Power Management for a Distributed Wireless Health Management Architecture

Sankalita Saha [1], **Bhaskar Saha** [1], **and Kai Goebel** [2]

[1] *MCT/NASA Ames Research Center, Moffett Field, CA, 94035, USA*
Sankalita.Saha-1@nasa.gov
Bhaskar.Saha@nasa.gov
[2] *NASA Ames Research Center, Moffett Field, CA, 94035, USA*
Kai.Goebel@nasa.gov

ABSTRACT

Distributed wireless architectures for prognostics is an important enabling step in prognostic research in order to achieve feasible real-time system health management. A significant problem encountered in implementation of such architectures is power management. In this paper, we present robust power management techniques for a generic health management architecture that involves diagnostics and prognostics for a system comprising multiple heterogeneous components. Our power management techniques are based on online dynamic monitoring of the sensor battery discharge profile which enables accurate predictions of when the device should be put into low power modes. In our architecture, low power mode is achieved by run-time sampling rate modification through sleep states. Our experiments with a cluster of smart sensors for a hybrid diagnostics and prognostics architecture show significant gains in power management without severe loss in performance.

1 INTRODUCTION

Recent times have witnessed a sharp increase in complexity of modern aircraft and spacecraft systems and it is projected to further increase in future. Thus, it is expected that prognostics and health management for such systems would also increase in complexity and involve several challenges such as a) large amounts of sensor data due to presence of more components with elaborate interconnections and sensor instrumentation along with high frequency diagnostic sampling b) increasingly more complex algorithms incorporating higher expectations of system performance being deployed that exceed the capabilities of single-processor systems due to high memory computation requirements and c) robustness to failure of sensor modes and fast recovery. The most common architecture for such systems - mainly due to ease of development - is centralized i.e., a central processing device collects data from sensors and processes them, while executing various health management algorithms. However, such a system does not scale well for data intensive and complex health management systems and thus distributed system architectures are essential.

The distributed architecture (first proposed in (Saha *et al.*, 2008)) discussed in this paper comprises multiple wirelessly connected smart sensor devices. A smart sensor comprises a sensing device along with a microprocessor in order to enable functionalities beyond sensing such as low-weight signal processing before transmitting to a controller. These devices would monitor different parts of the distributed health management system and collaborate when computationally intensive prognostic algorithms or large amounts of data are involved that cannot be handled efficiently by a single processor/node. Recent advances in smart sensor technology combining the power of embedded computing devices with sensors and wireless transmission technology make the practical implementation of such systems feasible.

An important problem in wireless sensor networks is efficient power management which has been explored in various contexts in order to prolong battery life as well as reduce cooling requirements. With longer battery life, frequent replacement of sensors is avoided thereby leading to more autonomous systems and, hence, less maintenance costs. For many applications, replacement of sensors with dead batteries is possible only after significantly long intervals, thus, requiring aggressive battery life management. Most battery conservation algorithms for embedded devices rely on forcing the device to switch to a low-power mode which entails reduction in computational resources and hence reduced performance. Thus, an efficient power management technique involves balancing trade-offs between performance and operational costs.

In this paper we present regression based power management techniques that provide robust battery capacity management without significant loss in performance. Our battery management architecture relies on individual on line monitoring of battery discharge profile which in turn enables accurate prediction of remaining battery capacity. The computation load of the smart sensors are then adjusted dynamically to ensure longer battery life. The specific target architecture in our case is composed of a network of smart sensors from Sun Microsystems called SPOT (Small Programmable Object Technology) devices. However, the techniques developed in this paper are generic enough for use in other distributed wireless networks. The paper is organized as follows: in section 2 we discuss related work, followed by an overview of our distributed health management architecture in section 3. In section 4, we discuss our power management techniques in details. In section 5 we present the details of our experiments and present results followed by conclusions in section 6.

2 RELATED WORK

Most of the efforts for power conservation in embedded devices are focused on reducing the power consumption by forcing the embedded computing element to operate in a low power mode more well-known as *sleep* states. Dynamic voltage scaling (DVS) implements such low power modes by lowering the supply voltage and operating frequency when the computing load is less and has been explored extensively (M. Nakai and Shimura, 2005). In (Pillai and Shin, 2001; **?**), the authors present power management by using the real-time scheduler and task manager within the operating system to decide when to implement voltage scaling. However, forcing such low power modes affect the performance since the sampling rate and computing speeds are affected.

Wireless sensor networks present significantly more challenges compared to other embedded devices, since they support extensive wireless communication which affects the power consumption severely. Thus, a lot of focus for power conservation for embedded wireless devices has been based on communication patterns. For example, in (Anand *et al.*, 2003), the authors present an adaptive power management technique that adapts its power consumption behavior based on the communication access patterns. However, it requires considerable prior knowledge of the applications in order to exploit the access patterns for energy savings. Similarly, in (Chiasserini and Rao, 2000), the authors propose a method in which the sensor nodes are maintained in *sleep* mode unless woken up by a Radio Frequency (RF) signal when required to participate in an activity. Distributed power management techniques, where a cluster of sensors instead of one central monitor participate in battery capacity savings, have been explored as well. For example, in (R. Tynan and Hare, 2005) the authors use intelligent software agents to perform distributed power management. The use of timeouts and local voting amongst video sensor clusters in order to collaboratively decide when to shut down a sensor is demonstrated in (Zamora *et al.*, 2007).

An important consideration for efficient power management is accurate prediction of when to transition to a low power mode. Some of the methods discussed above implement this by either relying on off line computing and communication load analysis, or high-level operating system based indicators in order to make this decision. However, for sensor networks that offer relatively low-level computational power, such methods may become too expensive in terms of computation. An alternative method is to monitor battery charge and discharge level and use this information to manage power (Jayashree *et al.*, 2004). In this paper, the authors propose new communication protocols in order to schedule tasks efficiently based on battery life predictions. However, the authors restrict to an off line model of the battery that does not account for run-time change in discharge profile.

