Chapter 11

Monitoring Technologies in Mission-Critical Environment by Using Wireless Sensor Networks

Yuanyuan Zeng, Kai Xiang and Deshi Li

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/48185

1. Introduction

Monitoring in mission-critical environment with the deployment of wireless sensor networks (WSNs) is one of the most widely-used application areas. Mission-critical environment implies monitoring tasks happened in locations that are difficult to get access to or easy to lead destroy of the network deployment: e.g., scenarios in hazard and emergency monitoring situations such as chemical pollution, fire and explosion, environment-oriented pollutions and land security (Poovendran, 2010). In the near future it can be expected that surveillance areas will be equipped with a range of smart sensors to provide parts of overall event detection and service. Sensors are expected to be programmed to report periodically and also when they detect a sensor input that exceeds a threshold. WSNs are well suited to monitor physical signals, allowing detection, localization, tracking and reacting toward the mission-critical environment.

We focus on monitoring related technologies for WSNs in mission-critical environment: 1) the connectivity issues in mission-critical monitoring and the solutions. 2) The data collection in mission-critical monitoring are addressed. The data collection requires technologies that can guarantee performance such as timeliness, reliability, scalability and energy efficiency. It can be provided by designed routing, link scheduling and even cross-layer mechanisms. 3) In mission-critical environment, dynamic network topology leads more difficulties in event detection. The related recent detection models and frameworks are addressed. 4) A new promising way for mission-critical monitoring is to utilize wireless sensors and actuators together. The monitoring related technologies about wireless sensor and actuator networks are then talked about.

2. Connectivity issues

Wireless sensor networks based monitoring and surveillance systems are physical and engineered systems that bridge the cyber-world of computing and communications with the physical world whose operations are monitored, coordinated and controlled. The research on connectivity of wireless sensor networks usually aims to utilize dense deployment that helps to improve reliability and extends longevity of the network lifetime. For general WSNs, energy efficiency is one of the major constraints in wireless sensor networks. Sensors are very limited in their processing, computing and communication capabilities as well as the storage and power supply. Moreover, due to the sheer number of sensor nodes and the potentially mission-critical environment, it easily leads to node failures and network partitions after the deployment of sensor nodes. So, how to maintain connectivity with recovery under the mission-critical environment is very important and challenging issue.

2.1. Challenges for connectivity in mission-critical environment

In mission-critical environment, the systems are usually built upon wireless sensor networks that need to provide timely and valid information about the field. As emergency events spread throughout the surveillance area, it is possible that the sensing devices will be easily disconnected from the network or indeed be destroyed. The network topology changes rapidly in the surveillance scenarios as emergency events spread. And then may lead to failure of the network system eventually. In this context, connectivity maintenance is very critical in order to ensure timely, accurate, and scalable monitoring.

The "Routing Hole" problem is a very important and well-studied problem in network connectivity, where messages get trapped in a "local minimum" that can be incurred by network partitions. Actually, the incidence of routing holes increases as network density diminishes and the success rate of the greedy algorithm drops very quickly with network density.

2.2. Related work on routing hole

Some existing "face routing" algorithms are developed to bypass routing holes using geo-routing algorithms. GPSR (Karp et al., 2000) recovers "hole" by using the "right-hand rule" to route data packets along the boundary of the hole, which combining greedy forwarding and perimeter routing on a planar graph that represents the same connectivity as the original communication network. Some incremental improvements are proposed based on it (Yu et al., 2000; Powell & Nikoletseas, 2007). GEAR (Yu et al., 2000) picks a next hop node that minimizes some cost value such as distance or energy and gradually forms a better path to the destination by sending multiple data packets. At this point a temporary rescue mode is used to escape the local minimum. But a major pitfall of face routing algorithm is that no practical planarization algorithm is known. An interesting approach BOUNDHOLE algorithm (Fang et al., 2006) is proposed to uses the TENT rule to discover local minimum nodes and then "bounds" the contour or the routing holes. But this algorithm has a high overhead.

In the mission-critical application situations, holes feature prominently and can be expected to grow in size rapidly as monitoring events spread, thus demanding solutions that are robust and low complexity for quick reactions.

2.3. Dealing with routing hole in mission-critical monitoring

Given a wireless sensor network deployed in a surveillance area under mission-critical environment. Each sensor can adjust its maximal transmission ranges to one of the *k* levels: R_0, R_1, …., $R_{k-1}=R_{max}$ by using different transmission power level from P_0, P_1, till $P_{k-1}=P_{max}$. Actually, most motes support to work under multiple power levels nowadays. Initially, all sensors work in P_0 for minimal energy under this power level. A sensor node is programmed to trigger and transmit data after it detects event locally. During data transmissions, if a sensor node cannot find a next hop that satisfies the routing metrics, a routing hole occurs. Here, an approach by increasing transmission power is utilized to jump over the "hole". It tries to find another node as the next hop by increasing the transmission power gradually until the maximal power.

Each sensor with *k* levels of power setting: $\{P_0, P_1, P_2,\ldots P_{k-1}\}$ can be in *k* levels of maximal transmission range as: $\{R_0,R_1,\ldots R_{k-1}\}$. A function is defined to find appropriate transmission power by increasing the power as follows:

$$P = P_{cur+\iota+1}, \iota = 1,\ 2,\ 3\ldots k-1 \tag{1}$$

where, *cur* is the current number of transmission range level among *k* levels, ι is the count of unsuccessful delivery. When a sensor is switched to adaptation approach, it increases transmission power gradually in levels of setting, if cannot deliver the packet successfully. A node is eligible for power increase according to formula (1) until one of the following conditions is satisfied:

1. It finds a node in an eligible node as the next hop according to the routing metrics.
2. When $P=P_{max}$; In this case, it will try to find the relay according to routing metrics. Otherwise, no eligible relay is found.

The node that works in a larger transmission range could still be adapted to lower transmission power for energy efficiency, when it satisfies: the node is in a good connectivity with its neighbourhood that is larger then a predefined threshold. A function to find appropriate transmission range by decreasing transmission power is defined as:

$$P = P_{cur-\iota'}, \iota = 1,\ 2,\ 3\ldots k-1 \tag{2}$$

where, *cur* is the current number of transmission power level among *k* levels. ι' is the count of decrement. A node is eligible for power decrease until:

1. The minimum power has been reached.
2. There are two consecutive power levels such that at the lower level does not met the required routing metrics, but at the higher power level does.

3. There are two consecutive power levels such that at the lower level the required neighbourhood connectivity threshold does not met, but at the higher power level the required does.

An example of the solution toward sensor data transmissions when in routing hole is illustrated as shown in Figure 1.

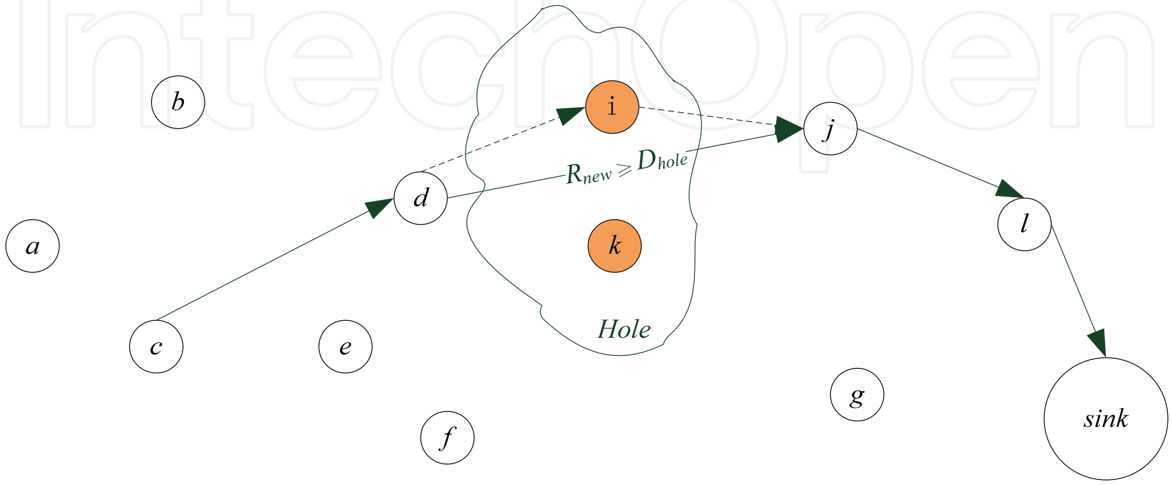


Figure 1. An illustration of the "Routing Hole Problem"

The node *d* can not find a suitable next hop and packets are stuck on node *i* and node *k*. So, node *d* is in local minimum. And we call that a "routing hole" occurs in the network. When we report "routing hole" on the hop of node *d*, the solution is to increase the transmission power and try to find the next hop with the higher power. As shown in Figure 1, node *d* increases its transmission power and find the next hop node. Actually, if the transmission range $R(R \leq R_{max})$ with the increased power $P(P \leq P_{max})$ exceeds the range of the "hole" (i.e., the adapted transmission range is larger than the maximal diameter of the hole), a new relay node *j* will be found located outside around the "hole". Then node *d* delivers the packets toward the new next hop *j* with higher power and jump over the "hole". In this way, the routing hole problem is solved by recover local connectivity by increased power.

3. Data collection technologies

The mission-critical monitoring systems are built upon wireless sensors that need to provide timely and accurate physical information. The rescue group relies on timely data update to be aware of on-time surveillance situations and reacts toward the emergency. In addition, as emergency events spread throughout the surveillance area, it is possible that the sensing devices will be easily disconnected from the network or indeed be destroyed. In this context, data collection requires a self-adaptive way in order to ensure timely, accurate, and scalable monitoring and surveillance. For a credible deployment of WSNs in mission-critical applications, the main properties of data collection that needs to be fulfilled are timeliness, self-adaptiveness, energy efficiency and scalability.

3.1. Challenges for data collection in mission-critical environment

The mission-critical surveillance applications are more challenging because of the time-varied connectivity and event scenarios. The network topology changes rapidly in the surveillance scenarios because of critical emergency events. The data packets have to be delivered effectively within the required time. The routes have to be adapted and reconfigured in order to continue operation for the network. The network can be in serious partitioned situations and not easy to be repaired, as emergency events destroy intensely in the surveillance area.

Power control schemes utilized in routing can help to increase the timeliness before the packet deadline especially for mission-critical situations. Increasing transmission power can effectively improve link quality and therefore reduce the number of transmissions to deliver a packet. The power increase reduces the throughput due to higher interference. Hence, supporting routing in mission-critical surveillance applications by power control alone is difficult.

3.2. Related work on application-specific communications

There are a lot of data collection related protocols and algorithms designed for ad hoc and sensor networks. Most of these schemes focus on energy efficiency and link node lifetime because of constrained node resources (Rogers et al., 2010).

Some sensor network system applications require real-time communication, typically for timely surveillance or tracking. The real-time routing design for general wireless sensor networks usually makes balance for real-time data delivery and energy consumption (He et al., 2003; Felemban et al., 2005; Chipara et al., 2006; Ahmed et al., 2008; Kim et al., 2009).

