Chapter 12
Clustering and Energy Consumption Minimization

12.1 Introduction

A large number of SNs are deployed in a WSN and SNs need to route data to BS. This creates a volume of data at BS and efforts need to be made to reduce the size of data. One effective approach is to explore the use of clustering WSN such that cluster members can send data to selected CH where data are aggregated and a single packet is forwarded by each CH. This leads to drastic reduction in data size.

12.2 Clustering

Topology management protocols for WSNs suffer from at least one of the following problems. It depends on location awareness (e.g., GAF: Geographic Adaptive Fidelity), or converges slowly (i.e., dependent on the network diameter). As energy efficiency is not the main goal of many protocols, e.g., Max–Min clustering maximizes the number of CHs and minimizes distance from each SN to CH and there is no focus on the quality of clustering, such as having CHs well-distributed in the network (e.g., LEACH [1]). For electing CHs, the primary parameter is the residual energy (E_r) with a SN and a secondary parameter being the communication cost that is used to break ties, such that it maximizes energy consumption and minimizes cost. Selecting CH reduces structural complexity in a WSN, and complex/energy consuming activities are delegated to a subset of SNs in the network. It not only reduces routing complexity but reduces wireless interference and preserves network capacity, while maintaining connectivity in a WSN. It also lessens routing complexity, reduces wireless interference, and preserves network capacity.

Besides reducing complexity, topology control approaches used in a WSN involve how to reduce radio power consumption. Most work concentrate on minimizing radio interference and reducing routing complexity while not worrying

D.P. Agrawal, *Embedded Sensor Systems*,
DOI 10.1007/978-981-10-3038-3_12

too much about its capacity. The net effect is routing selectivity is lost and incurs increased topology maintenance overhead. The literature is full of numerous theory/simulation results, and very few experimental results have been explored as algorithms are complicated and underlying assumptions in the algorithm are difficult to realize in practice. Moreover, wireless links usually vary in quality over time as wireless links do not follow binary value (good/bad) in nature and wireless links may be asymmetric. SNs are designed using low-speed CPUs, and it may not be possible to run complex algorithms. So, the overall objective should be to define a clustering approach that is completely distributed, has low message/processing overhead, terminates in O(*l*) iterations with *l* links, creates generates high-energy and well-distributed cluster heads (CHs), and can provide other characteristics such as balanced or dense clusters.

Hybrid, energy-efficient, distributed [2] clustering approach (HEED) (Fig. 12.1a) is proposed that it allows every SN to use information from its 1-hop neighbors only (within cluster range). HEED is distributed and energy-efficient as it

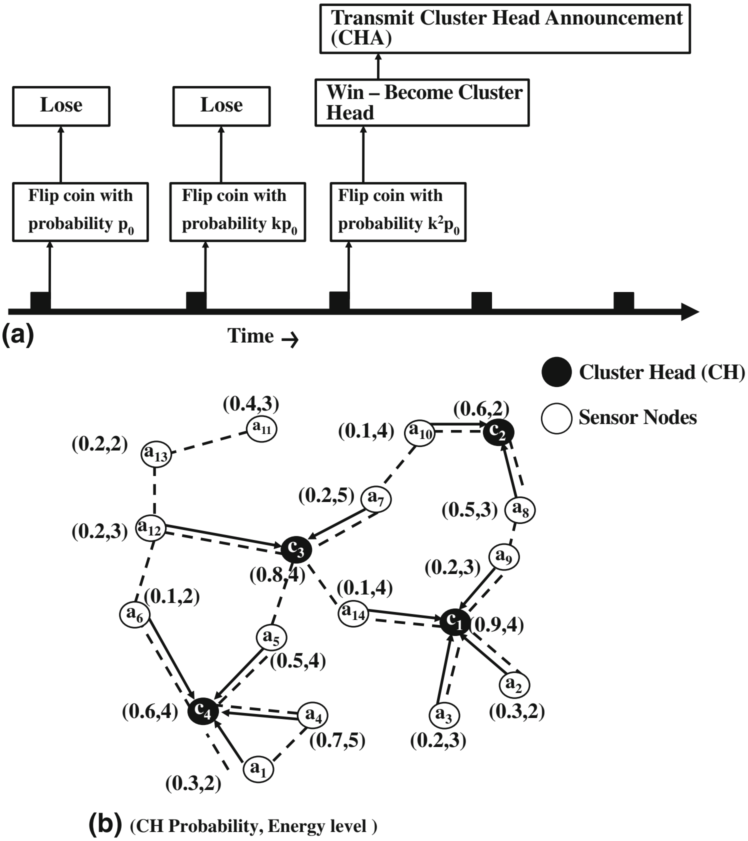


Fig. 12.1 **a** CH election algorithm in HEED, **b** example WSN in electing CH with HEED

elects those SNs as CHs that are rich in residual energy E_r, and reclustering is used to result in distributing energy consumption, with E_{max} being the maximum energy of each SN. The algorithm is implemented in three steps of initialization, main processing, and finalization. In initialization step, neighbors are discovered within the cluster range and initial cluster head probability $CH_{prob} = f(E_r/E_{max})$ is computed as the ratio of residual energy of a SN with maximum energy (Fig. 12.1b). In main processing, if a SN v receives message from few SNs to be a CH, then select one SN as CH that costs minimum energy consumption. If SN v does not have a CH, then elect to become a CH with CH_{prob} and adjust its probability by $CH_{prob} = \min(CH_{prob} * 2, 1)$ and repeat until CH_{prob} reaches 1 and a CH is found. In finalization step, join the CH if found; otherwise, elect SN to be a CH.

