locally and asynchronously. After the initial phase, during the action phase, if sensors choose to be active, they perform sensing, communication, and other tasks; otherwise, they switch to sleep mode to save energy.

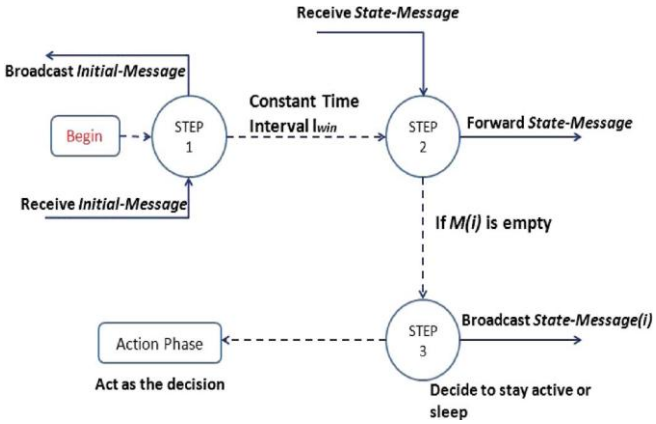

Figure 1: PTCP working

The PTCP runs during the initial phase of each slot. Sensors contend to sleep to save energy. If too many sensors choose to sleep, the requirement of probabilistic trap coverage will not be met. On the other hand, if too many sensors are active, it will be a waste of resources. Thus, a mechanism to coordinate sensors' decisions is desired. We, therefore, introduce *priority*. Every sensor has a unique ID. At the beginning of each time slot, each sensor is assigned a priority based on its residual energy and ID. Let *pri* denote the priority of sensor *i*. We define *pri = {Ei, IDi}*, where *Ei* is the residual energy of sensor *i*, and *IDi* is its ID. *pri*> *prj* if 1) *Ei* < *Ej* or 2)*E*(*i*) = *E*(*j*), and *IDi* < *IDj* . If a sensor has a lower priority than its neighbors, it has to make a decision after the sensors with higher priority. Sensor i start to broadcast its priority and location information to neighbors, the sensors within its transmission range. The information is packed as Initial-Message(i). At the same time, i will receive the Initial-Messages from its neighboring nodes too. Multihop communication is entertained in PTCP so i may receive messages whose sender is out of its transmission range. Sensor i should check the distance between the sender and itself. If the distance is greater than twice its diameter, the information is abandoned since they are impossible to be in the same circular graph; otherwise, sensor i should record the received information and forward it to neighbors to perform multihop communication. A time window is set for sensor i to wait for all Initial-Messages. The length of time window l_{win} is determined by sensor deployment density and the range D. It needs to guarantee that all sensors within the range of 2D are recorded during the time window. Since information broadcast is usually very fast, the time window should not occupy much time. When the time window ends, sensor i start to determine whether to sleep. It will broadcast its decision packed as State-Message(i). There are two kinds of State-Message: 1) State-Messagesleep(i) and 2) State-Messageactive(i), which denote the decision of sensor i, respectively. Here, we assume that Ci contains the recorded sensors from received Initial-Messages and that Mi contains sensors whose priority is higher than that of sensor i. If Mi is empty, i occupy the chance to make a decision since it is the sensor with the highest priority among the sensors within a distance of 2D. It will construct the circular graph and divide faces based on the information recorded in Ci. Note that sensor i itself is not contained in either Ci or Mi. Then, it employs Algorithm to determine whether the region within a distance of D is (D, ε)-trap covered. For the connectivity issue, sensor i also needs to guarantee that active sensors in a circular graph are connected if it chooses to sleep. Given the transmission range and the location information on all sensors in Ci, i can check whether all sensors in Ci are connected without i. If the region is covered and sensors in Ci are connected without i, sensor I will broadcast a State-Message$_{sleep(i)}$ since it does not need to be active; otherwise, it broadcasts a State-Message$_{active(i)}$ and chooses to stay active. If the region is still not (D, ε)-trap covered after sensor i chooses to stay active, it indicates that there are not enough sensors to provide (D, ε)-trap coverage and the lifetime of network terminates. If Mi is not empty, i have to wait for the State-Messages from other sensors in Mi. If i receive a new State-Message$_{sleep}$ whose sender is in set Ci, it will record the information, forward the message to its neighbors, and remove the sender from set Ci and, Mi if in it; otherwise, it will abandon the message since the message is useless. Sensor i can only make a decision when Mi is empty. Then, sensor i construct the circular graph based on Ci. All sensors in Ci are viewed as active when sensor i make a decision. Similarly, sensor i chooses to stay active in the PTCP if probabilistic trap coverage is not guaranteed or sensors in Ci will be disconnected without i. After decision making, sensor I will act as its choice, either in active mode or in sleep mode. In summary, the PTCP puts sensors into sleep mode in the order of priority/residual energy. In this way, sensors with less residual energy have the higher priority to switch to sleep mode and leave sensors with more residual energy to perform sensing tasks for energy balance, which can prolong the network's lifetime.

