, with topology shown in Figure 2 with a sensor configuration similar to the smart home from *KasterenA* dataset mentioned in Table 1 over a period of two years. We model the failure of each sensor using a Weibull distribution (a

[2]Refer papers cited in Table 1 for more detailed description of ADLs monitored in each home.

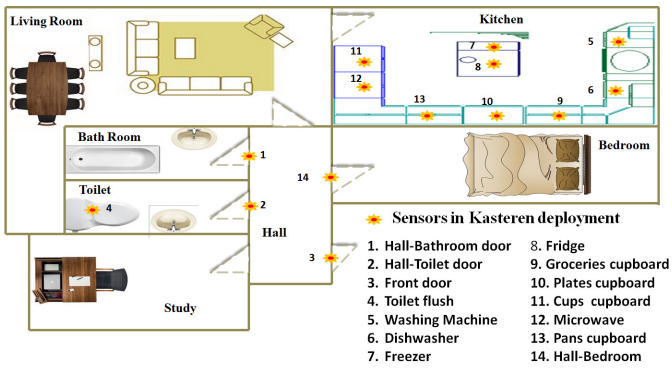


Figure 2: Sensor Layout in *KasterenA* dataset

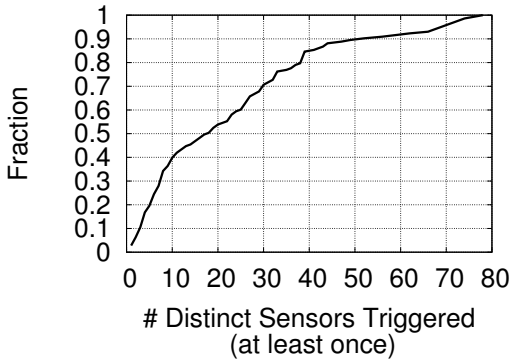


Figure 3: CDF of count of distinct sensors which triggered at least once for an ADL across all datasets in Table 1.

common choice of modeling failure of sensors) [29–31] with mean time to failure (MTTF) set to 1.0 $years$. Analytical calculations show that increasing the number of homes and sensors leads to an exponential increase in the number of sensor failures (results omitted for the sake of brevity). This exponential increase results in significantly high maintenance costs (operational expenditure) for the provider as they now need to devote a significant amount of manpower simply to monitor and service the deployment.

As mentioned in Section 1, our approach for mitigating the maintenance load leverages the key observation that a number of sensors are triggered when an ADL is performed by a given user, which would make a few sensors redundant for detecting the activity. To validate the above observation, in Figure 3 we plot the CDF of the number of sensors that are triggered by each ADL across all datasets in Table 1. From the figure, we see that fewer than 5% of ADLs trigger only one sensor; while for 50% of the ADLs, up to ten sensors are triggered per ADL. Intuitively, we can exploit the above *functional redundancy* amongst heterogeneous sensors, to delay the time to repair sensor failures in a home. As mentioned in Section 1, the task of ADL detection involves learning the sensor (and their temporal) activation patterns corresponding to the ADLs and then *detecting* the underlying activity based on the set of sensor events seen. However, note that ADL detection is not straightforward. For instance, a similar set of sensors are triggered by different ADLs as seen in the case of `Prepare_Breakfast` and `Prepare_Dinner` ADLs. Further, humans exhibit significant variability while performing the same ADL across different days, in terms of the set of sensors used as well as their sequence. Additionally, based on the user's behavioral patterns, the sensors co-relate differently for different ADLs, thereby implying that the set of redundant sensors must be learnt separately on a per ADL basis.

Hence, in this work we design a system called Idea, which (a) learns and leverages the functional redundancies among sensors leading to robust ADL detection in the presence of sensor failures (b) identifies when a sensor has failed, and (c) determines the impact of a failed sensor on the detection performance of all ADLs. Using these steps, Idea can significantly delay the Time-before-repair for a home resulting in significant savings in deployment maintenance costs.

Note that the alternate approach to reducing maintenance costs by deploying duplicate sensors beforehand as a way to achieve redundancy will not work due to the following reasons. First, adding 'n' sensors will improve redundancy by only log(n). We validated this observation using a tool available at [32]. Second, based on observations made by authors in [7], operational problems in smart homes such as limited number of power sockets and aesthetics also restrict the number of sensors that can be deployed in any home.

3. Idea SYSTEM DESIGN

In this section, we describe the design of Idea. At a high level, the functionality of Idea is divided into four key components (Figure 4): First, Idea includes an ADL Signature Generation phase to bootstrap the system with sensor events that are annotated with ADL labels, such that all possible ways of performing ADLs in the given home are captured at a central *IoT* gateway. The annotations can be done either generically for common human activities or on an application-domain basis by subject matter experts from the IoT service enterprise, or by competent residents of smart homes. Second, Idea assesses the impact of different sensor failures on the accuracy of detecting ADLs needed by the activity monitoring applications that run in the home. Third, as the system continues to run, Idea performs continuous ADL detection, and sensor failure detection, and raises an alert to the cloud when a sensor fails. Finally, on receiving sensor failure alerts, and determining the impact of the corresponding failure on ADL detection performance in the home, the maintenance scheduler on the cloud schedules maintenance visits for homes where critical sensors have failed[3].

3.1 ADL Signature Generation

The goal of the ADL signature generation component is to learn the sensory-temporal signatures of the various ADLs from annotated sensor events collected during the training phase. During the training phase, whenever the user performs an ADL, the generated set of sensor events are collected and annotated in the form of time-stamped event streams at the local gateway.

Sensory Signature Extraction

The key intuition behind sensory signature extraction is to identify frequently occurring subsets of sensor events across activi-

[3]For simplicity of description of Idea, we assume that the behavior of the resident exhibited in the learning phase does not change after learning. Since users' activity patterns change in reality, we note that the system can locally re-learn the ADLs when the system detects that significant behavioral change might have happened. We do not discuss this extension in the paper.

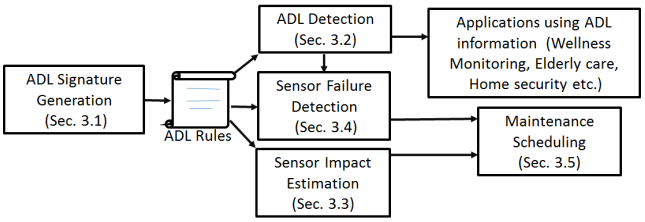


Figure 4: Solution Architecture of Idea

ties that are indicative of an underlying ADL. For example, in the home shown in Figure 2, when an elderly resident uses the toilet, the `Toilet Door`, `Bathroom Door` and `Flush` sensors are mostly triggered. Due to the inherent variability exhibited by humans regarding the set of sensors used and their corresponding sequences while performing ADLs, we focus on extracting frequent sensor *subsets* rather than sensor *sequences*. We employ a Frequent itemset mining (FIM) approach [33] for identifying frequently occurring subsets of sensors that are indicative of an ADL. The FIM algorithm takes as input a set of activities and the corresponding set of sensors that were triggered during the activity, and outputs frequently occurring subsets of sensors in the form of rules for each ADL. For example, we obtain the following rule for the `Prepare_Breakfast` ADL

`(R1)Fridge,Groceries_Cbrd,Plates_Cbrd` $\Rightarrow$
`Prepare_Breakfast<support:0.05,confidence:0.7>`

The rule attributes of R1 provide various measures of the frequency and predictive power of the rule. In particular, the support of a sensor subset X, $Supp(X)$ is defined as the proportion of activities in the training dataset that contain the sensor subset X. The predictive power of a rule is reflected in its *confidence* value, which for a given association rule of the form $A \Rightarrow O$, is defined as $Conf(A \Rightarrow O) = Supp(A \cup O)/Supp(A)$.

Note that FIM generates, not one, but a *set* of rules for a given ADL based on the number of sensors that are typically triggered by the ADL. For example, the following rules are obtained for *Prepare_Breakfast* ADL in the KasterenA home as shown in Figure 2,

`(R2) Groceries_Cbrd, Plates_Cbrd` $\Rightarrow$
`Prepare_Breakfast`
`(R3) Fridge, Groceries_Cbrd` $\Rightarrow$ `Prepare_Breakfast`
`(R4) Groceries_Cbrd` $\Rightarrow$ `Prepare_Breakfast`

In general, the number of rules generated for an ADL is a function of the redundancy exhibited by the triggered sensors. For example, in the *KasterenA* dataset, there is a single rule for the `leave_house` ADL which triggers a single sensor while the `Prepare_Dinner` ADL which triggers six sensors includes eighteen rules. By generating multiple association rules for each ADL, Idea is able to identify (heterogeneous) sensor subsets that are strongly co-related with each other on a per ADL basis, thus providing the provide functional redundancy for the task of detecting the ADL. For example, the sensor pair `[Freezer, Plates_Cbrd]` is strongly co-related with each other when occurring as part of `Prepare_Dinner` ADL, where as the sensor pair shows lower co-relation while occurring as part of `Prepare_Breakfast` ADL. This implies that the `Freezer` and `Plates_Cbrd` sensor pair are redundant with respect to `Prepare_Dinner` and a single sensor may be sufficient for ADL detection even if the other sensor fails. The above property holds for all rules for a given ADL that have the same support, but the set of sensors in one rule is a proper subset of the other. Thus Idea is able to systematically discover redundancies among sensors in ADL detection, using multiple association rules which in turn helps in estimating the importance of a sensor and in detecting sensor failures as described later.

