Sensor network overview

Distributed Systems, Fall 2008

David Volquartz Lebech

December 5, 2008

Contents

1 Introduction

Sensor networks are getting increasingly employed in very changing settings from everyday life situations to the battlefield of war. Sensor networks provide a link between the electronic computing world and the physical world by interpreting properties of the environments, they are deployed in. This interpretation – a fancy word for sensing – can be anything from simple temperature measurements, over more comprehensive climate data gathering, to advanced vision or sound analysis.

This paper surveys some of the interesting areas of sensor network research and tries to give an idea about what a sensor network is, and what problems previous research have tried to tackle. Instead of going into detail with a single topic, I will describe two interesting issues with sensor networks, that have been subject to a great deal of research.

In section 2, I will first shortly define what a sensor network is and what it is comprised of. In section 3 and 4, I will look at the *positioning* and *coverage* problems which are problems that arise when deploying sensor networks in a random setting. Each section will end by a short discussion on some of the issues with the presented research. In section 5, I conclude the paper and summarize the main points of my findings.

1.1 Motivation

Using Sun SPOTs for sensor networks [22, Only abstract is in English] was the title of my Bachelor's thesis, written together with Michael Thomassen in Spring 2008 at the Technical University of Denmark. The research for the thesis was very practical oriented and the purpose was exploring the possibilities of the Sun SPOT (Small Programmable Object Technology), a small, wireless sensor node that has a very easy to use Java programming interface (API) [1].

Our thesis only covered very basic sensor network theory and we did not go into much detail with previous sensor network research. Choosing this as my topic for the term paper thus seems logical and a relevant supplement to my knowledge in the field.

2 Sensor Networks

A *wireless ad-hoc sensor network*[1] is a network consisting of small, wireless and sometimes mobile devices, usually called *nodes*, that are characterized by their small size, low battery life and low computation power [8]. A sensor network is usually used to provide sensor coverage of a specific physical area of interest. The nodes in the sensor network are often distributed randomly across this area but with a big enough *node density* that they are able to communicate efficiently.

The term ad-hoc refers to the fact that nodes in a sensor network are *autonomous devices*. A sensor network as a whole has to be able to handle nodes coming and going and each individual node has to function independently of the other nodes in the network. In other words, a sensor network is a perfect example of a distributed system.

Although sensor networks can also be non ad-hoc and deployment can be managed manually, this is not preferable in at least two situations [21]:

- When the sensor network is deployed in remote or hostile areas.
- When the amount of nodes is big, manual deployment is expensive.

2.1 Algorithms

Every algorithm that is implemented in a sensor network can be categorized as either a *centralized*, *distributed* or *localized* algorithm. In a centralized algorithm, each node sends relevant information to a single point where the

[1] Throughout this paper, I use "sensor network" to imply a wireless ad-hoc sensor network

relevant computation for the algorithm is executed. A distributed algorithm exploits the fact that the system is ad-hoc and a distributed system. Every node does its own calculations, typically with some message passing to other nodes. A localized algorithm is a special case of a distributed algorithm that is run on a subset of the sensor network [16].

2.2 Communication

The communication protocol in a sensor network can be similar to the ones found in normal wireless networks, e.g. networks using the IEEE 802.11 protocols. In practice, however, protocols have been developed that better fit the scheme of sensor networks since power is a limiting factor, e.g. the IEEE 802.11.4 standard [8].

Another communication concern is how sensor data are routed through the network to get to either a base station or another node that is interested in the data. Several routing algorithms exist and as far as I know, they are all distributed. One example is the *Ad-hoc On-Demand Distance Vector* algorithm [17]. This algorithm is e.g. used as the routing algorithm for the Sun SPOT platform that I mentioned in the introduction. Research in optimizing routing algorithms for e.g. energy efficiency is a hot research topic and has enough material for several standalone papers, but I will not go into details with that in this paper.

3 Positioning

In most sensor network applications, knowing the position of individual nodes is important and relevant. Examples include environmental monitoring, movement tracking and location based routing [9]. If the nodes are placed according to a predefined pattern (i.e. deterministically) then there is no problem but as described in section 2, the nodes are usually distributed randomly, they could be moving or manual positioning could otherwise be impractical. Automatic positioning is thus a research problem of great interest.

