UNIT II

LOCALIZATION AND TIME SYNCHRONIZATION PROTOCOLS 10 hrs

Localization protocols: Approaches, Coarse-Grained Node Localization using Minimal Information, Fine-Grained Node Localization using Detailed Information-Network-Wide Localization-Theoretical Analysis of Localization Techniques

Time Synchronization Protocols: Traditional Approaches, Coarse-Grained Clock synchronization, Fine-Grained Clock Synchronization.

INTRODUCTION:

Wireless sensor networks are fundamentally intended to provide information about the spatio- temporal characteristics of the observed physical world. Each individual sensor observation can be characterized essentially as a tuple of the form < S_T_M >, where S is the spatial location of the measurement, T the time of the measurement, and M the measurement itself.

The location information of nodes in the network is fundamental for a number of reasons:

1. To provide location stamps for individual sensor measurements that is being gathered.
2. To locate and track point objects in the environment.
3. To monitor the spatial evolution of a diffuse phenomenon over time, such as an expanding chemical plume. For instance, this information is necessary for in-network processing algorithms that determine and track the changing boundaries of such a phenomenon.
4. To determine the quality of coverage. If node locations are known, the network can keep track of the extent of spatial coverage provided by active sensors at any time.
5. To achieve load balancing in topology control mechanisms. If nodes are densely deployed, geographic information of nodes can be used to selectively shut down some percentage of nodes in each geographic area to conserve energy, and rotate these over time to achieve load balancing.
6. To form clusters. Location information can be used to define a partition of the network into separate clusters for hierarchical routing and collaborative processing.
7. To facilitate routing of information through the network. There are a number of geographic routing algorithms that utilize location information instead of node addresses to provide efficient routing.
8. To perform efficient spatial querying. A sink or gateway node can issue queries for information about specific locations or geographic regions. Location information can be used to scope the query propagation instead of flooding the whole network, which would be wasteful of energy.

We should, at the outset, make it clear that localization may not be a significant challenge in all WSN. In structured, carefully deployed WSN (for instance in industrial settings, or scientific experiments), the location of each sensor may be recorded and mapped to a node ID at deployment time. In other contexts, it may be possible to obtain location information using existing infrastructure, such as the satellite-based GPS or cellular phone positioning techniques. However, these are not satisfactory solutions to all contexts. A-prior knowledge of sensor locations will not be available in large-scale and ad hoc deployments. A pure-GPS solution is viable only if all nodes in the network can be provided with a potentially expensive GPS receiver and if the deployed area provides

good satellite coverage. Positioning using signals directly from cellular systems will not be applicable for densely deployed WSN, because they generally offer poor location accuracy (on the order of tens of meters). If only a subset of the nodes have known location a priori, the position of other nodes must still be determined through some localization technique.

Key issues

Localization is quite a broad problem domain, and the component issues and techniques can be classified on the basis of a number of key questions.

1. What to localize? This refers to identifying which nodes have a priori known locations (called reference nodes) and which nodes do not (called unknown nodes). There are a number of possibilities. The number and fraction of reference nodes in a network of n nodes may vary all the way from 0 to n – 1. The reference nodes could be static or mobile; as could the unknown nodes. The unknown nodes may be cooperative (e.g. participants in the network, or robots traversing the networked area) or non-cooperative (e.g. targets being surveilled). The last distinction is important because non-cooperative nodes cannot participate actively in the localization algorithm.

2. When to localize? In most cases, the location information is needed for all unknown nodes at the very beginning of network operation. In static environments, network localization may thus be a one-shot process. In other cases, it may be necessary to provide localization on-the-fly, or refresh the localization process as objects and network nodes move around, or improve the localization by incorporating additional information over time. The time scales involved may vary considerably from being of the order of minutes to days, even months.

3. How well to localize? This pertains to the resolution of location information desired. Depending on the application, it may be required for the localization technique to provide absolute (x, y, z) coordinates, or perhaps it will suffice to provide relative coordinates (e.g. "south of node 24 and east of node 22"); or symbolic locations (e.g. "in room A", "in sector 23", "near node 21"). Even in case of absolute locations, the required accuracy may be quite different. The technique must provide the desired type and accuracy of localization, taking into account the available resources (such as computational resources, time-synchronization capability, etc.).

4. Where to localize? The actual location computation can be performed at several different points in the network: at a central location once all component information such as inter-node range estimates is collected; in a distributed iterative manner within reference nodes in the network; or in a distributed manner within unknown nodes. The choice may be determined by several factors: the resource constraints on various nodes, whether the node being localized is cooperative, and the localization technique employed, and, finally, security considerations.

5. How to localize? Finally, different signal measurements can be used as inputs to different localization techniques. The signals used can vary from narrowband radio signal strength readings or packet-loss statistics, UWB RF signals, acoustic/ultrasound signals, infrared. The signals may be emitted and measured by the reference nodes, by the unknown nodes, or both. The basic localization algorithm may be based on a number of techniques, such as proximity, calculation of centroids, constraints, ranging, angulation, pattern recognition, multi-dimensional scaling, and potential methods.

Localization approaches

Generally speaking, there are two approaches to localization:

2. Coarse-grained localization using minimal information: These typically use a small set of discrete measurements, such as the information used to compute location. Minimal information could include binary proximity, near–far information or cardinal direction information.

2. Fine-grained localization using detailed information: These are typically based on measurements, such as RF power, signal waveform, time stamps, etc., that are either real-valued or discrete with a large number of quantization levels. These include techniques based on radio signal strengths, timing information, and angulation.

The tradeoff that emerges between the two approaches is easy to see: while minimal information techniques are simpler to implement, and likely involve lower resource consumption and equipment costs, they provide lower accuracy than the detailed information techniques.

Coarse-grained node localization using minimal information

Binary proximity

Perhaps the most basic location technique is that of binary proximity – involving a simple decision of whether two nodes are within reception range of each other. Set of references nodes are placed in the environment in a non-overlapping manner. Either the reference nodes periodically emit beacons, or the unknown node transmits a beacon when it needs to be localized. If reference nodes emit beacons, these include their location IDs. The unknown node must then determine which node it is closest to, and this provides a coarse grained localization. Alternatively, if the unknown node emits a beacon, the reference node that hears the beacon uses its own location to determine the location of the unknown node.

An excellent example of proximity detection as a means for localization is the Active Badge location system meant for an indoor office environment. This system consists of small badge cards (about 5 square centimeters in size and less than a centimeter thick) sending unique beacon signals once every 15 seconds with a 6 meter range. The active badges, in conjunction with a wired sensor network that provides coverage throughout a building, provide room level location resolution. A much larger application of localization, using binary proximity detection, is with passive radio frequency identification (RFID) tags, which can be detected by readers within a similar short range. Today there are a large number of inventory-tracking applications envisioned for RFIDs. A key difference in RFID proximity detection compared with active badges is that the unknown nodes are passive tags, being queried by the reference nodes in the sensor network. These examples show that even the simplest localization technique can be of considerable use in practice.