From this discussion, one may observe that there is a distinct lack of work on exploiting dynamic battery life estimation for efficient power management. Such estimates can be made through simple yet accurate models

3 DISTRIBUTED HEALTH MANAGEMENT ARCHITECTURE

In this section we provide a brief overview of our distributed health management architecture. Further description of this architecture can be obtained in (Saha *et al.*, 2008; 2009). This architecture is comprised of a network of smart sensor devices that monitor the health of various subsystems or modules. The health management system comprises of mainly sensing, diagnostics and prognostics operations. The sensors collect component signals and monitor them using low-weight diagnostics algorithms. Prognostics operations are triggered based on user defined thresholds. An example of such a distributed prognostics system is shown in Figure 1.

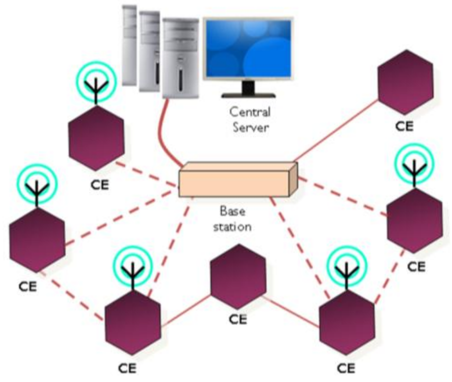

Figure 1: Overview of distributed prognostics system architecture. (Adapted from figure 1 in (Saha *et al.*, 2008).

The sensor devices – which we call computing elements (CEs) – consist of a sensor or a set of sensors and a communication device, i.e., a wireless transreceiver or wired communication capabilities besides an embedded processing element. Though in many instances wired sensor network may be preferable, in this paper we focus on wirelessly connected devices for enhanced flexibility. Such a wireless network has

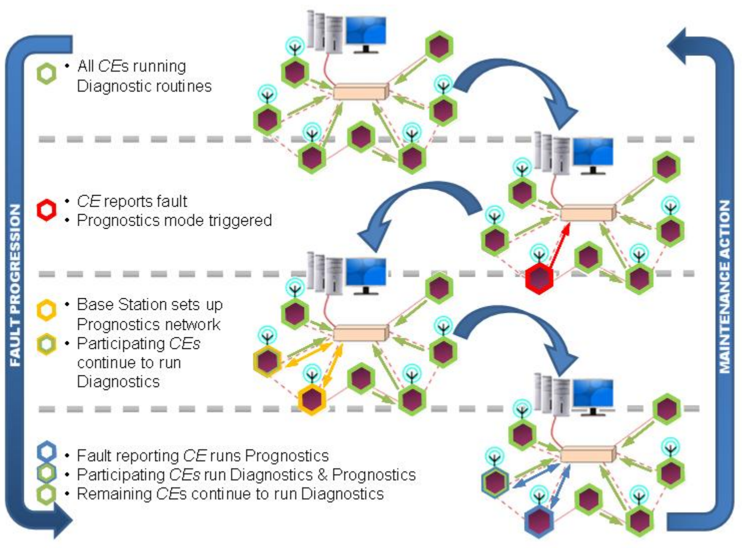
All CEs running Diagnostic routines
CE reports fault
Prognostics mode triggered
Base Station sets up Prognostics network
Participating CEs continue to run Diagnostics
Fault reporting CE runs Prognostics
Participating CEs run Diagnostics & Prognostics
Remaining CEs continue to run Diagnostics
FAULT PROGRESSION
MAINTENANCE ACTION

Figure 2: Flow diagram for diagnostics and prognostics operations in the distributed architecture. (Adapted from figure 2 in (Saha *et al.*, 2009).

both advantages and disdvantages. Major application domains for such a health management architectures are aircraft and spacecraft systems, where reduced physical weight of such monitoring structure is of prime importance and hence wireless systems are far more preferable. However, wireless networks can incur significant signal interference from other on-board interference. This problem is handled through various methods including specification of special communication frequency bands and is an important research area. However, we do not delve into detailed discussion of this problem since the main focus of this paper is power management issues.

As mentioned earlier, there are two main operating modes for a CE: diagnostics and prognostics (Saha *et al.*, 2008). The CEs are arranged as clusters that monitor and manage the health of a sub-system, and hierarchically the whole system. The default mode of operation for a CE is diagnostics where it monitors a given sub-system or component through a low weight diagnostics algorithm. During this monitoring, if a CE detects a critical condition, it raises a flag and starts the prognostics mode. In this mode it initiates the formation of a cluster of CEs that collaboratively performs the prognostics task. The prognostics task is expected to be computationally expensive involving complex algorithms. The amount of data also increases due to increase in the frequency of component signal sampling for more accurate prognostic estimates, which in turn, increases the memory requirements as well. If the CE does not have enough computational resource to perform the overheads of the distributed system management, it informs the base station of the prognostic task which then performs system management tasks, such as scheduling, synchronization, load distribution and so on. In the prognostics mode it is not necessary that all CEs within a cluster monitoring a subsystem participate in the prognostics task; some of them may lack the necessary computing power to support the additional new task. Also, the diagnostics operations continue uninterrupted in the prognostics mode. To ensure that a participating CE can support such multi-tasking efficiently the prognostics algorithms need to be distributed efficiently.

In many cases the sensor capabilities of the CEs may not be utilized at all, i.e., they could act as monitors for the rest of the system - schedule tasks, detect failures and initiate recovery, provide access to resources such as an external database and so on. These CEs are specially designated as base stations. The base station is also, typically, connected to a more power computing resource (to aid in collection and storage of system data) which in our case was a PC. Since the experiments presented in this paper were carried out in a laboratory setting, suitable integration steps would be involved to scale this architecture for full-scale aerospace applications. Thus, in practical deployment, more powerful computing resource could be on-board computers or communication devices that relay information to computers on ground stations instead of a laptop .

