A Balanced Energy Consumption Sleep Scheduling Algorithm in Wireless Sensor Networks

Zhuxiu Yuan[1], Lei Wang[1], Lei Shu[2], Takahiro Hara[2], Zhenquan Qin[1]

[1]School of Software, Dalian University of Technology, China
[2]Dept. Multimedia Engineering, Osaka University, Japan
[1]zhuxiu.yuan@gmail.com, [2]lei.shu@ieee.org, [1]lei.wang@ieee.org, [2]hara@ist.osaka-u.ac.jp, [1]zhqqin33@gmail.com

***Abstract*—Network lifetime is one of the most critical issues for wireless sensor networks (WSNs) since most sensors are equipped with non-rechargeable batteries with limited energy. To prolong the lifetime of a WSN, one common approach is to dynamically schedule sensors' active/sleep cycles (i.e., duty cycles) with sleep scheduling algorithm. In this paper, we propose a new sleep scheduling algorithm, named EC-CKN (Energy Consumed uniformly-Connected *K*-Neighborhood) algorithm, to prolong the network lifetime. The algorithm EC-CKN, which takes the nodes' residual energy information as the parameter to decide whether a node to be active or sleep, not only can achieve the $k-$connected neighborhoods problem, but also can assure the k awake neighbor nodes have more residual energy than other neighbor nodes at the current epoch. Based on the algorithm EC-CKN, we can obtain the state transition probability at the n'th epoch, and upper bound and lower bound of the network lifetime by Markov chain and Markov decision chain.**

***Index Terms*—Network lifetime; Energy consumption; Connected k-neighborhood problem; Duty cycle; Sleep scheduling**

I. INTRODUCTION

Wireless sensor networks (WSNs) are normally powered by batteries with limited energy, which are difficult or impossible to be recharged or replaced. A common approach for saving the sensor nodes' energy is to select a subset of nodes to remain active and let others go to sleep in a given epoch. CONNECTED K-NEIGHBORHOOD (CKN), described exactly in Appendix, is a distributed sleep scheduling algorithm proposed in [2] which can reduce the number of active nodes efficiently, keep the network k-connected, and optimize the geographic routing performance. Supporting geographic routing performance But CKN algorithm cannot ensure the network energy is uniformly consumed.

Motivated by the above issue, in this paper, we address the question: *How do we design a new sleep scheduling algorithm based on CKN that can balance the energy consumption to prolong network lifetime further?* Satisfying all those requirements that CKN algorithm holds with a new decentralized sleep scheduling algorithm is challenge. In the light of the discussions for the question, we proposed a new sleep scheduling algorithm, named EC-CKN, to prolong the network lifetime. In EC-CKN algorithm, the major difference from CKN is that it takes the nodes' residual energy information as the parameter to decide whether a node to be active or sleep. EC-CKN algorithm inherits all the major properties of CKN algorithm, e.g., achieving the $k-$connected neighborhoods problem. Meanwhile, it also makes a significant new contribution by assuring the k active neighbor nodes have more residual energy than other neighbor nodes at the current epoch. A theoretical analysis on the energy consumption of the new EC-CKN algorithm is given to show the correctness of the new contribution.

This paper is organized as follows. Section II shows the network model. Section III presents EC-CKN algorithm. Section IV analyzes the energy consumption of EC-CKN. Section V shows the simulation results about CKN and EC-CKN. Finally, we conclude this paper in Section VI.

978-1-4244-9538-2/11/$26.00 ©

II. NETWORK MODEL

A. Communication Network Model

A multihop WSN is modeled by a graph $G = (S, E)$, where the set $S = \{s_1, s_2, \cdots, s_N\}$ denotes the set of $N = |S|$ sensor nodes in the network, and a directed link $(s_i, s_j) \in E$ iff these two nodes s_i and s_j can communicate with each other directly without relaying. s_i is the sender and s_j is the receiver of link (s_i, s_j). s_j is the neighbor of s_i. And s_j is the 2-hop neighbor of the node s_i, if $(s_i, s_j) \notin E$ and there exits another node s_r satisfying (s_i, s_r) and $(s_r, s_j) \in E$ or (s_j, s_r) and $(s_r, s_i) \in E$. We will use $l_{i,j}$ instead of the link notation (s_i, s_j) in the following sections. Each node s_i has the transmission radius t_{ri}, and the interference radius f_{ri}. Assume the set S of sensor nodes are deployed with a homogeneous Poisson process in a two dimensions plane $\mathcal{A}$. Each wireless sensor node is only equipped with a single radio interface, and has the same initial energy E_{init}. Time is divided into epoches, and each epoch is T. In each epoch, the node will first transmit packets, and then run the sleep schedular to decide the state of the next epoch as shown in Fig 1. Suppose all packets are the same size and the transmission time of a packet is t. The ***network lifetime*** is defined as the average time that sensor nodes in the network run out its energy from the beginning