Centroid calculation

The same proximity information can be used to greater advantage when the density of reference nodes is sufficiently high that there are several reference nodes within the range of the

unknown node. Consider a two-dimensional scenario. Let there be n reference nodes detected within the proximity of the unknown node, with the location of the i^{th} such reference denoted by (x_i,y_i). Then, in this technique, the location of the unknown node (x_u,y_u) is determined as

$$x_u = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$y_u = \frac{1}{n}\sum_{i=1}^{n} y_i$$

This simple centroid technique has been investigated using a model with each node having a simple circular range R in an infinite square mesh of reference nodes spaced a distance d apart. It is shown through a simulation that as the overlap ratio R/d is increased from 1 to 4, the average RMS error in localization is reduced from 0.5d to 0.25d.

Geometric constraints

If the bounds on radio or other signal coverage for a given node can be described by a geometric shape, this can be used to provide location estimates by determining which geometric regions that node is constrained to be in, because of intersections between overlapping coverage regions. For instance, the region of radio coverage may be upper-bounded by a circle of radius R_{max}. In other words, if node B hears node A, it knows that it must be no more than a distance R_{max} from A. Now, if an unknown node hears from several reference nodes, it can determine that it must lie in the geometric region described by the intersection of circles of radius R_{max} centered on these nodes. This can be extended to other scenarios. For instance when both lower R_{min} and upper bounds R_{max} can be determined, based on the received signal strength, the shape for a single node's coverage is an annulus; when an angular sector $(\theta_{min}, \theta_{max})$ and a maximum range R_{max} can be determined, the shape for a single node's coverage would be a cone with given angle and radius.

Although arbitrary shapes can be potentially computed in this manner, a computational simplification that can be used to determine this bounded region is to use rectangular bounding boxes as location estimates. Thus the unknown node determines bounds $(x_{min},y_{min}, x_{max},y_{max})$ on its position. Figure 2.1 illustrates the use of intersecting geometric constraints for localization.

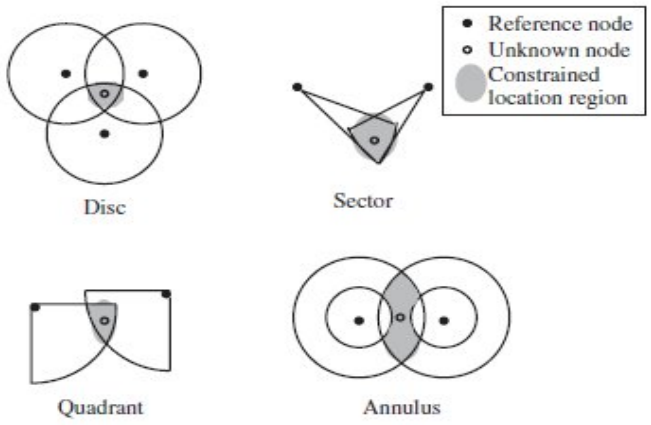


Fig 2.1: Localization using intersection of geometric constraints

Localization techniques using such geometric regions were first described by Doherty *et al.* One of the nice features of these techniques is that not only can the unknown nodes use the centroid of the overlapping region as a specific location estimate if necessary, but they can also determine a bound on the location error using the size of this region.

When the upper bounds on these regions are tight, the accuracy of this geometric approach can be further enhanced by incorporating "negative information" about which reference nodes are *not* within range.

Approximate point in triangle (APIT)

A related approach to localization using geometric constraints is the approximate point-in-triangle (APIT) technique. APIT is similar to the above techniques in that it provides location estimates as the centroid of an intersection of regions. Its novelty lies in how the regions are defined – as triangles between different sets of three reference nodes (rather than the coverage of a single node). This is illustrated in Figure 3.2. It turns out that an exact determination of whether an unknown node lies within the triangle formed by three reference nodes is impossible if nodes are static because wireless signal propagation is non-ideal. An approximate solution can be determined using near–far information, i.e. the ability to determine which of two nodes is nearer a third node based on signal reception. One caveat for the APIT technique is that it can provide erroneous results, because the determination of whether a node lies within a particular triangle requires quite a high density of nodes in order to provide good location accuracy.

Identifying codes

There is another interesting technique that utilizes overlapping coverage regions to provide localization. In this technique, referred to as the identifying code construction (ID-CODE) algorithm, the sensor deployment is planned in such a way as to ensure that each resolvable location is covered by a unique set of sensors.

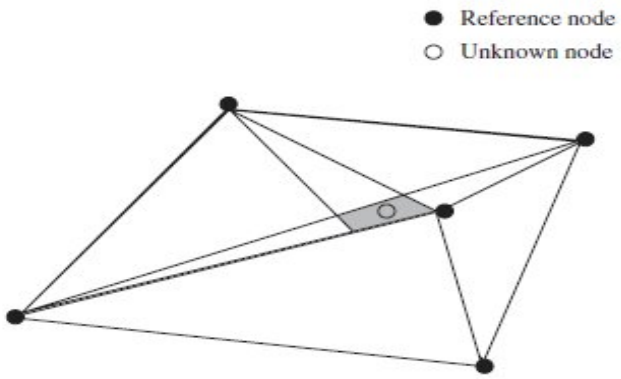
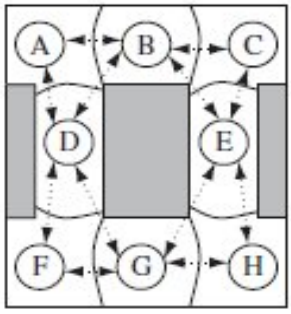


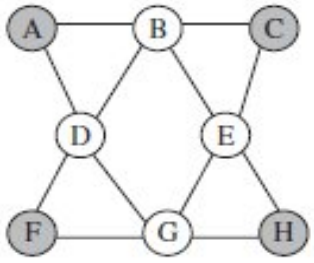
Fig2.2: The approximate point-in-triange (APIT) technique

The algorithm runs on a deployment region graph G= (V, E) in which vertices V represent the different regions and the edges E represent radio connectivity between regions. Let B(v) be the set of vertices that are adjacent to v, together with v itself. A set of vertices $C \in V$ is referred to as an *identifying code*, if, for all $u, v \in V$, $B(v) \cap C \neq B(u) \cap C$.

It can be shown that a graph is distinguishable, i.e. there exists an identifying code for it, if and only if there are no two vertices u_ v such that B(u)=B(v). The goal of the algorithm is to construct an identifying code for any distinguishable graph, with each vertex in the code corresponding to a region where a reference node must be placed. Once this is done, by the definition of the identifying code, each location region in the graph will be covered by a unique set of reference nodes. This is illustrated in Figure.



Node locations



Connectivity graph

Transmitters A, F, C, H provide unique IDs for all node locations

V:	A	B	C	D	E	F	G	H
ID:	A	A,C	C	A,F	C,H	F	F,H	H

Fig 2.3: Illustration of the ID-CODE technique showing uniquely identifiable regions

While the entire set of vertices V itself is an identifying code, such a placement of a reference node in each region would clearly be inefficient. On the other hand, obtaining a minimal cardinality identifying code is known to be NP-complete. The algorithm ID-CODE is a polynomial greedy heuristic that provides good solutions in practice. There also exists a robust variant of this algorithm called r-ID-CODE that can provide robust identification, i.e. guaranteeing a unique set of IDs for each location, even if there is addition or deletion of up to r ID values.