Thus, HEED consists of discovering neighbors, computing ($CH_{prob,}$ cost), elect to become CH and resolve ties, and finally select CH. The HEED has the properties like completely distributed, and clustering terminates with iterations as follows:

$$N_{iter} \leq \left\lceil \log_2 \frac{1}{p_{min}} \right\rceil + 1, \tag{12.1}$$

where P_{min} is the minimum node degree of SNs. The processing overhead is O (n) per SN where n is # SNs and the message overhead is O(1) per node = # iterations required. The CHs are well-distributed. P_r and there exists at least one CH in any area of size

$$\left(2 + \frac{1}{\sqrt{2}}\right) r_c \times \left(2 + \frac{1}{\sqrt{2}}\right) r_c. \tag{12.2}$$

HEED produces a connected multi-hop CH graph asymptotically, and its performance comparison for a field of size 100 × 100 with 1000 SNs is shown in Fig. 12.2 [2], with initial energy E_r of 2 J, and each round consists of 5 TDMA frames. So, the next question is how the network density and initial bootstrap parameters change the rate of energy consumption and lifetime of WSN. Clustering scheme has been implemented [3] on a simple simulator without modeling contention or message losses due to collisions. A WSN in an 500 × 500 m area is modeled as a Poisson point process with intensity λ varied from 0.0002 to 0.002 with communication range r_c of 90 m, and initial probability of being a CH in round 0 be denoted as p_0. In the ith round, a SN becomes a CH with probability $k^i p_0$. This leads to each SN having an average of 5–50 neighboring SNs. The number of CHs as a function of SN density is shown in Fig. 12.3a while b illustrates the impact of p_0.

Another clustering scheme [3] performs the sequence of time synchronization, neighbor discovery, cluster selection, gateway selection, and routing as a part of setup phase as shown in Fig. 12.4 and needs to be repeated at regular intervals. The clustering algorithm has been tested on 42 Mica2 SNs test bed [3], with the maximum one-hop neighbors of 15. B-MAC is adopted with DSDV-like table-driven proactive query–response approach. Link-level measurements are

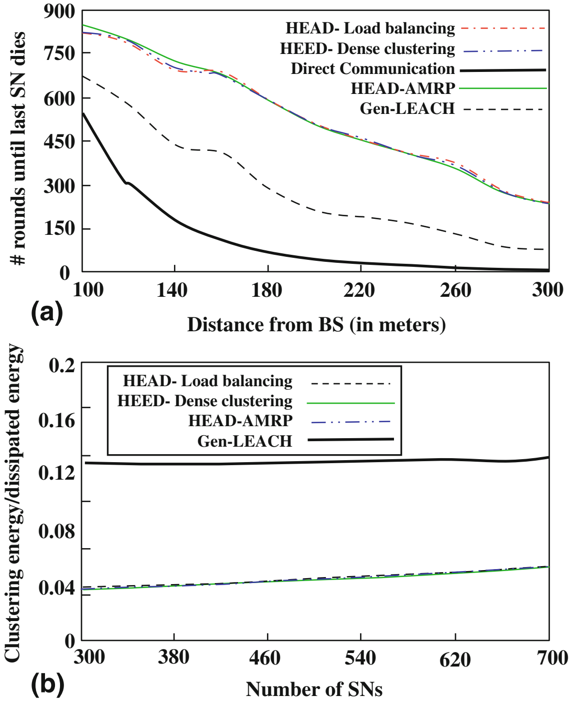


Fig. 12.2 **a** Rounds before SN dies as a function of distance to the BS, **b** energy consumption in forming CH as a function of no. of SNs

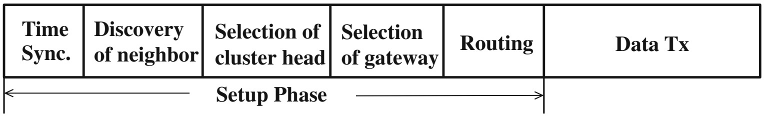


Fig. 12.3 Set-up phase for clustering [3]

used to select routing parents (Fig. 12.5). As CHs consume a lot more energy, it is desirable to recluster occasionally. Using δ as controlling power saving degree, Fig. 12.5c shows the effect of network density and δ on the reclustering period with $p_0 = 0.032$, $k = 2$, and duty cycle $X = 50\%$. To balance energy, frequent reclustering is required as fewer CHs are used here. CHs are tried for varying duty cycles ($X = 2$–45%). Radio is selected as 19.2 Kbps with a packet payload of 36 bytes. Every SN transmits packets with probability α%, and α is varied for two types of scenarios: low data rate experiment, with SN remains idle most of the time with a very brief periods of activity (e.g., earthquake detection) and α = 0.1 to 1; high data rate experiment with larger active periodicity (e.g., temperature monitoring) and α = 10–100. Experimental default parameters are shown in Table 12.1 and clustering overhead is given in Table 12.2. 11 out of 42 SNs are selected as CHs and