Figure 2 shows, in detail, the typical execution flow in our health management architecture. As mentioned earlier, each CE monitors different components or subsystems such as battery health, actuator faults, health of electronic components and so on. It can also be responsible for diagnostic monitoring of a sub-system comprising multiple components. In most cases the raw data collected is refined using diagnostics algorithms, and only a summary is reported to the base station. But, in many cases, when the CE does not have enough computing power - for example, in order to support heavy sampling rate of data collection - or remaining battery life, it can periodically send packets of raw data which can then be analyzed offline.

The base station monitors all the CEs and coordinates tasks. Also, the base station maintains information regarding CE resource availability. As mentioned earlier, the base station may not coordinate all the tasks in local prognostics operation which may be handled by the CE that raised the flag or another local leader/server may be chosen. This decision of leader selection in a distributed prognostic operation is based on multiple factors, including resource availability, proximity of the participating CEs etc and is a topic for future research.

3.1 Hardware Description

The smart sensor device used in our architecture is the Sun Microsystems SPOT device. A free range Sun SPOT is a small, wireless, battery powered experimental platform built by stacking a Sun SPOT processor board with a sensor board and battery as shown in figure 3. The smaller base station consists of just the processor board in a plastic housing. In terms of processing power, each Sun SPOT has a 180MHz 32-bit ARM920T core processor with 512 Kb RAM and 4 Mb Flash. The Sun SPOTs communicate using radio channels. The processor board has a 2.4 GHz radio with an integrated antenna on the board. The radio is a TI CC2420 (formerly ChipCon) and is IEEE 802.15.4 compliant. Each processor board has a USB interface which can be connected to a host PC to charge the battery. In our implementation, the free ranging SPOT devices act as CE while the SPOT base station acts as the default base station.

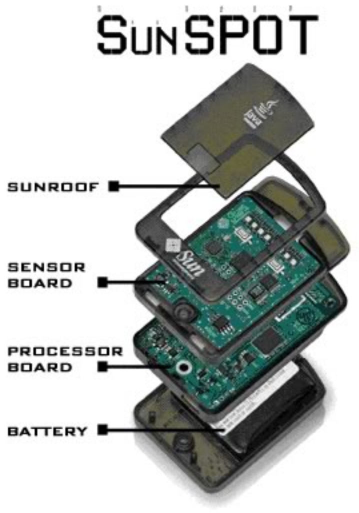

Figure 3: Anatomy of a free ranging Sun SPOT device (courtesy of www.sunspotworld.com).

Each free ranging SPOT is powered using a rechargeable, 720 mAh Lithium-ion (Li-ion) battery. The SPOT devices provide for automatic power management through an Atmel Atmega88 microcontroller. However, this controller does not provide aggressive battery management. It mainly ensures two functions: a) forcing a SPOT device into deep sleep when it is inactive for a long interval b) turning off the SPOT device when the battery capacity falls below critical level, which is approximately 3.2V. In our power management architecture, this controller is used in order to probe the battery to read current being drawn and corresponding voltage drop at regular intervals in order to monitor the battery usage. The controller is also used to induce *shallow sleep* modes in order to conserve battery. During *shallow sleep*, the processor clock is stopped and is restarted when an interrupt is received. The main difference between *shallow* and *deep* sleep mode lies in being able to react to external stimuli: during *shallow sleep* the CE can react to an external stimuli such as radio, while in *deep sleep* no such activity is allowed. The base station SPOT does not have its own battery and instead draws power via the USB connection to the host PC.

3.2 SUN Spot based Health Management Architecture

In the system considered in this paper, four free ranging CEs are involved in addition to the base station. All CEs perform diagnostics and one of the CE performs particle filter based prognostics routine on a Li-ion battery health data set from an offline source in collaboration with the base station. The number of particles used in the prognostics routine is 300, which is significantly high. Further details of this algorithm can be obtained in (Saha and Goebel, 2008), while details of its distributed implementation can be obtained in (Saha *et al.*, 2008). Note that this battery prognostics algorithm is distinct from the battery discharge model based power management routine presented in this paper. It is an example of a health management application and can be replaced by any other prognostics routine such as electronics prognostics. After startup of the system, during initialization, the base station communicates with remote CEs to gather information regarding available resources. The CEs collect vibration and temperature data and send the data to the base station periodically. Two of the CEs also run frequency analysis of the collected data by executing FFT (Fast Fourier Transform) on the dataset.

In our system, prognostics is triggered by the base station after it detects an anomaly. Based on resource information, it selects one of the CEs (say CE1) to collaborate in prognostics on battery health data. It allocates task share to this CE and acts as a leader for the prognostics routine as well. CE1 now performs the prognostics sub-task in addition to its diagnostics task. The remaining free ranging CEs continue with their diagnostic operation. The base station now performs only its share of the prognostics task besides scheduling and oversight of the prognostics task and collection of diagnostics data from the two CEs. Once the required maintenance has been performed and the prognostics task is over, the base station informs CE1, which then returns to its diagnostics mode. In addition to the diagnostics and prognostics task, all CEs also run individual low-weight battery management algorithm as detailed in section 4.2.

4 POWER MANAGEMENT

The core of our power management methodology is based on monitoring the rate of drop in CE battery capacity using low computation-weight regression techniques. When the battery capacity decreases to certain critical levels, the CEs are dynamically configured to low power modes. In order to achieve low power modes we adjust the system sampling rate of the CEs by periodically forcing the device to go shallow sleep. However, this affects the performance of the system in two ways: a) it reduces the rate at which a CE collects sensor data and runs diagnostic algorithms which we call the *diagnostics frequency*, and b) if sleep modes are introduced during the prognostics routine, it affects the execution speed and, hence, time required for evaluation of prognostic estimates. When the CE battery capacity reaches critical levels, the device is forced to go into shallow sleep for brief periods whose duration is determined by *diagnostics frequency* and maximum delay that can be tolerated for a prognostics update.