Fine-grained node localization using detailed information

We now examine techniques based on detailed information. These include triangulation using distance estimates, pattern matching, and sequence decoding. Although used in the large-scale GPS, basic time-of-flight techniques using RF signals are not capable of providing precise distance estimates over short ranges typical of WSN because of synchronization limitations. Therefore other techniques such as radio signal strength (RSS) measurements and time difference of arrival (TDoA) must be used for distance-estimation.

Radio signal-based distance-estimation (RSS)

To a first-order approximation, mean radio signal strengths diminish with distance according to a power law. One model that is used for wireless radio propagation is the following:

$$P_{r,dB}(d) = P_{r,dB}(d_0) - \eta 10 \log\left(\frac{d}{d_0}\right) + X_{\sigma,dB}$$

Where, $P_{r\_dB}(d)$ is the received power at distance d and $P(d_0)$ is the received power at some reference distance d_0, the path-loss exponent, and $X_{\sigma.dB}$ a lognormal random variable with variance σ^2 that accounts for fading effects. So, in theory, if the path-loss exponent for a given environment is known the received signal strength can be used to estimate the distance. However, the fading term often has a large variance, which can significantly impact the quality of the range estimates. This is the reason RF-RSS-based ranging techniques may offer location accuracy only on the order of meters. RSS-based ranging may perform much better in situations where the fading effects can be combatted by diversity techniques that take advantage of separate spatio-temporally correlated signal samples.

Distance-estimation using time differences (TDoA)

As we have seen, time-of-flight techniques show poor performance due to precision constraints, and RSS techniques, although somewhat better, are still limited by fading effects. A more promising technique is the combined use of ultrasound/ acoustic and radio signals to estimate distances by determining the TDoA of these signals. This technique is conceptually quite simple, and is illustrated in Figure 2.4. The idea is to simultaneously transmit both the radio and acoustic signals (audible or ultrasound) and measure the times T_r and T_s of the arrival of these signals respectively at the receiver.

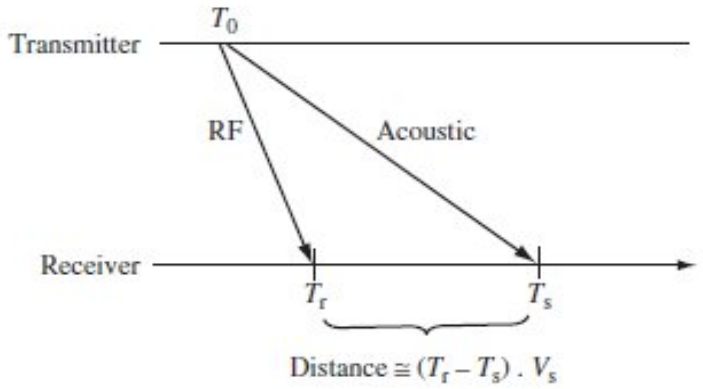


Fig 2.4: Ranging based on time difference of arrival

Since the speed of the radio signal is much larger than the speed of the acoustic signal, the distance is then simply estimated as $(T_s - T_r) \cdot V_s$, where V_s is the speed of the acoustic signal. One minor limitation of acoustic ranging is that it generally requires the nodes to be in fairly close proximity to each other (within a few meters) and preferably in line of sight. There is also some uncertainty in the calculation because the speed of sound varies depending on many factors such as altitude, humidity, and air temperature. Acoustic signals also show multi-path propagation effects that may impact the accuracy of signal detection. The basic idea is to send a pseudo-random noise sequence as the acoustic signal and use a matched filter for detection, (instead of using a simple chirp and threshold detection).

On the whole, acoustic TDoA ranging techniques can be very accurate in practical settings. For that distance can be estimated to within a few centimeters for node separations fewer than 3 meters. Of course, the tradeoff is that sensor nodes must be equipped with acoustic transceivers in addition to RF transceivers.

Triangulation using distance estimates

The location of the unknown node (x_0, y_0) can be determined based on measured distance estimates $\hat{d}_i$ to n reference nodes $\{(x_1, y_1),\ldots, (x_i, y_i),\ldots, (x_n, y_n)\}$. This can be formulated as a least squares minimization problem. Let d_i be the correct Euclidean distance to the n reference nodes, i.e.:

$$d_i = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2}$$

Angle of arrival (AoA)

Another possibility for localization is the use of angular estimates instead of distance estimates. Angles can potentially be estimated by using rotating directional beacons, or by using nodes equipped with a phased array of RF or ultrasonic receivers. Angulation with ranging is a particularly powerful combination. In theory, if the angular information provided to a given reference node can be combined with a good distance estimate to that reference node, then localization can be performed with a single reference using polar coordinate transformation. While the accuracy and precision with which angles in real systems can be determined are unclear, significant improvements can be obtained by combining accurate ranging estimates with even coarse-grained angle estimates.

Pattern matching (RADAR)

An alternative to measuring distances or angles that is possible in some contexts is to use a pre-determined "map" of signal coverage in different locations of the environment, and use this map to determine where a particular node is located by performing pattern matching on its measurements. This technique requires the prior collection of empirical measurements (or high-fidelity simulation model) of signal strength statistics (mean, variance, median) from different reference transmitters at various locations. It is also important to take into account the directional orientation of the receiving node, as this can result in significant variations. Once this information is collected, any node in the area is localized by comparing its measurements from these references to determine which location matches the received pattern best. This technique has some advantages, in particular as a pure RF technique it has the potential to perform better than the RSS-based distance-estimation and the triangulation approach we discussed before. However, the key drawback of the technique is that it is very location specific and requires intensive data collection prior to operation; also it may not be useful in settings where the radio characteristics of the environment are highly dynamic.

RF sequence decoding (ecolocation)

The ecolocation technique uses the relative ordering of received radio signal strengths for different references as the basis for localization. It works as follows:

1. The unknown node broadcasts a localization packet.

2. Multiple references record their RSSI reading for this packet and report it to a common calculation node.
3. The multiple RSSI readings are used to determine the ordered sequence of references from highest to lowest RSSI.
4. The region is scanned for the location for which the correct ordering of references (as measured by Euclidean distances) has the "best match" to the measured sequence. This is considered the location of the unknown node.

In an ideal environment, the measured sequence would be error free, and ecolocation would return the correct location region. However, in real environments, because of multi-path fading effects, the measured sequence is likely to be corrupted with errors. Some references, which are closer than others to the true location of the unknown node, may show a lower RSSI, while others, which are farther away, may appear earlier in the sequence. Therefore the sequence must be decoded in the presence of errors. This is why a notion of "best match" is needed.

The best match is quantified by deriving the n,n−1/2 pair-wise ordering constraints (e.g. reference A is closer than reference B, reference B is closer than reference C, etc.) at each location, and determining how many of these constraints are satisfied/violated in the measured sequence. The location which provides the maximum number of satisfied constraints is the best match. Simulations and experiments suggest that ecolocation can provide generally more accurate localizations compared with other RF-only schemes, including triangulation using distance estimates. Intuitively, this is because the ordered relative sequence of RSSI values at the references provides robustness to fluctuations in the absolute RSSI value.