B. Energy Model

Our energy model of sensor nodes is based on the first order radio model [5] where the radio dissipates E_{elec} to power

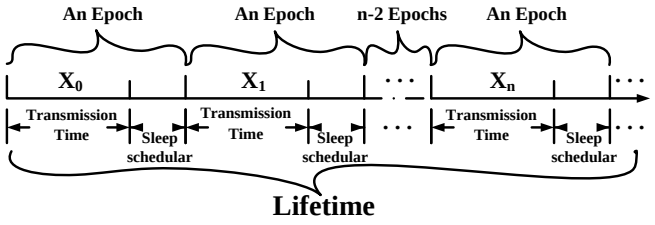


Fig. 1. A node s_u's lifetime consists of many epoches. Each epoch includes packet transmission time and sleep scheduling algorithm's execution time.

the transmitter or receiver circuitry, and ϵ_{amp} for the transmit amplifier, while the energy loss is due to channel transmission. The consumed energy to transmit a l-bit message over distance d is $E_T(l,d)$:

$$E_T(l,d) = E_{elec} \cdot l + \epsilon_{amp} \cdot l \cdot d^2 \quad (1)$$

and the consumed energy to receive this message is $E_R(l)$:

$$E_R(l) = E_{elec} \cdot l. \quad (2)$$

And E_I, the energy consumed by nodes with the radio in the idle mode, approximates to the radio in the receiving model [6].

III. Energy Consumed Uniformly Connected k-Neighborhood Algorithm

We develop a new sleep scheduling algorithm to prolong the network lifetime further. Meanwhile, it can still achieve all properties of CKN.

A scalable distributed solution to EC-CKN problem based on node's current residual energy information is challenging for several reasons.

ALGORITHM 1. ENERGY CONSUMED UNIFORMLY-CKN (EC-CKN)
(Run the following at each node s_u *)*

1. Get the information of current remaining energy $Erank_u$;
2. Broadcast $Erank_u$ and receive the energy ranks of its currently awake neighbors N_u. Let R_u be the set of these ranks.
3. Broadcast R_u and receive R_v from each $s_v \in N_u$.
4. If $|N_u| < k$ or $|N_v| < k$ for any $s_v \in N_v$, remain awake. Return.
5. Compute $E_u = \{s_v | s_v \in N_u \text{ and } Erank_v > Erank_u\}$;
6. Go to sleep if both the following conditions hold. Remain awake otherwise.
   - Any two nodes in E_u are connected either directly themselves or indirectly through nodes which is in the s_u's 2-hop neighborhood that have $Erank_v$ larger than $Erank_u$;
   - Any node in N_u has at least k neighbors from E_u.
7. Return.

We address these challenges by proposing the new algorithm EC-CKN. The pseudo-code above depicts EC-CKN Algorithm, which is repeated at each scheduling epoch on each node. The algorithm takes an input parameter K, the required minimum number of awake neighbors per node. In EC-CKN, a node s_u broadcasts its current residual energy information $Erank_u$ (Step 1) and computes a subset E_u of neighbors having $Erank > Erank_u$ (Step 5). Before s_u can go to sleep it makes sure that all nodes in E_u are connected by nodes with $Erank > Erank_u$ and each of its neighbors has at least k neighbors from E_u (Step 6). These requirements ensure that if a node has less than k neighbors, none of its neighbors goes to sleep and if it has more than k neighbors, at least k neighbors of them decide to remain awake. Note that these requirements are easy to keep by computing locally with 2-hop neighborhood information. The current residual energy is exchanged in Steps 2 and 3.

IV. Analysis of EC-CKN Algorithm

For the algorithm EC-CKN, each node could have four states: InitState, AwakeState, SleepState and DeadState. Let $SS = \{\text{InitState} = 0, \text{AwakeState} = 1, \text{SleepState} = 2, \text{DeadState} = 3\}$ be the set of node's states, and $m = |SS|$ is the capacity of the states. Nodes can turn into AwakeState and SleepState from InitState, AwakeState and SleepState respectively. And DeadState can be only transformed from AwakeState and SleepState. Fig 2 shows the Markov state transition probability graph in the algorithm EC-CKN, in which nodes are the states of the nodes and the weights of edges are the transition probability between two states at the n'th epoch.

