# Distributed Detection using Parley with Soft Decisions

Robert E. Van Dyck
National Institute of Standards and Technology
Gaithersburg, MD 20899
e-mail: vandyck@antd.nist.gov

*Abstract* — **In this paper, we propose an extension to the nth root parley distributed detection algorithm of Swaszek and Willet. Instead of making a single "hard" decision at each sensor node, a two bit quantizer is used to choose the hypothesis and also to provide a confidence measure of this decision. These "soft" decisions are broadcast to all nodes, and they are used to create a stopping rule that reduces the number of parleys. For the Bayesian criterion, the probability of error is unchanged, and it is equal to that of a central processor; for the Neyman-Pearson criterion, the receiver operating curve is essentially the same as that of a central processor. The performance is also compared to that obtained using one bit decision makers and the majority fusion rule. Simulation results are provided for the Gaussian shift in mean problem assuming an ideal channel and the binary symmetric channel.**

## I. Introduction

With the coming availability of low cost, short range radios along with advances in wireless networking, it is expected that smart sensor networks will become commonly deployed [1]. In these networks, each node will be equipped with a variety of sensors, such as acoustic, seismic, infrared, *etc.* These nodes may be organized in clusters such that a locally occurring event can be detected by most, if not all, the nodes in the cluster. Each node will have sufficient processing power to make a decision, and it will be able to broadcast this decision to the other nodes. One node may act as the cluster master, and it may also contain a longer range radio; however, we presently do not need to assume this.

While the design of the physical and medium access control (MAC) layers are important considerations in a wireless smart sensor network, in this paper we focus only on the distributed detection problem. The goal is to develop simple, efficient algorithms that fuse the local decisions into a global decision for the cluster. This decision can then be transferred out of the cluster over a multi-hop wireless network, perhaps using mobile ad-hoc network (MANET) routing algorithms. For power consumption and low probability of intercept considerations, it is desired that the number of bits transmitted among the local nodes be minimized. At the same time, the global probability of error must also be minimized, preferably allowing the cluster to achieve the same probability of error as a central processor.

The proposed algorithm is an extension of the nth root parley method of Swaszek and Willet [2]. It is important to point out that their algorithm does provide the same probability of error as an optimal central processor. This is done by multiple iterations, called parleys, of hard decision data from each local node to all its neighbors. One advantage of their method is that there is no single fusion center that can be a point of failure. However, a potential disadvantage is that a number of iterations must be completed before a consensus decision is achieved.

Our basic idea is to use a two bit quantizer to send soft decisions from each node to all the others. One bit determines the hypothesis chosen, while the other one indicates how close the likelihood ratio is to the threshold. The same *a posteriori* probability calculations from the original algorithm are performed, but now the requirement of a consensus is relaxed. Given N nodes, the algorithm requires that only some proportion of these nodes agree on the same hypothesis, while the other nodes must all have a decision in the quantization bin closest to the threshold. The simulation results show that the number of parleys can be substantially reduced, with essentially no effect on the probability of error.

In the next section, the basic parley algorithm is briefly described. Section III discusses the quantization and the selection of the early stopping rules, and Section IV provides the analytical performance results of a central detector for the Gaussian shift in mean problem. Section V contains the simulation results, while Section VI provides conclusions and future areas of study.

## II. Parley Algorithm

For completeness, we describe the binary hypothesis parley algorithm given in [2]. Assume each of $N$ sensors measures a phenomenon and computes a likelihood function $\Lambda(r_i)$, for each sensor $i = 1, \cdots, N$. The optimum centralized test is given by

$$\prod_{i=1}^{N} \Lambda(r_i) \underset{H_0}{\overset{H_1}{\gtrless}} \lambda \tag{1}$$

where $\lambda$ is a threshold. The decentralized test performed at each sensor during each iteration is given by

$$\Lambda(r_i) \underset{u_{i,m}=0}{\overset{u_{i,m}=1}{\gtrless}} \lambda_{i,m}. \tag{2}$$

In Eq. (2), $u_{i,m}$ is the hard decision and $\lambda_{i,m}$ is the local threshold, both for the $i$th sensor at iteration $m$. These thresholds need to be determined.