Network-wide localization

Issues & Challenges

So far, we have focused on the problem of *node localization*, which is that of determining the location of a single unknown node given a number of nearby references. A broader problem in sensor systems is that of *network localization*, where several unknown nodes have to be localized in a network with a few reference nodes. While the network localization problem can rarely be neatly decomposed into a number of separate node localization problems (since there may be unknown nodes that have no reference nodes within range), the node localization and ranging techniques described above do often form an integral component of solutions to network localization.

The performance of network localization depends very much on the resources and information available within the network. Several scenarios are possible: for instance there may be no reference nodes at all, so that perhaps only relative coordinates can be determined for the unknown nodes; if present, the number/density of reference nodes may vary (generally the more reference nodes there are, the lower the network localization error); there may be just a single mobile reference. Information about which nodes are within range of each other may be available; or inter-node distance estimates may be available; inter-node angle information may be available. Some network localization approaches are centralized, in which all the available information about known nodes, and the inter-node distances or other inter-node relationships are provided to a central node, where the solution is computed.
Such a centralized approach may be sufficient in moderate-sized networks, where the nodes in the network need to be localized only once, post-deployment. Other network localization approaches are distributed, often involving the iterative communication of updated location information. There may be several ways to measure the performance of network localization. If the ground truth is available,

these can range from the full distribution/ histogram of location errors, to the mean location error, to the percentage of unknown nodes that can be located within a desired accuracy. Alternatively some localization approaches provide an inherent way to estimate the uncertainty associated with each node's calculated location.

Constraint-based approaches

Geometric constraints can often be expressed in the form of linear matrix inequalities and linear constraints. This applies radial constraints (two nodes are determined to be within range R of each other), annular constraints (a node is determined to be within ranges [R_{min}, R_{max}] of another), angular constraints (a node is determined to be within a particular angular sector of another), as well as other convex constraints. Information about a set of reference nodes together with these constraints (which provide the inter-node relationships amongst reference as well as unknown nodes) describes a feasible set of constraints for a semi-definite program. By selecting an appropriate objective function for the program, the constraining rectangle, which bounds the location for each unknown node, can be determined.

When using bounding rectangles, a distributed iterative solution can be used. In this solution, at each step nodes broadcast to their neighbors their current constrained region, which is calculated based on the overheard information about their neighbors' constrained regions at the previous step. If continued for a sufficient number of iterations, or until there is no longer a significant improvement in the bounds, this can provide a solution that is near or at optimal. Network localization can also be performed in the presence of mobile reference/ target nodes. If the mobile node is a reference and able to provide an accurate location beacon, then it can substantially improve localization over time, because each new observation of the moving beacon introduces additional constraints. In theory the location error can be reduced to an arbitrarily small quantity if the moving beacon is equally likely to move to any point in the network. If the mobile node is a non-cooperative target, then the distributed iterative algorithm can be extended to provide simultaneous network localization and tracking with performance that improves over time.

RSS-based joint estimation

If radio signal strengths can be measured between all pairs of nodes in the network that are within detection range, then a joint maximum likelihood estimation (MLE) technique can be used to determine the location of unknown nodes in a network. In the joint MLE technique, first an expression is derived for the likelihood that the obtained matrix of power measurements would be received given a particular location set for all nodes; the objective is then to find the location set that maximizes this likelihood. The performance of this joint MLE technique has been verified through simulations and experiments to show that localization of the order of 2 meters is possible when there is a high density of unknown nodes, even if there are only a few reference nodes sparsely placed.

Iterative multilateration

The iterative multilateration technique is applicable whenever inter-node distance information is available between all neighboring nodes (regardless of whether it is obtained through RSS measurements or TDoA or any other approach). The algorithm is quite simple. It applies the

basic triangulation technique for node localization in an iterative manner to determine the locations of all nodes. One begins by determining the location of an unknown node that has the most reference nodes in its neighborhood. In a distributed version, the location of any node with sufficient references in its neighborhood may be calculated as the initial step. This node is then added to the set of reference nodes and the process is repeated. Figure 2.5 shows an example of a network with one possible sequence in which unknown nodes can each compute their location so long as at least three of their neighbors have known or already computed locations. Note that a version of iterative multilateration can also be utilized if only connectivity information is available. In such a case, a centroid calculation could be used at each iterative step by the unknown nodes, instead of using distance based triangulation. The iterative multilateration technique suffers from two shortcomings: first, it may not be applicable if there is no node that has sufficient (≥3 for the 2D plane) reference nodes in its neighborhood; second, the use of localized unknown nodes as reference nodes can introduce substantial cumulative error in the network localization (even if the more certain high-reference neighborhood nodes are used earlier in the iterative process).

Collaborative multilateration

The key insight is to determine collaborative sub-graphs within the network that contain reference and unknown nodes in a topology such that their positions and inter-node distances can be written as an over-constrained set of quadratic equations with a unique solution for the location of unknown nodes (which can be obtained through gradient descent or local search algorithms). Used in conjunction with iterative multilateration, this technique is generally useful in portions of the network where the reference node density is low.

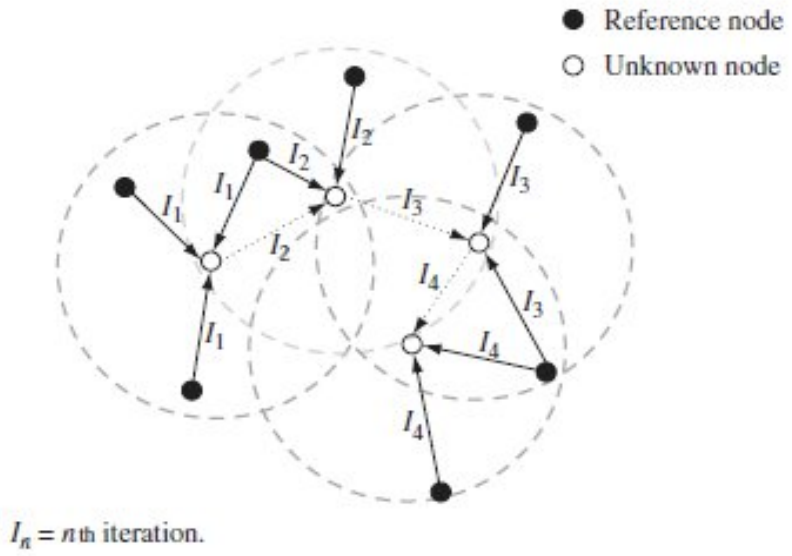


Fig 2.5: Illustration of sequence of iterative multilateration steps

Multi-hop distance-estimation approaches

An alternative approach to network localization utilizes estimates of distances to reference nodes that may be several hops away. These distances are propagated from reference nodes to unknown nodes using a basic distance-vector technique. There are three variants of this approach:

DV-hop: In this approach, each unknown node determines its distance from various reference nodes by multiplying the least number of hops to the reference nodes with an estimated average distance per hop. The average distance per hop depends upon the network density, and is assumed to be known.