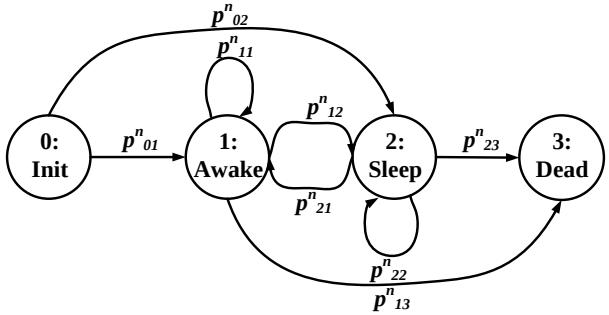


Fig. 2. The Markov state transition probability graph of the EC-CKN algorithm

Notations used in this section: N_u is the set of s_u's neighbors, and $N_u^{'}$ is a set of s_u's 2-hop neighbors. E_u and $E_u^{'}$ are the subsets of N_u and $N_u^{'}$ having $Erank > Erank_u$. $|N_u|$, $|N_u^{'}|$, $|E_u|$ and $|E_u^{'}|$ are the number of the elements in N_u, $N_u^{'}$, E_u and $E_u^{'}$, respectively.

Theorem 1: The network lifetime in EC-CKN increases when the ratio of the size of network (N) and k, N/k increases.

Proof: The maximum network lifetime in computed by the lifetime of the node s_u which is the last one to be dead. Suppose that s_u consumes the equal energy, e, during each epoch when it is awake. Then the lifetime of s_u can be expressed as:

$$L_{s_u} = \frac{\pi E_{init}(t_{ru})^2}{e\mathcal{A}} \cdot \frac{N}{k}$$

Therefore, we can find that the network lifetime in EC-CKN increases when N/k ratio increases. ■

Theorem 2: The sleep probability of a node s_u under EC-CKN algorithm is

$$P_{sleep}(|E_u|) = Prob_1 \cdot Prob_2, \tag{3}$$

where $Prob_1$ is defined in Eq (5) and $Prob_2$ is defined in Eq (6).

And the awake probability of s_u is

$$P_{awake}(|E_u|) = 1 - P_{sleep}(|E_u|). \tag{4}$$

Proof: The two conditions which decide s_u whether to sleep or not when s_u and its neighbors all have at least k neighbors can be interpreted as the following corresponding conditions: (1) the graph $G_{E_u^{'}}$ composed of the nodes and potential links in $E_u^{'}$ is connected and (2) the graph G_{E_u} composed by the nodes and potential links in E_u is k-connected and each node in the set $N_u - E_u$ has at k-neighbors in E_u.

The probability that the graph $G_{E_u^{'}}$ is k-connected is

$$\begin{aligned} Prob_1 \leq & Prob(G_{E_u^{'}} \ is \ connected) \\ \leq & P(|N_u|_{min} > 0) \\ = & P(|N_u| \geq 1). \end{aligned} \tag{5}$$

$|N_u|_{min}$ is the minimum degree in the graph $G_{E_u^{'}}$. And

$$P(|N_u|_{min} \geq n_0) = (1 - \sum_{n=0}^{n_0-1} \frac{(\rho_1 \pi {t_r}^2)^n}{n!} \cdot e^{-\rho_1 \pi {t_r}^2})^{|E_u^{'}|},$$

where $\rho_1 = \frac{|E_u^{'}|}{4\pi {t_r}^2}$ is the node density in graph $G_{E_u^{'}}$.