Beyond the above, the data collection design needs to consider application characteristics and requirements of the network system. Sivrikaya et al. proposed a stochastic routing mechanism for public safety applications such as emergency evacuations or rescue applications in wireless sensor networks based on Markov chains (Sivrikaya et al., 2009). Yang et al proposed a shortest path routing for mobile ad hoc and sensor networks by considering topology dynamics, i.e., the network topology changes due to energy conservation and node mobility (Yang et al., 2010). Li et al. proposed a solution to delay-bounded and emergency-efficient emergency event monitoring problem by an event detection model and a warning delivery model (Li et al., 2010). Li and Fen proposed a dynamic adaptive cooperative routing for emergency data in wireless sensor networks (Li & Fen, 2009). Byun et al developed a self-adaptive intelligent system for building energy saving and context-aware smart services (Byun & Park, 2011), where they proposed an energy-efficiency self-clustering sensor network and a node type indicator based routing protocol that considers the application requirement. Tseng et al. proposed a distributed 2D navigation algorithm to direct evacuees to an exit while helping them avoid hazardous areas (Tseng et. al., 2006). We have proposed a delay-sensitive routing scheme in wireless sensor networks for building fire monitoring (Zeng et al., 2011), which is adaptive to fire emergency scenarios such as node failure and dynamic network.

Beyond routing, being an essential part of data collection problems, media access control (MAC) protocols that handle with interference problems have received intense research attention. The proposed MAC layer schemes mainly include collision-based schemes such as CSMA/CA based schemes, and scheduling based schemes, as well as hybrid schemes. The collision-based schemes make collision-avoidance scheduling with node information of interference range usually two communication hops away. The scheduling based schemes are mostly based on TDMA slot allocation schemes. Sobral et al. proposed hybrid contention/TDMA-based (HCT) MAC, which is specially designed to work with ad-hoc wireless networks organized in clusters, providing timely bounded communications both inside and outside the clusters by resource reservation(Sobral et al.,2008).

There are some literatures with cross-layer design for data collection. The methods with joint optimal scheduling, routing and power control are proposed to achieve goals such as: fair rate allocation, minimize overhead and maximal resource utilization, etc. Recently, Lu et al. proposed to minimize average communication latency for the active flows with emergency efficiency by joint scheduling and routing in wireless sensor networks (Lu & Krishnamachari, 2007). Joseph et al. considered the problem of obtaining jointly optimal power control, routing and scheduling policies to ensure a fair utilization of network resources for energy harvesting sensor networks (Joseph et al., 2009).

3.3. Self-adaptive data collection framework

3.3.1. Intelligent data collection control model

An intelligent data collection control framework is used to circumvent the problem according to varied network scenarios. As illustrated in Fig. 2, the data collection control with adapted strategies is based on dynamic surveillance scenarios. The controller (sink) works as the gateway to control the data collection of the local sensor networks, which is used to collect physical data and interact with the Internet. During this, the controller gets the global knowledge of the emergency events occur. The controller updates and adapts the strategies of power, routing and scheduling in the network to achieve intelligent data collection.

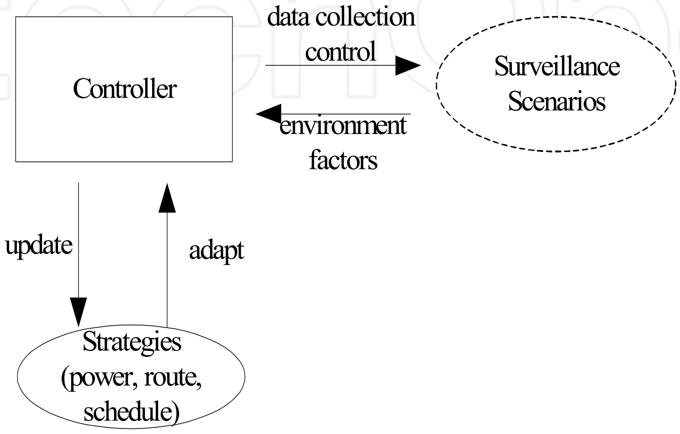


Figure 2. The intelligent data collection control framework

3.3.2. *Mission-critical environment model*

Considering characteristics of mission-critical surveillance, we use a finite-state discrete state set to model it. Usually, in most mission-critical surveillance, the scenarios of physical objectives could be in different finite states as: start, expand, diminish, etc. The set of finite discrete hazard states can be represented as S={0,1,….K-1}. By partitioning the range of the received data delivery ratio of the controller into finite number of intervals, then physical objective model is constructed with finite hazard states. The number of set S is related with the partitions. The partitioning is executed by thresholds of the data delivery ratio. Let $\Gamma_0=0<\Gamma_1<\Gamma_2 \ldots<\Gamma_K=1$ be the thresholds of the data delivery ratio. The hazard scenario is in state k if the network data delivery ratio is between Γ_k and Γ_k+1.

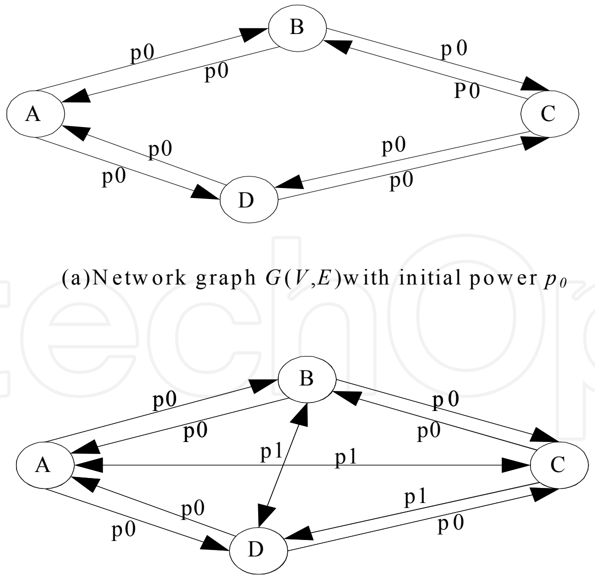
3.3.3. *Elements of data control*

The network system is formulated by a tuple <S, A, R, $\check{T}$>, where S is the discrete hazard state space. A is the discrete action space that is dependent on strategies taken. R: $S\times A \rightarrow$R is the cost function, which implies the quality of a state-action combination of the WSN-based network control system. $\check{T}$: $S\times A \rightarrow \Delta S$ is the state transition function, where ΔS is the probability distribution over state space S. The controller receives a performance-oriented signal from WSN in state $s_k \in S$, and selects a strategy $a_k \in A$ based on it.

For data collection control in mission-critical monitoring, the strategies include sets of selected power, route and schedule, i.e., a set of tuple pair (p_i, $node_i$, $slot_i$), where $p_i \in$ [p_{min}, p_{max}], $node_i \in$ [$node_1$, $node_N$] and $slot_i \in$ [$slot_0$, $slot_{L-1}$]. Once the strategy is taken, the network system produces new performance signal according to the action. Then the controller receives the update cost $R \in$R, which is used to evaluate the effectiveness of the control strategy. It is obvious that a higher power increases network connectivity and helps to find real-time delivery routes. In turn, it also increases the interference with larger interference ranges.

3.3.4. *Search for strategy set*

The search of strategy set for a WSN can use a multi-edge graph $G'(V,E';P)$ based on original network graph $G(V,E)$, which is shown in Fig. 3. For a connected network graph $G(V, E)$, each node can transmit data packets on each level of available transmission power. The each edge between u and v with $p_u \in$ [p_{min}, p_{max}] is drawn to form a multi-edge. It is well known that the derivation of all feasible scheduling in a single-channel wireless network is an NP-hard problem (Jain et al., 2003). Our problem is more difficult because of delay and interference control. Intuitionally, the number of possible concurrent transmission tuple set grows exponentially according to the increased number of allocations on directional links, power and slots. The search process is executed on the multi-graph G' and to find a "good" subset of transmission tuple series that satisfy delay and interference.



(a)Network graph $G(V,E)$with initial power p_0

(b)Network graph G' $(V,E;P)$ with two power levels for each node

Figure 3. The multi-edge graph *G′*

3.3.5. Q-learning based data collection control

The objective of the *Q*-learning based control algorithm is to learn the optimal transmission strategy (on power, route and schedule) that minimizes the cost function (i.e., gets maximal expected rewards). Let $Q_t(s_k,a_k)$ be the strategy quality. Essentially, the *Q*-value means the expected reward when taking strategy *a* under state *s*. The *Q*(*s*,*a*) value are estimations of the optimal *Q**(*s*, *a*), which implies the total discounted cost of taking strategy a in state s and then following the optimal control from then on. Accordingly, the optimal control strategy can be derived by selecting the set of power-relay-slot vector pair where the *Q**-value is minimized.

The learning procedure is to update the *Q*-value. When a packet transmission beginning at time *t* is finished at time *t*+1, the *Q*-value for state-strategy pair is updated by:

$$Q_{t+1}(s_t,a_t) = (1-\alpha_t)Q_t(s_t,a_t) + \alpha_t(R_t + \beta \min_{\alpha_{t+1}} Q_t(s_{t+1},a_{t+1})) \tag{3}$$

In Formula (3), α_t is the learning rate and in the range of (0, 1). β is the discount factor and is in the range of (0, 1) too. A constant learning factor is used so that the learning procedure can track the mission situations.

There is one data collection tree formed toward the sink during the transmission. Considering the multi-sink situation in the network, multi-agent *Q*-learning is utilized to find out the optimized strategy to improve system performance for all sink-oriented data collection trees. The motivation can be expressed as shown in Formula (4). Formula (5) shows the way to calculate the expectation of strategy cost. *a* is the selected strategy at state *s*. $\pi_i(a)$ is the probability for data collection tree toward sink *i* to select a specific strategy *a*, which achieves the equilibrium. Π_i is the set of strategies available. In this case, $(a_1 \ldots a_n)$ is the joint action of trees under the equilibrium. $R_i(s, a)$ is tree *i*'s total cost for the state given that agents follow the equilibrium strategies.

$$\min_{\pi^i \in \Pi^i} E[R^i(s,a)] \qquad for\,any\,tree\,i \tag{4}$$

$$E[R^i(s,a)] = \sum_i \pi^i(s,a) R^i(s,a) \tag{5}$$

The multi-sink data collection control process forms a stochastic non-cooperative game, i.e., multi-agent *Q*-learning process. Formula (4) shows the objective to achieve minimal cost for each data collection tree in the network. To achieve this, each tree finds a Nash equilibrium strategy when responses to the other trees for current state (Hu & Wellman, 2003). Let *NashQi* be the current agent's cost in current state for the selected equilibrium. According to Hu & Wellman (Hu & Wellman, 2003), we get Formula (6). *m* is the number for players in the game. In order to calculate the Nash equilibrium, each agent *i* needs to know the other agents' *Q*-values. Then, each agent observes the other agents' immediate rewards and actions. So, agent *i* can update its *Q*-value according to other agents' *Q*-values, as shown in Formula (7).

$$\begin{aligned} Q^i_{t+1}(s_t, a^1_t, \ldots., a^m_t) &= (1-\alpha_t) Q^i_t(s_t, a^1_t, \ldots., a^m_t) \\ &+ \alpha_t (R^t_i + \beta NashQ^i_t(s_{t+1}, a^1_{t+1}, \ldots., a^m_{t+1})) \end{aligned} \tag{6}$$

$$NashQ^i_t(s,a) = \sum_1^m \pi^i(s,a) Q^i_t(s,a) \tag{7}$$

To minimize the system cost, the multi-agent *Q*-learning algorithm has to explore possible strategies randomly and greedily choose the "good" strategy. As shown in Formula (8), γ is a constant factor between 0 and 1. The learning policy satisfies the GLIE (Greedy in the Limit with Infinite Exploration) property.

$$\pi^i_{t+1}(a^*) = \begin{cases} \pi^i_t(a) + \gamma(1-\pi^i_t(a)), & if\ a^* = \arg\underset{a}{NashQ}(s^i_t, a) \\ \gamma \pi^i_t(a), & if\ a^* \neq \arg\underset{a}{NashQ}(s^i_t, a) \end{cases} . \tag{8}$$

In Algorithm 1: Line 1-6 is network initialization. In line 4, |A| is the number of control strategies. Line 7-12 is *Q*-learning procedure. In line 9, $R_1, \ldots R_m$ denote the rewards for tree 1 to tree *m*. $a_1 \ldots a_m$ denote the strategy taken by tree 1 to tree *m*. Line 10 is the *Q*-value update of each user according to Formula (7). The time complexity and space requirement of this learning algorithm is high when agent number is big. For 2-player Nash *Q*-learning, it has exponential worst-case time complexity. The space complexity is also exponential in the number of users.