3.1 Concepts and methods

This section introduces important concepts and methods related to positioning and is a make-up of the information found in [16, 11, 18, 9, 20].

Depending on the application, the nodes in a sensor network can be positioned either according to a *global/absolute*, *relative* or *local* coordinate system. In global positioning, the position of nodes are put into some kind of global reference system, e.g. relative to the GPS system. Relative positioning means that nodes know their position relative to the network they are in but are not necessarily coherent with coordinate systems of other networks. Local positioned nodes only know their position according to their immediate surroundings and are not necessarily coherent with the entire network.

Usually, absolute and relative positioning is carried out with the help of a number of *anchor nodes* (also called *beacon* or *seed* nodes), that is, nodes that know their relative or absolute coordinates beforehand, either because they have been manually positioned or typically because they are connected to a GPS receiver. Other nodes can turn into anchor notes during the positioning process for further refinement.

Positioning can be based on nodes with or without *signaling* abilities. Received signal strength indication (RSSI), angle of arrival (AOA), time of arrival (TOA) and time difference of arrival (TDAO) are examples of signaling data that are often used for positioning in sensor networks. RSSI values can be obtained from the radio signals of the nodes, a feature that is usually already present in the nodes due to the wireless communication, but often with very low accuracy. AOA, TOA and TDOA requires sophisticated hardware (antenna arrays and/or ultrasound) which increases production cost and lowers battery life while at the same time requiring line of sight (LOS) between the nodes. For the same reasons, built-in GPS receivers are usually also ruled out.

Another approach for measuring range is using only *hops* or *mere connectivity*, i.e. using only information about the network topology. Methods using connectivity are often regarded as non range-based techniques in contrast with methods using RSSI or TOA which are considered range-based.

The common goal for all range-based techniques is (as the name implies) to obtain some kind of measure for either range, angle or both. This measurement can then be used to do a *trilateration*, *multilateration* or *triangulation*[2]. Shortly described, trilateration is the process of finding positions based on (at least 3) distances (e.g. from TOA, RSSI), multilateration uses TDOA and triangulation uses (at least 2) angles and a baseline distance (e.g. from AOA)[3].

3.2 Algorithms research

Several algorithms have been proposed for solving the positioning problem. In this section, I will describe a few of them.

The algorithm presented in [18] uses the basic RSSI as measurement of distance. The authors acknowledge the low accuracy of RSSI but make some assumptions to improve optimism about their approach: Dense interconnectivity and limited mobility in the sensor network. The paper describes an approach that the authors call *coorporative ranging* where positions are first calculated locally (using trilateration[4]) for every node in the network and then propagated slowly, ending up with a global consensus about the positions of the nodes.

Using the *Assumption Based Coordinates* (ABC) and *Triangulation via Extended Range and Redundant Association of Intermediate Nodes* (TERRAIN) algorithms their results show as low as 5% error on the positions. This is

[2] The literature sometimes seems to use the terms interchangeably.

[3] Clear definitions are difficult to find so the descriptions are based on Wikipedia [4, 2, 3].

[4] The authors use the term triangulation but based on the description it is actually trilateration

counterintuitive with the fact that RSSI values can have a 50% measure error as the authors themselves state. However, it is worth remembering that the results are under the assumption of a dense network. [19] is a work by the same author (Savirese *et al.*) which is based on some of the same concepts (trilateration and TERRAIN) but with a more in-depth discussion. The results in this paper show errors of 33% but is not directly comparable with [18].

Both [11] and [20] use algorithms that are based on a mathematical concept called *multidimensional scaling* (MDS). [20] describes an algorithm called *MDS-MAP* which is interesting since it is based only on mere connectivity of nodes. The algorithm works in three steps. First, the distance between all pairs of nodes are calculated, resulting in a distance matrix. Second, "classical MDS" is applied to the distance matrix and a relative network map is constructed. Third, given a few anchor nodes, the absolute positions for the nodes can be found based on the relative map.

One of the drawbacks of the MDS-MAP algorithm is that it is centralized. [11] describes an MDS algorithm that is distributed but uses the standard RSSI measurement for distance instead of mere connectivity. The paper states that sensor networks usually are deployed in *anisotropic* conditions, i.e. surroundings with obstructions or similar changing environments throughout the network. The authors furthermore say, that other algorithms often lack the ability to work under anisotropic conditions and based on this observation, they describe an MDS algorithm that apparently performs well under these conditions.