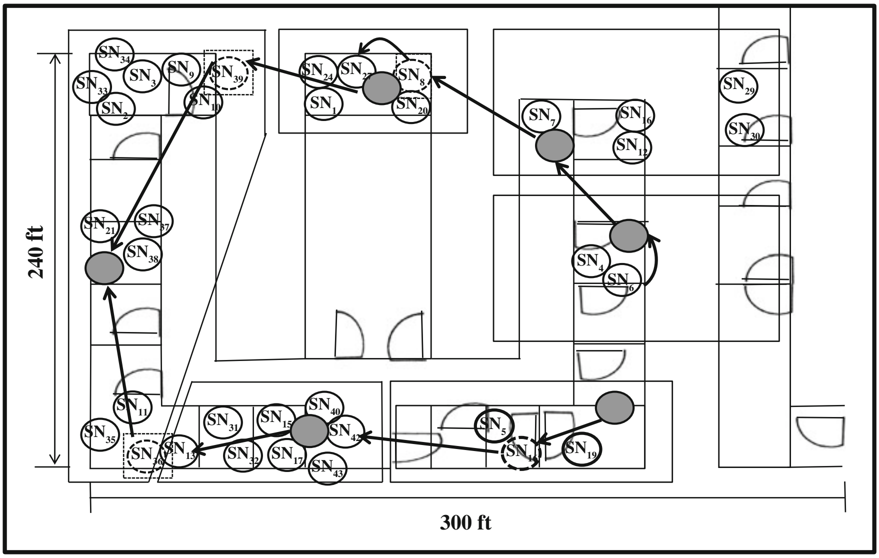


Fig. 12.4 Experimental test bed [3]

achieve roughly 3 times of energy efficiency of B-MAC. About half the overhead is due to routing and clustering consumes about remaining half. With time synchronization of 17 h leads to synchronization error of up to 10 s. During the neighbor discovery phase, SNs periodically broadcast beacon signals to discover neighbors and receiving SNs use this to determine link quality. In cluster selection phase, few SNs adjacent to current CHs are selected as new CHs with probability $k^i p_0$ in the ith round where p_0 is probability of being CH in round 0. When a SN becomes a CH candidate, it broadcasts a *CH Advertisement* (CA) message till the timer expires. In cluster selection phase, SNs select their CHs and two adjacent CHs choose their gateway SN. A proactive routing algorithm is used to determine path to BS via CHs and gateway nodes. Throughput is illustrated in Fig. 12.6; energy efficiency in topology control and routing is illustrated in Fig. 12.7, and the results depend on the values of p_0 and constant k. A parameter δ is used as a controller that reflects power saving desired by the designer, and $\delta = 1$ means all SNs follow the same duty cycle while $\delta = 0$ means all SNs are turned off and only CH maintain a duty cycle.

12.3 Sensor Properties and Resource Constraints

WSNs have lower transmission distances (<10 m), lower bit rates (typically < kbps) and limited battery capacity [4]. Many different SNs have been developed including MIT μAMPS with 59–206-MHz processor, 2 radios, capable

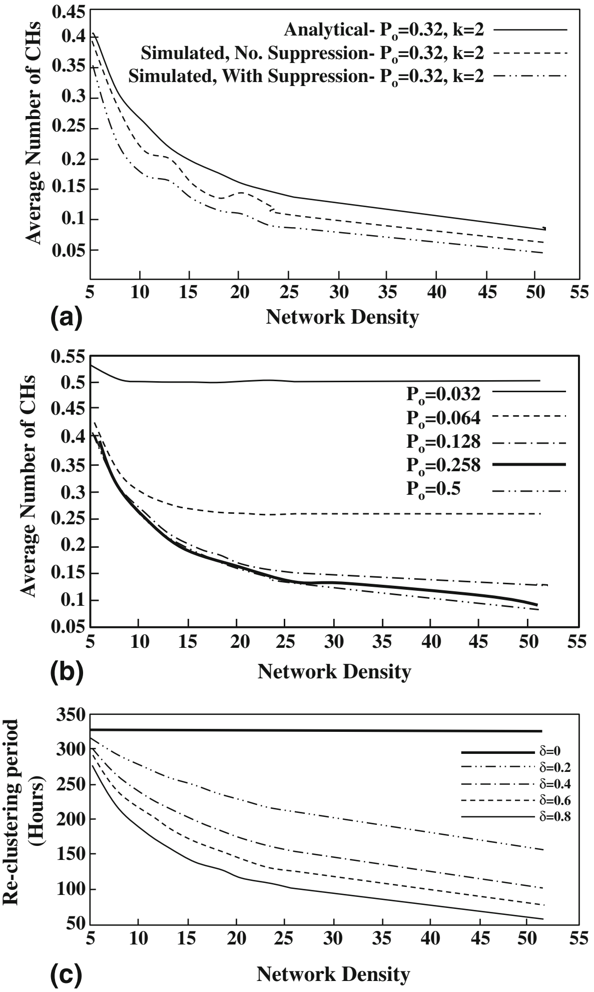


Fig. 12.5 **a** Number of CHs as a function of SN density for $p_0 = 0.032$ and $k = 2$, **b** Impact of p_0 on number of CHs in a WSN, **c** Impact of network density and δ on reclustering period

of transmitting at 1 Mbps, and 4 KB RAM; Berkeley MICA motes with 8-bit, 4-MHz processor; 40-kbit CSMA radio; 4-KB RAM, and TinyOS -based operating system. However, circuit gains are nearing flat as circuit tricks and voltage scaling provided a large part of the gains (Fig. 12.8).