Temporal Signature Extraction

We observe in the datasets that several ADLs (especially the related ones) trigger the same set of sensors and differ only in their temporal features such as time(hour) of day of occurrence and duration. For example, `Prepare_Breakfast` and `Prepare_Dinner` use a similar set of sensors as shown in Figure 2 while differing only in their time of day occurrence, peaking at 11 am and 8 pm respectively. Similarly, both `Brushing` and `Sleeping` use the Toilet door at roughly the same time of day around 10 PM, but significantly differ in the activity durations ranging from one minute for `Brushing` to about six hours for `Sleeping`. This indicates that temporal features such as time-of-day of occurrence as well as activity duration play a significant role in disambiguating ADLs when the sensory signatures alone are not discriminative enough, either due to sensor overlap among ADLs or due to the failure of sensors. Consequently, for each rule, Idea identifies the set of activities in the training set that match the rule, and determine the time of day of occurrence as well as duration of the activity. Based on the collected samples across days, Idea then estimates the *most probable* time of day of occurrence as well as duration based on the modal value and standard deviations of the samples.

At the end of this step, Idea generates for each ADL a set of sensory-temporal signatures of the form $< S_i, S_j ... S_n, Mean\_Time\_of\_Day, Mean\_Duration \Rightarrow ADL >$.

3.2 ADL Detection

Once the ADL signatures are generated, Idea processes the sensor streams collected at the IoT gateway at run-time to detect ADLs based on matching the incoming sensor stream against the sensory temporal signatures learnt in the previous step. We assume that the incoming sensor event stream is cut into activities using an event segmentation model [22, 34, 35] such that each activity, T, potentially corresponds to an underlying ADL undertaken by a single user. The detection process consists of several scorer modules, each extracting specific features from the activity T and scoring the various ADLs that the activity could potentially map to. For each ADL_k, we denote the set of rules that are used to detect the ADL_k as R_k . Idea computes a closeness of match for the incoming activity with each rule in the rule base to obtain the following set of scores.

Sensor Scoring: We define that an activity T matches a rule if the set of sensors in the rule is a subset of the sensors in the activity. Let $R(T)$ denote the set of rules matching activity T. For each $ADL_k \in R_k(T)$, Idea computes the sensory score of T as the sum of the confidences of the matching rules multiplied by the prior probability of occurrence of the ADL_k that is estimated from the training dataset. Note that the above formulation handles the case where the same sensor set is used in multiple ADLs. In these cases, the confidence would be higher for the more frequently occurring ADL such as `Drinking` compared to the less frequent one such as `Prepare_Breakfast`, and consequently, the sensory score for the more frequent ADL would be higher compared to the less frequent ADL.

Temporal Scoring: Idea next computes the temporal scores by matching the activity T against the temporal features viz. time of day and duration of the rule set. We say that an activity temporally matches a temporal feature of a rule if it lies within a threshold k number of standard deviations from the modal value of the rule's feature. By experimenting with multiple datasets, we determine that $k = 1.5$ yields the best accuracy. For each rule R_i, we extract

temporal features and denote the modal hour of day as $Tod(R_i)$ and average duration of the ADL as $Dur(R_i)$. For example, in KasterenA dataset, the modal ToD and duration for `Brushing` ADL are 10 PM and one minute respectively. For a given input activity T, we compute the hour of day (ToD), by extracting the hour from the timestamp of the first event in T, and duration as the time elapsed between the first and last sensor events. After identifying the set of temporally matching rules, Idea computes the time of day score as the sum of the confidences of the matching rules. Similarly, the duration score is computed as the sum of the confidences of the rules matching the duration.

Idea computes the composite scores for each ADL_k as a weighted sum of the individual scores. Empirically, we observe that temporal features possess a stronger discriminating power in detecting ADLs compared to sensor usage, and consequently we set higher weights for the temporal scores compared to the sensor score weight (ratio of 1000:1). Finally, Idea assigns the ADL label corresponding to the highest composite score to the input activity.

3.3 Impact Estimation

The goal of impact estimation is to identify critical sensors for each ADL. We define a sensor as critical for a given ADL if the sensor is necessary to maintain the detection accuracy above a configured threshold. When a critical sensor for an ADL fails, there are two possibilities that arise for the corresponding input activities: (1) no ADL is detected, and (2) the ADL is misclassified as another ADL. The goal of Idea is to minimize both possibilities.

The impact estimation component takes as input the annotated event stream collected during the training phase along with the sensory temporal signatures learnt in Section 3.1. From the annotated data, Idea estimates the functional redundancy, for ADL detection, among sensor patterns that are triggered by the ADL as well as their corresponding proportions. For example, in the KasterenA dataset, the following subset of sensor patterns are triggered by the `Prepare_Breakfast` ADL in the corresponding proportions
$\{Fridge, Groceries\_Cbrd, Plates\_Cbrd\} = 0.05$,
$\{Groceries\_Cbrd, Plates\_Cbrd\} = 0.05$,
$\{Fridge, Plates\_Cbrd\} = 0.06$.

For a given [sensor, ADL] pair, Idea estimates the impact of sensor failure as follows. In the first step, Idea determines if the sensor is applicable for the ADL by checking if any of the sensory signatures for the ADL include the sensor. If no such signature exists, the impact is determined to be zero, as the sensor is not used for recognition of the given ADL; Idea also classifies the sensor as non-critical. In the above example, a `Toilet Flush` sensor is non-critical for `Prepare_Breakfast` as it is not part of any sensory signature for `Prepare_Breakfast`. In the second step, Idea determines if there are other sensors that can be used to detect the ADL. This is achieved by scanning the rule base to check if there are any sensory signatures for the ADL that do no include the given sensor. If no such signature exists, the ADL can no longer be detected in the absence of the sensor, and Idea classifies the sensor as critical for the ADL. For example, in KasterenA, only the `Front Door` that is used for detecting the `Leave_House` ADL, and in its absence, no `Leave_House` instance can be detected.

In the third step, Idea determines if the remaining set of sensors alone can detect the ADL and estimates the resulting detection accuracy. In this case, Idea estimates the *detection accuracy* with and without the sensor and returns the change in the estimated detection accuracy as the impact of losing the sensor on the ADL.

Estimating Detection Accuracy: Given an ADL, Idea estimates its detection accuracy by choosing each distinct sensor pattern of the ADL and estimating its accuracy individually as follows. For a given sensor pattern and ADL, Idea determines if the rules matching the sensor pattern can accurately detect the ADL. If the sensor pattern matches only the single given ADL, then Idea assumes all these instances can be correctly detected. On the other hand, if the sensor pattern matches multiple ADLs, Idea determines if any of the other matching ADLs overlaps with the given ADL in the temporal domain. We define two ADLs to overlap in the temporal domain if any of their rules have temporal features (time of day or duration) that are within a threshold number of standard deviations of each other. In the case of overlapping ADLs, Idea chooses to be conservative, in estimation of functional redundancy and assumes all these instances of the ADL will be misclassified, and returns a detection accuracy of zero. This conservative design enables Idea to proactively maintain failed sensors, and prevent the application QoS from falling below the required threshold. Finally, Idea estimates the overall detection accuracy for the ADL by summing the individual estimates of each sensor pattern multiplied by their corresponding proportion. Algorithm 1 summarizes the impact estimation approach. At the end of this step, Idea computes the aggregate impact of a sensor as the sum of the impacts of the sensor for each individual ADL, and returns critical sensors both on a per ADL as well as aggregate basis.

Algorithm 1: Compute Impact of Sensor loss on an ADL

```
ComputeImpact{ADL,S_id}
impact ← 0
for SensorPattern P ∈ Adl do
    if S_id ∈ P then proportion ← ||P|| ÷ ||ADL||
    noLoss ← EstAccuracy(P, ADL)
    wLoss ← EstAccuracy(P − {S_id}, ADL)
    δ ← (noLoss − wLoss)/noLoss
    impact ← impact + proportion × δ
    ;
end
Return impact
```

Note that for ease of explanation we described impact estimation for a single sensor failure. However, the same calculation is easily extendable to estimating the impact of multiple sensor failures by carrying out the calculations for a set of failed sensors instead of a single one. We evaluate performance of Idea under multiple sensor failures in Section 4.