And then, the probability of the condition (2) is

$$\begin{aligned} Prob_2 = & Prob(|E_u| \geq k+1) \cdot Prob(G_{E_u}^k) \cdot \\ & Prob({E_u}^k_{N_u - E_u}) \end{aligned} \tag{6}$$

where $Prob(G_{E_u}^k)$ the probability that graph G_{E_u} is k-connected can be expressed as:

$$Prob(G_{E_u}^k) \leq (1 - \sum_{n=0}^{k-1} \frac{(\rho_2 \pi {t_r}^2)^n}{n!} \cdot e^{-\rho_2 \pi {t_r}^2})^{|E_u|},$$

and the probability that the node $v \in (N_u - E_u)$ has at least k neighbors in E_u is:

$$Prob({E_u}^k_{N_u - E_u}) = (1 - \sum_{n=0}^{k-1} \frac{(\rho_3 \pi {t_r}^2)^n}{n!} \cdot e^{-\rho_3 \pi {t_r}^2})^{|N_u - E_u|}.$$

where $\rho_2 = \frac{|E_u|}{\pi {t_r}^2}$, $\rho_3 = \frac{|E_u|+1}{\pi {t_r}^2}$ and $|N_u - E_u|$ is the number of the elements in the set $N_u - E_u$. ■

Theorem 3: For the nonuniform energy consumption network, the probability from state i to state j at the n'th epoch time is:

$$p_{ij}(n) = \begin{cases} P_{awake}(|E_u^{n-1}|) & j = 1; \\ P_{sleep}(|E_u^{n-1}|) & j = 2. \end{cases} \tag{7}$$

where $i, j \in 1, 2, 3, 4$ and

$$|E_u^{n-1}| = \begin{cases} \frac{|N_u| \sum_i^m \sum_j^m p_{0i}^{n-1} p_{0j}^{n-2} (D_{n-2} + d_{n-1})}{\max(D_{uv}^{n-2} + d_{n-2}) - \min(D_{uv}^{n-2} + d_{n-2})} & if \ i, j = 1, 2; \\ \frac{|N_u| \sum_i^m \sum_j^m p_{0i}^{n-1} p_{0j}^{n-2} D_{n-2}}{\max(D_{uv}^{n-2} + d_{n-2}) - \min(D_{uv}^{n-2} + d_{n-2})} & otherwise. \end{cases} \tag{8}$$

$d_n = \alpha e(w_u^n, T_2) - \beta e(w_v^n, T_2)$ is the energy consumption difference between the node s_u and its neighbor s_v, if $X_u^n = 0$ or $1, \alpha = 1$, and if $X_u^n = 3, \alpha = 0$, which is the same with β.

Proof: Let $X_u = \{X_u^0, X_u^1, \cdots, X_u^n\}$ denote the first $n+1$ states of s_u, $X_u^n \in \{0, 1, 2, 3\}$ and the first state is InitState, that is, $X_u^0 = 0$. $W_u = \{w_u^0, w_u^1, \cdots, w_u^n\}$ is the set of the transmission saturations, and w_u^n denotes the transmission saturation in the n'th epoch. If $X_u^n = 2$, $w_u^n = 0$.

According to the algorithm EC-CKN, if the node s_u has less than k neighbors or any s_u's neighbors has less than k neighbors, s_u keeps AwakeState. s_u could go to sleep iff (i) any two nodes in E_u are connected either directly or indirectly through s_u's 2-hop neighborhoods that have $Erank > Erank_u$, and (ii) any node in N_u has at least k neighbors from E_u. The sleep probability in the n'th epoch is $P_{sleep}(|E_u^{n-1}|)$ derived from Eq (3) and the awake probability is $P_{awake}(|E_u^{n-1}|)$ derived from Eq (4).

Therefore, s_u's state probability (the sleep probability and the awake probability) is determined by $|E_u^n|$. In order to compute the $|E_u^n|$, we introduce a new transition chain firstly.

$$D_{uv} : D_{uv}^0, d_0, D_{uv}^1, d_1, \cdots, D_{uv}^n, d_n \tag{9}$$

where $s_v \in N_u$. D_{uv}^n is the difference of current residual energy between s_u and s_v, that is, $D_{uv}^n = Erank_u^n - Erank_v^n$. $d_n = \alpha e(w_u^n, T_2) - \beta e(w_v^n, T_2)$ is the difference of the energy consumption in the n'th epoch, where the factors α and β depend on the state of node s_u and s_v. If $X_u^n = 0$ or $X_u^n = 1$, $\alpha = 1$, and if $X_u^n = 3$, $\alpha = 0$, which is the same with β. Then, $D_{uv}^n = D_{uv}^{n-1} + d_{n-1}$.