Our methodology was developed in a two-phase manner. First, the CE battery discharge profile was studied to understand how the capacity variation can be modeled based on usage. Battery usage included computation, communication and in-built sensor probing loads. This profile was then used to create a regression model for battery discharging. Our power management algorithm uses this model to predict how long the CE battery would last. This prediction scheme along with system thresholds (such as minimum *diagnostics frequency*) are used to determine when and for what duration a CE should be forced into low power mode, thereby extending battery life. Each CE has its individual prediction model based on its discharge profile. This is because for the exact same computation load, different CEs may show slightly different discharge profiles. Such variation in discharge profile may exist for batteries with exact same specifications. Since it is expected that different CEs will handle different diagnostics computation loads, the various thresholds would vary between different CEs. Also, the prognostics computation load is non-periodic in nature (arising only when a flag is raised due to a critical fault), thus leading to different CE performance requirements. Thus, the power management routines are customized for individual CEs; each CE performs its own prediction of its remaining battery life based on its regression model and based on the *diagnostics frequency* and/or maximum prognostics update delay decides when to move into shallow sleep mode.

In the following sections, we first provide details of our battery discharge profile analysis and regression model, followed by the power management methodology.

4.1 State-of-Charge Estimation for Sun SPOT Li-ion Battery and Regression Modeling

After analyzing the battery discharge profile, it was observed that the CEs that only collected sensor data had insignificant computation load and, hence, did not warrant sophisticated power management techniques. However, for CEs that executed frequency analysis using FFT, the computational load was non-trivial but not significantly high. The computational load handled by the CE that executed prognostics along with sensor data collection was highest and, hence, only this CE was chosen to test our power management routines. Implementing such routines for the lightweight diagnostics algorithms would result in unnecessary overloading of the CEs. For more complex diagnostics routines, the same power management routines can be deployed by the CEs. The prognostics routine starts after a few diagnostics routines (based on a prognostic trigger threshold) and continue till the CE shuts down due to low battery.

There are several published techniques in literature to estimate the state-of-charge (SOC) for Li-ion batteries, however, these are not directly applicable in our case since battery health monitoring is not the main goal. It is just an ancillary function that helps us better schedule our main task, which is to monitor the subsystem being supervised by the CEs and perform prognostics if required. Thus, we need a SOC estimation algorithm that is very fast and consumes the bare minimum of computational and memory resources.

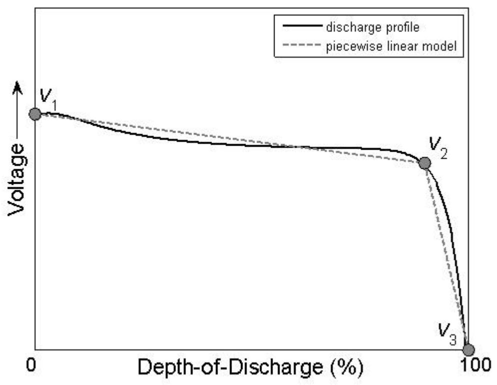

Figure 4: Typical discharge curve of a Sun SPOT Li-ion battery with piecewise linear model fit.

In order to achieve this, we built a simple piecewise linear model of the battery discharge and used least-squares regression on the training data to identify the model parameters. We approximate the discharge curve by two linear sections defined by the points V_1, V_2 and V_3 as shown in figure 4. Since the start and end points V_1 and V_3 are well defined, it only remains to find the position of V_2.

Although figure 4 displays the cell voltage as a function of the depth-of-discharge (DOD) during constant current discharge cycles, this curve can represent the decrease in cell voltage with time. DOD is essentially computed as $100\% - \mathrm{SOC}$. Figure 5 shows the plot of voltage vs. time for a specific Sun SPOT device discharge cycle (running particle filter based prognostics application ((Saha and Goebel, 2008))). The current drawn from the battery has a lot of jitter as is characteristic of logic circuits, but it may be viewed as a uniform random distribution with a stationary mean. The jitter on the cell voltage is smoothed out by moving window averaging (solid blue line). The coordinates of the point V_2 are computed by minimizing the squared error between the smoothed voltage curve and the piecewise linear model (dashed red line). The points V_1 and V_3 are not present in figure 5 since it shows zoomed in view of the plot. Figure 6 shows the plot of the squared error vs. the time index.

Having estimated the coordinates of the point V_2 from the discharge cycles used for training and correlating the time axis to DOD, we found the SOC at V_2,

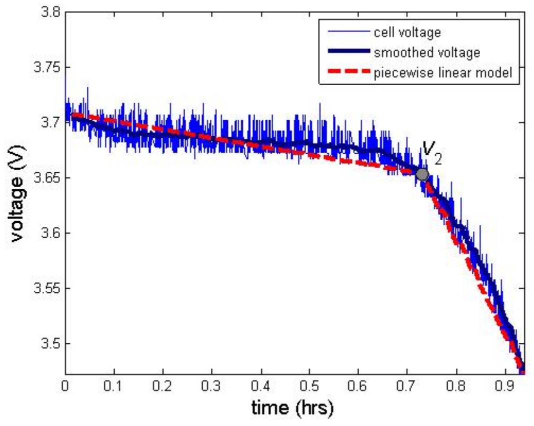

Figure 5: Model identification from a specific discharge cycle.