3.4 Sensor Failure Detection

We now describe how Idea detects sensor failures, while continuously monitoring ADLs in a smart home. Algorithm 2, shows the three techniques employed by Idea to generate failure alerts for both periodic and event-driven sensors. The first technique is for detecting the failure of sensors that send events (of their state) periodically. For such sensors, Idea raises a failure alert if a threshold amount of time has elapsed since last state report by the sensor. The second and third techniques are for event-driven sensors, which generate events only when certain activity is performed. For e.g., a working Toilet-door contact sensor reports an `"open"` (or `close`) event only when the door is opened (closed). Idea's task for such sensors that fail to report data over a period of time, is to differentiate between whether the corresponding ADL did not actually happen, or the sensor has failed. In our current design, we consider only fail-stop failures[4]. The second technique focuses on sensors that are identified as critical by the impact estimation component

[4]Our approach, *in principle*, can be adapted to sensors giving inconsistent values when they fail, but we did not have sufficient

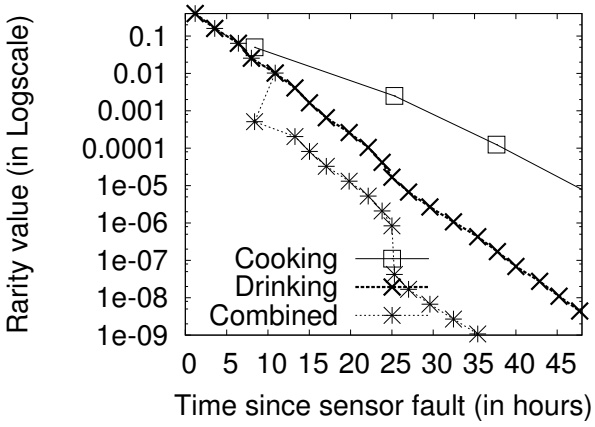


Figure 5: Detection of missing Fridge sensor in *KasterenA* dataset using three ADLs.

in the previous step. In particular, the second technique focuses on those critical sensors that have a 100% impact on detection accuracy, i.e. the ADL can no longer be detected in the absence of the sensor. We call these sensors *grave* sensors. For example, the *Front Door* sensor is a *grave* sensor for `Leave House` ADL in *KasterenA* dataset. For such sensors, Idea raises a failure alert if the time elapsed since the last detection of the *ADL* exceeds a threshold that is based on previously observed ADL occurrence data.

The third technique is for the rest of the sensors that do not satisfy the criteria of the previous two techniques. The approach for such sensors, involves computing a *rarity score* (ρ). The ρ essentially, is the probability that the sensor has not triggered, while certain related activities (ADLs) that the sensor participates in have been detected by the detector using a combination of other sensors. If the ρ value falls below a *rarity threshold* (τ) we raise a sensor failure alert. We now explain alert generation in the context of an example. In Figure 5, we show the amount of time it will take to generate an alert for missing `Fridge` sensor in *KasterenA* dataset using occurrence traces of two activities `Drinking` and `Cooking` from *KasterenA* dataset. On X-axis, we plot the time since the Fridge sensor stopped generating events. On Y-axis, we plot ρ values. The points on each line represent the time when an ADL was detected since the `Fridge` failed.

First, for every tuple <$\mathtt{s_i}$, $\mathtt{ADL_k}$> we calculate the $Pr(\overline{\mathtt{s_i}}|\mathtt{ADL_k})$, probability of *non-occurrence* of $\mathtt{s_i}$ given $\mathtt{ADL_k}$ has occurred. Specifically, in *KasterenA* dataset,
$Pr(\overline{\texttt{Fridge}}|\texttt{Cooking}) = 0.05$ and $Pr(\overline{\texttt{Fridge}}|\texttt{Drinking}) = 0.4$ respectively. Assuming, that τ is set to 0.1, we observe that the sequence `[Drinking, Cooking]` was detected over last 6 hours without a single `Fridge` event. The ρ for the above sequence is defined as $Pr(\overline{\texttt{Fridge}}|\texttt{Sequence})$. We observe that the sensor trigger for a given activity are independent of other activities and hence,
$Pr(\overline{Fridge}|Sequence)$
$= Pr(\overline{\texttt{Fridge}}|\texttt{Drinking}) \times Pr(\overline{\texttt{Fridge}}|\texttt{Cooking})$. Note that, for each detected activity in which an expected (based on prior history) sensor has not triggered, besides sensor failure, two other possibilities could have happened:

- Idea could had misclassified the activity itself. We counter this by weighting the probability of non-occurrence of the sensor for a given activity with the detection accuracy of the ADL ($\mathtt{Acc_k}$). This introduces a slack in alert generation times. Based on experimentation, ADL detection accuracy, $Acc_{\texttt{Drinking}} = 0.93$ and $Acc_{\texttt{Cooking}} = 0.89$.
Hence, $\rho = \{Pr(\overline{Fridge}|Drink) \times Acc_{Drink}\} \times \{Pr(\overline{\texttt{Fridge}}|\texttt{Cook}) \times Acc_{Cook}\} = 0.02$.

- Or (2), The user behavior has changed and she has stopped using the specific sensor for conducting an activity. In this case, we assume that the behavior of the user will not change for all activities at the same time, and the presence of sensor triggers caused by other activities will inform us that the sensor has not failed.

Further, we periodically re-learn all the rules to account for behavioral changes [5] We raise a sensor failure alert when $\rho < \tau$. Note that, we track ρ across multiple activities as it leads to a faster alert generation than considering a single activity. For example, from Figure 5, when τ is set to 0.01 or lower, tracking only `Drinking` ADL will generate an alert around the 18 hour mark while tracking only `Cooking` will generate an alert after 24 hours.

We define that a false failure alert for a sensor occurs when a sensor signal is received sometime after Idea raises a failure alert for the sensor. We note that a high value of τ will lead to larger number of false alerts since a a smaller number of ADLs with non-occurrence of a specific sensor will cause the ρ to quickly fall below τ. On the other hand a lower τ reduces false alerts but increases the alert generation times. We evaluate this tradeoff and present results in later sections(Section 4.4).

Algorithm 2: Missing Sensor alert generation

Data: S : Set of all sensors, t_{max} :, Maximum time allowed since last firing, τ : Rarity threshold
Phase 1 - Detect Stoppage of Periodic Sensors
forall the $\mathtt{s_{periodic}} \in \mathtt{S}$ **do**
  if *Time since last* $\mathtt{s_{periodic}}$ *event* $> t_{max}$ **then** generateAlert($\mathtt{s_{periodic}}$) ;
end
Phase 2 - Detect Stoppage for Grave Sensors
forall the $\mathtt{s_{critical}} \in \mathtt{S}$ **do**
  Data: $\mathtt{ADL_K}$ for which $\mathtt{s_{critical}}$ is critical
  if *Time since last* $\mathtt{ADL_k} > t_{max}$ **then** generateAlert($\mathtt{s_{critical}}$) ;
end
Phase 3 - Detect Stoppage using Rarity Score (ρ)
forall the $\mathtt{s_i} \in \mathtt{S} - \mathtt{S_{critical}} - \mathtt{S_{periodic}}$ **do**
  Data: $\mathtt{Seq} = ADL_1, ADL2, \ldots$, sequence of ADLs detected since last s_i event
  $\rho = Pr(\overline{s_i}|Seq) \times Acc_{Seq}$
  if $\rho < \tau$ **then** generateAlert($\mathtt{s_i}$) ;
end

3.5 Maintenance scheduling

The maintenance scheduling component in the cloud collects all sensor failure alerts across the various homes, and periodically determines the homes that need to be visited by maintenance personnel, and the order in which the visits should occur. To build the maintenance schedule, the system expects for each candidate home,

datasets to explore this extension.

[5] In a deployment setting, we could also ask the user/maintenance personnel to provide feedback when false alerts occur to trigger re-learning of rules.

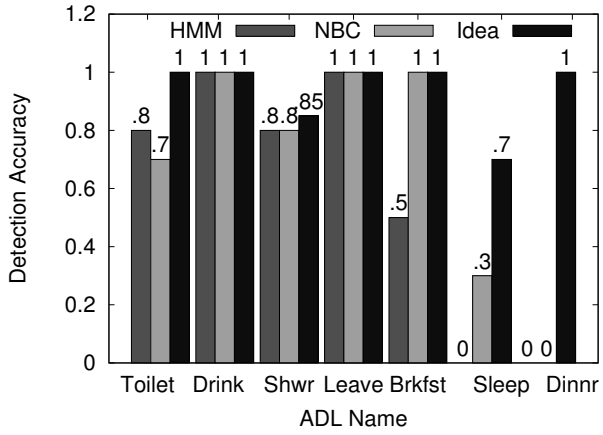


Figure 6: ADL Detection accuracy of Idea, NBC and HMM for the *KasterenA* dataset with no failed sensors.