According to the chain (9), we could compute the probability that s_u's neighbors have larger residual energy than s_u's. The expected residual energy difference $\mathcal{E}(D_{uv}^n)$ in the n'th epoch can be interpreted as:

$$\mathcal{E}(D_{uv}^n) = \begin{cases} \sum_i^m \sum_j^m p_{0i}^{n-1} p_{0j}^{n-1} (D_{uv}^{n-1} + d_n) & if \ i, j = 1, 2, \\ \sum_i^m \sum_j^m p_{0i}^{n-1} p_{0j}^{n-1} D_{uv}^{n-1} & otherwise. \end{cases} \tag{10}$$

And then, the average probability that s_u has the less residual energy than its neighbors' at the n'th epoch is:

$$p_{E_v > E_u}^n = \frac{\mathcal{E}(D_{uv}^n)}{\max(D_{uv}^{n-1} + d_{n-1}) - \min(D_{uv}^{n-1} + d_{n-1})}, \tag{11}$$

And,

$$|E_u^n| = p_{E_v > E_u}^n \cdot |N_u| \tag{12}$$

Then, the probability from the state i to state j at the n'th epoch is

$$p_{ij}(n) = \begin{cases} P_{awake}(|E_u^{n-1}|) & j = 1; \\ P_{sleep}(|E_u^{n-1}|) & j = 2. \end{cases} \tag{13}$$

■

As we know, each node has the same initial energy $D_{uv}^0 = 0$. Each node will choose a random number in $[0,1]$ as w_u^n. Therefore, the set $w^0 = \{w_1^0, w_2^0, \cdots, w_N^0\}$ subjects to the random distribution on $[0,1]$, where N is the total number of the nodes in the network. So $p_{E_u<E_v}^0 = 1/2$.

$$|E_u^0| = \frac{1}{2}|N_u| \tag{14}$$

The initial state transition probability matrix during the first EC-CKN is

$$p_{ij}(0) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ P_{awake}(|E_u^0|) & 0 & 0 & 0 \\ P_{sleep}(|E_u^0|) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{15}$$

According to the above results, the probability from the state i to state j at the n'th epoch depends on the value of the $|E_u^{n-1}|$. From Eq (10), we can figure out $|E_u^{n-1}|$ involved in $p_{0,i}^{n-1}$. And according to the *Chapman-Kolmogorov Equation* for the n'Step transition probabilities, $p_{ij}^n = \sum_k^m p_{ij}^{n-1} p_{kj}(n)$. Therefore, the computation of $p_{ij}(n)$ is recursive, and can be obtained from the $p_{ij}(0)$.

Theorem 4: Under EC-CKN, the upper bound network lifetime L_{EC-CKN}^{upper} is

$$L_{EC-CKN}^{upper} = \frac{E_{init}}{\min \sum_i \sum_a \sigma_{ia} e(w_u^n, T_2)}, \tag{16}$$

where σ_{ia} is the steady-state probability that the chain will be in state i and action a will be chosen under the policy Φ, and $e(w_u^n, T_2)$ is the energy consumption of the node s_u in the time T_2.

Proof: Now we construct a Markov State Decision chain for each node s_u:

$$X_u^0, a_0, X_u^1, a_1, \cdots, X_u^n, a_n,$$

where $X_u^n \in \{0,1,2,3\}$ denotes the state at the n'th epoch, and $a_n \in A$ is the action under the state X_u^n. $A = \{a_1^n, a_2^n\}$ is the set of the possible actions.

$$a_n = \begin{cases} a_1^n = e(w_u^n, T_2), & for\ \ X_u^n = 0\ \ or\ \ 1; \\ a_2^n = 0, & for\ \ X_u^n = 3. \end{cases} \tag{17}$$

The policy Φ is a set of numbers $\Phi = \{\phi_{u,i}(a), a \in A, i = 0, \cdots, m\}$ with the interpretation that if the chain of s_u is in state i, then action a is to be chosen with probability $\phi_i(a)$.

$$\begin{cases} 0 \le \phi_{u,i}(a) \le 1, & for\ all\ i, a \\ \sum_a \phi_{u,i}(a) = 1, & for\ all\ i \end{cases} \tag{18}$$

Under the policy Φ, the sequence of states X_u^n where $n = 0, 1, \cdots$, constitutes a Markov chain with transition probability $p_{ij,u}(\Phi, n)$ given by

$$\begin{aligned} p_{ij,u}(\Phi, n) &= p_{\Phi,u}\{X_u^{n+1} = j | X_u^n = i\} \\ &= \sum_a p_{ij,u}(n) \phi_{i,u}(a) \end{aligned}$$

where the last equality follows by conditioning on the action chosen when the state is i. And for the policy Φ, let $\sigma_{i,a}$ denote the steady-state probability that the chain will be in state i and action a will be chosen when the policy Φ is employed.