A consensus occurs if all nodes choose the same hypothesis. When this happens, the product of all the local likelihood functions is either greater or less than the product of the local

[0]This work was supported by the Advanced Research and Development Activity (ARDA) under contract number 706400.

thresholds. The main concept of the parley algorithm is given by the following proposition. "If the thresholds at each round of the parley satisfy $\prod_{i=1}^{N} \lambda_{i,m} = \lambda$, a consensus decision, if reached, matches the centralized decision exactly" [2]. From this, one can show that the optimal local thresholds are

$$\lambda_{i,m+1} = (\lambda \prod_{k=1}^{N} \frac{Pr(s_{k,m} \le \Lambda(r_k) \le t_{k,m}|H_0)}{Pr(s_{k,m} \le \Lambda(r_k) \le t_{k,m}|H_1)})^{1/N} \times \frac{Pr(s_{i,m} \le \Lambda(r_i) \le t_{i,m}|H_1)}{Pr(s_{i,m} \le \Lambda(r_i) \le t_{i,m}|H_0)}. \quad (3)$$

Here, $s_{k,m}$ and $t_{k,m}$ are the minimum and maximum values of $\Lambda(r_k)$, given the past hard decisions $u_{k,n}, n = 1, \cdots, m-1$.

Given the probability density functions (pdfs) for each hypothesis, one can compute the pdfs of the likelihood ratios conditioned on each hypothesis, and use this to select $\lambda$. Details are given below. The minimum and maximum values are chosen for each node's likelihood function, and the parley process begins. At each iteration, the likelihood function at each node is computed and compared to the local threshold. This threshold is calculated using the *a posteriori* probabilities determined by numerically integrating the pdfs of the other nodes' likelihood functions between the minimum and maximum values, as shown in Eq. (3). A hard decision is made, and this value is then broadcast to all other nodes. Since each node knows the previous minimum and maximum values, it can reduce these values for the next iteration by using the single information bit received from each of the other sensor nodes.

## III. Quantization and Early Stopping Rules

Figure 1 shows the pdfs of a likelihood function, $\Lambda(r_i)$, conditioned on the two hypotheses. Consider setting the threshold, say, $\lambda_{i,m} = 1$; then, it is straight-forward to numerically calculate $Pr(u_{i,m} = 0)$ and $Pr(u_{i,m}) = 1$ for each hypothesis. As an alternative, one can quantize the likelihood function using multiple bits. Since there is more information retained, performance should not be decreased. The question arises as to how to determine the quantizer thresholds and reproduction values to obtain this good detection performance. In general, the design is dependent on the fusion rule used, as well as the particular system architecture [3], *cf.* pages 107-110. For example, Longo *et al.* [4] suggest using the Bhattacharyya distance of the joint index space as a design criterion.

One needs to be somewhat careful, because the original nth root parley algorithm achieves the same probability of error of the central detector. While we want to reduce the number of iterations required for consensus, it is mandatory that the error performance remain essentially constant. To do this, we use a two bit quantizer where the first bit partitions the decision space identically to the original algorithm. That is, the thresholds $\lambda_{i,m}$ are retained, and above each one corresponds to the decision for $H_1$, while below is the decision for $H_0$. The second bit is used to segment each of these regions in two. For example in the Gaussian shift in mean problem, define a scale factor $SF$ such that the other two thresholds are given by

$$\lambda_{i,m,L} = \frac{\lambda_{i,m}}{SF} \quad (4)$$

$$\lambda_{i,m,H} = \lambda_{i,m} \times SF. \quad (5)$$

Since each received two bit $u_{i,m}$ still allows one to infer the *a posteriori* probabilities, Eq. (3) can still be used.