Algorithm PTCP
1. Define pri = {Ei, IDi} as the priority of sensor i. pri >prj if Ei < Ej or (Ei == Ej and IDi < IDj);
2. Set timeout threshold tτ .
3. At the beginning of each time slot, i turn into active mode;
4. Broadcast Initial-Message (i) to neighbors;
5. While Time window not end do
6. if Receive new Initial-Message(j) and distance(i, j) <2D then
7. Record Initial-Message(j);
8. Broadcast Initial-Message(j) to neighbors;
9. end if
10.end while
11. Define set Ci contains the recorded sensors from received Initial-Message;
12. Define set Mi contains sensors whose priority is greater than pri;
13. while Ci= ∅
14. % Update Ci and Mi when receiving State-Message from sensors in Ci
15. if Receive new State-Message(j) then
16. if distance(i, j) < 2D
17. Record State-Message(j);
18. Broadcast State-Message(j) to neighbors;
19. if j decides to sleep then
20. Remove j from Ci;

21. end if
22. if j ∈ Mi then
23. Remove j from Mi;
24. end if
25. end if
26. end if
27. % Clear Mi if timeout
28. if Mi = ∅ and exceed timeout threshold tτ then
29. Ci = Ci −Mi;
30. Let Mi = ∅;
31. end if
32. % Start to decide if Mi is empty
33. if Mi == ∅ then
34. Assume set Fi as the faces who are covered by sensor i;

5. Advantages of Proposed method

- The proposed method efficiently tracks the target using TOA even in higher noise rates.
- TCP minimizes amount of active sensors and hence energy balance is attained.
- Performance gain, throughput, and better navigation control is ensured through the theoretical study when compared with the state-of -art solutions.
- The number of stable nodes can be found with the energy constraint.

6. Conclusion

With unknown target and mobile sensor locations, we need to estimate the locations of the target and the mobile sensors first. Based on a more general TOA measurement model, convex optimization algorithms through SDP relaxation are developed for localization. Here is provided a sequential algorithm and a joint weighted localization algorithm before controlling the mobile sensor movement to follow the target. For the navigation of mobile sensors, the cubic law is applied. With these it's supposed to provide an efficient approach towards tracking. Along with this an energy efficient algorithm called PTCP is being used. The lower bound of lifetime acquired by the protocol is proven to be nearly half the optimum lifetime. This makes the environment much efficient towards tracking approach.

References

[1] P.H. Tseng, K.T. Feng, Y.C. Lin, and C.L. Chen, "Wireless Location Tracking Algorithms for Environments with Insufficient Signal Sources," IEEE Trans. Mobile Computing, vol. 8, no. 12, pp. 1676-1689, Dec. 2009.
[2] T. Li, A. Ekpenyong, and Y.F. Huang, "Source Localization and Tracking Using Distributed Asynchronous Sensors," IEEE Trans. Signal Processing, vol. 54, no. 10, pp. 3991-4003, Oct. 2006.
[3] Y. Zou and K. Chakrabarty, "Distributed Mobility Management for Target Tracking in Mobile Sensor Networks," IEEE Trans. Mobile Computing, vol. 6, no. 8, pp. 872-887, Aug. 2007.
[4] R. Rao and G. Kesidis, "Purposeful Mobility for Relaying and Surveillance in Mobile Ad Hoc Sensor Networks," IEEE Trans. Mobile Computing, vol. 3, no. 3, pp. 225-231, Mar. 2004.
[5] Enyang Xu, Zhi Ding and Soura Dasgupta, "Target Tracking and Mobile Sensor Navigation in Wireless Sensor Networks", IEEE Trans. Mobile Computing, vol. 12, no. 01, Jan.2013