The algorithm works by first having some start anchor node send its coordinates through the entire network (a *flooding* of the network). The message will then eventually reach some end anchor nodes that transmit their position data back to the start anchor nodes. Using RSSI and MDS, relative positions are found for neighboring nodes and during the flooding of the network the absolute positions are propagated.

The above algorithms, and indeed most positioning algorithms in general, do not take *mobility* into account. [9] describes an algorithm that can be used when the nodes in a sensor network are moving. In fact, the authors find that the algorithm performs better when the nodes are moving.

The algorithm presented in the paper is based on a positioning scheme used for robotics called *Monte Carlo localization* (MCL), although the authors underline that there are significant differences between positioning for robots and sensor networks, e.g. a robot usually has control over its movement while a sensor network node does not.

Cut down to the bone, the algorithm works in two phases: In the *prediction* phase, a set of *possible* positions are calculated based on the previous position and in the *update* phase, the position is calculated based on new measurements of the surroundings, i.e. anchor nodes.

One of the quite clever things about the algorithm is, that it utilizes the fact that the nodes are moving. By looking at which anchor nodes are visible before and after each iteration of the algorithm, and knowing an upper bound to its moving speed, a node can filter and predict its position within certain bounds. On the downside, the algorithm requires about 1 anchor node per 10 nodes to produce good results and message passing requirements are not discussed in detail.

[16] is a decent general survey of the solutions for the positioning problem which also touches upon some of the above mentioned algorithms.

3.3 Discussion

A common trait for most of the positioning algorithms is that the authors tend to praise their own method and emphasize how the others have shortcomings and drawbacks. This is immediately apparent by just reading the introduction and conclusion of the research papers, I reference. Whether this is just academic pride or an indication of how difficult the positioning problem really is, I do not know, but there is certainly a lot of difficulties involved with automatic positioning.

It is difficult to directly compare performance between the algorithms since they make different assumptions and network settings. In fact, only [9] offers a direct comparison between positioning methods, namely the Monte Carlo localization, Amorphous and Centroid. Another paper, [12], offers an in-depth analysis of three algorithms, including the one presented in [19]. The paper concludes that the choice of algorithm must depend on the specific sensor network that applications are built for. Algorithms that work only by hop-counting or mere connectivity are very cost efficient but algorithms based on TOA or TDOA can be more precise but expensive and requires line of sight.

All the papers use simulations to produce their results. Indeed, physical implementation is expensive and time consuming but would probably give different and better – or worse, depending on your point of view – results. Even though a lot of randomness is built into many of the simulations, a real sensor network is much more unreliable than a simulation. [19] supports this claim in its conclusion.

The energy problem for sensor networks is important and optimizing algorithms for energy efficiency is a key factor of success for a real life sensor network. This issue is not addressed by most of the papers. However, [11] states that further research needs to be done with this in mind and the authors also propose an on demand positioning algorithm to lower the need for constant updating positions.

Overall, the positioning problem has a variety of solutions, none of them being all-purpose and well suited for every situation. It is apparent from the above discussion, that a lot of problems still persist which makes this a hot research topic.

4 Coverage

One of the arguments for employing automatic positioning in a sensor network is the ability to label sensor data for certain physical areas. But what if an area does not even have a sensor in it? How do we find out if it has or not? The coverage problem seeks to answer the question about how well a sensor network covers a specific area, or put another way, the *quality of service* of a sensor network [15].