denoted as V_2, SOC, to be approximately 10% (90% DOD) with a cell voltage, $V_{2,E}$, of about 3.66V. We use these observations to further simplify our prediction task, which is to predict the occurrence of V_2, i.e. the time $t_{critical}$ after which the cell voltage starts to drop sharply. In our experiments, we consider the end-of-discharge (EOD) the same as $t_{critical}$. The prediction happens in two steps. In the first step we estimate the current SOC, SOC_i, from the current cell voltage, E_i, (averaged over a small time window to smooth out the jitters):

$$\begin{aligned} SOC_i = & \\ & V_{2,SOC} + (V_{1,SOC} - V_{2,SOC}) \\ & \times (E_i - V_{2,E})/(V_{1,E} - V_{2,E}). \end{aligned} \quad (1)$$

Substituting the values learnt from the training data, this equation reduces to:

$$SOC_i = 10 + 183.67(E_i - 3.66). \quad (2)$$

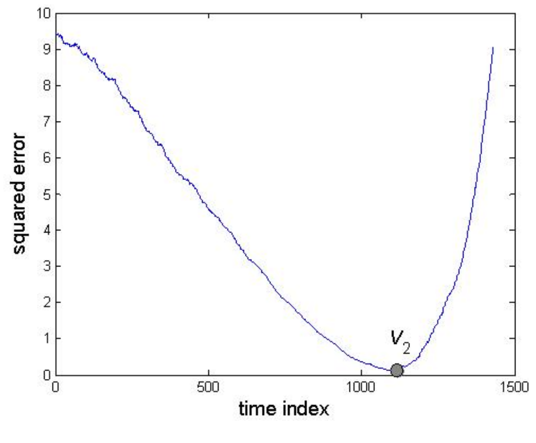

Figure 6: Least square regression for a specific discharge cycle.

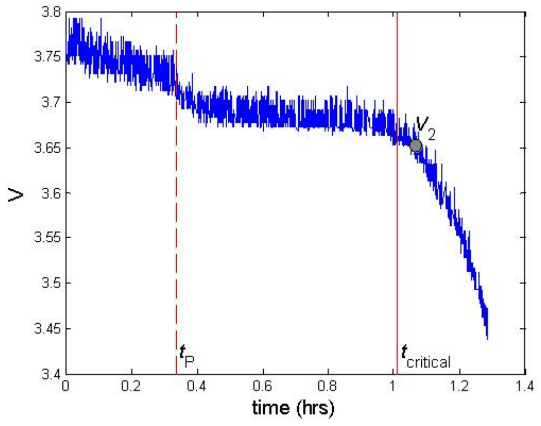

Figure 7: Predicting $t_{critical}$ for a test case.

Having determined the current SOC, we simply integrate the expected current profile over time to get the rate of charge depletion and extrapolate it from the current time t_i till we reach 10% SOC. Assuming stationary distribution of the expected current with mean I_{avg}, this step reduces to:

$$t_{critical} = t_i + (C/I_{avg}) \times (SOC_i - 10)/100, \quad (3)$$

Here, C represents the charge capacity of the battery (0.72 Ahr in our case). Thus the actual prediction exercise is reduce to two simple calculations of complexity order O(1). Although, this method is simplistic, the results are definitely usable as shown in figure 7. The prediction is made at time instant t_P, when the CE starts to run the prognostics routine. It is desired to predict $t_{critical}$ or EOD which gives us the time window $(t_{critical} - t_P)$ within which the health management task may be run reliably.

Figure 8 shows one such example where the initial prediction at $t_{P,1}$ of $t_{critical}$, denoted by $t_{critical,1}$ in the plot, is determined to be too soon for the completion of the prognostic activity being undertaken. Subsequently we introduce sleep states to prolong the battery and make another prediction at $t_{P,2}$. The new value of $t_{critical}$ is denoted by $t_{critical,2}$ in the plot. In this case $t_{critical,2}$ is far enough out in the future to complete the health management routine. If that is not the case, we can prolong the sleep states and make another prediction and so on. It should be noted that $t_{critical,2}$ agrees well with the computed location of V_2, indicating the robustness of our simplistic approach.

4.2 Regression Based Battery Power Management

As mentioned earlier, each individual CE executes its own power management routine. If a CE participates in prognostics, the power management routine is initiated whenever the prognostics routine is started since it is expected that the prognostics routine would cause significant rise in current consumption by the device and hence would reduce the battery life earlier.

As mentioned in the previous section, two regression models are developed. In the first case, a static

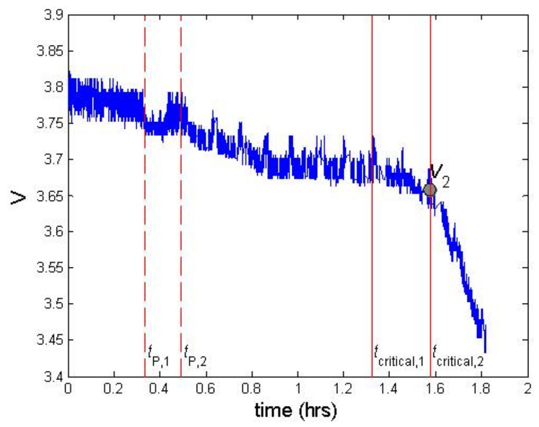

Figure 8: Predicting $t_{critical}$ with sleep modes introduced.

model (*model A*) is created for the discharge profile which provides only one EOD value. This model considered the battery discharge profile involving prognostics load. Though this model leads to significant saving in battery life, it does not model the effects of introducing the sleep mode which would change the current consumption. In the second case, the model is dynamic (*model B*) and adapts to accommodate changes in the discharge profile due to introduction of shallow sleep modes. There are essentially two forms of low power mode that we implement. In the first mode, the CE goes to shallow sleep for time t_{sleep} during every iteration. For the second mode, the CE goes to shallow sleep after multiple iterations (*n*) of prognostics and diagnostics. For the dynamic regression model based scheme (*model B*), both t_{sleep} and *n* vary during run-time. Based on the two regression models and the two low power modes, three power management routines are formulated.