Algorithm 1: Multi-agent Q-learning based Control

```
for $i$=1…$m$ //$m$ agents
    Let $t$=0, get the initial state $s_0$
    for all $s \in S$ and $a \in A$
        $Q_0^i(s, a^1, ..., a^m) = 0, \pi_0^i(a) = 1/|A|$
    endfor
endfor
while (network execution condition is TRUE)
    Choose action $a_i^t$ according to (8)
    Observe R1,…,Rm, a1,…, am
    Update $Q_t^i$ for $i$=1…$m$ according to (7)
    $t$=$t$+1
endwhile
```

3.4. Distributed cross-layer data collection

3.4.1. Network models

Given a WSN deployed in a surveillance area with n heterogeneous sensors and m controllers (i.e., sinks) that connected to Internet. Each sensor can adjust its maximal transmission ranges to one of the levels: r_0, $r_1 \ldots r_{max}$ by using different transmission power levels from p_0, p_1, till p_{max}. Initially, all controllers work in default power p_0. A directed graph G (V, E) is used to model the network system. An end-to-end data delivery bound T_{max} is given as the maximum acceptable delay timeliness in reporting hazard event packets to the controller.

For each directed link e (u, v) in G, our data collection tries to find a data delivery route from data source s to sink d that satisfies the following:

- The found route delay from s to d is within the bound T_{max}.
- Within the power vector, each link on the path makes feasible schedule, i.e., interference-free scheduling.
- There exists a minimal end-to-end power allocation on each path link on this route:{p_0,p_1,p_2,….p_n} ,while $p_i \in$ [p_{min},….p_{max}], $i \in$ [0,1,….n].
- The data collection scheme is adaptive to the surveillance environment scenarios.

The physical model of interference is formed by signal to noise ration. Transmission from u can be successfully received by a receiver node v at slot t, if it satisfies:

$$\eta(p) = \frac{G_{uv} p_{uv}}{N_0 + \sum_{(x, y \in \tau \backslash \{(u,v)\})} G_{xv} p_{xy}} \geq \eta_{th} \tag{9}$$

In which, τ is the set of concurrent transmissions; p_{uv} is the power level set at the transmitter of node u for link (u, v). G_{uv} is the channel gain for (u, v) depending on path loss, channel fading and shadowing. η_{th} is a given threshold determined by QoS requirements such as bit error rate. N_0 is thermal noise power.

In protocol model, transmission methods are used to evaluate the interference impact. A node receiving from a neighbor, should be spatially separated from any other transmitter by at least a distance *D* defined by interference area. The "Interference Area" (IA) is defined as the maximal range that two concurrent transmissions would interfere with each other. For irregularity of radio, it is difficult to estimate IA by hops, because node *a* can reach node *b* doesn't mean node *b* can reach node *a*. In this case, "Possible Interference Area" (PIA) is defined as 2-hop neighborhood to estimate IA.

Assume that time is slotted into a non-overlapping equal time period called frame. The frame is divided into non-overlapping equal time periods calls time slots. Each time slot is a time slice enough for a data packet transmission and corresponding ACK. The frame structure is defined as shown in Fig. 4. Each frame includes three parts:

- Control part: A start beacon is broadcasted out and used for local time synchronization. It exchanges newly assigned slots of itself and its neighbors. In this case, each node exchanges the allocated slot of itself and its neighbors and then every node knows the allocated slots information among its 2-hop neighborhood. During this part, we use contention-based mechanism. Based on local allocated slots information in PIA, each node makes the interference-aware link slot allocation.
- Schedule part: The node schedules the assigned slot for the current link.
- ACK part: The node acknowledges the allocated slot in this part. If there is no flow on the allocated slot for continuous frames, the slot can be recycled. If interference occurs on allocated slot, then this slot is tagged as "interference slot". And then the node will choose the other available slots in control part in the next frame.

Each node has the same default frame length initially. In each frame, the default schedulable slot length is *L*.

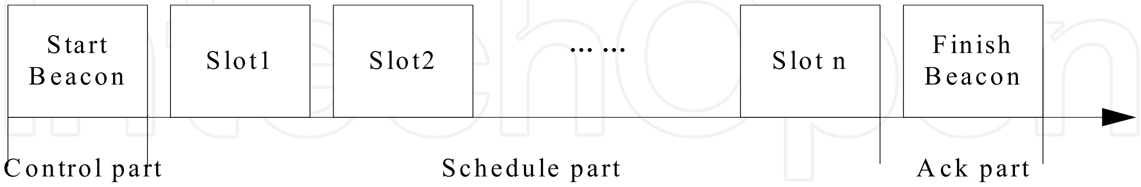


Figure 4. The frame structure

3.4.2. Cross-layer data collection schemes

The cross-layer can combine power control and schedule in data collection process and shift traffic towards the real-time delivery direction. Each node needs to maintain: 1) frame length and slot assignment information of it. 2) slot assignment information of its neighbors and 2-hop neighbors.

An admission control based method is used on each on-line link to make decisions on power allocation and schedule to achieve a real-time delivery with minor interference.

4. In "control" part: the node collects the slot allocation information. The per-link admission control is executed according to the delay deadline and interference constraints as follows:

- Calculate the available slots: the slots that are not being used by the other links among the node's possible interference area.
- Calculate the time left for end-to-end data delivery, which is denoted as slack. If satisfies $slack - t\_sche > 0$, the link schedule is admitted. The link schedule time t_sche is calculated by: $t\_sche = t_c + t_s$, where t_c is the time waiting for the next link schedule. t_s is the time of scheduling slots for current link. The number of allocated schedule slots should satisfy the link flow.
- If the allocated slots are not being used for continuous number of frames, it can be recycled for other transmissions.

5. In "scheduling" part: the node tries to schedule the feasible slot for transmission, i.e., the real-time and interference-free slots are found with enough flow.
6. In "Ack" part: the node acknowledges the slot allocation among its 2-hop neighbors. In this part, the problems of slot reuse and the interference are dealt with.

- Slot reuse: if allocated slots are not used by certain number of continuous frames, they can be reused for other transmissions.
- Interference: if current link schedule interferences with other transmissions, the failure slots will be tagged as "unavailable" and then to choose other slots in the next frame.

The distributed cross-layer data collection is described in Algorithm 2. Line 1-21 describes the distributed cross-layer data collection algorithm, where the data delivery can combine the metrics of the existing real-time routing in WSNs. Line 22-34 is link admission control function. Line 1-5 shows that a node selects the next relay by metrics. If it finds a next hop satisfying admission control, then record the next hop with its assigned slot. Line 6-15 shows that the node will increase its power if cannot find a next hop satisfying successful admission control. The node increases the power by levels until it reaches the maximal power, and tries admission control each time. If the link admission control is successful with current power, then breaks out of the loop. Line 16-18 shows that the routing will be blocked, if the node fails to find a next hop satisfying admission control even with the maximal power. Line 23-28 shows how to assign slots for current link e. In line 24, $L \backslash\ slot(e')$ is the available slots except interference slots of current link. Line 27-28 is constraint condition that the time left for node data routing should be larger than the schedule time. Line 29-33 shows the return value of the function: if admission control is successful, the return value is 1. Otherwise, the return value is 0. It is obvious that algorithm 2 is with linear time complexity and suitable for mission-critical wireless sensor networks. For selection of L parameter, it tends to guarantee an interference-free transmission among local interference area. So, we can give an initialized L as $|\Delta|^2+1$, Δ is max degree in the network graph.

Algorithm 2: Distributed cross-layer data collection

```
For each node $u$ selects the next relay with $p_{cur}$
  If reaches $v$
    If $link\_admission\_control(e(u,v); p_{cur})$
        record allocated slots for $e$
        return
    else
        do
           increase $p_{cur}$ to $p_{cur}+1$ to reach $v'$
           $p_{cur} \leftarrow p_{cur}+1$
             If $link\_admission\_control(e\ (u,v');p_{cur})$
               record allocated slots for $e$
               return
             endif
        while $p_{cur}<=p_{max}$
        enddo
        If  $p_{cur}>p_{max}$
           routing is blocked
        endif
    endif
  endif
endfor
int $link\_admission\_control(e(u,v);p_i)$
{
   $slot(e) \leftarrow$ subset of $(L \setminus slot(e'))$ with size $Flow_e$;
   // $L$ is the schedulable slot number in a frame
   //$e'$ is the 2-hop neighborhood PIA links of $e$
    s.t.
    $slack - t\_sche > 0$
    If $slot(e)!=\Phi$
       return 1
    else
       return 0
    endif
}
```

3.4.3. Data collection with priorities

In multi-event sensor-based cyber-physical surveillance systems, there are different emergency levels for data collection. Two priorities for events can be defined in emergency missions as: *"high"* and *"ordinary"*. For example, we define event priority as *"high"* priority, when it happens at nodes in dangerous conditions such as short of energy. The event as *"high"* priority can be defined as it happens on specific positions that in crowded venues or that is easy to collapse in surveillance area. The priority of data packets in hazard shows the different levels of importance for data delivery. The priority of event shows the different levels of importance and emergency for multi-event mission-critical data delivery. In this case, the data collection requests with high priority event always have more advantages on real-time data delivery when compared with an ordinary event.

According to surveillance scenarios, the node with high event priority could double its frame as shown in Fig. 5. If a node with high priority event data packets cannot find its next hop with interference-aware schedule, this node broadcasts a request to try to double its number of slots in the frame. The double frame request is broadcasted out in the control part of a frame. During the control part, if current node receives other requests from neighbors, only the one sent out earlier is admitted. The other requests are blocked. The node doubles its frame will notify its 2-hop neighbors. These 2-hop neighbors will update their frame length to keep local synchronization. For the synchronization among 2-hop neighbors and their neighbors, the double frame method can guarantee that there is at least one successful synchronization chance during every two previous frames. For a node in the network, it can easily reconnect to the network during continuous two timers of cycles among its neighborhood.