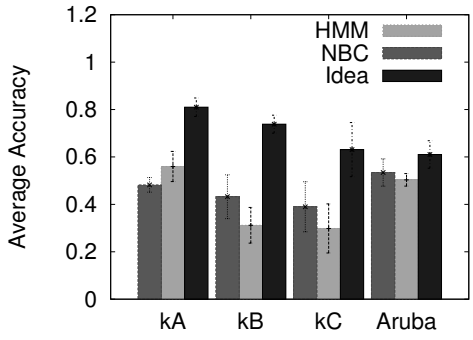


Figure 7: ADL Detection accuracy of NBC, HMM and Idea across multiple datasets described in table 1 with no sensor failures.

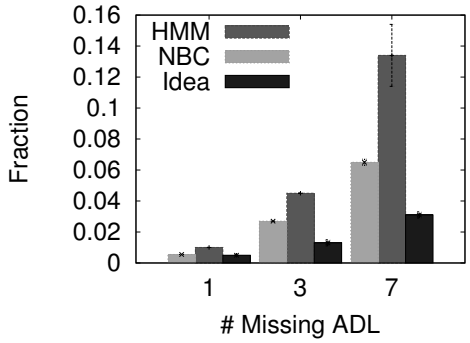


Figure 8: Reduction in Detection accuracy of NBC, HMM and Idea across all datasets in the presence of sensor failures

information on (a) impact of sensor failure, (b) location of the candidate home, and c) the maximum allowable reduction in accuracy for each ADL (or aggregated across all ADLs). The maintenance operation itself is simple and involves replacing *all* failed sensors in the home. Note that the main cost in maintenance involves the mechanic's hourly billing rate, and hence Idea optimizes for labor cost in the optimization formulation. Assuming that each mechanic's billing rate is same, Idea adds additional constraints on the maximum distance that can be traversed by each mechanic and then attempts to minimize the number of mechanics required to fix all the *critical* sensor failures. Thus minimizing the overall cost of maintenance while ensuring that the reduction in ADL detecting accuracy does not degrade below the required threshold.

We model the problem of finding optimal routes for maintenance personnel as a mixed integer program for Traveling Sales Man (TSP) problem. The solution to the problem is used for scheduling maintenance jobs periodically.

3.6 Prototype

Idea adopts a hybrid-deployment architecture wherein the service provider deploys an IoT gateway at the home to collect and process sensor data locally for enabling resident-centered functions like ADL monitoring, while sensor maintenance scheduling are centralized and run on the enterprise cloud. The centralized operation of maintenance scheduling allows the maintenance service provider to leverage failure information from all homes in the deployment to plan maintenance dispatches optimally. In our current implementation, the IoT gateway is emulated by an application that collects sensor data, performs ADL detection and sensor failure detection, and sends alerts to the cloud server that runs the maintenance scheduling component. We have implemented the components in Java, with the frequent itemset mining for ADL signature generation carried out using R packages. We use a CPLEX [36] solver for solving the mixed integer programming solution. The cloud server is a node hosted on a commercial cloud platform [37].

4. EVALUATION

In this section, we evaluate the performance of Idea. We begin by evaluating the constituent components of Idea to show the efficacy of our design choices. We then demonstrate the overall performance of Idea in increasing the time-before repair, and consequently the significant reduction in maintenance overhead across a variety of data sets. We begin by describing our methodology.

4.1 Methodology

We run experiments using the datasets listed in Table 1. For bench marking ADL detection performance, we use 80% of the dataset for training, and the remaining 20% for testing. In case we need to run an experiment for a much longer time than the available trace length, we re-create a longer trace by replaying the same trace of the home in a loop (e.g. for reliable detection of failed sensors in Section 4.4).

For statistical confidence, we randomly choose a different starting point in the trace for each run of the experiment; For example, the trace may start on the morning of Monday for one run, while starting at Thursday afternoon for the next run. Unless otherwise stated, we run each experiment 20 times and present the aggregate results. To evaluate the effects of sensor failure on ADL detection, we emulate a sensor failure by creating a trace with the failed sensor removed, and running the detector on this trace. To emulate sensor failures over large deployments in Section 4.5, we generate traces with failed sensors using the data sets as follows. We iterate over the trace, select a sensor at random and determine whether the sensor will fail at a specific time using a Weibull distribution, with mean time to failure of 1 year. Once sensor failure is determined, we remove its future instances from the trace. We present our evaluation results next.

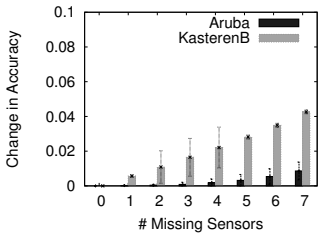
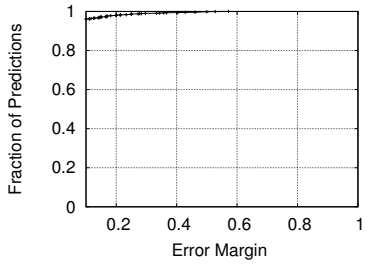
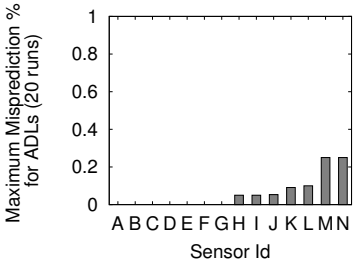


Figure 9: Reduction in ADL detection accuracy of Idea for *Aruba* and *KasterenB* datasets with increasing number of sensor failures.



(a) CDF of estimation error in estimating sensor failure impact on detection accuracy for all sensors and ADLs across all datasets.



(b) Maximum estimation error for prepare_breakfast across all sensors for *KasterenA* dataset.

Figure 10: Benchmarking impact estimation

4.2 ADL Detection

We first demonstrate the efficacy of the ADL detection component in Idea. We compare the performance of Idea with two other detectors (NBC [25] and HMM [22]) that have been used in the past for ADL detection. Briefly, NBC learns a probabilistic model of the ADL from the underlying sensor event data, while HMM learns a sequential model for each ADL based on the sequence of sensor events that constitute the ADL. In particular, we evaluate the detection accuracy of the three schemes, both in the presence and absence of sensor failures. Figure 6 shows the detection accuracy of Idea, HMM and NBC for each individual ADL in *KasterenA* dataset when there are no sensor failures. As can be seen from the plot, Idea provides the highest detection accuracy across all ADLs. We observe similar results for other datasets.

In Figure 7, we present the average accuracy of the three schemes across all ADLs for several representative datasets with no sensor failures. We find that Idea performs better than NBC and HMM schemes in each data set. NBC performs worse compared to Idea since the NBC model assumes independence among the sensor events, while Idea is designed to extract the dependencies among sensors on a per ADL basis. Similarly, HMM performs worse, since it is based on *sequences* of human activities, whereas human activities are inherently variable and can happen out of sequence several times. On the other hand, Idea is designed to seamlessly handle human induced variabilities and achieves higher detection performance.

We next evaluate detection performance in the presence of sensor failures. Figure 8 depicts the aggregate drop in detection accuracy (compared to the baseline case of no sensor failures) for all possible combinations of 1, 3 and 7 failed sensors across all datasets. We observe that the drop in aggregate accuracy is the least for Idea compared to NBC and HMM, and Idea performs significantly better compared to the other schemes with increasing number of sensor failures. This lower deterioration in performance can be explained in terms of Idea's ability to effectively extract redundancy among sensors and leverage it for better detection.

To further explain, in Figure 9, we plot the aggregate change in detection accuracy of Idea with increasing number of sensor failures for *Aruba* and *KasterenB* respectively . We find that the deterioration in performance with sensor failures is lower for *Aruba* than *KasterenB* datasets. On deeper investigation, we found that *Aruba* has higher redundancy compared to *KasterenB*, thus, demonstrating that Idea effectively leverages sensor redundancies for ADL detection.