DV distance: If inter-node distance estimates are directly available for each link in the graph, then the distance-vector algorithm is used to determine the distance corresponding to the shortest distance path between the unknown nodes and reference nodes.

Euclidean propagation: Geometric relations can be used in addition to distance estimates to determine more accurate estimates to reference nodes. For instance consider a quadrilateral ABCR, where A and R are at opposite ends; if node A knows the distances AB, AC, BC and nodes B and C have estimates of their distance to the reference R, then A can use the geometric relations inherent in this quadrilateral to calculate an estimated distance to R. Once distance estimates are available from each unknown node to different reference nodes throughout the network, a triangulation technique can be employed to determine their locations. Through simulations, it has been seen that location errors for most nodes can be kept within a typical single-hop distance. In a comparative simulation study of these approaches, it has been shown that the relative performance of these three schemes depends on factors such as the radio range and accuracy of available distance estimates.

Refinement

Once a possible initial estimate for the location of unknown nodes has been determined through iterative multilateration/collaborative multilateration or the distance-vector estimation approaches, additional refinement steps can be applied. Each node continues to iterate, obtaining its neighbors location estimates and using them to calculate an updated location using triangulation. After some iteration, the position updates become small and this refinement process can be stopped.

Force-calculation approach

An inherently distributed iterative approach to network localization is to use a physics-based analogy. Each node first picks a reasonable initial guess as to its location, which need not be very accurate. If $d_{i,j}$ is the calculated distance between the two nodes as per their current positions, $\hat{d}_{i,j}$ the estimated distance and $u_{i,j}$ the unit vector between them, and H_i is the set of all neighboring nodes of i, then a vector force on a link and the resultant force on a node can be respectively defined as

$$\overrightarrow{F_{i,j}} = (d_{i,j} - \hat{d}_{i,j})\overrightarrow{u_{i,j}}$$

$$\overrightarrow{F_i} = \sum_{j \in H_i} \overrightarrow{F_{i,j}}$$

Each unknown node then updates its position in the direction of the resulting vector force in small increments over several iterations (with the force being recalculated at each step). However, it should be kept in mind that this technique may be susceptible to local minima.

Multi-dimensional scaling

Given a network with a sparse set of reference nodes, and a set of pair-wise distances between neighboring nodes (including reference and unknown nodes), another network localization approach utilizes a data analysis technique known as multi-dimensional scaling (MDS) [193]. It consists of the following three steps:

1. Use a distance-vector algorithm (similar to DV-distance) to generate an n×n matrix M, whose (i, j) entry contains the estimated distance between nodes i and j.

2. Apply classical metric-MDS to determine a map that gives the locations of all nodes in relative coordinates. The classical metric MDS algorithm is a matrix-based numerical technique that solves the following least squares problem: if the estimated distance matrix M can be expressed as the sum of the actual distance matrix D and a residual error matrix E (i.e. M=D+E), then determine possible locations for all n nodes, such that the sum of squares of the elements of E is minimized.

3. Take the position of reference nodes into account to obtain normalized absolute coordinates.

Reference-less localization

In some scenarios, we may encounter sensor networks that are deployed in such an *ad hoc* manner, without GPS capabilities, that there are no reference nodes whatsoever. In such a case, the best that can be hoped for is to obtain the location of the network nodes in terms of relative, instead of absolute, coordinates. While such a map is not useful for location stamping of sensor data, it can be quite useful for other functions, such as providing the information required to implement geographic routing schemes.

Algorithm is described as a progression of three scenarios with successively fewer assumptions:

1. All (and only) nodes at the boundary of the network are reference nodes.
2. Nodes at the boundary are aware that they are at the boundary, but are not reference nodes.
3. There are no reference nodes in the network, and no nodes are aware that they are at the boundary.

In the first scenario, all nodes execute a simple iterative algorithm for localization. Unknown interior nodes begin by assuming a common initial coordinate (say [0,0]), then at each step, each unknown node determines its location as the centroid of the locations of all its neighbors. It is shown that this algorithm tends to "stretch" the locations of network nodes through the location region. When the algorithm converges, nodes have determined a location that is close to their nearest boundary nodes. Figure 2.6 gives an example of a final solution.