$$\sigma_{ia} = \lim_{n\to\infty} P_\Phi\{X_u^n = i, a_n = a\}$$

The vector $\sigma = (\sigma_{ia})$ satisfy:

$$\begin{array}{ll} (i) & \sigma_{ia} \ge 0 \qquad\qquad\qquad\qquad for\ all\ i, a \\ (ii) & \sum_i \sum_a \sigma_{ia} = 1 \\ (iii) & \sum_a \sigma_{ja} = \sum_i \sum_a \sigma_{ia} p_{ij}(a) \quad for\ all\ j \end{array} \tag{19}$$

Equations (i) and (ii) are obvious, and Equation (iii) follows as the left-hand side equals the steady-state probability of being in the state j and the right-hand side is the same probability computed by conditioning on the state and action chosen one stage earlier.

Suppose that a reward $R(Y_i^n, a_i^n) = R(Y_i, a) = a$ is earned whenever action a_i^n is chosen in state i at the n'th epoch. Since $R(Y_i^n, a_i^n)$ would represent the reward earned at time n, the expected average reward per epoch under Φ can be expressed as:

$$\begin{aligned} \mathcal{E}(\Phi) &= \lim_{n\to\infty} \mathcal{E}_\Phi[\frac{\sum_n \sum_{i=0}^m R(Y_i^n, a_i^n)}{m}] \\ &= \sum_i \sum_a \sigma_{ia_i^n} R(Y_i^n, a_i^n) \\ &= \sum_i \sum_a \sigma_{ia_i} R(Y_i, a) \end{aligned} \tag{20}$$

Therefore, $\mathcal{E}(\Phi)$ can be interpreted as the following linear program:

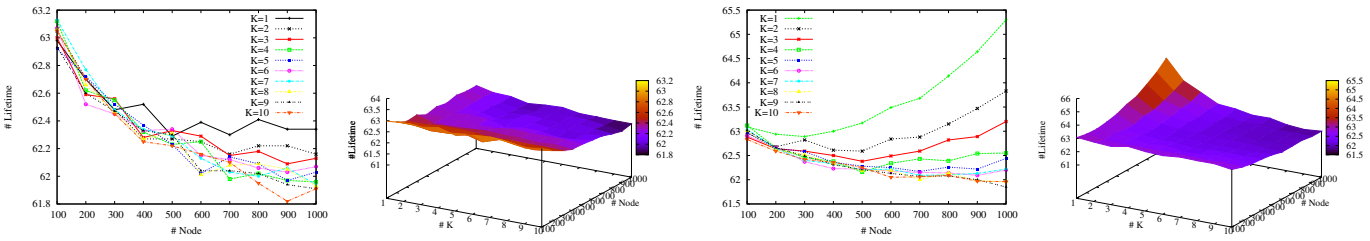
$$\min \sum_i \sum_a \sigma_{ia} R(Y_i, a)$$

$$\begin{array}{c} subject\ to\ \sigma_{ia} \ge 0, \qquad for\ all\ i, a \\ \sum_i \sum_a \sigma_{ia} = 1 \\ \sum_a \sigma_{ja} = \sum_i \sum_a \sigma_{ia} p_{ij}(a), \quad for\ all\ j \end{array} \tag{21}$$

$\mathcal{E}(\Phi)$ is a special case of linear programm and can be solved by a standard linear linear programming algorithm known as *the simplex algorithm*. The simplex algorithm solves this linear program by moving from an extreme point of the feasibility region to a better extreme point until the optimal is reached. So we can figure out the lower bound and upper bound of the lifetime by the linear programming Eq (21). ■

V. Simulation

Simulation Setup. In NetTopo [1], we conduct extensive simulation experiments. The studied WSN has the network size: 800×600m^2. The number of deployed sensor nodes are increased from 100 to 1000 (each time increased by 100). The value of k is changed from 1 to 10 (each time increased by 1). For every number of deployed sensor nodes, we use 100 different seeds to generate 100 different network deployment. A source node is deployed at the location of (50, 50), and a sink node is deployed at the location of (750, 550). The transmission radius for each node is 60m.