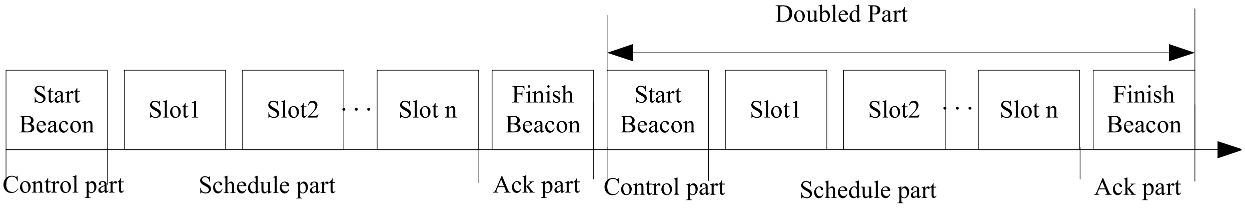


Figure 5. The structure of doubled frame

In Algorithm 3: Line 1-6 shows that each node selects the next hop until it cannot find the next hop with successful admission control. Line 7-15 shows the node increases its power level gradually to try to find an interference-aware next hop with a feasible schedule until reaches p_{max}. Line 16-23 shows if the node reaches p_{max} and cannot find a satisfying next hop with high event priority, then tries to double the frame length continuously to find a satisfying next hop. In lines 24-27, if it finds such a next hop with feasible schedule by double frame, the node records the slot information. Lines 34-39 show if no other frame double requests or current double request wins the tie, then both nodes on current link doubles its frame length. Otherwise, the function returns 0 to imply the failure of double request. Line 40-42 shows if current link nodes double the frame, they broadcast notification to 2-hop neighbors. Those nodes that receive such messages will update their frame length to keep synchronization.

The algorithm has linear time complexity, and easy to be performed in WSNs. The double frame scheme guarantees to deliver the high priority packets with high probability. A node with doubled slots can halve the frame till its default frame length and provide more chances for other data packets when necessary.

Algorithm 3: Distributed priority-based data collection

```
For each node $u$ selects the next relay with $p_{cur}$
 If can reach $v$ with $p_{cur}$
  If $link\_admission\_control(e(u,v);p_{cur})$
    record assigned slots for $e$
    return
  else
    do
      increase $p_{cur}$ to $p_{cur}$ +1 to reach $v'$
      $p_{cur} \leftarrow p_{cur} +1$
      If $link\_admission\_control(e\ (u,v');\ p_{cur})$
          record assigned slots for $e$
          return
     endif
    while $p_{cur} <= p_{max}$
    enddo
    If  $p_{cur} > p_{max}$ && $event\_priority ==$ "high"
            do
                  $p_{cur} \leftarrow p_{max}$
                  If  !$double\_frame(e\ (u,\ v'))$
                        break
                  endif
            while !$link\_admission\_control\ (e(u,\ v');\ p_{cur})$
             enddo
              If $link\_admission\_control(e(u,v);\ p_{cur})$
                  record assigned slots for $e$
                  return
           endif
    endif
  endif
 endif
endfor
int $double\_frame(e(u,v))$
{
  If no other requests or current request wins the tie
      $Frame\_\ length(u)= Frame\_length(u)*2$
      $Frame\_length(v) = Frame\_length(v)*2$
  else
       return 0
  endif
  updates frame length among 2-hop neighbors
  return 1
}
```

3.4.4. Comparisons and simulation results

We verify the above data collection schemes for sensor based cyber-physical system in mission-critical hazard surveillance applications by simulations using ns2 network simulator. We consider the building fire hazard as a case study. To generalize and simplify the surveillance blueprint, we use a grid topology. The simulation results can represent performance for a general case. We choose 100 nodes that are distributed as grid topology in a 100m×100m area (Zeng et al., 2011). Each node can work under 3 power levels. We place 1-4 sinks on the corner of the simulation areas, respectively. We simulate and test our algorithms with an epidemic building fire model. Within the simulated area, a fire start point breaks out randomly 30s after the simulation is started. We define "high" priority hazard event when the event source node is with less than 10% energy. We use some metrics for performance evaluations. End-to-end delay is the total time needed for a data packet sent out till it is received correctly by a sink node. The packet delivery miss ratio (use "miss ratio" in the following paragraph) is the ratio of all packets missed because of the delay bound and interference versus the total packets sent out. The energy efficiency is evaluated by average residual energy in the network.

Fig. 6 shows the Q-learning performance as delay bound increases. As we increase the delay bound, the average residual energy increases too. More low power feasible transmissions available can decrease the energy consumption in the network when we relax the delay bound. In 2-agent learning case, two sinks are selected on the diagonal corner of the area. The source node reports data to the sink periodically until it fails. We set the end-to-end delay bound as 60ms and test the energy efficiency when compares 1-agent Q-learning and 2-agent Q-learning performance. Fig. 7 shows the average residual energy of 1-agent Q-learning and 2-agent Q-learning. From the result, we can observe that performance of 2-agent Q-learning is better than 1-agent Q-learning case because of one more sink. Fig. 8 shows the end-to-end delay as delay bound increases from 15ms to 100ms. The priority-based algorithm has slightly less delay, because it increases the delivery probability of the critical event in fire. Fig.9 shows the miss ratio performance. Fig. 10 shows the node average residual energy. We consider the high priority of critical event occurred by low-energy nodes in our simulations, those nodes deliver data packets with high priority and then achieves better energy efficiency in the network.

We then compare our distributed algorithms with RPAR (Chipara et al., 2006) and EAR (Zeng et al., 2011). Fig. 11 shows the end-to-end delay results comparison. We observe that our two distributed algorithms achieve much better performance compared with the other two mechanisms. Because we consider not only real-time delivery but also the interference-aware transmission, more packets are delivered within the delay bound successfully. Fig. 12 shows the miss ratio of the four mechanisms. Fig. 13 shows average residual energy.

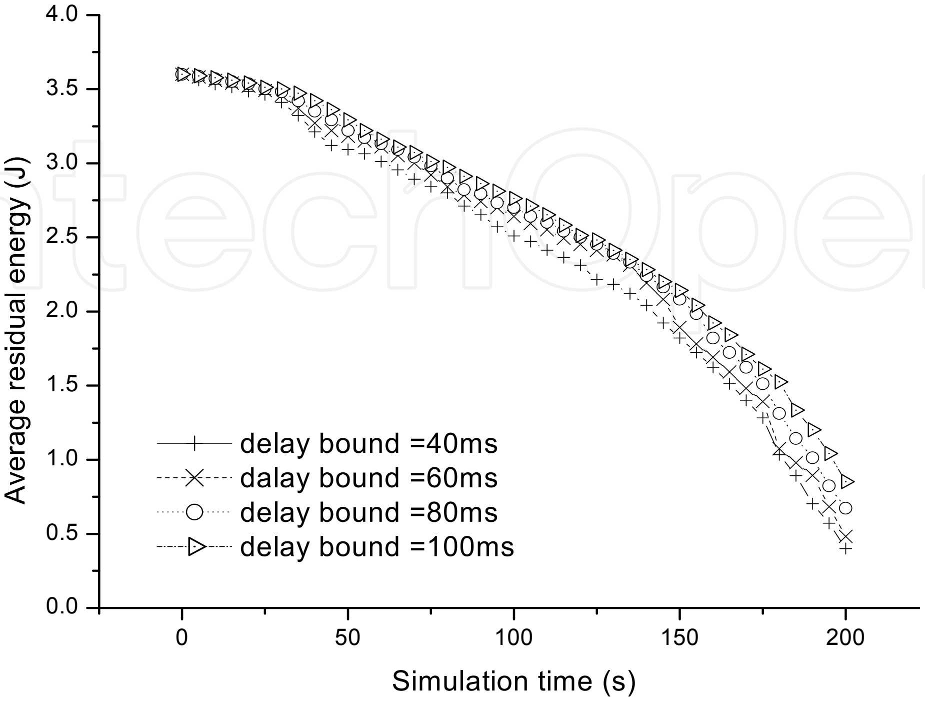


Figure 6. Energy efficiency by Q-learning as delay bound increases.

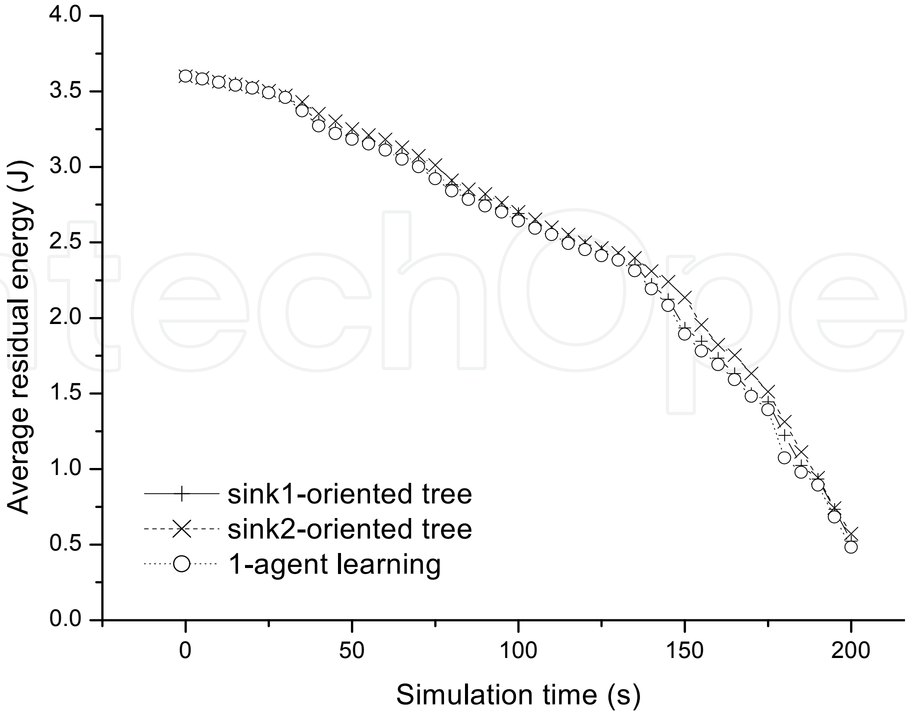


Figure 7. Energy efficiency of 2-agent Q-learning (delay bound=60ms).