Before describing the routines we define the following parameters that are specified by the system user

- f_{diag}: This is the minimum *diagnostics frequency* for the health management system.
- δ_{prog}: This is the maximum delay that can be tolerated in obtaining a prognostics update for a single iteration. The inverse of δ_{prog} is defined as the minimum *prognostics frequency* f_{prog}.

Note the values for the above parameters are application-specific and should decided based on user requirements. We also assume that an average value for the computation time for the diagnostics routines t_{diag} and each prognostics iteration t_{prog} is available. These computation times also include average communication times. In case of problems in communication, due to sudden interference, significant deviations in the actual performance can occur. Also, in our system, only a single prognostic activity is included, for more complex systems, there may be multiple other prognostic activities which may interrupt the operations thereby causing changes in f_{diag} and f_{prog} values. Thus, f_{diag} and f_{prog} are mainly used for guiding calculations for t_{sleep} as accurately as possible. The three power management routines are presented below.

Simple power management

In this method, a CE is forced to go to shallow sleep at every iteration for a brief period of time Note that in this case f_{diag} and f_{prog} are same. Thus in this case f_{diag} and f_{prog} are same. The power management routine consists of the following steps:

- Using *model A*, the EOD is calculated.
- Based on t_{diag}, t_{prog} and f_{diag}, the sleep time t_{sleep} is calculated.
- The total number of diagnostics and prognostics updates *N* that can be obtained before the battery reaches EOD is calculated.
- The total number of diagnostics and prognostics iterations *N* that can be achieved for this scheme is calculated as follows:

$$N \times (t_{diag} + t_{prog} + t_{sleep}) = EOD - t_P. \quad (4)$$

- Shallow sleep is introduced during every prognostics iteration for time t_{sleep}.

Our goal in implementing power saving routines is to maximize *N*, i.e., ensure more health management iterations. *N* is directly proportional to remaining battery life and execution speed. Higher values of t_{sleep} result in more battery life, at the cost of low execution speeds, hence the trade-offs between these two factors need to be investigated. The next power management schemes, provide more parameters that can be varied to investigate such trade-offs more comprehensively.

Parameterized power management

As observed in the *simple* power management scheme, the only parameter that can be adjusted to meet system constraints is t_{sleep}. For many complex systems, this may not be sufficient to meet all system performance requirements. For example, in many cases f_{diag} may need to be much higher compared to f_{prog} and, thus, the device may have to be forced to avoid sleep modes during diagnostics. The following scheme caters to such cases by providing more parameters for performance adjustments. In this method, the device goes to shallow sleep for a few iterations, *m*, after *n* regular iterations while diagnostic operations – involving only probing sensors – is not stopped during sleep. Regular iterations involve both diagnostics and prognostics routines.

- Using *model A*, the EOD is calculated.
- Based on $t_{diag}, t_{prog}, f_{diag}$ and δ_{prog}, the sleep time t_{sleep} and *m* and *n* are calculated.
- The total number of diagnostics and prognostics iterations *N* can be achieved for this scheme as follows:

$$N \times (n \times (t_{diag} + t_{prog}) + m \times (t_{diag} + t_{sleep})) = EOD - t_P. \quad (5)$$

Table 1: Test matrix for parameter variation for different power management routines.

Algorithm		$f_{diag}(Hz)$	$\delta_{prog}(secs)$	$N_{diag}(Hz)$	$N_{prog}(Hz)$
Simple		0.4	2.5	1127	1127
Parameterized	a	0.4	50	1800	900
	b	0.4	40	1700	1500
Dynamic	a	0.4	50	1925	1650
	b	0.4	120	2160	300
	c	0.4	40	1620	900

Since in this case the total number of diagnostics is update is different from the total number of prognostics updates, we define them separately as:

$$N_{diag} = N \times n \quad (6)$$

$$N_{prog} = N \times m \quad (7)$$

- The CE goes to shallow sleep mode after *n* prognostics iteration for time t_{sleep} for *m* iterations. During the sleep iterations, the diagnostics updates are not stopped.

In this case we have three parameters – t_{sleep}, *n* and *m* – that can be adjusted in order to explore trade-offs between execution time and battery life. Note that N_{diag} and N_{prog} are equal to each other for the *simple* power management scheme.

Dynamic power management

In this scheme, we extend the *parameterized* power management scheme by including dynamic updates to the value of EOD so that the change in current profile due to sleep modes is taken into consideration. In this method, during the sleep iterations the diagnostics updates are not stopped, only prognostics operations that drain more battery capacity are stopped. The details of the scheme is given below.

- Using the *model B*, the initial EOD is calculated.
- Using this initial value of EOD combined with $t_{diag}, t_{prog}, f_{diag}$ and δ_{prog}, the sleep time t_{sleep}, *m* and *n* are calculated.
- The CE goes to shallow sleep mode after *n* prognostics iteration for time t_{sleep} for *m* iterations. During this sleep iterations, the diagnostics updates are not stopped.
- As the shallow sleep modes are introduced, the battery discharge profile gets updated and, hence, new values of EOD (EOD_i) are obtained. For every new value of EOD_i, t_{sleep}, *m* and *n* are updated.
- For a new value of EOD_i, *N* is calculated as follows:

$$N \times (n \times (t_{diag} + t_{prog}) + m \times (t_{diag} + t_{sleep})) = EOD_i - t_P. \quad (8)$$

N_{diag} and N_{prog} are as defined in *parameterized* power management scheme.

Note that it is not useful to increase the t_{sleep} value frequently based on new calculations of EOD. This is because a high value of t_{sleep} will cause an increase in the value of δ_{prog}. This is not desirable as we progress further into the prognostics routine when more frequent updates – and hence less delays – would be required for critical prognostics decisions. In our experiments, we update the EOD value twice for the duration of the prognostics routine.