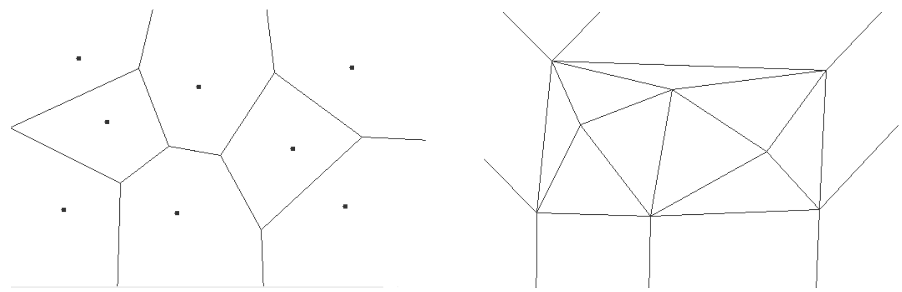
Different coverage problems exist. One example is the *Art Gallery Problem*: How to determine the number of observers necessary for covering an art gallery so that every point is monitored by at least one observer [15, 10]. Another way of looking at the coverage problem is how well an area is covered *over time*. Because of the need for low energy consumption, it is easy to imagine a network where several sensors cover the same physical area but are sleeping most of the time and then waking up at different times to provide a frequent flow of sensor data. How this type of coverage is determined will not be discussed in further detail here but is described in [21].

4.1 Concepts and methods

In the coverage problem, it is assumed that the position of every node is already known beforehand. This means that the coverage problem is actually building on top of the positioning problem unless of course the nodes in the sensor network are distributed manually [15].

Often, we are interested in finding out if an area is covered by more than one sensor. This is called *k-coverage* where k is the number of sensors covering the area. The reasons for $k > 1$ being preferable are, among others, better fault-tolerance, an inherited property of the positioning problem (recall that 3 or more neighbor nodes are required for trilateration) or simply better environmental monitoring [10].

In the papers I have read, the authors basically only use two different



(a) (b)

Figure 1: (a) Example of Voronoi Diagram (b) Delaunay triangulation corresponding to (a). Both are constructed using [7]

approaches to solving the coverage problem. The first is concerned with finding the *maximal breach path* and the *maximal support path*, which are measures for worst and best case coverage, respectively [15]. The other approach regards the coverage problem as a decision problem where the goal is to determine whether every node in the sensor network is k-covered.

While both methods rely on mathematics and computational geometry, they are different in their implementations. The first uses the *Voronoi diagram* and *Delaunay triangulation*. Simply put, a Voronoi diagram splits an area (e.g. a sensor network) into several polygons, where each point inside the polygon (i.e. the area that the polygon covers) is closest to the same node (see figure 1(a)). In a Delaunay triangulation, each triangle is constructed such that a circumscribed circle for the triangle does not contain any nodes except the nodes that the triangle touches (see figure 1(b)). A Voronoi diagram an Delaunay triangulation are duals of each other [15, 13].

4.2 Algorithms research

The coverage problem is not as extensively researched as the positioning problem. I have looked at the proposed algorithms in [15, 13, 10] but will not explain them in detail since they all contain quite a bit of mathematics and geometry.

In [15], an algorithm is presented that use a combination of the Voronoi diagram, Delaunay triangulation and graph algorithms. The algorithm finds the maximal breach path and maximal support path, it is centralized and is shown to be optimal and with a worst case time complexity of $O(n^2 \log n)$ (n is number of nodes), assuming that position data is present for every node in the network. The maximal breach path and maximal support path are then used heuristically to improve coverage and the results show decent improvements to the coverage.

Although acknowledging [15] as “pioneering work”, [13] claims that it has some problems. One of the problems is, that relying on e.g. Voronoi diagrams is inefficient and the correctness of using the diagrams is not proven. [13] proposes a localized algorithm that solves what the authors call the *best coverage problem* which is in fact the same as finding the maximal support path. The algorithm is distributed and uses Delaunay triangulation which the authors this time prove to be correct and actually find an optimal solution.

In the paper, three algorithms are presented based on the same concepts. The first solves the best coverage problem optimally while the two others are extensions, e.g. for better energy consumption. All algorithms are shown to have $O(n \log n)$ time complexity (n is number of nodes). Unfortunately, the work is purely based on theory and no simulation results are given and it is thus impossible to verify that the given solution is efficient.

Finally, [10] presents an algorithm that is concerned with solving the k-coverage problem. The paper describes two related problems, k-Non-unit-disk-Coverage (k-NC) and k-Unit-disk-Coverage (k-UC). In k-NC, all sensors have different coverage radii while in k-UC they are the same. The authors propose centralized algorithms for solving the problems which “can be easily translated to distributed protocols”. However, the authors do not describe what “easy” means.