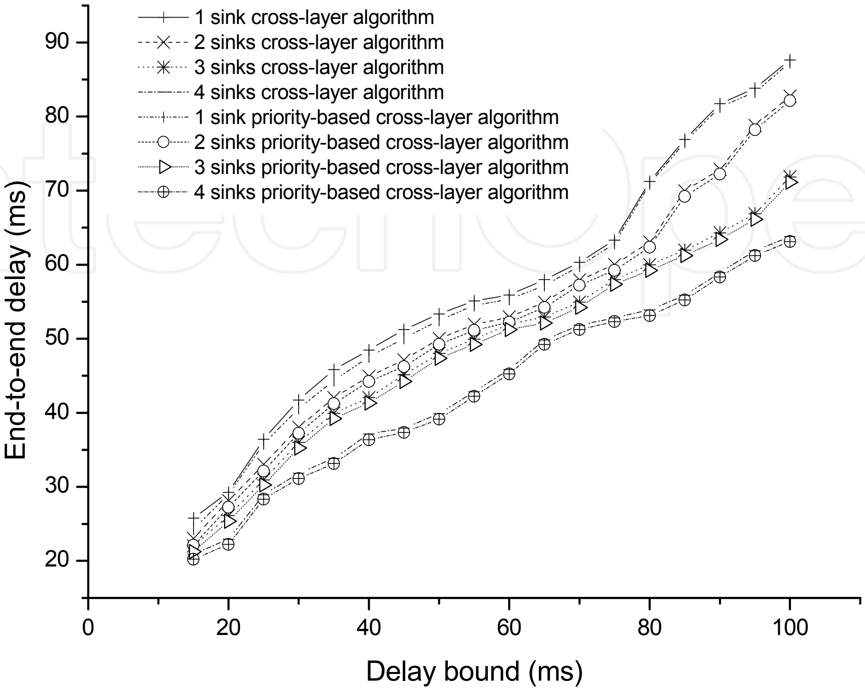


Figure 8. End-to-end delay as delay bound increases

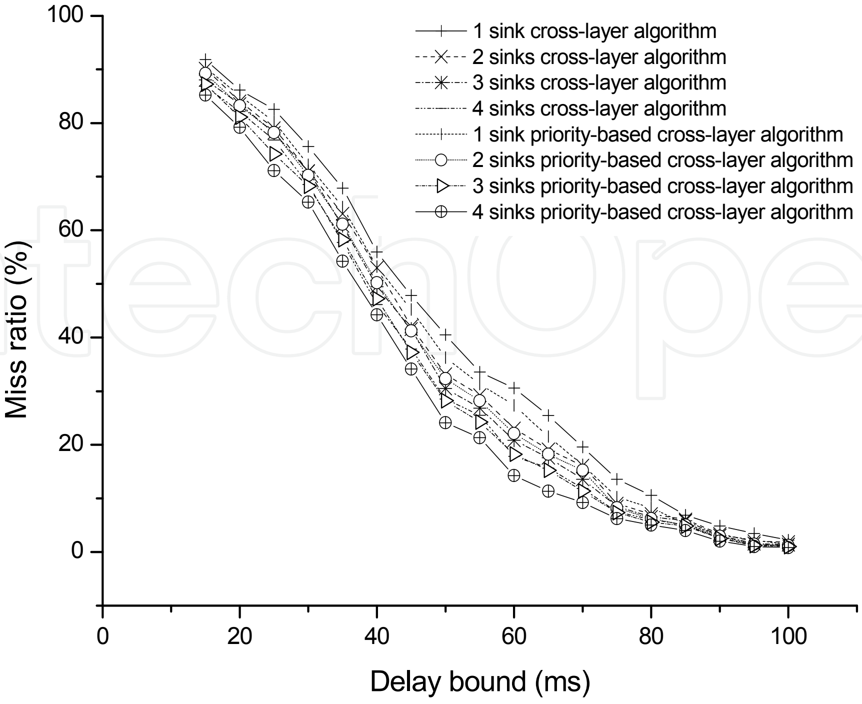


Figure 9. Miss ratio as delay bound increases

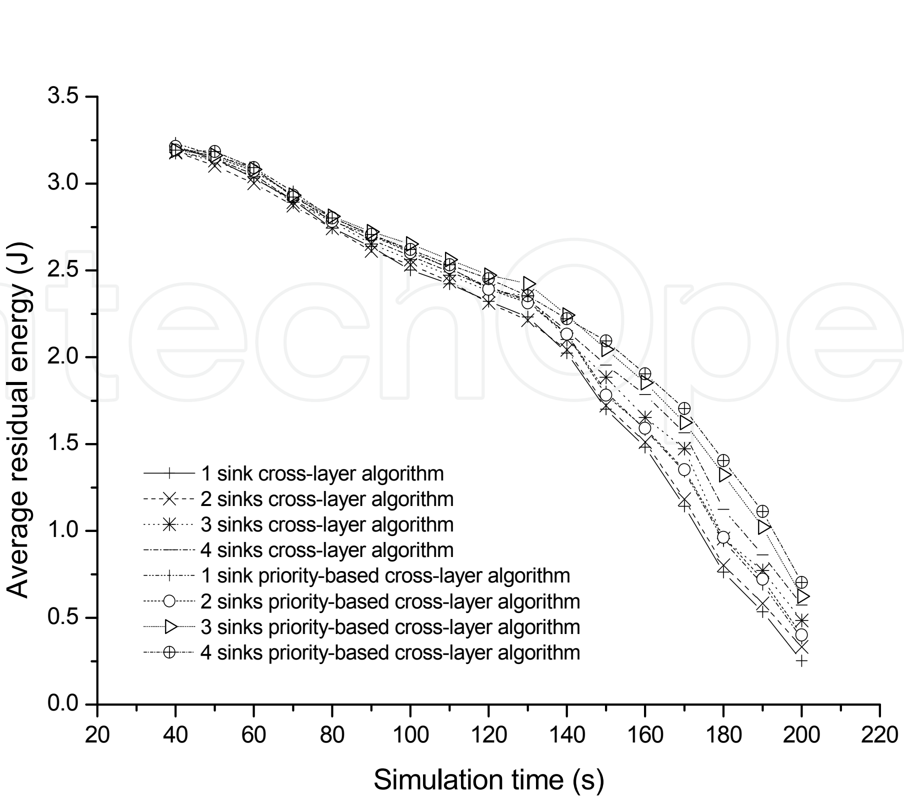


Figure 10. Energency efficiency with/without priorities (bound =60ms).

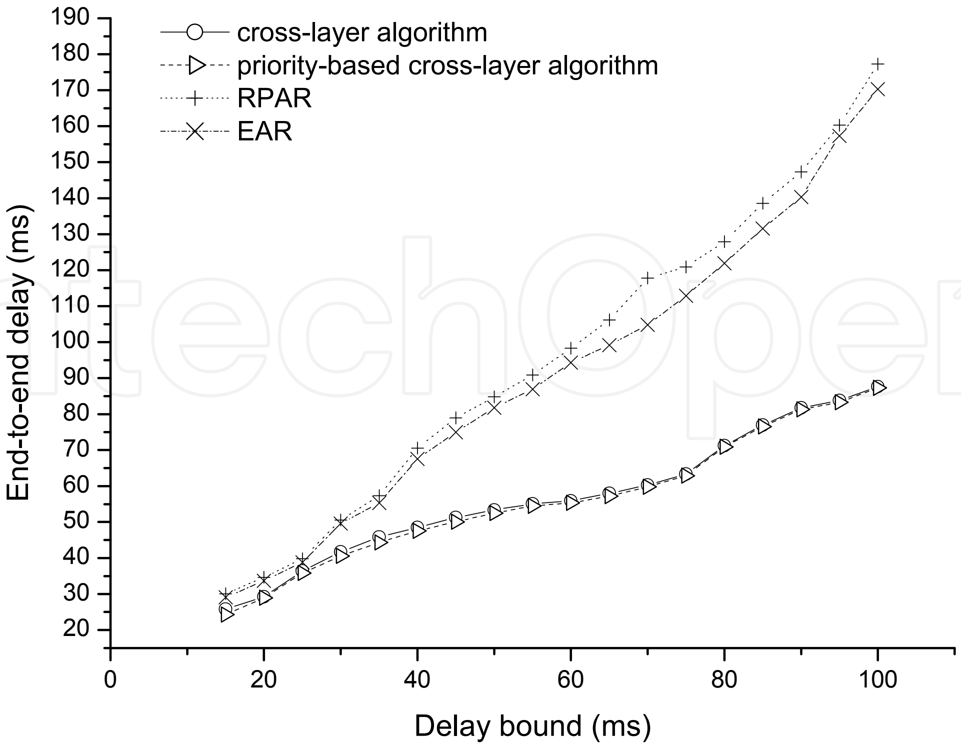


Figure 11. Comparison of end-to-end delay

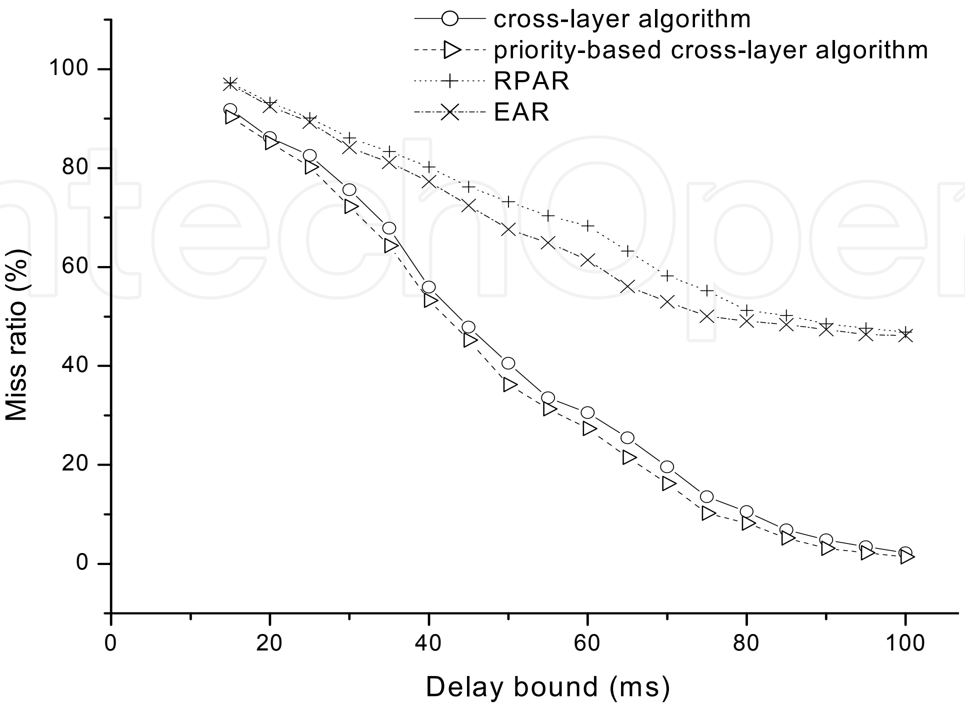


Figure 12. Comparison of Miss ratio

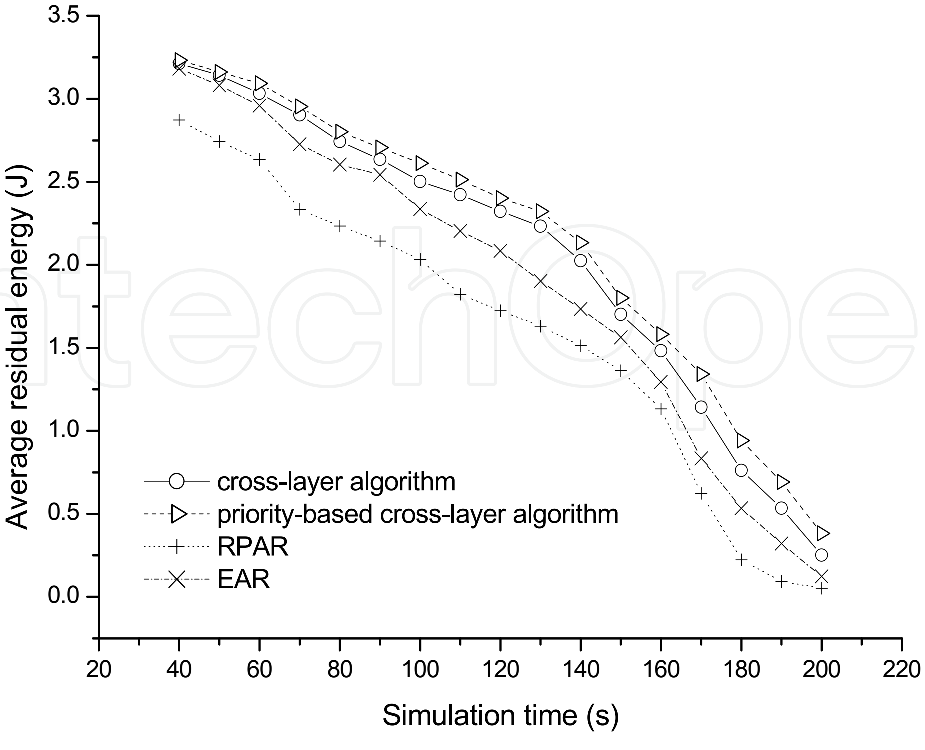


Figure 13. Comparison of energy efficiency.