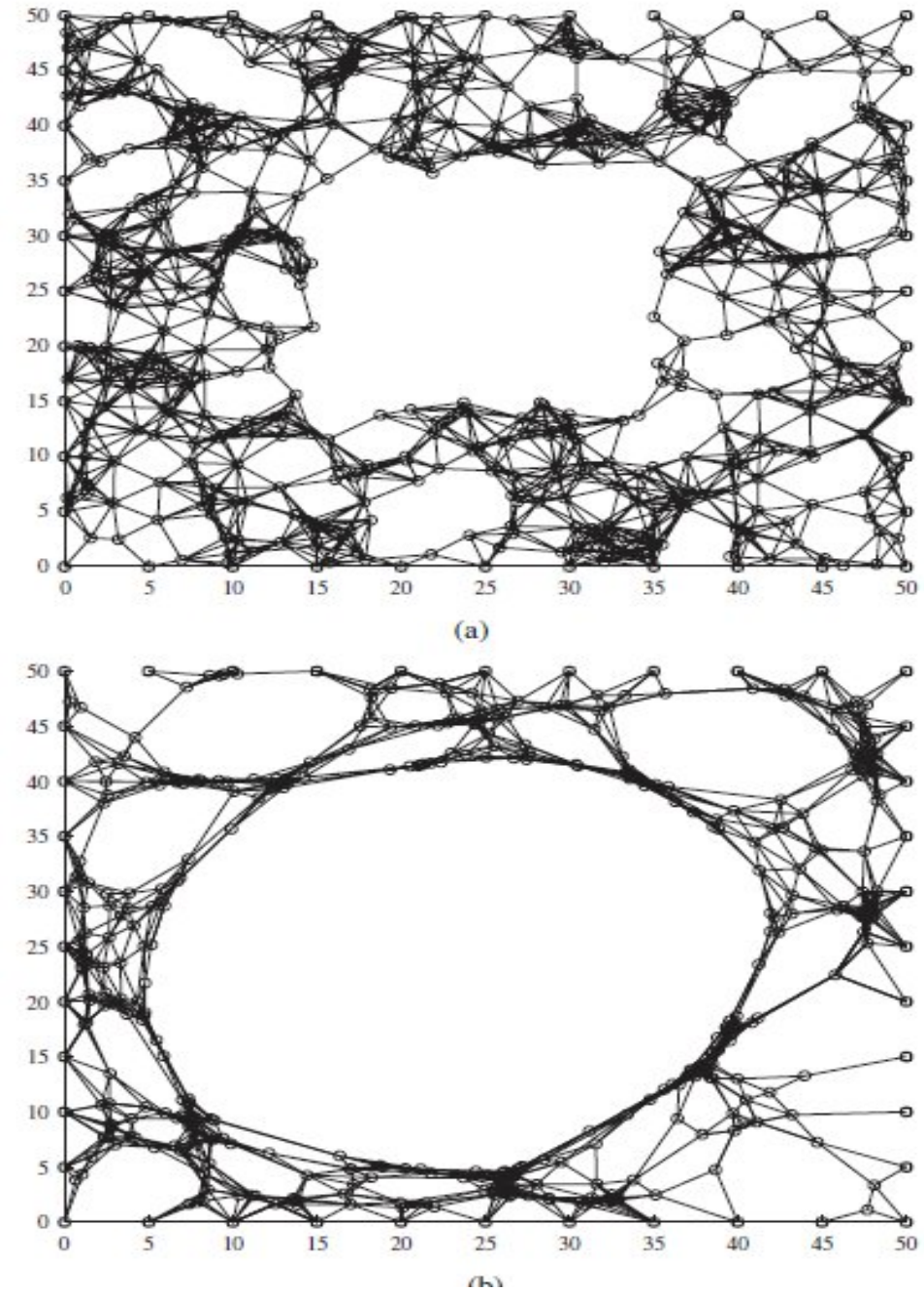


Fig 2.6: An illustration of the reference-less network localization technique assuming boundary node locations are known: (a) original map and (b) obtained relative map

While the final solution is generally not accurate, it is shown that for greedy geographic routing it results in only slightly longer routing paths and potentially even slightly better routing success rates (as non-ideal positions can sometimes improve over the local optima that arise in greedy geographic routing).

The second scenario can be reduced approximately to the first. This can be done by having the border nodes first flood messages to communicate with each other and determine the pair-wise hop-counts between themselves. These hop counts are then used in a triangulation algorithm to obtain virtual coordinates for the set B of all border nodes by minimizing

$$\sum_{i,j \in B} (\text{hops}(i, j) - \text{dist}(i, j))^2$$

Where hops (i, j) is the number of hops between border nodes i, j, and dist(i, j) their Euclidean distance for given virtual coordinates. An additional bootstrapping mechanism ensures that all nodes calculate consistent virtual coordinates. Finally, the third scenario can be reduced to the second. Any node that is farthest away from a common node in terms of hop-count with respect to all its two-hop neighbors can determine that it is on the border. This hop-count determination is performed through a flood from one of the bootstrap nodes.

Theoretical analysis of localization techniques

Cramer–Rao lower bound

One theoretical tool of utility in analyzing limitations on the performance of localization techniques is the use of the Cramér–Rao bound. The Cramér–Rao bound (CRB) is a well-known lower bound on the error variance of any unbiased estimator, and is defined as the inverse of the Fisher information matrix (a measure of information content with respect to parameters). The CRB can be derived for different assumptions about the localization technique (e.g. TOA based, RSS-based, proximity-based, node/network localization). The CRB has been used to investigate error performance of K-level quantized RSSI-based localization. A special case is K = 2, which corresponds to proximity information (whether the node is within range or not). The lower bound can be improved monotonically with K, with about 50% improvement if K is large compared with just using proximity alone. On the other hand, K =8 (three bits of RSS quantization) suffices to give a lower bound that is very close to the best possible. It is also found that the MLE estimator, which is a biased estimator, provides location errors with variance close to that observed with the CRB. CRB analysis has also been used to investigate the performance of network localization under different densities, and shown to give similar trends to the iterative / collaborate multilateration technique. The CRB-based analysis suggests that localization accuracy improves with network density, with diminishing returns once each node has about 6–8 neighbors on average. It also suggests, somewhat surprisingly, that increasing the fraction of beacon nodes from 4% to 20% does not dramatically decrease the localization error (under the assumptions of uniform placement, high-density, low-ranging error).

Unique network localization

There is a strong connection between the problem of unique network localization and a mathematical subject known as rigidity theory.

Consider a sensor network with n nodes (m reference nodes and n−m unknown nodes) located in the 2D plane, with edges between neighboring nodes. Information is available about the exact location coordinates of the reference nodes, and the exact Euclidean distance between all neighboring nodes. This network is said to be uniquely localizable if there exists only one possible assignment of (x, y) coordinates to all unknown nodes that is consistent with all the available information about distances and positions.

The key result concerning the conditions for a network to be unique localizable is the following:

A network N *is uniquely localizable if and only if the weighted grounded graph* G N *corresponding to it is globally rigid.*

There are two terms here that need to be explained – weighted grounded graph and global rigidity. The *weighted grounded graph* G N is constructed from the graph described by network N (with each edge weighed by the corresponding distance) by adding additional edges between all pairs of reference nodes, labeled with the distance between them (which can be readily calculated, since reference positions are known).

We shall give an intuitive definition of global rigidity. Consider a configuration graph of points in general position on the plane, with edges connecting some of them to represent distance constraints. Is there another configuration consisting of different points on the plane that preserves all the distance constraints on the edges (excluding trivial changes, such as translations, rotations, and mirror images)? If there is not, the configuration graph is said to be globally rigid in the plane. Figure 2.7 gives examples of non-globally rigid and globally rigid configuration graphs. There exist polynomial algorithms to determine whether a given configuration graphs is globally rigid in the plane, and hence to determine if a given network is uniquely localizable. However, the problem of realizing globally rigid weighted graphs (which is closely related to actually determining possible locations of the unknown nodes in the corresponding network) is

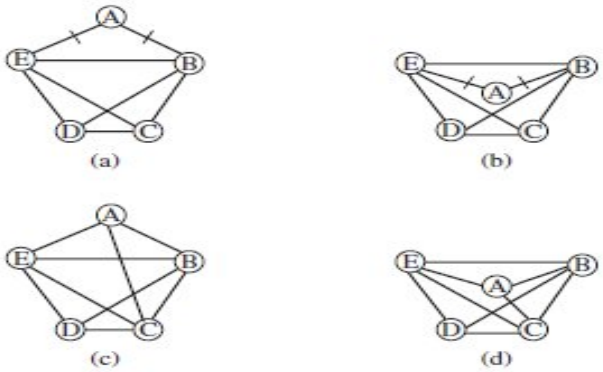


Fig 2.7: Examples of configuration graphs that are not globally rigid ((a),(b)) and that are globally rigid ((c),(d))

NP-hard. While this means that in the worst case there exist no known tractable algorithms to solve all instances, in the case of geometric random graphs, with at least three reference nodes within range of each other, there exists a critical radius threshold that is $o\left(\frac{\sqrt{\log n}}{n}\right)$, beyond which the network is uniquely localizable in polynomial time with high probability.

Time Synchronization Overview

Given the need to coordinate the communication, computation, sensing, and actuation of distributed nodes, and the spatio-temporal nature of the monitored phenomena, it is no surprise that an accurate and consistent sense of time is essential in sensor networks. In this chapter, we shall discuss the many motivations for time synchronization, the challenges involved, as well as some of the solutions that have been proposed.