The authors prove that if the coverage radius of every node intersects with at least k other nodes' coverage radii, then the sensor network is k-covered. This is somewhat intuitive and the simulation results are also as could be expected, i.e. increasing the number of sensors or the average sensing range in a specific network size linearly increases k-coverage. The time complexity is decent, $O(nd \log d)$ where n is number of nodes and d is the maximum number of neighboring nodes for any node, which in cases with a low neighbor count approaches linear time $O(n)$, while the worst case complexity is $O(n^2 \log n)$. This is exactly the same as in [15].

4.3 Discussion

Although I have only covered very few papers dealing with the coverage problem, the mathematical foundation seems much more solid (because of the formal proofs) than for the positioning problem. This claim is probably unjust and too generalizing but nevertheless, this is indeed the case for the papers I have read and surveyed. In the same breath, it is strange that none of the positioning papers discuss time and space complexity of their algorithms except [20], that only shortly mentions the complexity of multidimensional scaling.

Having a good mathematical foundation is a nice starting point but unfortunately, the coverage problem is – like the positioning problem – mostly dealt with theoretically, i.e. through simulations, and real world networks are different than simulations. Another problem is that centralized approaches might work well in simulations but not in real sensor networks. [6] also concludes that distributed methods need further research.

Even though the coverage problem is an interesting topic, it is – in my opinion – questionable how relevant the problem really is, compared to the positioning problem. Given that nodes in a sensor network are very low cost, and under the assumptions of a uniform distribution of the sensors to an area, good coverage is achieved automatically by just increasing the number of deployed sensors sufficiently. This is somewhat backed up by the results in [10] where relatively few nodes per area are needed for a decent (≥ 2) k-coverage.

Probably the most relevant coverage algorithms are those that measure and optimize coverage with respect to time. As already mentioned, an algorithm is described in [21] and a general survey of the coverage problem can be found in [6].

5 Conclusion

In this paper, I have surveyed some of the research related to sensor networks. In particular, I have looked at the positioning problem and the coverage problem and presented a short overview of the two fields. Both fields are heavily researched and a lot of interesting solutions exist.

The positioning and coverage problems are not by definition related to distributed systems. But one of the reoccurring questions is whether a solution should be centralized or distributed. For very large sensor networks, a centralized method is intuitively not a good choice since it induces a large message overhead. And if the network is mobile, centralization is entirely impractical. On the other hand, and inefficient distributed algorithm can have trouble coordinating results, possibly creating even more message passing and overhead. However, a distributed algorithm is preferable in most cases because it suits the self-organizing structure of the sensor network better.

It is peculiar that no centralized routing algorithms exist for sensor networks while several centralized methods exist for the positioning and coverage problem in sensor networks. An explanation for this could be, that routing simply has been around for a longer time than sensor networks. Another explanation could be, that centralization fits better into positioning and coverage, even though it is against the distributed nature of sensor networks.

Unfortunately, most of the results in the research are only based on simulations and assumptions that do not fit exactly into real life settings. Energy optimization is often just theoretical and most solutions are not very fault tolerant, e.g. concerning normal conditions like (big) fluctuations in the radio range for each sensor node. An analysis of the trade-offs between calculating positions/coverage and manual deployment or between centralized and distributed computation could be very interesting in a real life implementation. The problem is of course that all this would be an expensive and large scale project to undertake.

Perhaps this fact is the reason why there is only a few (publicly) available papers on real sensor network implementations. And in fact, the positioning and coverage problems do not directly relate to many of these. There is at least two reasons for this:

- Many real applications need very precise control over where the sensors are deployed because the *purpose* of the sensor is more important than its position, e.g. "the-sensor-that-monitors-feature-3" is more important than "the-sensor-at-position-x-y" and feature3's position may already be well known. This thus renders both coverage and positioning algorithms superfluous.

- Manual node deployment can optimize coverage (i.e. not overestimate the number of nodes needed) and is most of the time the preferred way of distributing sensor nodes in a real life setting.

One example out of many is described in [14, 5]. A sensor network is set up on Great Duck Island for monitoring the habitats of Storm Petrels. The sensor network does not use positioning or coverage algorithms for anything but discusses energy requirements and communication because the deployment time of the project is several months at the time.