4. Event detection technologies

Wireless sensor networks can be utilized continuously to monitor the physical phenomenon with the deployment in surveillance areas. While the sensor only reports its local information and data that may be uncertain and unreliable, effective processing, collaboration and analysis of data streams become necessary and important. According the characteristic of mission-critical application-specific environment, the design of detection approaches is facing more challenges. The detection models and architectures with good performance are needed that can effectively detect the event from data sensed within the timeliness and accuracy requirements.

4.1. Challenges for mission-critical detection

To achieve the goal of event detection especially emergency detection, we require the continuous assessment of the state of the surveillance area, forecasting the possibility of events. The volume of data that is gathered form the monitoring network usually can be enormous, particularly with high bandwidth devices such as video cameras. According to this large amount of sensing data, an efficient model that can process well the data is needed. In a majority of cases the information is insufficient and the inter-communications between different sensors detecting the emergency event are inadequate, which leads to detection inaccuracy and a potentially huge loss of life and resources. This is even worse for the complex and varied environment.

4.2. Related work on WSN event detection

A lot of work has been done for event detection in wireless sensor networks. The work can be categorized into different categories according to the approaches.

Firstly, sensing coverage is an important problem in WSNs and surveillance applications. One individual sensor is easy to get weak and noisy signal. The multi-sensor information fusion and aggregation algorithms have been proposed to make event detection by using the coverage of sensors. In this case, combining the multiple sensor capability to provide the enough detecting reliability by increased coverage obtained from multiple sensors. The basic objectives of this category approaches are to minimize the probabilities of detection errors while meeting the specific constrains of communication cost and energy efficiency. In which, energy saving is usually achieved by ensuring at least k nodes are awake to cover the detection location, leaving the other nodes asleep (Xing et al., 2005).

Some model-based approaches are proposed by using statistical analysis. Zhuang et al. proposed to use a weighted moving average based approach to remove outlier sensor data as noise (Zhuang et al., 2007). Gupchup et al. proposed to use well-established Principal Component Analysis to build a compact model of the observed phenomena (Gupchup et al., 2009). The Principal Component Analysis model is used to determine a single or a sequence of measurements that are dissimilar to the normal behaviour of the system. There are some solutions using a probabilistic noise distribution to mathematically model the sensing area and detect events (Rachlin et al., 2005).

Other solutions utilize machine learning based approaches to provide event detection. This category of approaches makes feature classification and exaction from sensing data (Benbasat & Paradiso, 2007; Eriksson et al., 2008; Kang et al., 2008). There are some approaches that propose to utilize Hidden Markov Models to deduce events (Singh et al., 2008).

Event detection by using WSNs is also application-specific. Especially for mission-critical applications, they need to impose stringent requirements for event detection accuracy and network lifetime, etc. Keally et al. proposed a confident event detection approach in WSNs that can adapt the detection capability with the runtime observations to save energy (Keally et al., 2010). There is some related work that combines activity recognition into sensor based event detection in body networks (Keally et al., 2011).

4.3. Detection cases for mission-critical monitoring

4.3.1. An integrated fire emergency response system: FireGrid

The FireGrid system (Upadhyay et al., 2009) aims to leverage a number of modern technologies such as wireless sensor networks and grid computing to aid building fire emergency response. As shown in Fig.14, to detect the building fire and make responses toward hazard events, there are different visions with different technologies in the system to communicate among fire steward, remote management, and collaborate with smart devices in the building such as sprinklers, evacuation indicators, etc. The building modelling is achieved by blueprints, maps and scenarios. The emergency response is dependent on the physical information that is achieved by sensor networks through sensing, transmission and then analysis and decision making upon the data. The decision making of responders is based on a knowledge based system and planning by remote experts, which makes the responses by sensor data processing and super-real-time simulations.

4.3.2. The CodeBlue infrastructure

The CodeBlue infrastructure (Lorincz et al., 2004) is an ad hoc sensor network infrastructure for emergency medical care-tracking the patients and first response application system, which is introduced by Harvard University in collaboration with various medical facilities. CodeBlue is designed to provide routing, naming, discovery, and security for wireless medical sensors, PDAs, PCs, and other devices that may be used to monitor and treat patients in a range of medical settings. The goal is to enhance first-responders' ability to access patients on scene, ensure seamless transfer of data among caregivers and facilitate efficient allocation of hospital resources. Fig.15 shows the CodeBlue system architecture. The wearable wireless sensors are put on patients' body to collect heart rate, blood oxygen saturation, and hearts electrical activity, etc., continuously. The sensing data can be displayed in real time and integrated into system record. The smart devices are programmed to alert medical personnel when vital signs fall outside of the normal conditions. Any adverse change in patient status can then be signalled to a nearby emergency terminal or paramedic.

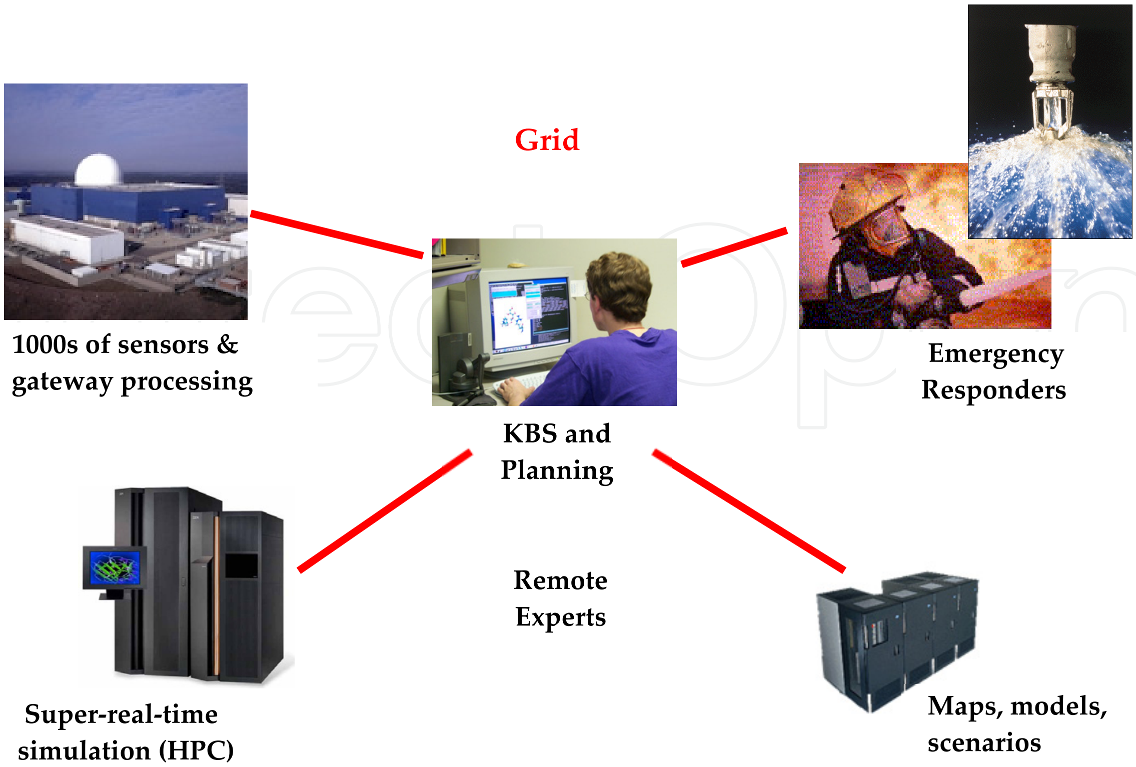


Figure 14. The FireGrid visions

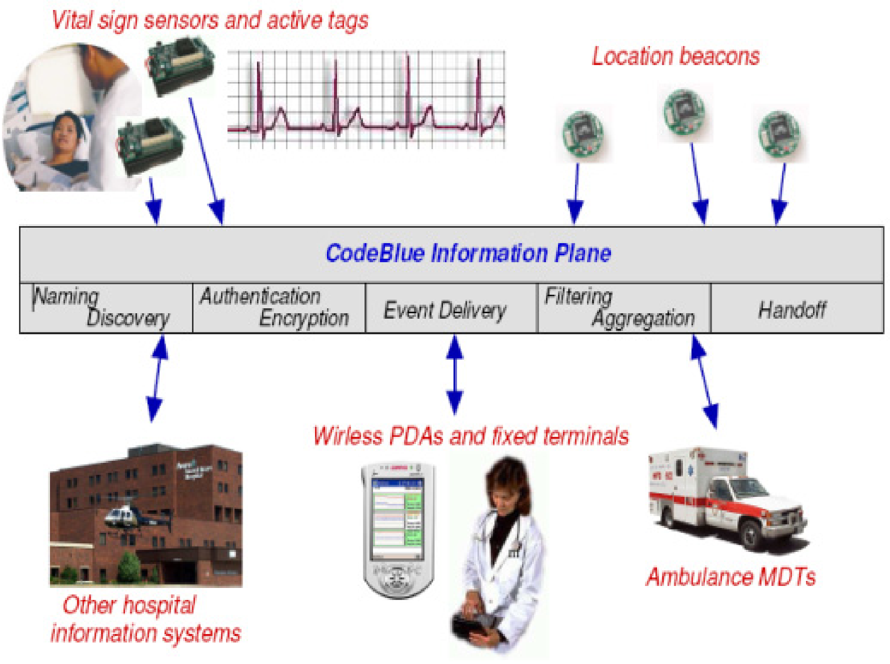

Vital sign sensors and active tags
Location beacons
CodeBlue Information Plane
Naming Discovery
Authentication Encryption
Event Delivery
Filtering Aggregation
Handoff
Other hospital information systems
Wirless PDAs and fixed terminals
Ambulance MDTs

Figure 15. The CodeBlue system architecture

4.3.3. The NEMBES facilities management

For NEMBES project on Facilities management within buildings, the objective is to facilitate a multi-user environment precise data aggregation, acquisition and interpretation that can provide successful building automation (Zeng et al., 2009). We take building fire as a case study. We then design a real-time simulator for detecting and handling building fire emergency scenarios. The goals of this simulator are to provide for: 1) a dynamic virtual test-bed for population routing and networking algorithms during emergencies, 2) identification of building features that impact on evacuation scenarios, such as corridors prone to congestion, 3) visualising real-world emergency situations and predicting outcomes to inform rescue personnel as to the best rescue strategy or possible danger areas.

The underlying world model for this simulation is an object-based 2.5 dimension "building". Each floor of the building is a 2D collection of world objects, with the floors arranged in a spacial collection (ground floor, first floor, second floor etc). Stairs, fire escapes and elevators provide a mechanism for agents to travel between floors. This 2-and-a-half dimension model was chosen as it simplifies agent behaviour computations and allows for very clear visualisation of the emergency as it unfolds. The underlying building objects have analogues within the Industry Foundation Classes building model objects, such as walls, doors and so on. The simulation features multiple agents with dynamic behaviours navigating a building during an emergency. These agents are driven by a Sense->Plan->Act cycle and have basic memory. The two main classes of Agent are "Occupant" agents (persons present in the building, primarily driven by environmental cues such as direction signs or following crowds) and "Firefighter" agents (primarily driven by individual instructions, such as radio contact or personal "compass" direction). Agents will have steering and crowding mechanisms to accurately reflect real-life population movement. The underlying physical model of the world combined with such measures will provide useful knowledge as to areas in the building with excessive traffic and poor movement flow, or parts of a building which are of high-importance for evacuation (e.g. a main corridor).Fig. 16 shows a screenshot of our simulation for building fire.