Distributed wireless sensor networks need time synchronization for a number of good reasons, some of which are described below:

1. For time-stamping measurements: Even the simplest data collection applications of sensor networks often require that sensor readings from different sensor nodes be provided with time stamps in addition to location information. This is particularly true whenever there may be a significant and unpredictable delay between when the measurement is taken at each source and when it is delivered to the sink/base station.
2. For in-network signal processing: Time stamps are needed to determine which information from different sources can be fused/aggregated within the network. Many collaborative signal processing algorithms, such as those for tracking unknown phenomena or targets, are coherent and require consistent and accurate synchronization.
3. For localization: Time-of-flight and TDoA-based ranging techniques used in node localization require good time synchronization.
4. For cooperative communication: Some physical layer multi-node cooperative communication techniques involve multiple transmitters transmitting in-phase signals to a given receiver. Such techniques [105] have the potential to provide significant energy savings and robustness, but require tight synchronization.
5. For medium-access: TDMA-based medium-access schemes also require that nodes be synchronized so that they can be assigned distinct slots for collision free communication.
6. For sleep scheduling: As we shall see in the following chapters, one of the most significant sources of energy savings is turning the radios of sensor devices off when they are not active. However, synchronization is needed to coordinate the sleep schedules of neighboring devices, so that they can communicate with each other efficiently.
7. For coordinated actuation: Advanced applications in which the network includes distributed actuators in addition to sensing require synchronization in order to coordinate the actuators through distributed control algorithms.

Traditional approaches

Time synchronization is a long-studied subject in distributed systems, and a number of well-known algorithms have been developed for different conditions. For example, there is a well-known algorithm by Lamport that provides a consistent ordering of all events in a distributed system, labeling each event x with a distinct time stamp L_x, such that:

(a) $L_x \neq L_y$ for all unique events x and y,

(b) if event x precedes event y within a node $L_x<L_y$, and

(c) if x is the transmission of a message and y its reception at another node, $L_x <L_y$.

These Lamport time stamps do not provide true causality. Say the true time of event x is indicated as T_x; then, while it is true that $T_x <T_y \rightarrow L_x <L_y$, it is not true that $L_x <L_y \rightarrow T_x <T_y$. Such a true causal ordering requires other approaches, such as the use of vector time stamps.

A fundamental technique for two-node clock synchronization is known as Cristian's algorithm. A node A sends a request to node B (which has the reference clock) and receives back the value of B's clock, T_B. Node A records locally both the transmission time T_1 and the reception time T_2. This is illustrated in Figure 2.8

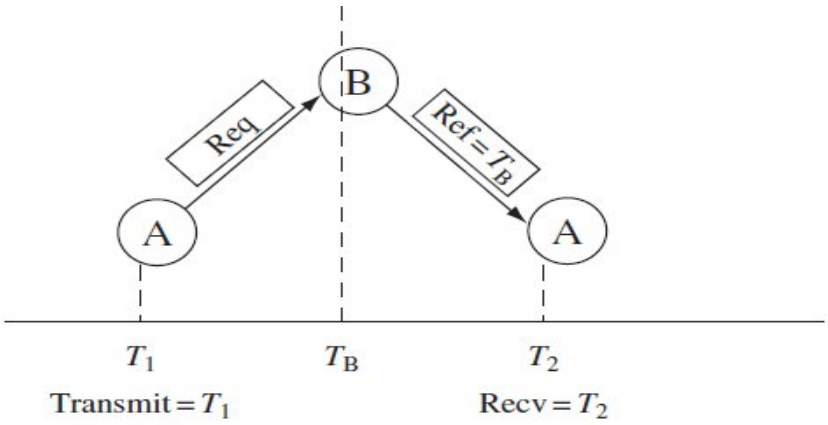


Fig 2.8: Cristian's synchronization algorithm

In Cristian's time-synchronization algorithm, there are many sources of uncertainty and delay, which impact its accuracy. In general, message latency can be decomposed into four components, each of which contributes to uncertainty:

- Send time – which includes any processing time and time taken to assemble and move the message to the link layer.
- Access time – which includes random delays while the message is buffered at the link layer due to contention and collisions.
- Propagation time – which is the time taken for point-to-point message travel. While negligible for a single link, this may be a dominant term over multiple hops if there is network congestion.
- Receive time – which is the time taken to process the message and record its arrival.

Good estimates of the message latency as well as the processing latency within the reference node must be obtained for Cristian's algorithm. The simplest estimate is to approximate the message propagation time as $(T_2 - T_1)/2$. If the processing delay is known to be I, then a better estimate is $(T_2 - T_1 - I)/2$. More sophisticated approaches take several round-trip delay samples and use minimum or mean delays after outlier removal.

The network time protocol (NTP) is used widely on the Internet for time synchronization. It uses a hierarchy of reference time servers providing synchronization to querying clients, essentially using Cristian's algorithm.

Fine-grained clock synchronization

Several algorithms have been proposed for time synchronization in WSN. These all utilize time measurement-based message exchanges between nodes from time to time in order to synchronize the clocks on different nodes.

Reference broadcast synchronization (RBS)

The reference broadcast synchronization algorithm (RBS) exploits the broadcast nature of wireless channels. RBS works as follows. Consider the scenario shown in Figure 2.9, with three nodes A, B, and C within the same broadcast domain. If B is a beacon node, it broadcasts the reference signal (which contains no timing information) that is received by both A and C simultaneously (neglecting propagation delay). The two receivers record the local time when the reference signal was received. Nodes A and C then exchange this local time stamp through separate messages. This is sufficient for the two receivers to determine their relative offsets at the time of

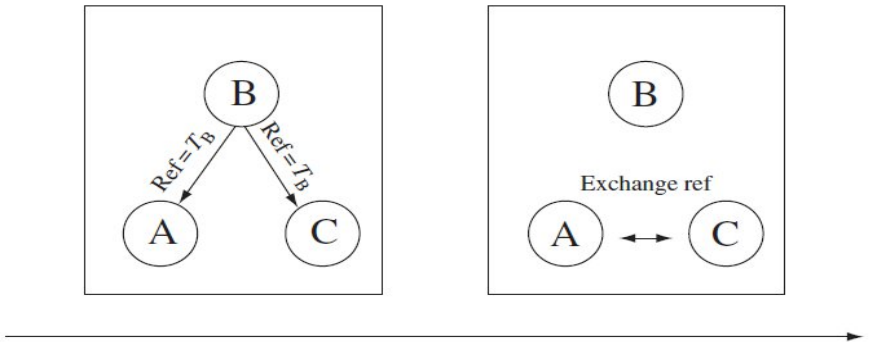


Fig 2.9: The reference broadcast synchronization (RBS) technique

reference message reception. This basic scheme, which is quite similar to the Cesium Spray mechanism for synchronizing GPS-equipped nodes, can be extended to greater numbers of receivers. Improvements can be made by incorporating multiple reference broadcasts, which can help mitigate reception errors, as well as by estimating clock drifts.

A key feature of RBS is that it eliminates sender-side uncertainty completely. In scenarios where sender delays could be significant (particularly when time stamping has to be performed at the application layer instead of the link layer) this results in improved synchronization. RBS can be extended to a multi-hop scenario as follows. If there are several separate reference beacons, each has its own broadcast domain that may overlap with the others. Receivers that lie in the overlapping region (in the broadcast domains of multiple references) provide "bridges" that allow nodes across these domains to determine the relationship between their local clocks, e.g., if nodes A and C are in range of reference B and nodes C and D are in range of reference E, then node C provides this bridge. In a large network, different paths through the temporal graph, representing connections between nodes sharing the same reference broadcast domain, provide different ways to convert times between arbitrary nodes. For efficiency, instead of computing these conversions *a priori* using global network information, these conversions can also be performed locally, on-the-fly, as packets traverse the network.