Sensor networks is a hot research topic and this paper is not at all exhaustive. Since communication is the foundation for every network, and thus also a sensor network, it could be interesting to compare different routing algorithms and communication protocols. Much of the research in this field is concerned with energy efficiency because this is a key factor for a successful and long living sensor network. A general survey of energy efficient solutions could be another topic for future research in sensor networks.

For the future, I hope to see even more research under real life conditions. Sensor networks are an exciting topic and with the development of even smaller, more energy efficient and multipurpose sensor nodes, the possible applications for sensor networks keep growing.

References

[1] Homepage for sun small programmable object technology (sun spot). `http://www.sunspotworld.com/`.

[2] Wikipedia: Multilateration. `http://en.wikipedia.org/wiki/Multilateration`.

[3] Wikipedia: Triangulation. `http://en.wikipedia.org/wiki/Triangulation`.

[4] Wikipedia: Trilateration. `http://en.wikipedia.org/wiki/Trilateration`.

[5] Martha Baer. The ultimate on-the-fly network. *Wired Magazine*, (11.12), December 2003. http://www.wired.com/wired/archive/11.12/network.html.

[6] Mihaela Cardei and Jie Wu. Energy-efficient coverage problems in wireless ad-hoc sensor networks. *Computer Communications*, 29(4):413–420, February 2006.

[7] Paul Chew. Voronoi/delaunay applet. `http://www.cs.cornell.edu/home/chew/Delaunay.html`.

[8] George Coulouris, Jean Dollimore, and Tim Kindberg. *Distributed Systems Concepts and Design*, chapter 16. Addison-Wesley, 4th edition, 2005.

[9] Lingxuan Hu and David Evans. Localization for mobile sensor networks. In *Tenth Annual International Conference on Mobile Computing and Networking (MobiCom 2004)*, pages 45–57, Sep.-Oct. 2004.

[10] Chi-Fu Huang and Yu-Chee Tseng. The coverage problem in a wireless sensor network. *Mobile Networks and Applications*, 10(4):519–528, August 2005.

[11] Xiang Ji and Hongyuan Zha. Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, pages 2652–2661, March 2004.

[12] Koen Langendoen and Niels Reijers. Distributed localization in wireless sensor networks: a quantitative comparison. *Computer Networks*, 43(4):499–518, November 2003.

[13] Xiang-Yang Li, Peng-Jun Wan, and Ophir Frieder. Coverage in wireless ad-hoc sensor networks. *IEEE Transactions on Computers*, 52(6):753–763, 2003.

[14] Alan Mainwaring, Joseph Polastre, Robert Szewczyk, David Culler, and John Anderson. Wireless sensor networks for habitat monitoring. In *1st ACM international workshop on Wireless sensor networks and applications*, pages 88–97, September 2002.

[15] Seapahn Meguerdichian, Farinaz Koushanfar, Miodrag Potkonjak, and Mani B. Srivastava. Coverage problems in wireless ad-hoc sensor networks. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1380–1387, April 2001.

[16] D. Niculescu. Positioning in ad hoc sensor networks. *IEEE Network*, 18(4):24–29, July-Aug. 2004.

[17] C. Perkins, E. Belding-Royer, and S. Das. Rfc 3561: Ad hoc on-demand distance vector (aodv) routing, July 2003. `http://tools.ietf.org/html/rfc3561`.

[18] Chris Savarese, Jan Rabaey, and Jan Beutel. Locationing in distributed ad-hoc wireless sensor networks. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2037–2040, 2001.

[19] Chris Savarese, Jan Rabaey, and Koen Langendoen. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In *USENIX 2002 Annual Technical*, pages 317–327, June 2002.

[20] Yi Shang, Wheeler Ruml, Ying Zhang, and Markus P. J. Fromherz. Localization from mere connectivity. In *Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 201–212, June 2003.

[21] Sasa Slijepcevic and Miodrag Potkonjak. Power efficient organization of wireless sensor networks. In *IEEE International Conference on Communications*, pages 472–476, June 2001.

[22] Michael Thomassen and David Lebech. Anvendelse af sun spots til sensornetværk (eng. using sun spots for sensor networks). `http://ix.cs.uoregon.edu/~lebech/cis630/SunSPOT.pdf`, June 2008.