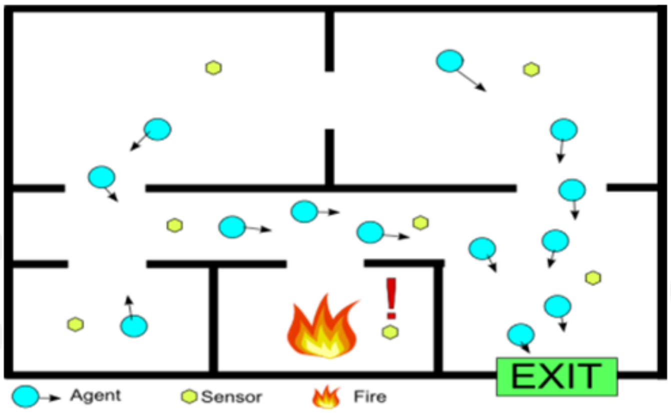


Figure 16. The NEMBES building fire simulator

4.3.4. Mobile sensing architectures

By using the mobility of people with cell phones, bicycles, cars, there are many platform and testbed projects are proposed based on mobile sensing networks.

BikeNet (Eisenman et al., 2007) is a platform of Mobile Sensing by using bicycles. In BikeNet, the real-time social networking is built among the cycling community. The bike area networking is a multifaceted sensing system that explores personal, bicycle and environmental sensing based on smart sensor motes and sensor-enabled Nokia N80 mobile phones. The mobile sensing data is collected through real-time and delay-tolerant communications to a networked repository, which is used to infer and find routes with low CO2 levels. There is provision of a web portal for cycling community that can access and share the real-time data of best routes with better air quality. Fig. 17 shows the BikeNet network architecture.

CarTel (Hull et al., 2006) is a distributed vehicle sensor network platform proposed by MIT Computer Science and Artificial Intelligence laboratory, which combines sensing, processing, analysis and visualization. CarTel platform is involved in several applications including commute and traffic portal, traffic mitigation using new predictive delay models, Pothole Patrol, fleet testbed and Wi-Fi monitoring, cars as mules etc. CarTel provides a simple query-oriented programming interface, handles large amounts of heterogeneous data from sensors, and handles intermittent and variable network connectivity. CarTel nodes rely primarily on opportunistic wireless (e.g., Wi-Fi, Bluetooth) connectivity—to the Internet, or to “data mules” such as other CarTel nodes, mobile phone flash memories, or USB keys—to communicate with the portal. Fig. 18 shows CarTel in traffic delay estimation application (Thiagarajan et al., 2009).

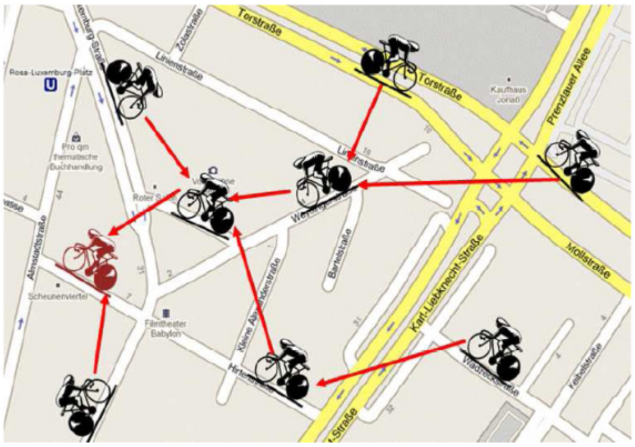


Figure 17. The BikeNet system architecture

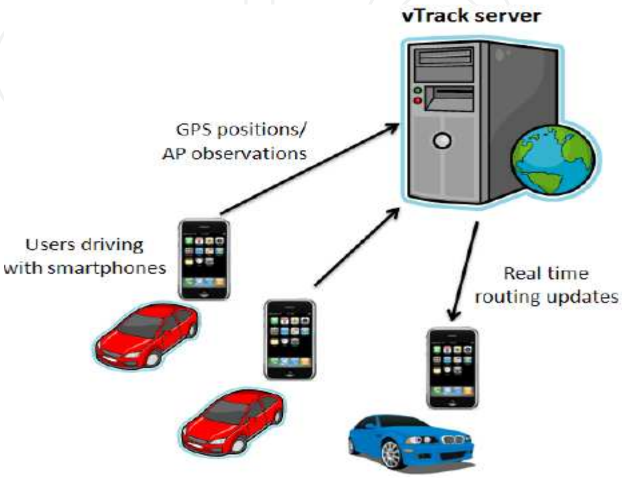


Figure 18. An illustration of traffic delay estimation application

5. Monitoring by using wireless sensors and actuators

Recent advances in pervasive computing, communication and sensing technologies are leading to the emergence of wireless sensor and actuator networks (WSANs). A wireless sensor and actuator network is a group of sensors and actuators that are geographically distributed and interconnected by wireless networks. Sensors gather information about the state of the physical world, and the actuators react to the information by performing appropriate actions. In most of the mission-critical monitoring applications, it also expects to respond to the sensed data and event by performing corresponding actions upon the system. WSANs enable the application systems to sense, interact, and change the physical world, e.g., to monitor and manipulate the temperature and lighting in a smart office or the speed and direction of a mobile robot. WSANs are widely used in mission-critical applications such as home automation, environmental monitoring and industrial control system, etc.

5.1. Architecture

5.1.1. Network-based architecture

There are two architectures called semi-automated and automated architectures for WSANs, respectively.

In semi-automated network architecture, one or more controller entities explicitly exist in the network. The controller is functional modules embedded in the base stations or separated nodes equipped with sufficient capabilities. The sensors collect the data and transmit to the controller. Then the controller produces control commands and sends them to the actuators according to the sensed information. The actuators perform the actions according to the commands they receive.

In automated network architecture, there is no explicit controller entity in the network. In this case, the collected data of sensors will be transmits directly to the actuators. The actuators will make decision and perform the actions according to the data sensed. In this kind of architecture, the actuators serve not only the executor of actions but also the controller.

5.1.2. Application-specific architecture

The architecture design problem is always an application-specific open issue. The concept of service-oriented sensor network architecture was proposed that encompass a set of sensor services and enables rapid scalable systems based on reusable services (Rezgui & Eltoweissy, 2007). For deploying this idea in WSANs need to handle problems such as the functionality interface and probabilities of service, as well as identifying the differences of sensor and actuator services.

There are usually QoS requirements for WSANs, the architecture needs to be designed with in mind the QoS demand and resource constraints of sensor and actuators respectively. The QoS-aware architecture design should consider the communication and coordination protocols within its framework, which is usually achieved by using cross-layer design to effectively optimize the network performance. The requirements imposed on WSANs are varied and depend on the applications. Considering mission-critical monitoring applications, WSANs should have high scalability to support timely data collection, event detection and task exectution. These requirments should be best addressed at the network architecture. The well investigated and appropirate architecture will benefit the network protocoals and other solutions.

5.1.3. Software-based architecture

In order for the wireless sensor and actuator networks to function, software-based architecture should be proposed and effectively manage unreliable dynamic distributed communication and coordination with the resource constraints, hostile environments and mission-critical tasks. The software-architecture can be achieved by middleware that can provide with a set of services and solutions involved in the network application (Chaczko et al., 2011). The software-based architecture is based on layered model, i.e., a set of services can interact with highly scalable central services. All the services such as node management service, task service, configuration service etc., are integrated through layers in the architecture.

There is another solution to design the software-based architecture based on collaborative operations between the sensors and actuators, enabling distributed sensing and action performing. The network functions are categorized such as: localization, communication, target tracking and standardization of services/interfaces. The data flow of sensors and actuators are categorized and analyzed. Then, the communication flow among network components: sensors and actuators are defined by the access method, physical link and formulation. The network software paradigm will be designed based on the function flow, data flow and communication flow.

5.2. Coordination

The coordination in WSANs is responsible for effective data sensing and action performing with limited resources by fully making use of node mobility. From the control perspective, mission-critical monitoring application systems are inherently real-time systems in the sense that control actions should be performed on the physical systems through valid node coordination. From the function perspective of coordination, it includes sensor-actuator coordination and actuator-actuator coordination, which is mainly responsible for network collaboration and task allocation. When a sensor gets the event packet, it will decide which actuator it reports to. Each actuator takes charge for its local real-time data collection from nearby sensors to form clusters in the monitoring filed, i.e., the actuator acts as a cluster head in the local cluster. And the clusters would vary with the dynamic topology, residual energy and reliability of mobile actuators. The objective of actuator-actuator communication and coordination is to select the best actuators to perform appropriate reactions towards the event. Each actuator acts as a collector by receiving event data from a subset of the sensors. However, one actuator acting as a data collector may not be able to act in its event area, i.e., the event area may not be totally covered by the action range of the actuator or the actuator has no enough energy to do so, etc.. For this reason, before action, actuator-actuator coordination is required to make decision on the choice of optimal actuators to perform tasks by considering the acting capability of actuators such as acting coverage, energy level and completion time.

5.2.1. Coordination in mission-critical environment

There is an optimal strategy for actuator coordination that is formulated as a mixed integer non-linear program. The objective of the optimization problem is to find minimal number and velocity of appropriate actuators to finish the task that resides in the surveillance area. Each occurring event e in the event space Ω can be characterized by {$F(e)$, $Pr(e)$, $S(e)$,$D(e)$},where $F(e)$ describes the event type, $Pr(e)$ the priority, $S(e)$ the event area, and $D(e)$ the action completion bound, i.e., the maximum allowed time from the instant when the event is collected to the instant when the associated action needs to be completed. The following notations are used in the problem formulation:

$X(e)$is a binary vector $[x_a^{(e)}]$ whose element is equal to 1, if actuator a acts on $S(e)$.
$V(e)$ is a vector $[v_a^{(e)}]$ whose element represents the velocity assigned to actuator a.
$T_a^{\Omega,(e)}$ is the time that actuator a needs to complete the task associated with event e when a is part of an acting team.
$d_a^{(w)}$ is the distance between actuator a and the center of the event area S when the acting range of actuator a is not within the event area.
$T_a^{M,(e)}$ is time need by actuator a to reach the event area.
T^c is the coordination time for actuators.
η_a^f is the performing rate (m^2/s) of actuator a acting on an event type f.
S_a^c is the subset of coordinating actuators when an event e occurs.

The problem can be cast as follows:

Find:

$$\mathrm{X}^{(e)} = [x_a^{(e)}], V^{(e)} = [v_a^{(e)}] \tag{10}$$

Minimize:

$$\sum_{a \in S_a^c} x_a^{(e)} * v_a^{(e)} \tag{11}$$

Subject to:

$$T_a^{M,(e)} * v_a^{(e)} = d_a^{(w)}, \forall a \in S_a^c \tag{12}$$

$$0 \le v_a^{(e)} \le v_a^{\max} \tag{13}$$

$$T_a^{M,(e)} + T_a^{\Omega,(e)} + T^c \le D^{(e)} \tag{14}$$

$$\sum_{a \in S_a^{f,(e)}} x_a^{(e)} \geq 1 \tag{15}$$

$$\sum_{a \in S_a^c} x_a^{(e)} \cdot \eta_a^f \cdot T^{\Omega(e)} \geq S^{(e)} \tag{16}$$

The other promising ways to go are using feedback control, swarm intelligence and machine learning could be used to make design on coordination control algorithms. The distributed algorithms make design on heuristic methods in optimizing network performance with flexible resource management in dynamic and unpredictable environments.