While energy needs functionality, speed, they continue to climb at the rate of 10× increases in gate count every 7 years) and in frequency every 9 years. Speed power efficiency has indeed gone up 10 times every 2.5 years for μPs and DSPs in 1990s and varies from 100 mW/MIP to 1 mW/MIP since 1990. IC processes have provided 10 times improvements every 8 years since 1965. Power consumption is being lowered for a given function and performance, such as DSP is reduced by 1.6 times every year since the early 1980s and most optimistic projections are 60 pJ/op (about 20 times) for a given function and performance (Table 12.3).

Table 12.1 Setting for MICA2 experiments

TinyOS experiment parameters	Default
Initial cluster probability (p0)	0.032
Constant multiplier (k)	2
Cluster head and gateway duty cycle (X)	Varied
Member node duty cycle (δ = 0)	0
Transmission power (full power)	10 dBm
Communication bandwidth (CC1000 radio)	19.2 Kbps
Preamble size	Varied
Payload size	36 bytes
Radio transmission power consumption	60 mW
Radio receiving power consumption	45 mW
Channel sampling power consumption	5.75 mW

Table 12.2 Clustering set up overhead cost

Phase	Average time (s)	Average energy (J)
Time synchronization	10	0.235
Neighbor discovery	20	0.469
Cluster selection	60	1.288
Gateway selection	20	0.429
Routing	120	2.583
Total	230	5.004

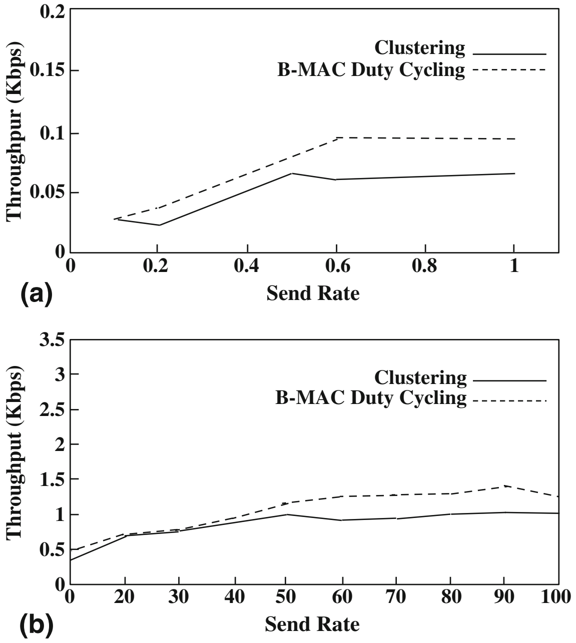


Fig. 12.6 **a** Throughput for low rate SNs, **b** throughput for high rate SNs

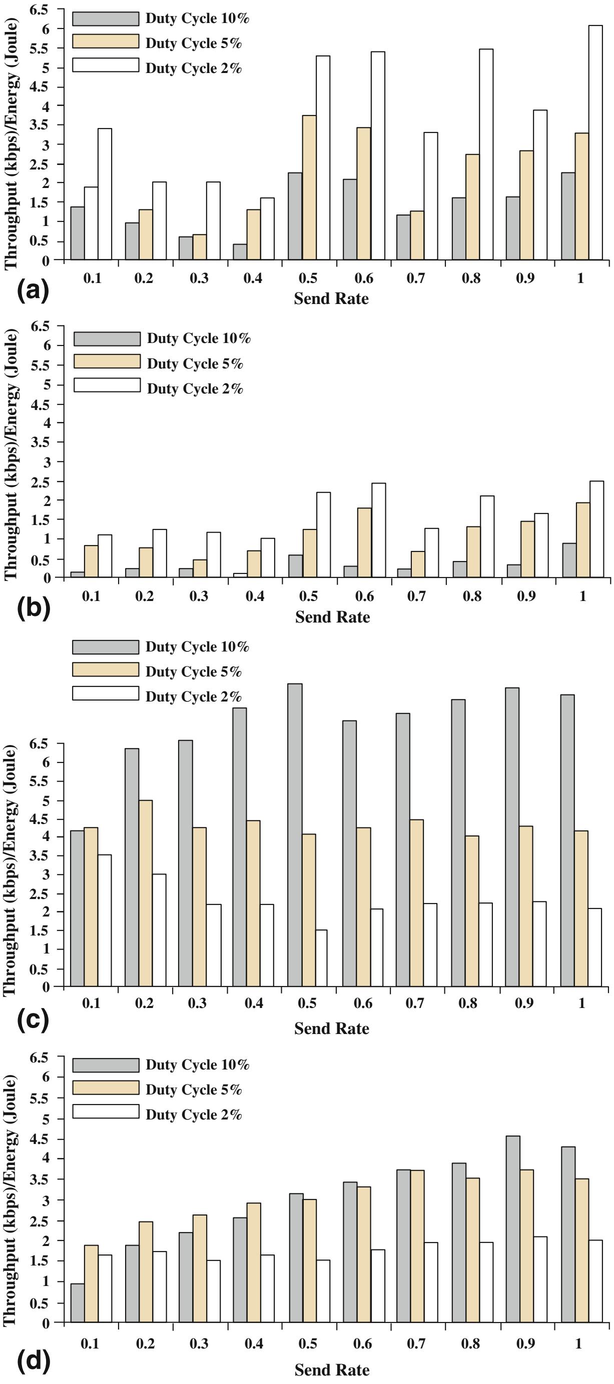


Fig. 12.7 Energy efficiency **a** topology control in low rate, **b** B-MAC in low rate, **c** topology control in high rate, **d** B-MAC in high rate