5 EXPERIMENTS AND RESULTS

In this section we present our experiments with the three power management techniques presented in section 4.2 for the health management architecture outlined in section 3.2 comprising of 4 CEs and 1 base station. As observed from the descriptions of the routines, the design space that can be explored by varying the different parameters and their combinations are huge and cannot be fully explored within the scope of this paper. Therefore, we examined a limited set of parameter variations. The values of both the f_{diag} and f_{prog} obtained for operation without power management are 0.5 Hz. Note that these values are used mainly for calculating the t_{sleep} values, and hence, are assumed to not have any variations. However, during practical implementation these frequencies do not remain constant due to disturbances in the communication channel. The computation and memory overhead for the power management algorithms was negligible and did not add any overhead to the existing framework. All the CEs were charged to a voltage of 3.8V before starting the health management routines.

Table 1 presents our experimental test matrix along with the resultant values of N_{diag} and N_{prog} from the experiments. The parameter variations for the different power management schemes are outlined by the different f_{diag} and δ_{prog} values. Figure 9 shows the battery lifetime for the different power management schemes as outlined in table 1 compared to lifetime with no power management scheme.

Switching to low power mode clearly shows savings in battery life. However, the savings come at the expense of lower *diagnostic* and *prognostic* frequencies. Also, when the total battery lifetime values are compared with the values obtained for N_{diag} and N_{prog} in table 1, it may be observed that the *simple* power management routine yields very low total number of iterations. This is because the device goes to sleep at every iteration that hampers the execution profile. However, for the *parameterized* scheme, the N_{diag} and N_{prog} values are much higher for comparable total battery life. This implies that it is more efficient to make the device go to sleep for brief bursts of

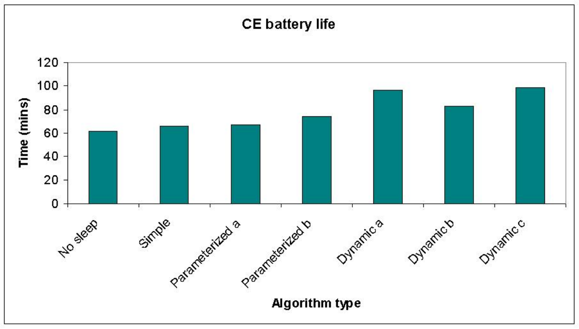

Figure 9: Battery life with different power management schemes compared with no power management scheme.

time after longer intervals of execution as against letting the device sleep at more frequent intervals. The reason behind this could be that bursts of sleep periods allow the battery to significantly recover its capacity thereby extending its battery life with lower penalties on performance. However, in order to determine when to start these burst periods as well its duration, accurate prediction of EOD is required. Our *dynamic* power management aims to achieve that by periodically updating the value of EOD as the current discharge profile changes. For the *dynamic* power management scheme, it was observed that increasing t_{sleep} was more efficient than increasing the value of *m* in terms of achieving higher N_{diag} and N_{diag} for comparable battery lifetimes.

Figure 10 shows the expected voltage and current discharge profile for a CE with diagnostics and prognostics routine, while figure 11 shows the voltage and current profiles for the *dynamic* power management schemes. The current discharge profile clearly shows the dip in current values which occur during sleep iterations. An increase in voltage level corresponding to a dip in current value during sleep iterations is also visible. Also, compared to the battery life of the CE with no power management, a significant increase in battery life is observed in the device with power management.

6 CONCLUSIONS AND FUTURE WORK

This paper explored regression based power management schemes for wireless sensor architecture implementing a distributed health management system. The system involved multiple smart sensors implementing diagnostics operation along with off line battery prognostics routines. Our proposed power management schemes are centered around monitoring of battery discharge profile and using that information for accurate regression based prediction of device battery EOD. This accurate prediction enables efficient low power mode transitions, since the sensors transition into sleep modes only when required. The EOD prediction based method also enables efficient calculation of the time periods for which the device can sleep, thereby ensuring minimal loss in system performance. Three low computation-weight power management schemes were explored that involved on line prediction of EOD as well as dynamically updating EOD based on change

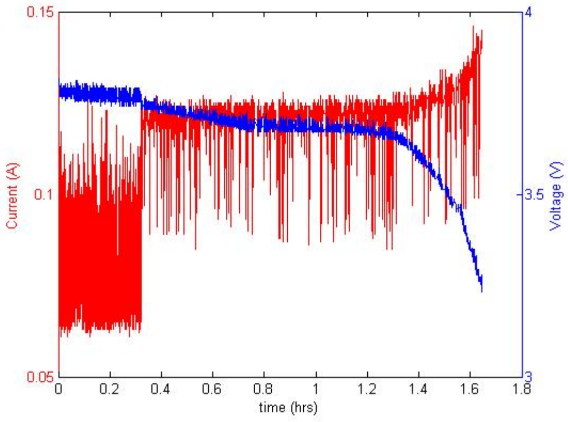

Figure 10: Current and discharge profile for CE with no power management schemes.

in battery discharge profile. Our power management schemes provide multiple parameters that can be adjusted for different system constraints, and the user should make a careful trade-off analysis of system performance requirements such as speed cost etc., in order to set threshold values for the different parameters.

As observed from the experimental results, increased sleep durations by the device may help in extending battery life of the device but hampers system performance. The results also show that bursts of sleep iterations after several regular health management task iterations provided better trade-off between performance and battery life.

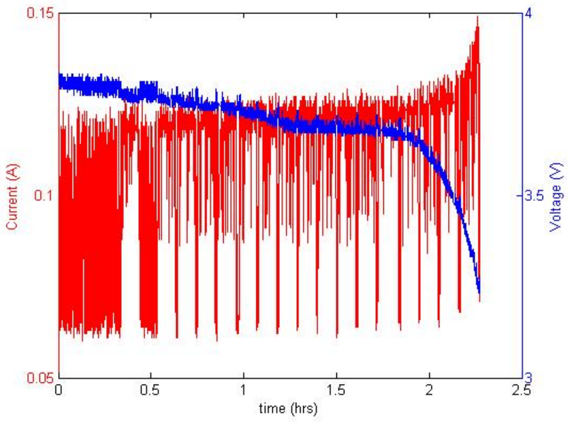

Figure 11: Current and discharge profile for CE with *dynamic* power management schemes with varying f_{diag}. The dip in current values correspond to sleep states.