(a) Lifetime of CKN (2D) (b) Impact of k on CKN (3D) (c) Lifetime of EC-CKN (2D) (d) Impact of k on EC-CKN (3D)

Fig. 3. The network lifetime of running CKN and EC-CKN, respectively. In Fig 3(a), the ten lines of the lifetime results in CKN have a lot intersections, which means the energy consumption in CKN based WSN is not managed towards the energy balancing direction. However, in Fig 3(c), the ten lines of the lifetime results in EC-CKN present smooth changing when the number of nodes and the value of k are changed and the lifetime increases when the ratio N/k increases. This point clearly reflects that the energy consumption in EC-CKN based WSN is well managed towards the energy balancing direction. Furthermore, simulation results in Fig 3(b) and 3(d), also reveal that decreasing the value of k (let more nodes sleep) can definitely help to prolong the network lifetime in EC-CKN based WSN, but not in CKN based WSN.

Energy consumption and network lifetime comparison between CKN and EC-CKN algorithm. The network lifetime in both CKN and EC-CKN based WSNs are represented by the number of epoches. We conduct simulation for CKN and EC-CKN based WSNs in Fig 3. Results in Fig 3(a) and Fig 3(c) confirm that the energy consumption of EC-CKN based WSN is better managed and balanced than CKN based WSN. Results in Fig 3(b) and Fig 3(d) reveal the impact of changing the value of k: decreasing the value of k in EC-CKN based WSN can prolong the network lifetime, particularly when the network nodes are densely deployed.

VI. Conclusion

When deploying real WSNs for practical applications, having a good sleeping scheduling algorithm to balance sensor nodes' energy consumption and a reasonable length for epoch time towards energy saving is extremely important. In this research work, we made the following major contributions for supporting the realistic WSNs applications: 1) A new sleep scheduling algorithm, named EC-CKN, is proposed to balance the energy consumption and prolongs the network lifetime. 2) Solid simulation work is conducted, which proved the energy consumption in EC-CKN based WSN is well balanced.

Acknowledgments

Lei Shu's research in this paper was supported by Grant-in-Aid for Scientific Research (S)(21220002) of the Ministry of Education, Culture, Sports, Science and Technology, Japan. This work is partially supported by Natural Science Foundation of China under Grant No. 61070181.

References

[1] L. Shu, C. Wu, Y. Zhang, J. Chen, L. Wang, M. Hauswirth, "NetTopo: Beyond Simulator and Visualizer for Wireless Sensor Networks", in Proceedings of the Second International Conference on Future Generation Communication and Networking, December 13-15, 2008.

[2] S. Nath, P.B. Gibbons, "Communicating via fireflies: geographic routing on duty-cycled sensors", in Proceedings of the 6th international conference on Information processing in sensor networks (IPSN 2007), April 25-27, 2007.

[3] W. Wang, Y. Wang, X.Y. Li, W.Z. Song, O. Frieder, "Efficient interference-aware TDMA link scheduling for static wireless networks", in Proceedings of the 12th Annual International Conference on Mobile Computing and Networking (Mobicom 2006), eptember 24-29, 2006.

[4] C. Bettstetter, "On the minimum node degree and connectivity of a wireless multihop network", in Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc 2002), June 9-11, 2002.

[5] W.R. Heinzelman, A. Chandrakasan, H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks", in Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS 2000),January 4-7, 2000.

[6] V. Raghunathan, C. Schurgers, S. Park, M. Srivastava, B. Shaw, "Energy-Aware Wireless Microsensor Networks", IEEE Signal Processing Magazine, 2002

Appendix

We review Connected K-Neighborhood (CKN) algorithm in [2] in this appendix.

CONNECTED K-NEIGHBORHOOD (CKN) ALGORITHM
(Run the following at each node s_u *)*

1. Pick a random rank $rank_u$;
2. Broadcast $rank_u$ and receive the ranks of its currently awake neighbors N_u. Let R_u be the set of these ranks.
3. Broadcast R_u and receive R_v from each $s_v \in N_u$.
4. If $|N_u| < k$ or $|N_v| < k$ for any $s_v \in N_v$, remain awake. Return.
5. Compute $C_u = \{s_v | s_v \in N_u \text{ and } rank_v > rank_u\}$;
6. Go to sleep if both the following conditions hold. Remain awake otherwise.
   - Any two nodes in C_u are connected either directly themselves or indirectly through nodes which is in the s_u's 2-hop neighborhood that have $rank_v$ larger than $rank_u$;
   - Any node in N_u has at least k neighbors from C_u.
7. Return.