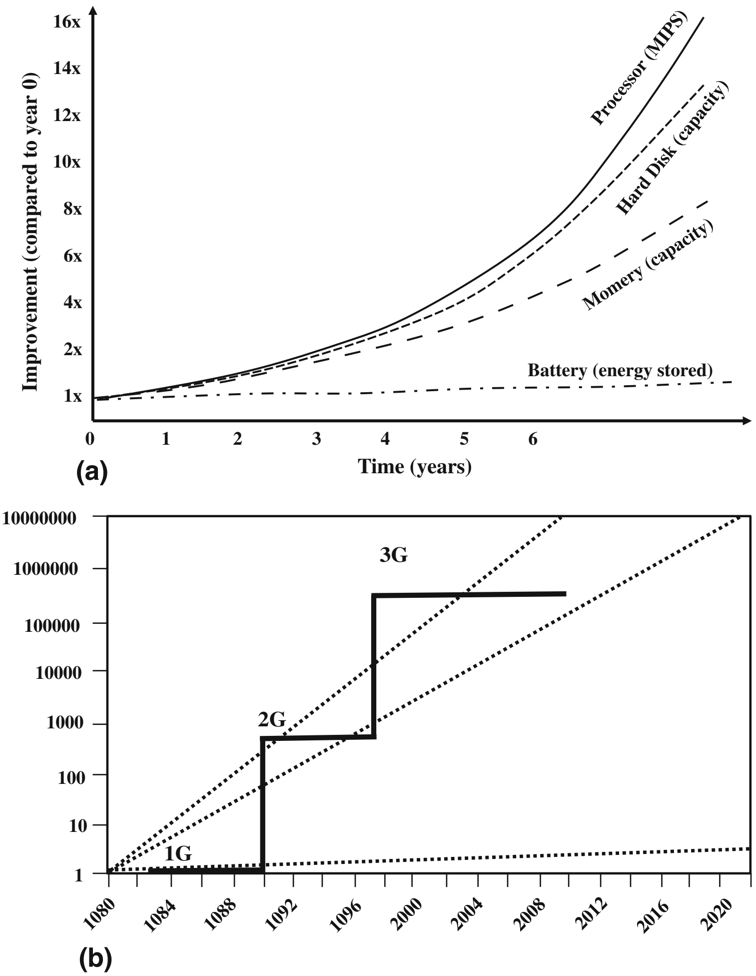


Fig. 12.8 **a** Processor, memory, and battery improvements, **b** processor performance and battery capacity

Table 12.3 Properties of sensor nodes

Radio mode	Power consumption (mw)
Transmit	14.88
Receive	12.50
Idle	12.36
Sleep	0.016

Power efficiency (or energy efficiency) η_P is defined as the ratio of signal energy per bit E_b to noise power spectral density N_0 required at the receiver for a certain BER (bit error rate) and is given by

$$\eta_P = E_b/N_0. \tag{12.3}$$

High power efficiency requires low (E_b/N_0) for a given BER and the bandwidth efficiency

$$\eta_B = \text{bit rate}/\text{bandwidth} = R_b/\text{W bps/hz}. \tag{12.4}$$

The ratio of throughput data rate to bandwidth occupied by the modulated signal typically ranges from 0.33 to 5, and there is a trade-off between the two as for a given BER. Adding FEC reduces $\eta_{B,}$ but reduces required η_P. Modulation schemes with larger number of bits per symbol have higher η_B, but also require higher η_P. Projected computation cost in 2004 is 60 pJ/op, and minimum thermal energy for communications is 20 nJ/bit transmitting for 100 m at a bandwidth of 1.5 GHz is equivalent of 300 ops and 2 nJ/bit at 1.5 GHz for 10 m is equivalent of 0.03 ops (Fig. 12.9a). There exists a clear trade-off between significant processing energy versus communication cost. A reduction in supply voltage decreases speed (Fig. 12.9b) and the supply voltage ought to be reduced when slower speed can be tolerated.

To combat slow operation, another alternative is to use architectural techniques such as concurrency and pipelining via compiler techniques. Parameter of Rockwell WINS SN of Fig. 1.10a is shown in Fig. 12.10b. The processor consumes 360 mW power doing repeated transmit/receive, and transducer takes 23 mW. Shutdown is attractive for many wireless applications due to low duty cycle of many subsystems, and issues to be addressed are the cost of restarting as latency versus power trade-off remains that increases in latency (response time) and power consumption due to start-up. So, the question is, when to shut down for optimal gain and how to select idle time threshold or use some predictive approach and when to wake up for optimal performance or depend on on-demand service or utilize predictive approach? (Fig. 12.11). Two main approaches are reactive versus predictive. In a reactive approach, the SN goes to reduced power mode after the SN has been idle for a few seconds/minutes and restarts on demand. In predictive mode, computation history is used to predict whether $T_{\text{block}}[i]$ is large enough satisfying $T_{\text{block}}[i] \geq T_{\text{cost}}$. It has been observed that it is better to lower voltage than to complete shutdown in case of digital logic. For example, a task with 100 ms deadline requires 50 ms CPU time at full speed. A normal system gives 50 ms computation, 50 ms idle/stopped time. At half speed/voltage system gives 100 ms computation, 0 ms idle and thus requires the same number of CPU cycles but 1/4 in energy consumption. So, the voltage gets dictated by the tightest (critical) timing constraint both on throughput and latency. So, it is better to change voltage dynamically and use voltage to control the operating point on the power versus speed curve. As the power and clock frequency are functions of voltage, the main