Future work will look into more complex and interactive systems involving multiple prognostic routines to test the robustness of the schemes. Other battery life prediction techniques for the SPOT devices such as particle filter based schemes will also be explored.

NOMENCLATURE

SOC	State of Charge
EOD	End of Discharge
DOD	Depth of Discharge
V_1	Start point of discharge curve
V_2	Knee point of discharge curve
V_3	End point of discharge curve
$V_{i,E}$	Voltage at point V_i
$V_{i,SOC}$	SOC at V$_i$
I_{avg}	Mean current
t_i	Current Time
C	Charge capacity
SOC_i	Current at SOC
$t_{critical}$	Time at which SOC is 10% of full value
t_P	prediction start time
t_{diag}	Computation time for diagnostics
t_{prog}	Computation time for prognostics
f_{diag}	Minimum diagnostics frequency
f_{prog}	Minimum prognostics frequency
δ_{prog}	Maximum prognostics delay
N	Total number of diagnostics and prognostics iterations
N_{diag}	Total number of iagnostics iterations
N_{prog}	Total number of prognostics iterations
t_{sleep}	Amount time spent in shallow sleep (in ms)
m	number of sleep iterations for parameterized and dynamic power management schemes
n	number of iterations before sleep iterations start

REFERENCES

(Anand *et al.*, 2003) M. Anand, E. B. Nightingale, and J. Flinn. Self-tuning wireless network power management. In *9th Annual International Conference on Mobile Computing and Networking*, 2003.

(Chiasserini and Rao, 2000) C. F. Chiasserini and R. R. Rao. A distributed power management policy for wireless ad hoc networks. In *IEEE Wireless Communications and Networking Conference*, 2000.

(Jayashree *et al.*, 2004) S. Jayashree, B. S. Manoj, and C. SivaRamMurthy. On using battery state for medium access control in ad hoc wireless networks. In *10th Annual International Conference on Mobile Computing and Networking*, 2004.

(M. Nakai and Shimura, 2005) K. Seno T. Meguro T. Seki T. Kondo A. Hashiguchi H. Kawahara K. Kumano M. Nakai, S. Akui and M. Shimura. Dynamic voltage and frequency management for a low-power embedded microprocessor. 40:28–35, 2005.

(Pillai and Shin, 2001) P. Pillai and K. G. Shin. Real-time dynamic voltage scaling for low-power embedded operating system. In *Proceedings of the eighteenth ACM symposium on Operating systems principles*, 2001.

(R. Tynan and Hare, 2005) D. Kane R. Tynan, D. Marsh and G. M. P. Hare. Agents forwireless sensor network power management. In *International Conference on Parallel Processing Workshops*, 2005.

(Saha and Goebel, 2008) B. Saha and K. F. Goebel. Uncertainty management for diagnostics and prognostics of batteries using bayesian techniques. In *Proceedings of IEEE Aerospace Conference 2008*, 2008.

(Saha *et al.*, 2008) S. Saha, B. Saha, and K. F. Goebel. Distributed prognostics using wireless embedded devices, 2008.

(Saha *et al.*, 2009) B. Saha, S. Saha, and K. F. Goebel. A distributed prognostic health management architecture. In *Proceedings of MFPT 2009*, 2009.

(Zamora *et al.*, 2007) N. H. Zamora, J.C. Kao, and R. Marculescu. Distributed power-management techniques for wireless network video systems. In *International Conference on Design, Automation and Test in Europe*, 2007.

Sankalita Saha is a research scientist with Mission Critical Technologies at the Prognostics Center of Excellence, NASA Ames Research Center. She received the M.S. and PhD. degrees in Electrical Engineering from University of Maryland, College Park in 2007. Prior to that she obtained her B.Tech (Bachelor of Technology) degree in Electronics and Electrical Communications Engineering from the Indian Institute of Technology, Kharagpur in 2002.

Bhaskar Saha received a Ph.D. degree from the School of Electrical and Computer Engineering at Georgia Institute of Technology, Atlanta, GA, USA in 2008. He received the M.S. degree also from the same school and his B. Tech. (Bachelor of Technology) degree from the Department of Electrical Engineering, Indian Institute of Technology, Kharagpur, India. He is currently a Research Scientist with Mission Critical Technologies at the Prognostics Center of Excellence, NASA Ames Research Center. His research is focused on applying various classification, regression and state estimation techniques for predicting remaining useful life of systems and their components, as well as developing hardware-in-the-loop testbeds and prognostic metrics to evaluate their performance. He has been an IEEE member since 2008 and has published more than 10 papers on these topics.

Kai F. Goebel received the degree of Diplom-Ingenieur from the Technische Universitt Munchen, Germany in 1990. He received the M.S. and Ph.D. from the University of California at Berkeley in 1993 and 1996, respectively. Dr. Goebel is a senior scientist at NASA Ames Research Center where he leads the Diagnostics Prognostics groups in the Intelligent Systems division. In addition, he directs the Prognostics Center of Excellence and he is the Associate Principal Investigator for Prognostics of NASA's Integrated Vehicle Health Management Program. He worked at General Electric's Corporate Research Center in Niskayuna, NY from 1997 to 2006 as a senior research scientist. He has carried out applied research in the areas of artificial intelligence, soft computing, and information fusion. His research interest lies in advancing these techniques for real time monitoring, diagnostics, and prognostics. He holds eleven patents and has published more than 100 papers in the area of systems health management.