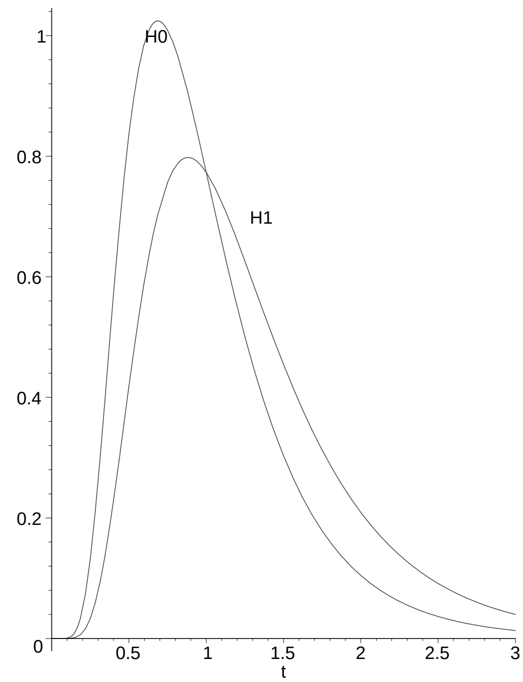


Figure 1: Probability density functions of the likelihood function conditioned on each hypothesis. These are denoted $\Lambda(r_i|H_0)$ and $\Lambda(r_i|H_1)$, respectively.

To reduce the number of parleys, while maintaining a good probability of error, one needs to develop an early stopping rule. While one could conceive of using maximum entropy or other information-theoretic ideas, we choose some simple heuristics that lead to good results. Firstly, we relax the condition that all $N$ nodes are in consensus to the rule that $X \times N$ nodes agree on the same hypothesis, where $X$ is typically in the range of 0.6 to 0.8. Secondly, if $X \times N$ nodes agree on hypothesis zero and the remaining nodes have likelihood ratios in the range $\lambda_{i,m} < \Lambda(r_i) < \lambda_{i,m,H}$, then the algorithm stops and says that the hypothesis is zero. Conversely, if $X \times N$ nodes agree on hypothesis one and the remaining nodes satisfy $\lambda_{i,m,L} < \Lambda(r_i) < \lambda_{i,m}$, the algorithm stops and says that the hypothesis is one. Otherwise, the next parley is conducted using the new *a posteriori* probabilities. Note that in Figure 1, $Pr\{\lambda_{i,m,L} < \Lambda(r_i|H_1) < \lambda_{i,m}\} = Pr\{\lambda_{i,m} < \Lambda(r_i|H_0) < \lambda_{i,m,H}\}$, when $\lambda_{i,m} = 1.0$.

## IV. Performance of Central Detector

The Gaussian shift in mean problem is a useful starting point for performance comparisons, since it is quite easy to calculate the probability of error for the central detector. This error is a lower bound on the performance of any distributed detection algorithm. The hypotheses are given by

$$p_R(r) = \frac{1}{\sqrt{2\pi}\sigma} \exp \frac{-(r \pm s)^2}{2\sigma^2}, \quad (6)$$

where the mean is $s$ for hypothesis one and $-s$ for hypothesis zero. Alternatively, one can shift these so that for hypothesis zero the mean is zero and for hypothesis one the mean is $m = 2s$. Defining the likelihood function [5] according to

$$\Gamma(\mathbf{R}) = \frac{\prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp \frac{-(R_i - m)^2}{2\sigma^2}}{\prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp \frac{-R_i^2}{2\sigma^2}}, \quad (7)$$

it is straight-forward to show that a sufficient statistic is

$$l = \frac{1}{\sqrt{N}\sigma}\sum_{i=1}^{N} R_i \overset{H_1}{\underset{H_0}{\gtrless}} \frac{\ln\eta}{d} + \frac{d}{2}, \qquad (8)$$

where $d := \sqrt{N}m/\sigma$, and $\eta$ depends on the costs and *a priori* probabilities. After a little manipulation, one can show that

$$H0: \ l = N(0,1) \qquad (9)$$

$$H1: \ l = N(\sqrt{N}m/\sigma, 1). \qquad (10)$$

To calculate the probability or error, Pr(e), for the central detector, let us assume that *a priori* probabilities of the two hypotheses are equal. Then, $Pr(e) = erfc_*(\ln\eta/d + d/2)$, where

$$erfc_*(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} \exp(-x^2/2)\, dx. \qquad (11)$$