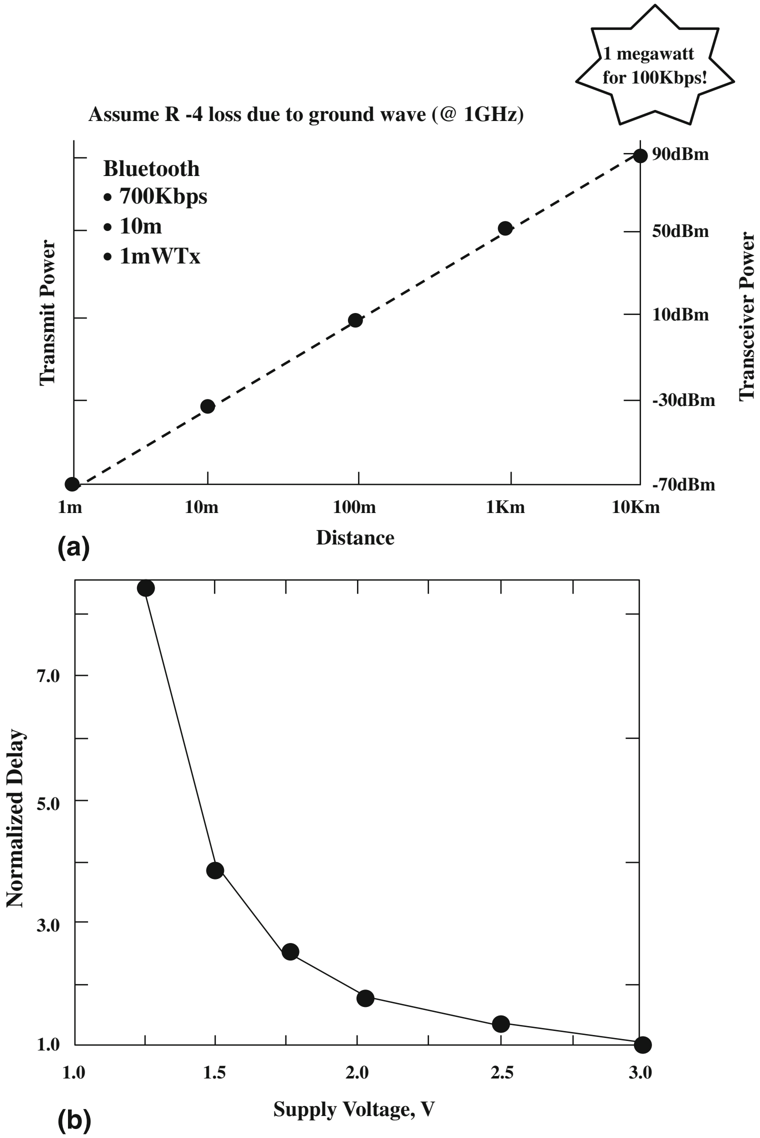
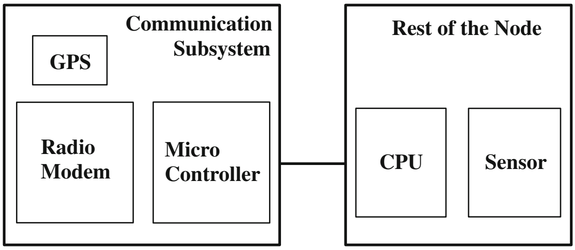


Fig. 12.9 **a** Power consumption as a function of distance, **b** slowdown by reducing supply voltage

challenge is algorithmic and need to schedule the voltage variation as well either through compiler or OS or the hardware (Fig. 12.12).

Advanced Configuration and Power Management Interface (ACPI) allows OS/drivers of SN to be in sync regarding power states, and a standard way is used for the system to describe its device configuration and power control hardware



(a)

Processor	Seismic Sensor	Radio	Power (mW)
Active	On	Rx	751.6
Active	On	Idle	727.5
Active	On	Sleep	416.3
Active	On	Removed	383.3
Active	On	Removed	360.0
Active	On	TX (36.3 mW)	1080.5
		TX (27.5 mW)	1033.3
		TX (19.1mW)	986.0
		TX (13.8 mW)	942.6
		TX (10.0 mW)	910.9
		TX (3.47 mW)	815.5
		TX (2.51 mW)	807.5
		TX (1.78 mW)	799.5
		TX (1.32 mW)	791.5
		TX (0.955 mW)	787.5
		TX (0.437 mW)	775.5
		TX (0.302 mW)	773.9
		TX (0.229 mW)	772.7
		TX (0.158 mW)	771.5
		TX (0.117 mW)	771.1

(b)

Fig. 12.10 **a** Power measurements on Rockwell WINS SN, **b** parameters of Rockwell WINS SN

interface to the OS. Common functions are registered through interface and the system controls the events, processor power and clock control, and thermal management. Information on devices, resources, and control mechanisms is described using tables (actually linked to a table of tables) and power management capabilities