4.3 Impact Estimation

The impact estimation component enables identifying *critical* sensors for each ADL by computing the severity of the sensor failures in terms of deterioration in ADL detection accuracy. We benchmark the performance of impact estimation in terms of the estimation error. For this, we compare the impact estimated by the model on the training data against the actual change in ADL detection accuracy obtained by running the detector twice on the test data, once with all sensors included, and again with the failed sensor removed from the trace. In Figure 10(a), we plot the CDF of error between the estimated impact and the actual impact obtained by running the detector. As can be seen from the plot, for more than 95% of the activities the error in estimation is zero. This result shows that impact estimation is able to accurately predict the change in recognition accuracy, thereby allowing us to identify critical sensors, and reduce scheduling of "avoidable" maintenance visits. Figure 10(b), depicts the maximum estimation errors (over all sensors in 20 runs) for a specific ADL in *KasterenA* dataset. We have obtained similar performance on other datasets as well and present *KasterenA* as an exemplar. Seven out of fourteen sensors (50%) have zero error in the impact estimate. The `Groceries_Cbrd` and `Plates_Cbrd` sensors have a maximum error of 25% for `Prepare_Breakfast`. This is due to the fact that the remaining `Prepare_Breakfast` sensors such as fridge are shared with other ADLs such as `Drinking` and `Snacking` whose sensors overlap with `Prepare_Breakfast` in the absence of an critical sensor like `Groceries_Cbrd` or `Plates_Cbrd`. In this case, Idea assumes all these instances will be misdetected and overestimates the impact.

4.4 Sensor Failure Detection

We next evaluate the responsiveness and tradeoffs involved in sensor failure detection and alert generation in Idea. Figure 11 shows the time taken by the failure detection module to detect the failure of the *Fridge* sensor in *KasterenB* dataset, by analyzing the (*non*-)occurrence of *Fridge* events during the performance of four ADLs: `Eating`, `Cooking`, `Storing Groceries` in isolation. As can be seen from the plot, the alert for the failed Fridge sensor is generated the earliest (12 hours) by monitoring `Eating` activity. This is due to the combined effects of high triggering probability(0.95) for the $[sensor, ADL]$ pair and high frequency of the ADL (every 11.2 hrs). We also observe that `Cooking` provides the next best detection time of 13 hours, which is 4 hours less than that of `Drinking` ADL although the `Drinking` ADL (once every 0.81 hours) is more frequent than `Eating` event (once every

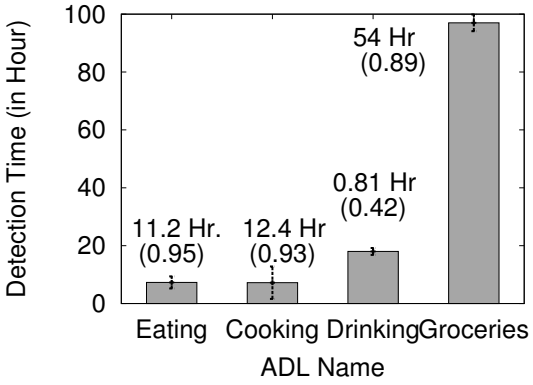


Figure 11: Time taken to detect failure of fridge sensor using a single ADL in *KasterenB* dataset. The numbers denote the occurrence frequency of each ADL and occurrence probability of the Fridge sensor(in brackets) for that ADL

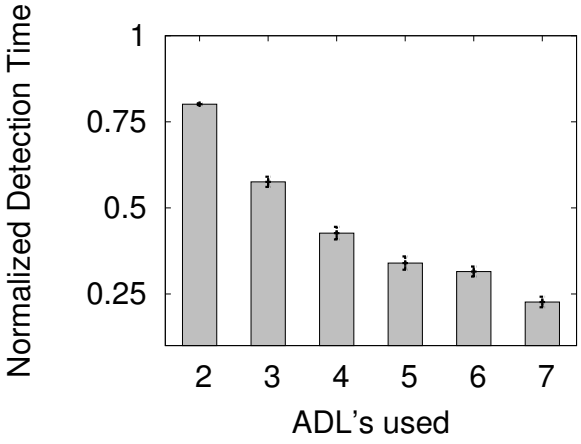


Figure 12: Time taken to detect failed sensors using multiple ADLs, with best ADL combination time normalized to single best ADL time.

12.4 hours). This is due to the fact that the probability of triggering fridge to Drinking is only 0.43, compared to 0.93 probability of triggering fridge for Cooking.

We next evaluate the improvements in failure detection times when multiple ADLs are combined and used together to detect a sensor failure. In Figure 12, we present the time taken to raise an alert from the time of occurrence of a failure, as a function of the maximum number of ADLs that are collectively monitored over all datasets, normalized to the time taken by the single best ADL to raise an alert. Particularly, we present the best possible time for alert generation by an ADL combination of specific size. As expected, the figure shows that the alert generation time decreases with an increasing number of ADL combinations. More interestingly, even having just combinations of three ADLs instead of one leads to halving of the time to detect a failure and raise an alert, showing the benefits of the collective monitoring scheme of Idea.

False positive trade-offs: Next we evaluate the effect of the rarity threshold on the sensor failure false (positive) alert rate. In Figure 13, we present the effect of the rarity threshold on the time taken to detect a sensor failure across all datasets. For the median case, compared to a baseline rarity threshold of 0.01, it takes 17× more time to detect sensor failures when the rarity threshold is set to 0.001. The detection time increase to 123× for the same failure when the rarity metric is set to 0.0001 (compared to baseline of 0.01). We next present the effect of the rarity threshold on false alert rates.

In our experiments, we found that with a rarity threshold of 0.001 we had 23× fewer false positive alerts compared to a threshold of 0.01, while the number of false alerts decreased 318× when the threshold was changed to 0.0001. The above results are expected, as a low value for rarity threshold implies that more instances of non firing events (when a ADL occurs) will have to be observed before the alert can be raised. Thus, a lower rarity threshold, will lead to a delay in raising a failure alert (false positive) while minimizing the number of false alerts (false positive).

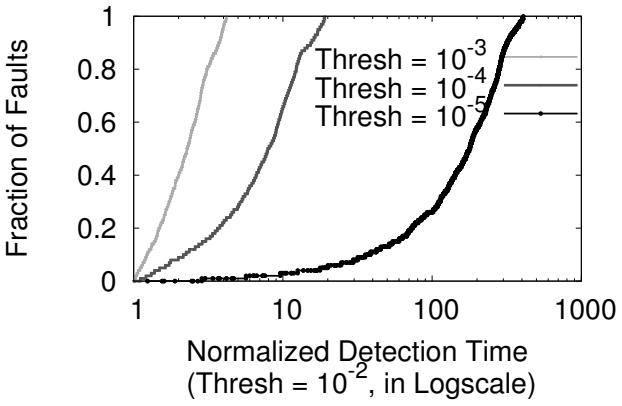


Figure 13: Normalized (to rarity threshold = 0.01) increase in time taken to detect sensor failures for different values of rarity threshold

4.5 End-to-end: Maintenance Scheduling

We now benchmark the performance of Idea on maintenance efficacy in terms of the number of maintenance visits and per-home maintenance inter-arrival times.

Methodology

To quantify the maintenance benefits, we begin by emulating the geo-graphical distribution of homes over a large city, in our case, the Manhattan region of NY city. We generate home locations (latitude, longitude) of ten thousand homes over a 400 sq miles region of Manhattan as shown in Figure 14 using the City and Network Generator(CING) [38] that generates representative traces of the population distribution, mobility patterns as well as spatial distribution of homes and offices. We then randomly assign to each home a set of sensors identical to one of the testbed homes listed in Table 1. We emulate error probabilities of individual sensors in each home using the well accepted Weibull distribution [29, 30] with mean time to failure (MTTF) of 1 year for a period of 2 years. Once a sensor is determined as failed, it remains failed till a maintenance person visits the home and fixes the sensor. Finally, we collate the failures across the ten thousand homes and determine the maintenance visit schedules for each day based on the alerts received the previous day. We consider and alert as critical if the

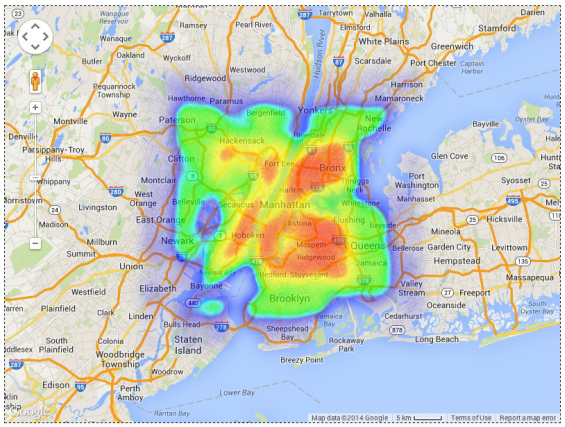


Figure 14: Home Location Distribution in NY

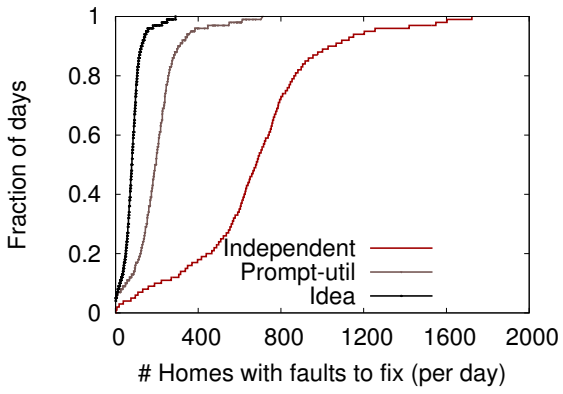


Figure 15: Number of homes with critical faults over 10K homes

impact of new sensor failures, along with potential prior failures is greater than 5%. We investigate the performance of Idea against three other schemes for scheduling maintenance visits:

- **Independent:** Any failed sensor in each home has to be fixed the next day.
- **Prompt-Util:** Maintenance is done only for sensors that are necessary for detecting one or more ADLs in the home. Any sensor that is part of at least one ADL signature (in Idea) is considered eligible for maintenance.
- **Periodic:** A maintenance person will periodically visit each home, and fix all the failed sensors on the day of visit.