In particular, it is noted that the basic RBS scheme is composed only of a series of independent pair-wise synchronizations. This does not ensure global consistency in the following

sense. Consider three nodes A, B, and C in the same domain, whose pair-wise offsets are determined through RBS. There is no guarantee that the estimates obtained of $C_A(t) - C_B(t)$ and $C_B(t) - C_C(t)$ add up to the estimate obtained for $C_A(t) - C_C(t)$. An alternative technique has been developed for obtaining globally consistent minimum-variance pair-wise synchronization estimates, based on flow techniques for resistive networks.

Pair-wise sender–receiver synchronization (TPSN)

The timing-sync protocol for sensor networks (TPSN) provides for classical sender–receiver synchronization, similar to Cristian's algorithm. As shown in Figure 2.10, node A transmits a message that is stamped locally at node A as T_1. This is received at node B, which stamps the reception time as its local time T_2. Node B then sends the packet back to node A, marking the transmission time locally at B as T_3. This is finally received at node A, which marks the reception time as T_4. Let the clock offset between nodes A and B be Δ and the propagation delay between them is d. Then

$$T_2 = T_1 + \Delta + d$$
$$T_4 = T_3 - \Delta + d$$

Then,

$$\Delta = ((T_2 - T_4) - (T_1 - T_3))/2$$
$$d = ((T_2 + T_4) - (T_1 + T_3))/2$$

Network-wide time synchronization in TPSN is obtained level-by-level on a tree structure. Nodes at level 1 first synchronize with the root node. Nodes at level 2 then each synchronize with one node at level 1 and so on until all nodes in the network are synchronized with respect to the root.

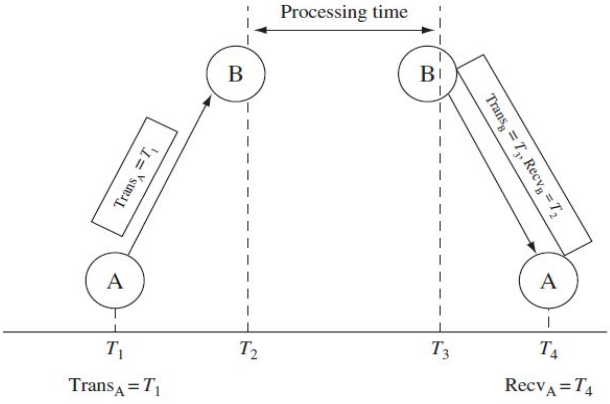


Fig 2.10: Basic sender – receiver synchronization technique used in TPSN

Assume that sender-side uncertainty can be mitigated by performing time stamping close to transmissions and receptions at the link layer. The synchronization error with the pair-wise sender–receiver technique can actually provide twice the accuracy of the receiver–receiver technique used in RBS over a single link. This can potentially translate to even more significant gains over multiple hops. The lightweight time synchronization (LTS) technique is also a similar tree-based pair-wise sender–receiver synchronization technique.

Flooding time synchronization protocol (FTSP)

The flooding time synchronization protocol (FTSP) aims to further reduce the following sources of uncertainties, which exist in both RBS and TPSN:

1. Interrupt handling time: This is the delay in waiting for the processor to complete its current instruction before transferring the message in parts to the radio.
2. Modulation/encoding time: This is the time taken by the radio to perform modulation and encoding at the transmitter, and the corresponding demodulation and decoding at the receiver.

FTSP uses a broadcast from a single sender to synchronize multiple receivers. However, unlike RBS, the sender actually broadcasts a time measurement, and the receivers do not exchange messages among themselves. Each broadcast provides a synchronization point (a global–local time pair) to each receiver. FTSP has two main components:

1. Multiple time measurements: The sender takes several time stamp measurements during transmission, one at each byte boundary after a set of SYNC bytes used for byte alignment. These measurements are normalized by subtracting an appropriate multiple of the byte transmission time, and only the minimum of these multiple measurements is embedded into the message. At the receiver too, multiple time measurements are taken and the minimum of those is used as the receiver time. This serves to reduce the jitter significantly in interrupt handling and the (de)coding and (de)modulation times. With as few as six time stamps, an order of magnitude improvement in precision can be obtained on a Mica Mote platform (from the order of tens of microseconds to the order of about one microsecond).
2. Flooded messaging: To propagate the synchronization information, a flooding approach is used. First, a single uniquely identifiable node in the network provides the global clock. The reception of each broadcast message allows a receiver to accumulate a reference synchronization point. When a receiver accumulates several reference points, it becomes synchronized itself (e.g. using a regression line to estimate the local clock drift). Nodes can collect reference points either from the global reference node, or from other nodes that are already synchronized. The frequency of the flooding provides a tradeoff between synchronization accuracy and overhead.

Predictive time synchronization

In the real world, clock drift can vary over time quite drastically due to environmental temperature and humidity changes. This is the reason clock drifts must be continually reassessed. The navel approach to this reassessment is tore-synchronize nodes periodically at the same interval. However, a static synchronization period must be chosen conservatively to accommodate a range of environments. This does not take into account the possibility of temporal correlations in clock drift. It will thus incur an unnecessarily high overhead in many cases.

This problem is addressed by the predictive synchronization mechanism in which the frequency of inter-node time sampling is adaptively adjusted. It has been determined through an empirical study that environments are characterized by a time constant T over which drift rates are highly correlated, which can be determined through a learning phase. Depending on the time sampling period S, a window of T/S prior sample measurements is used in this technique not only to predict the clock drift (through linear regression), but also to estimate the error in the prediction. A MIMD technique is used to adapt the sampling period: if the prediction error is above a desirable threshold, the sampling period S is reduced multiplicatively; and if it is below threshold, the

sampling period is increased accordingly. This adaptive scheme provides for robust long-term synchronization in a self-configuring manner.

2 Mark Questions

1. What is localization? Explain RSSI
2. Differentiate coarse and fine grained localization methods.
3. What is need for time synchronization?
4. Why does not TCP work well in ad hoc network?
5. What are the types of localization? Explain AoA
6. Differentiate coarse and fine grained time synchronization methods.
7. What is localization error?
8. Explain about synchronization phase?
9. Define access time and sending time
10. What is Hidden and Exposed Node Problem?
11. Differentiate coarse and fine grained localization methods?
12. What is the need for time synchronization?

12 Mark Questions

1. Explain in detail about different network topology in ad- hoc
2. List mobility models and explain how the mobility models influence the performance of the network protocols.
3. Describe in detail about the localization techniques in ad-hoc
4. Explain in detail about the various methods of time synchronization
5. Give detailed explanation for Fine-grained node localization using detailed information
6. Explain briefly Fine-grained clock synchronization.
7. Discuss in detail about Coarse-grained node localization using minimal Information.
8. Write notes on network localization.