5.2.2. Distributed task allocation

The distributed coordination in mission-critical monitoring, a negotiation-based algorithm can be utilized to make agreement among actuators about collaboration and task allocation.

When an actuator gets the last data packet of an event from local sensors, it broadcasts out a *Negotiate*(*actuator_id*, *Event_id*, *Event_priority*) message, and we call this actuator as negotiation invoker. The *Event_id* is the id of the event that the actuator collects; and the *Event_priority* is an application-specific identity used to discern the urgent degree of each event, such as “very high”, “high”, “medium”, “low”, and “very low”. The event priority is predefined according to the application requirements. The node that receives a *Negotiate* message will reply an *ActReply*(*actuator_id*, *energy*, *location*, *Event_id*, *Event_priority*) message with its own id, event id and priority. The negotiation is executed in the local area of the actuator invoking the process. Multiple sensors placed in the monitoring field could be used to detect the same event, so one or multiple actuators will get the sensing data of the exact event. Through the process of backward *ActReply* message, the data of the same event (with the same *Event_id*) will be aggregated during the backward process and forwarded back to the first actuator that invokes the negotiation. If there is more than one event sensed by an actuator, then after the data aggregation for the same event, a multi-event ordered task assignment should be involved. The task assignment on this aggregation actuator will be executed according to the event priority, i.e., the event with higher *Event_priority* rank will win. If there is a tie of the event priority, the event with lower *Event_id* will win. Beyond the above, the aggregation actuator will get to know the whole area and location of the detected event through multi-actuator communication. The event area could be represented by using the left upside coordinate (Ev_X_i, Ev_Y_i) and the right downside coordinates (Ev_X_j, Ev_Y_j) approximately. During negotiation, we could combine each sub event area gathered by each actuator into the whole event area.

6. Summary

Considering the changing and time-varied topology and application scenarios in surveillance areas, the monitoring applications under mission-critical environment are very challenging. In this chapter, we have given some of the main concern for design in mission-critical monitoring application scenarios, and bring forward efficient methods and solutions according to network connectivity, dynamic application scenarios. Major research challenges and open research issues in mission-critical monitoring applications of WSNs are also outlined.

Author details

Yuanyuan Zeng and Deshi Li
School of Electronic Information, Wuhan University, Wuhan, China

Kai Xiang
School of Information Management, Hubei University of Economics
School of Computer, Wuhan University, Wuhan, China

Acknowledgement

This work was supported Chinese National Science Foundation Project (no. 61103218), Hubei Province National Science Foundation Project (no.2011CDB446), Wuhan University Automatic Research Project (no.111093), and Chinese National Program on Key Basic Research Project (no. 2011CB707106).

7. References

Poovendran, R. (2010). Cyber-physical systems: close encounters between two parallel worlds, *Proceedings of the IEEE*, 98(8), pp. 1363–1366.

Karp, B.; & T. Kung, H. (2000). GPSR: Greedy Perimeter Stateless Routing for Wireless Networks, *Proceedings of ACM MobiCom*, pp. 243-254.

Yu, Y.; Estrin, D.; Govindan, R. (2011) Geographical and energy-aware routing: a recursive data dissemination protocol for wireless sensor networks, UCLA Computer Science Department Technical Report, pp. 1-11.

Powell, O.; Nikoletseas,S. (2007). Geographic routing around obstacles in wireless sensor networks, Technical report, Computing Research Repository.

Fang, Q.; Gao, J.; Guibas, L. (2006).Locating and bypassing holes in sensor networks, *Mobile Networks and Applications*, 11(2), pp. 187-200.

Rogers, A.; David, E.; R. Jennings, N. (2010). Self-organized routing for wireless microsensor networks, *IEEE Transactions on Systems, Man, and Cybernetics*, 35(3), pp.349-359.

He, T.; Stankovic, J.; Lu, C.; Abdelzaher, T.(2003). SPEED: A Stateless Protocol for Real-time Communication in Sensor Networks,*Proceedings of ICDCS'03*.

Felemban, E.; G. Lee, C.; Ekici, E.; Boder, R.; Vural, S.(2005). Probabilistic QoS Guarantee in Reliability and Timeliness Domains in Wireless Sensor Networks, *Proceedings of InfoCom*.

Chipara, O.; He, Z.; Xing, G.; Chen, Q. (2006). Real-time Power-aware Routing in Sensor Networks, *Proceedings of 14th IEEE International Workshop on Quality of Service*.

Ahmed, A.; Fisal, N. (2008). A Real-time Routing Protocol with Load Distribution in Wireless Sensor Networks, *Computer Communications*, 31(14), pp.3190-3203.

Kim, J.; Ravindran, B. (2009). Opportunistic Real-time Routing in Multi-hop Wireless Sensor Networks, *Proceedings of the 2009 ACM Symposium on Applied Computing*.

Sivrikaya, F.; Geithner, T.; Thruong, C.; A. Khan, M.; Albayrak, S. (2009). Stochastic Routing in Wireless Sensor Networks, *Proceedings of IEEE International Conference on Communications Workshop*.

Yang, S.; Cheng, H.; Wang, F. (2010). Genetic Algorithms with Immigrants and Memory Schemes for Dynamic Shortest Path Routing Problems in Mobile Ad hoc Networks, *IEEE Transactions on Systems, Man, and Cybernetics*, 40(1), pp. 52- 63.

Li, Y.; Ai, C.; Vu, C.; Pan, Y. et al. (2010). Delay-bounded and energy-efficient composite event monitoring in heterogeneous wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 21(9), pp. 1373-1385.

Li, H.; Fen, L.(2009). Dynamic adaptive cooperative routing for emergency data in wireless sensor networks, *Proceedings of IEEE International Conference on Intelligent Computing and Intelligent Systems*.

Byun, J. , Park, S. (2011). Development of a self-adapting intelligent system for building energy saving and context-aware smart services, *IEEE Transactions on Consumer Electronics*, 57(1), pp: 90-98.

Zeng, Y.; Sreenan, C.; Sitanayah, L.; Xiong, N.; Park, J.; Zheng, G. (2011). An emergency-adaptive routing scheme for wireless sensor networks for building fire hazard monitoring, *Sensors*, 11(3), pp.2899-2919.

Tseng, C. Y.; Pan, S. M.; Tsai, Y. Y. (2006). Wireless Sensor Networks for Emergency Navigation, *IEEE Computer*, 39(7), pp. 55-62.

Sobral M. & Becker, L. (2008). A Wireless Hybrid Contention/TDMA-based MAC for Real-time Mobile Applications, *Proceedings of SAC'08*.

Lu, G. & Krishnamachari, B.(2007). Minimum Latency Joint Scheduling and Routing in Wireless Sensor Networks, *Ad Hoc Networks*, 5(6), pp. 832-843.

Joseph, V.; Sharma, V.; Mukherji, U. (2009). Joint Power Control, Scheduling and Routing for Multihop Energy Harvesting Sensor Networks, *Proceedings of 4th ACM workshop on performance monitoring and measurement of heterogeneous wireless and wired networks*.

Jain, K.; Padhye, J.; Padmanabhan V. N; Qiu, L. (2003). Impact of interference on multi-hop wireless network performance, *Proceedings of ACM MobiCom'03*.

Hu J. & Wellman, M. P.(2003). Nash Q-learning for General-sum Stochastic Games, *Journal of Machine Learning Research*, 4, pp. 1039-1069.

Xing, G., Wang, X.; Zhang, Y.; Lu, C.; Pless, R.; Gill, C.(2005). Integrated coverage and connectivity configuration for energy conservation in sensor networks, *ACM Transactions on Sensor Networks*.

Gupchup, J.; Terzis, A.; Burns, R.C.; Szalay, A.S.(2009). Model-Based Event Detection in Wireless Sensor Networks, *Proceedings in Proceedings of CoRR*.

Zhuang, Y.; Chen, L.; Wang, X.; Lian, J. (2007). A Weighted Moving Average-Based Approach for Cleaning Sensor Data, *Proceedings in IEEE ICDCS*.

Rachlin, Y.; Negi, R.; Khosla, P. (2005). Sensing Capacity for Discrete Sensor Network Applications, *Proceedings in ACM/IEEE IPSN*.

Eriksson, J.; Girod, L.; Hull, B.; Newton, R.; Madded, S.; Balakrishnan, H.(2008). The Pothole Patrol: Using a Mobile Sensor Network for Road Surface Monitoring, *Proceedings in ACM MobiSys*.

Benbasat A.& Paradiso, J. (2007).A Framework for the Automated Generation of Power-Efficient Classifiers for Embedded Sensor Nodes, *Proceedins in ACM SenSys*.

Kang, S.; Lee, J.; Jang, H.; Lee, H.; Lee, Y.; Park, S.; Park, T.; Song,J.(2008). SeeMon: Scalable and Energy-efficient Context Monitoring Framework for Sensor-right Mobile Environment, *Proceedings in ACM MobiSys*.

Singh, A.; Ramakrishnan, C.; Ramakrishnan, I.; Warren, D.(2008) A Methodology for In-Network Evaluation of Integrated Logical-Statistical Models, *Proceedings in ACM SenSys*.

Zappi, P.; Lombriser, C.; Steifmeier, T.; Farella, E.; Roggen, D.; Benini, L.;Troster,G. (2008).Activity Recognition from On-Body Sensors: Accuracy-Power Trade-Off by Dynamic Sensor Selection, *Proceedings in EWSN*.

Keally, M.; Zhou, G.; Xing, G.(2010). Watchdog: Confident Event Detection in Heterogeneous Sensor Networks, *Proceedings of IEEE RTAS*.

Keally, M.; Zhou, G; Xing, G.; Wu, J.; Pyles, A. (2011). PBN: Towards Practical Activity Recognition Using Smartphone –Based Body Sensor Networks, *Proceedings of ACM SenSys*.

Upadhyay, R.; Pringle, G.; Beckett, G.; Potter, S.; Han, L.; Welch, S.; Usmani, A.; Torero, J. L. (2009). An Architecture For An Integrated Fire Emergency Response System For The Built Environment, *Fire Safety Science*,9,pp. 427-438.

Eisenman S. B.et. al.(2007). The BikeNet Mobile Sensing System for Cyclist Experience Mapping, *Proceedings of Sensys'07*.

Thiagarajan A.; et. al. (2009). Vtrack: Accurate, Energy-Aware Road Traffic Delay Estimation using Mobile Phones, *Proceedings of Sensys'09*, pp.85-98.

Zeng, Y.; Murphy, S.; Sitanayah, L.; Tabirca, T.; Truong, T.; Brown, K.; Sreenan. C.J. (2009). Building Fire Emergency Detection and Response Using Wireless Sensor Networks, *Proceedings of 9th Information Technology & Telecommunications Conference (IT&T)*.

Rezgui, A. & Eltoweissy, M.(2007) Service-oriented sensor-actuator networks: Promises, challenges, and the road ahead, *Computer Communications*, 30(13), pp.2627-2648.

Chaczko, Z.; Chiu, C.; Aslanzadeh, S.; Dune, T. (2011). Software Infrastructure for Wireless Sensor and Actuator Networks, P*roceedings of 21st International Conference on Systems Engineering (ICSEng)*,pp. 474-479.