We present the performance comparison results next.

Reactive Maintenance

In Figure 15, we present the CDF of the number of unique homes that need to be visited daily over the entire duration of the experiment. Observe that the number of homes is significantly fewer for Idea compared to other schemes. Specifically, we find that for more than 50% of the days Idea needs only 68 visits (less than 1% of the number of homes), while the independent scheme will require around 750 visits (11×) , and Prompt-Util needs around 134 visits (2×). Note that, Prompt-Util (a simplified version of Idea) itself leads to a significant reduction in the number of visits. In

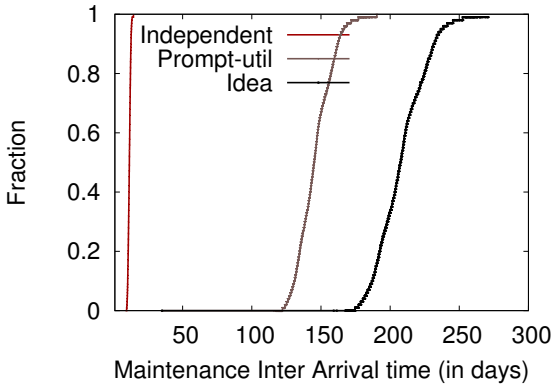


Figure 16: Inter maintenance visit time distribution for 10K homes

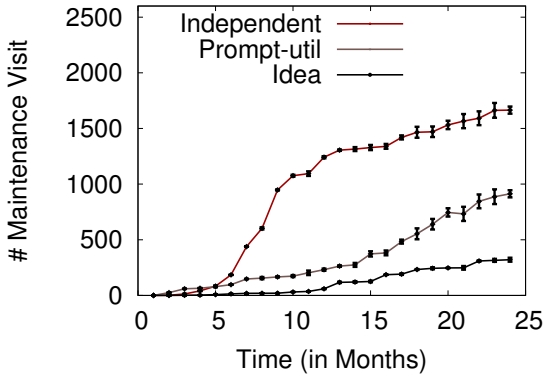


Figure 17: Maintenance visits with *Aruba* type deployments for 10K homes

Figure 16, we plot the CDF of the inter-maintenance visit times (in days) to individual homes. We find that for the average case, Idea needs to visit a home every 211 days, whereas Independent and Prompt-Util need maintenance visits every 18 and 128 days respectively.

We illustrate the effects of passing time on the maintenance workload in Figure 17 and Figure 18, the figure shows the per month average number of maintenance visits for the three schemes for homes with sensor configurations similar to *Aruba* and *KasterenB* respectively. The figure shows that the number of maintenance visits will increase with time, as expected, since more sensors will fail with increasing time. However, the overall visits for Idea are fewer than the other schemes. We also note that for *Aruba* the improvement in performance of Prompt-Util scheme over Independent is higher than that of *KasterenB* homes; this is due to the fact that *Aruba* homes have around 21% sensors not useful for ADL detection, while *KasterenB* has only 8% such sensors.

Periodic Maintenance

The benefit of a periodic visit scheme is the fixed number of maintenance visits; however, the scheme suffers from the possibility of critical sensor failures (leading to reduced detection accuracy of important ADLs) in the home till the next maintenance visit. In Figure 19(a) we depict the average number of sensor failures (both

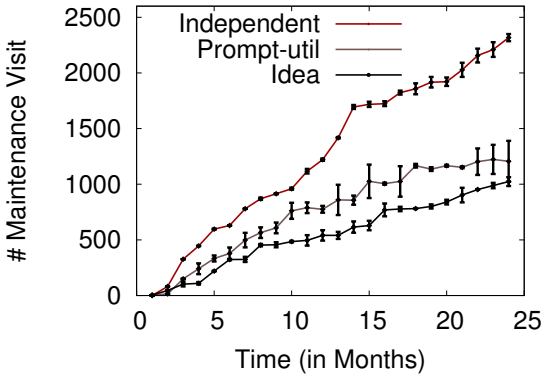


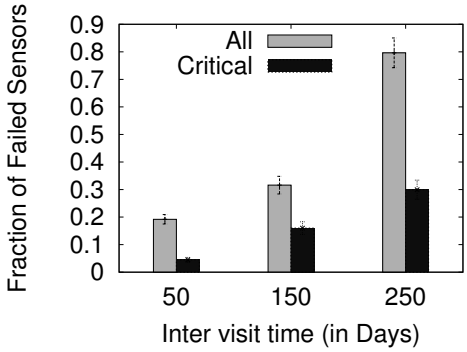
Figure 18: Maintenance visits with *KasterenB* type deployments

overall and critical) per home that will be present as a function of periodic maintenance times. As can be seen from the plot, with increased inter-visit time gap from 50 days to 250 days, the number of failed sensors also increases from 0.04 to 0.31. Figure 19(b) shows the number of days from the first critical failure till the arrival of maintenance operation. We find that on an average a home will have 19 days with failed critical sensors for a periodic visit of 50 days. The number goes up to 87 days for a visit gap of 250 days. Such under-performance of monitoring applications is clearly undesirable in a number of scenarios, esp. patient care and elderly care.

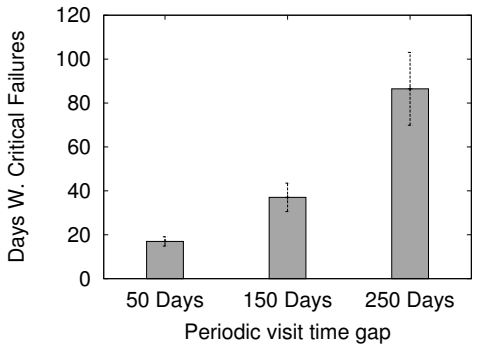
Effect of redundancy aware detector design on Maintenance load

We next benchmark the impact of Idea's redundancy aware detector design against other schemes such as NBC and HMM. In Figure 20, we plot the average inter-maintenance visit time (Time-before-repair) for the four schemes on a 1000 home *KasterenA* deployment. HMM and NBC improve Time-before-repair more than twice over the independent scheme having a Time-before-repair of 47 days. On the other hand, Idea has a approximately 2× longer Time-before-repair of 207 days compared to 101 and 117 days for NBC and HMM schemes respectively. This is expected as detection performance of Idea degrades more gradually compared to NBC, HMM as shown in Figure 8. We note that techniques such as SMART [27] that also attempt to reduce maintenance visits by relying on off-the-shelf NBC and HMM detectors will also generate lesser number of maintenance visits by using Idea instead.

Such significant reductions in maintenance overhead have a direct implication on resource provisioning for an IoT enterprise. In practice, the total distance that can be covered by a maintenance person on any given day is limited. We consider this constraint in our MIP formulation and allow the use of multiple people to cover the tasks at hand. We conduct two sets of experiments - one that limits the total distance traveled per day to 40 Km and the other that limits it to 20 Km (Figure 21). Under each of these settings we compute the number of maintenance people needed to complete "'X%"' of the tasks on any given day. In Figure 21(b), we find that under a 40 Km distance limit per day on the KasterenB dataset, we observe that our approach requires about 2.5x fewer maintenance personnel to cover 10% of the tasks and about 3.3x fewer personnel to cover all the tasks. In FigureFigure 21(a), we show similar results for 20 KM distance limit. These results demonstrate that our



(a) % failed sensors to be fixed



(b) Days with critical failures

Figure 19: Periodic maintenance scheme performance.

approach translates into significant (over one order of magnitude in dense deployments) savings in maintenance resources (costs), thereby making IoT enterprises more profitable.