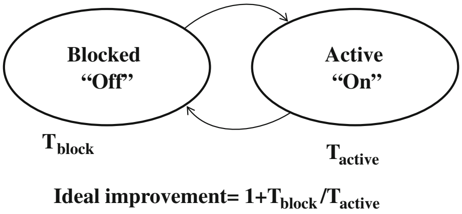


Fig. 12.11 Blocked and active state of Rockwell WINS node

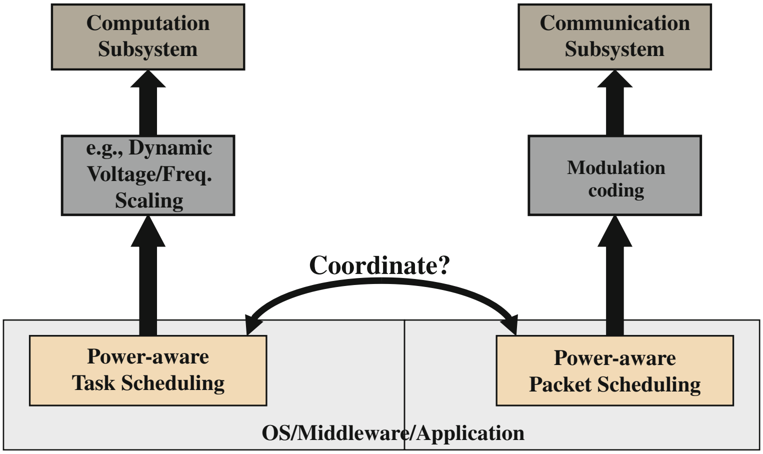


Fig. 12.12 Communication subsystem of Rockwell WINS node

and requirements for each device are described. There are specific methods for setting and getting the power state, and hardware resource settings are managed by appropriate methods. Thus, ways are provided for application, operating system, and hardware to efficiently exchange energy/power and performance related information. This facilitates a continuous dialogue/adaptation between OS and applications, and power-aware OS services are implemented by providing a software interface to low-power devices. A power-aware API to the end user enables one to implement energy-efficient RTOS services and applications. The applications’ interface enables the application to pass on RT (round trip) information to OS (period, deadlines), WCET (worst-case execution time), hardness by creating new threads. It also predicts when OS time is expected to finish a given task depending on the conditions of the environment (application dependent and not yet implemented). OS is also able to predict and tell applications the time estimated to finish the task depending on the scheduling scheme used, and a task with hard deadline must be killed if its deadline is missed.

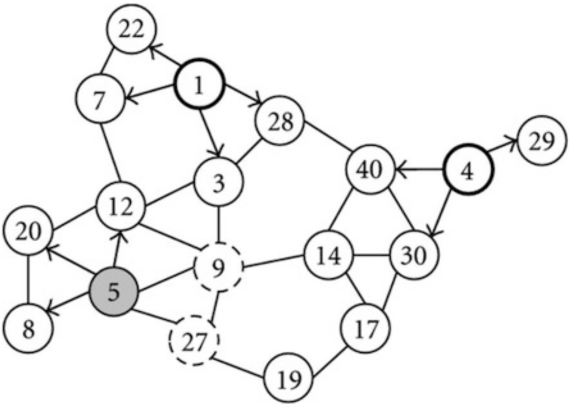
12.4 Conclusions

There are many issues critical to WSNs, and clustering is very useful, especially for randomly deployed WSNs. The question has been addressed as to how to do it effectively with minimal information about neighboring SNs. What is the impact of multi-hop communication and how to keep the process distributed in nature. Reducing supply voltage could be another effective solution in order to minimize energy consumption.

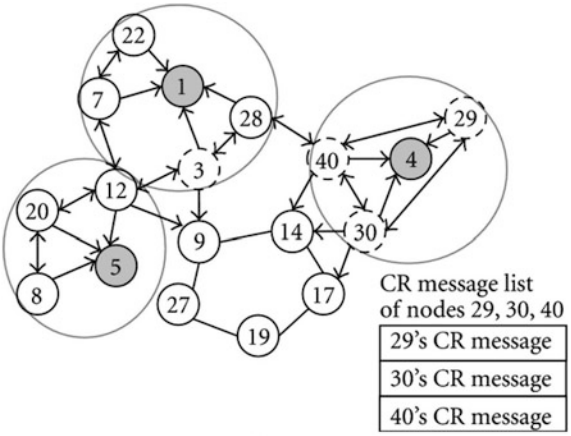
12.5 Questions

Q.12.1. What is an optimal cluster sizes for wireless sensor networks?

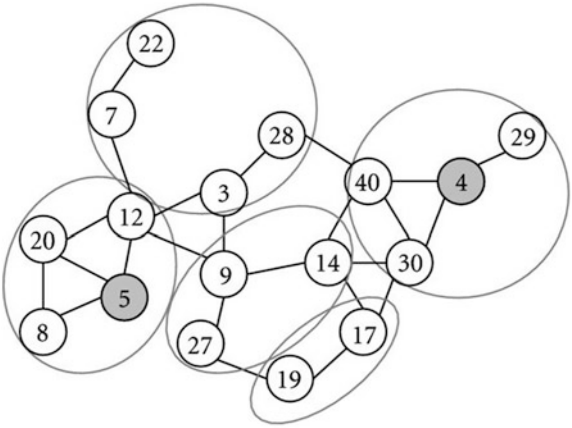
Q.12.2. In the following examples of WSNs, how many clusters will be formed using (i) LEACH, (ii) HEED, and (iii) topology control schemes?