5. RELATED WORK

We compare and contrast Idea with prior literature in the areas of diagnostics in smart homes, large scale asset management tools, robust sensor network deployment, fault detection in homogeneous sensor networks and ADL detection in smart homes.

Smart home fault diagnostics

The closest work to ours is SMART [27] that leverages application level semantics to identify *non-fail stop* failures. In the training phase, SMART learns multiple classifiers for recognizing the same set of activities based on different subsets of sensors. At run time, SMART analyzes the relative change in recognition accuracy of the different classifier instances, and uses this information to identify the failed sensors. We believe we make the following distinctions from SMART 1) SMART retrofits failure detection on state of the art detectors such as NBC and HMM by viewing the classifiers as black boxes and using relative changes in accuracy between the different classifiers for detecting a failure. On the other hand, Idea is a white box integrated approach that combines ADL detection with sensor failure detection and functional redundancy estimation for determining maintenance scheduling 2) SMART trains and simultaneously runs O(Number of Sensors) classifiers just for failure detection, with this ensemble of classifiers having no impact on the ADL detection accuracy. On the other hand, Idea runs a single classifier, that improves both ADL detection accuracy as well as sensor failure detection. 3) SMART cannot detect failures of sen-

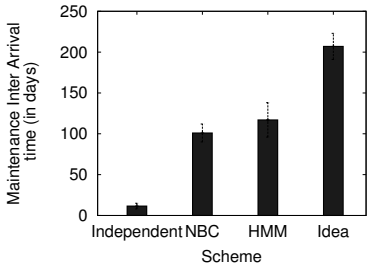
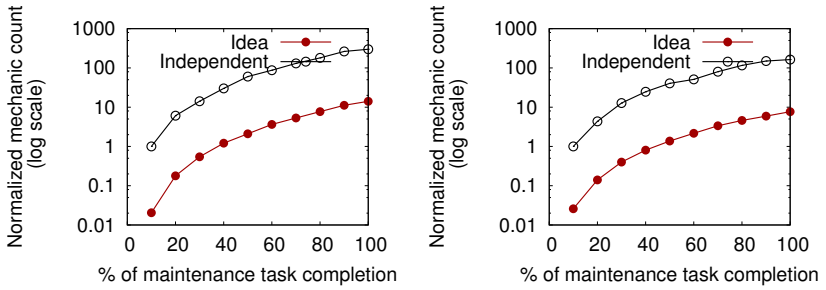


Figure 20: Average inter-maintenance visit times for a deployment of 1000 *KasterenA* homes across four schemes viz. Independent, HMM, NBC and Idea.



(a) 20 Km (b) 40 Km

Figure 21: Distribution of number of mechanics necessary to repair "'X%"' of faults in a day, normalized to number of mechanics necessary to fix 10% faults, with a constraint of (a) 20 Km and (b) 40 Km travel radius

sors that are not frequently used in any of the activities. On the other hand, Idea will detect the failures of these sensors as well.

The authors in [28] present a complementary technique to detect failures by learning the regular patterns of sensor firings with respect to electrical appliance usage, and raising an alert when there is significant deviation from the regular firing interval distribution. This is an example of approaches that require extra electrical sensing infrastructure, and can be leveraged in Idea as well.

Commercial asset management software

Existing enterprise asset management products such as [39, 40] automate the maintenance of large scale device deployments. Akin to Idea such software allows (automatic as well as manual) logging of fault tickets about various assets being maintained, and provides sophisticated algorithms for scheduling of maintenance operations. Idea enables complementary functionality by learning the inherent redundancies in capabilities of various sensors, which allows for significantly extending the time-before-repair.

Robust homogeneous sensor network deployments

There has been considerable research [8–15] in understanding and enhancing redundancy in *homogeneous* sensor deployments to enhance the lifetime of deployments, from both an energy efficiency and fault tolerance perspective. [41] and [42] enable applications to specify at a high level the relevant contexts and context changes, while the system optimizes and manages the sensing based on the available resource state. Similarly, [43] proposes an energy efficient sensing framework by exploiting the relationship among contexts, for example between driving and being at home, as well as the relationship between the cost of sensing and inferring a context. The key distinction of Idea from such prior art, lies in the fact that it focuses on systematically understanding the functional redundancies amongst *heterogeneous* sensors resulting from resident behavior. Furthermore, home deployments involve heterogeneous event-driven sensors that are triggered by human action as opposed to continuous sensing systems for which the prior approaches were developed.

Fault detection in homogeneous sensor networks

A number of works [44–46] address the problem of fault detection in sensor networks. The authors in [47] provide a survey of fault management approaches in sensor networks. Akin to Idea, authors in [48, 49] a) identify sensor faults by identifying outliers among neighboring sensors to build a community of trust among nodes to provide high data integrity and detect faulty sensors and b) prompt for human assistance to improve fault detection. However, in contrast to Idea prior work does not deal with finding outliers in heterogeneous sensor deployments.

Smart home ADL detection

A number of recent works [20, 21, 23, 24] have addressed the problem of detecting ADLs from sensor firings in smart homes using machine learning based approaches Hidden Markov Models, Naive Bayes Classifier, Conditional Random Fields etc.. Specifically, the authors in [23] propose using hidden semi-Markov models for activity recognition while incorporating duration of activities as a feature for classification. Similarly, Roy et al. [50] address the problem of ADL recognition in multi-inhabitant environments by combining smart phone and infrastructure sensors and learning spatio-temporal constraints to infer the attributes such as location of the user.

However, the above body of work does not customize the design of their classifiers to address the unique problems and opportunities associated with sensor faults in smart home deployments.

6. CONCLUSION

In this paper, we design Idea a novel sensor failure management system for IoT-enabled smart environments. Our approach is based on learning and leveraging the functional redundancy available among sensors for robust ADL detection in the presence of sensor failures. Specifically, our approach leverages the fact that there is inherent redundancy when combinations of sensors monitor *activities of daily living* (ADLs) in smart environments, and the utility or impact of each sensor depends on the activity being monitored and the rest of the sensors available for detecting the activities. Based on estimated impact of sensor failures, the system schedules necessary maintenance visits. Results show that Idea significantly increases the *time-before-repair* of homes, thereby reducing the maintenance overhead of keeping such IoT deployments functioning as expected.

7. ACKNOWLEDGMENT

We thank our shepherd Aruna Balasubramanian and other anonymous reviewers whose feedback helped bring the paper to its final form.

8. REFERENCES

[1] Telehealth and self-monitoring in the elder-care markets: A global analysis. http://blog.bccresearch.com/telehealth-and-self-monitoring-in-the-elder-care-markets-a-global-analysis-0.

[2] V Rialle, C Ollivet, C Guigui, and C Hervï£¡. What do family caregivers of alzheimer's disease patients desire in smart home technologies? *Methods of information in medicine*, 47:63–69, 2008.

[3] G et. al. Acampora. A survey on ambient intelligence in healthcare. *Proceedings of the IEEE*, 101(12):2470–2494, 2013.

[4] S Robben, M Pol, and B Krose. Longitudinal ambient sensor monitoring for functional health assessments: a case study. In *UbiComp*, 2014.

[5] G. LeBellego, N. Noury, G. Virone, M. Mousseau, and J. Demongeot. A model for the measurement of patient activity in a hospital suite. *Information Technology in Biomedicine, IEEE*, 10:92–99, 2006.

[6] Basma M. Mohammad El-Basioni, Sherine Mohamed Abd El-Kader, and Hussein S. Eissa. Independent living for persons with disabilities and elderly people using smart home technology. *International Journal of Application or Innovation in Engineering & Management*, 2014.

[7] Timothy W. Hnat, Vijay Srinivasan, Jiakang Lu, Tamim I. Sookoor, Raymond Dawson, John Stankovic, and Kamin Whitehouse. The hitchhiker's guide to successful residential sensing deployments. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys '11, pages 232–245, New York, NY, USA, 2011. ACM.

[8] Shouzhi Huang and Xuezeng Zhao. Redundant nodes elimination in wireless sensor networks. In *Information Computing and Applications*, pages 48–58. Springer, 2013.

[9] Chi-Fu Huang and Yu-Chee Tseng. The coverage problem in a wireless sensor network. In *Proceedings of the 2Nd ACM International Conference on Wireless Sensor Networks and Applications*, WSNA '03, pages 115–121, New York, NY, USA, 2003. ACM.

[10] Di Tian and Nicolas D. Georganas. A coverage-preserving node scheduling scheme for large wireless sensor networks. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 32–41. ACM Press, 2002.

[11] Mihaela Cardei, My T Thai, Yingshu Li, and Weili Wu. Energy-efficient target coverage in wireless sensor networks. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 3, pages 1976–1984. IEEE, 2005.

[12] Habib M. Ammari and Sajal K. Das. Fast track article: Scheduling protocols for homogeneous and heterogeneous k-covered wireless sensor networks. *Pervasive Mob. Comput.*, 7(1):79–97, February 2011.

[13] Honghai Zhang and Jennifer C Hou. Maintaining sensing coverage and connectivity in large sensor networks. *Ad Hoc & Sensor Wireless Networks*, 1(1-2):89–124, 2005.

[14] Yong Gao, Kui Wu, and Fulu Li. Analysis on the redundancy of wireless sensor networks. In *IN PROC. 2ND ACM INTL. WORKSHOP ON WIRELESS SENSOR NETWORKS AND APPLICATIONS (WSNA*, pages 108–114. ACM Press, 2003.

[15] Rajagopal Iyengar. Low-coordination topologies for redundancy in sensor networks. In *In ACM MobiHoc*, pages 332–342, 2005.

[16] Benchmark datasets; datasets for activity recognitions. https://sites.google.com/site/tim0306/datasets.

[17] Wsu casas dataset. http://ailab.wsu.edu/casas/datasets/.

[18] K.D. Feuz, D.J. Cook, C. Rosasco, K. Robertson, and M. Schmitter-Edgecombe. Automated detection of activity transitions for prompting. *Human-Machine Systems, IEEE Transactions*, 2014.

[19] Prafulla Nath Dawadi, Diane Joyce Cook, and Maureen Schmitter-Edgecombe. Automated clinical assessment from smart home-based behavior data. *IEEE Journal of Biomedical and Health Informatics*, 2015.

[20] Tim van Kasteren, Athanasios Noulas, Gwenn Englebienne, and Ben Kröse. Accurate activity recognition in a home setting. In *Proceedings of the 10th International Conference on Ubiquitous Computing*, UbiComp '08, pages 1–9, New York, NY, USA, 2008. ACM.

[21] Parisa Rashidi, Diane J. Cook, Lawrence B. Holder, and Maureen Schmitter-Edgecombe. Discovering activities to recognize and track in a smart environment. *IEEE Trans. Knowl. Data Eng.*, 23(4):527–539, 2011.

[22] Aaron S. Crandall and Diane J. Cook. Using a hidden markov model for resident identification. In *Sixth International Conference on Intelligent Environments, IE 2010, Kuala Lumpur, Malaysia, July 19-21, 2010*, pages 74–79, 2010.

[23] TLM Van Kasteren, Gwenn Englebienne, and Ben JA Kröse. Activity recognition using semi-markov models on real world smart home datasets. *Journal of ambient intelligence and smart environments*, 2(3):311–325, 2010.

[24] Emmanuel Munguia Tapia, Stephen S Intille, and Kent Larson. *Activity recognition in the home using simple and ubiquitous sensors*. Springer, 2004.

[25] T. van Kasteren and B. Krose. Bayesian activity recognition in residence for elders. In *3rd IET International Conference on Intelligent Environments (IE 07)*, pages 209–212, 2010.

[26] L Liao, D Fox, and H Kautz. Location-based activity recognition using relational markov networks. In *Proc. Int. Joint Conf. Artif. Intell.*, 2005.

[27] Krasimira Kapitanova, Enamul Hoque, John A. Stankovic, Kamin Whitehouse, and Sang H. Son. Being smart about failures: Assessing repairs in smart homes. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, pages 51–60, New York, NY, USA, 2012. ACM.

[28] Sirajum Munir, John Stankovic, et al. Failuresense: Detecting sensor failure using electrical appliances in the home. In *Mobile Ad Hoc and Sensor Systems (MASS), 2014 IEEE 11th International Conference on*, pages 73–81. IEEE, 2014.

[29] Hazewinkel Michiel ed. Weibull distribution. *Encyclopedia of Mathematics*, 2001.

[30] Curt Ronniger. Reliability analyses with weibull. http://www.crgraph.com/Weibull.pdf, 2012.

[31] Department of Defense USA. Electronic reliability design handbook. "http://reliabilityanalytics.com/reliability_engineering_library/MIL-HDBK-338-V1_Electronic_Reliability_Design_Handbook_15_Oct_1984/MIL-HDBK-338-V1_Electronic_Reliability_Design_Handbook_15_Oct_1984_pp_2.htm", 1984.

[32] Seymour Morris. Online redundancy calculator for effective failure rate of n active redundant units, with m required for success (with repair), assuming either exponential or Weibull failure distributions. http://reliabilityanalyticstoolkit.appspot.com/active_redundancy_integrate, 2010.

[33] Ehsan Nazerfard, Parisa Rashidi, and Diane J. Cook. Using association rule mining to discover temporal relations of daily activities. In *Proceedings of the 9th International Conference on Toward Useful Services for Elderly and People with Disabilities: Smart Homes and Health Telematics*, ICOST'11, pages 49–56, Berlin, Heidelberg, 2011. Springer-Verlag.

[34] P. Kodeswaran, R. Kokku, M. Mallick, and S. Sen. Demultiplexing Activities of Daily Living in IoT enabled Smarthomes (to appear). In *IEEE Infocom*, 2016.

[35] Nirmalya Roy, Archan Misra, and Diane J. Cook. Infrastructure-assisted smartphone-based ADL recognition in multi-inhabitant smart environments. In *2013 IEEE International Conference on Pervasive Computing and Communications, PerCom 2013, San Diego, CA, USA, March 18-22, 2013*, pages 38–46, 2013.

[36] Cplex optimizer. http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/.

[37] P Kodeswaran and S Sen. Intellihome: Activity detection in smart homes on IBM BlueMix. http://intellihome.eu-gb.mybluemix.net/.

[38] S. Mitra, S. Ranu, V. Kolar, A. Telang, A. Bhattacharya, R. Kokku, and S. Raghavan. Trajectory aware macro-cell planning for mobile users. In *Computer Communications (INFOCOM), 2015 IEEE Conference on*, pages 792–800, April 2015.

[39] Maximo:ibm enterprise asset management. http://www-03.ibm.com/software/products/en/maximoassetmanagement.

[40] Sap–enterprise asset management. http://go.sap.com/solution/lob/asset-management.html.

[41] Seungwoo Kang, Jinwon Lee, Hyukjae Jang, Hyonik Lee, Youngki Lee, Souneil Park, Taiwoo Park, and Junehwa Song. Seemon: Scalable and energy-efficient context monitoring framework for sensor-rich mobile environments. In *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services*, MobiSys '08, pages 267–280, New York, NY, USA, 2008. ACM.

[42] Seungwoo Kang, Youngki Lee, Chulhong Min, Younghyun Ju, Taiwoo Park, Jinwon Lee, Yunseok Rhee, and Junehwa Song. Orchestrator: An active resource orchestration framework for mobile context monitoring in sensor-rich mobile environments. In *PerCom*, 2010.

[43] Suman Nath. Ace: Exploiting correlation for energy-efficient and continuous context sensing. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, pages 29–42, New York, NY, USA, 2012. ACM.

[44] Jessica Staddon, Dirk Balfanz, and Glenn Durfee. Efficient tracing of failed nodes in sensor networks. In *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, WSNA '02, pages 122–130, New York, NY, USA, 2002. ACM.

[45] Sapon Tanachaiwiwat, Pinalkumar Dave, Rohan Bhindwale, and Ahmed Helmy. Secure locations: routing on trust and isolating compromised sensors in location-aware sensor networks. In Ian F. Akyildiz, Deborah Estrin, David E. Culler, and Mani B. Srivastava, editors, *SenSys*, pages 324–325. ACM, 2003.

[46] S. Rost and H. Balakrishnan. Memento: A health monitoring system for wireless sensor networks. In *Sensor and Ad Hoc Communications and Networks, 2006. SECON '06. 2006 3rd Annual IEEE Communications Society on*, volume 2, pages 575–584, Sept 2006.

[47] Lilia Paradis and Qi Han. A survey of fault management in wireless sensor networks. *Journal of Network and Systems Management*, 15(2):171–190, 2007.

[48] Nithya Ramanathan, Thomas Schoellhammer, Eddie Kohler, Kamin Whitehouse, Thomas Harmon, and Deborah Estrin. Suelo: Human-assisted sensing for exploratory soil monitoring studies. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys '09, pages 197–210, New York, NY, USA, 2009. ACM.

[49] Saurabh Ganeriwal, Laura K. Balzano, and Mani B. Srivastava. Reputation-based framework for high integrity sensor networks. *ACM Trans. Sen. Netw.*, 4(3):15:1–15:37, June 2008.

[50] N. Roy, A. Misra, and D. Cook. Infrastructure-assisted smartphone-based adl recognition in multi-inhabitant smart environments. In *Pervasive Computing and Communications (PerCom), 2013 IEEE International Conference on*, pages 38–46, March 2013.