Prob. 12.2(i): Example WSN



Prob. 12.2(ii): Example WSN



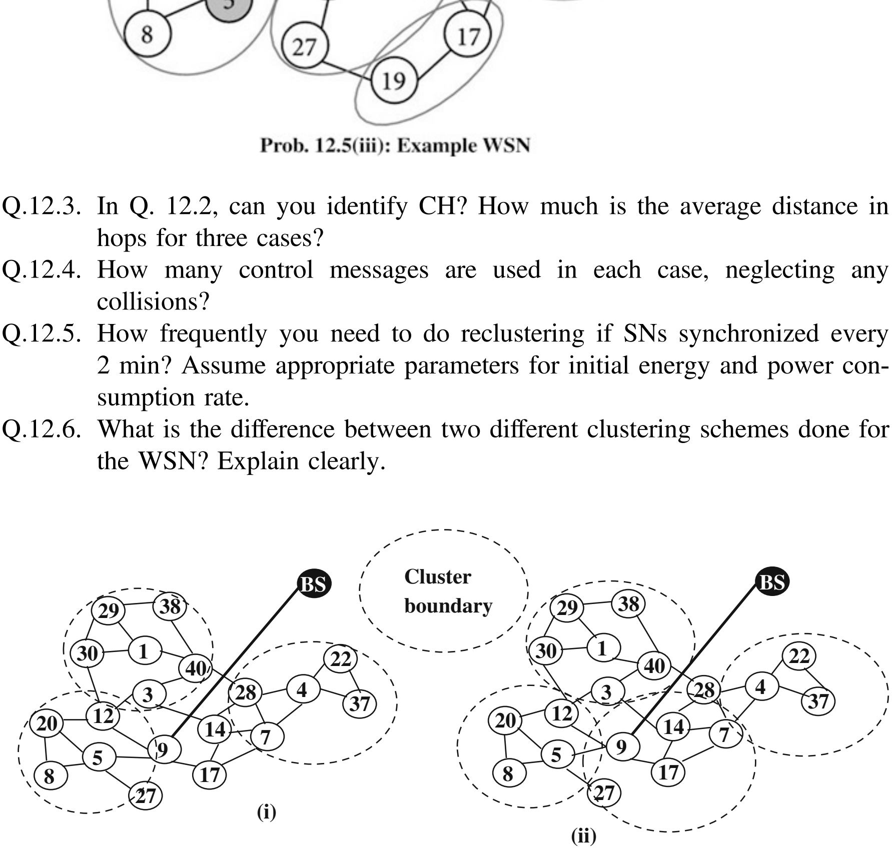
Prob. 12.5(iii): Example WSN

Q.12.3. In Q. 12.2, can you identify CH? How much is the average distance in hops for three cases?

Q.12.4. How many control messages are used in each case, neglecting any collisions?

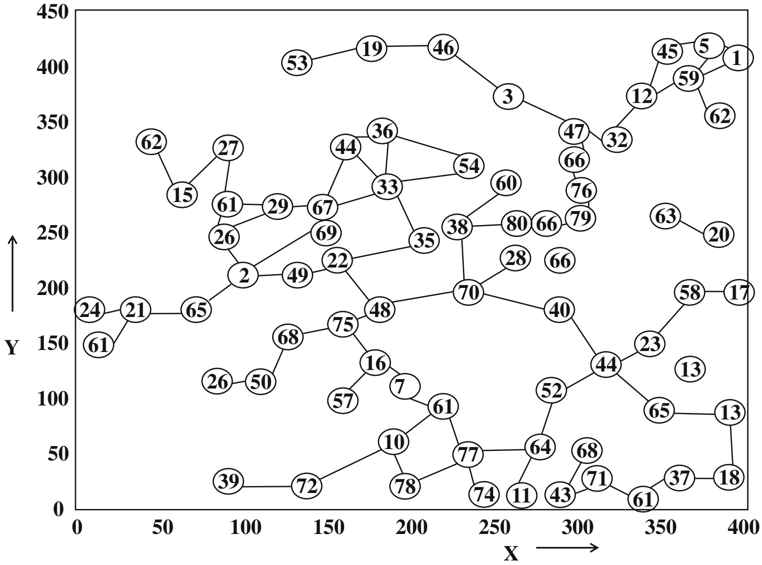
Q.12.5. How frequently you need to do reclustering if SNs synchronized every 2 min? Assume appropriate parameters for initial energy and power consumption rate.

Q.12.6. What is the difference between two different clustering schemes done for the WSN? Explain clearly.



Q.12.6

Q.12.7. In the following WSN, how can you group SNs to form clusters? Explain your answer carefully.



References

1. Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," Proceedings of the 33rd Hawaii International Conference on System Sciences—2000, https://pdos.csail.mit.edu/archive/decouto/papers/heinzelman00.pdf.
2. O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," IEEE Transactions on Mobile Computing, vol. 3, no. 4, pp. 366–379, Oct-Dec. 2004.
3. Ajit Warrier, Sangjoon Park, Jeongki Min, and Injong Rhee, "Impact of Topology Control on Energy Saving through Differential Duty Cycling," http://www4.ncsu.edu/~rhee/export/topology.pdf.
4. http://www.ics.uci.edu/~quasar/tutorial/hipc.ppt.