Table 1 shows the probability of error as a function of the number of sensors and the distance between the two density functions.

| Sensors | $m = 2s$ | | |
|---|---|---|---|
| N | 0.2 | 0.5 | 1.0 |
| 1 | 0.460 | 0.401 | 0.309 |
| 2 | 0.444 | 0.362 | 0.240 |
| 5 | 0.412 | 0.288 | 0.132 |
| 10 | 0.376 | 0.215 | 0.0569 |
| 20 | 0.327 | 0.132 | 0.0127 |
| 50 | 0.240 | 0.0385 | $2.03 \times 10^{-4}$ |
| 100 | 0.159 | $6.21 \times 10^{-3}$ | $2.87 \times 10^{-7}$ |
| 200 | 0.079 | $2.03 \times 10^{-4}$ | $\approx 0$ |

Table 1: Analytical computation of probability of error for central detector.

For the Neyman-Pearson criterion, one can also calculate the probability of detection $P_D$ as a function of the probability of false alarm, $P_F$, for a given number of sensors. The equations are given by [5] $P_D = erfc_*(\ln\eta/d - d/2)$, and $P_F = erfc_*(\ln\eta/d + d/2)$. Figure 2 shows the receiver operating curves for various numbers of sensors when $m = 2s = 0.5$.

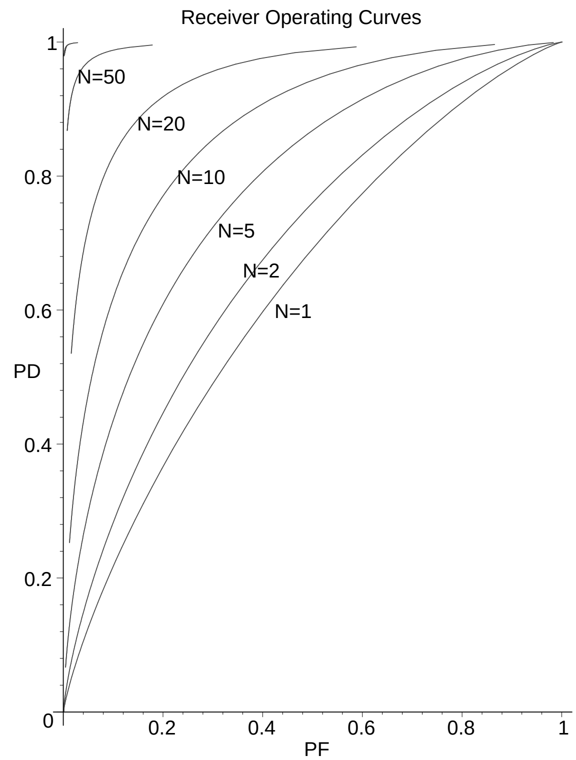


Figure 2: Analytical receiver operating curves (ROC) for the Gaussian shift in means problem. m = 0.5.

## V. Simulation Results

### Bayesian Criterion

Table 2 shows the average number of parleys as a function of the number of nodes for the case $m = 2s = 0.5$. The probability of error for each row is the worst case, usually obtained by the two bit algorithm when $SF = 1.5$. In all cases, the error probability is essentially identical for the one bit and two bit algorithms, and these numbers are very close to the optimal centralized detector error probabilities given in Table 1. Please note that the number of parleys for the original one bit algorithm is duplicated, since there is no early stopping rule for this case. For the cases of 100 and 200 sensors, the probability of error is very slightly lower than the central processor's; these results are most likely due to the relatively small number of trials.

Consider a cluster of 20 sensors. The one bit algorithm requires 7.35 iterations on average, for a total of approximately 147 bits. The extended algorithm, stopping with 12 nodes in agreement, needs 2.71 iterations for a total of 109 bits. Perhaps more importantly, the number of channel accesses is reduced from 147 to 54. This will certainly simplify the design of the MAC layer, as well as decrease the probability of intercept. It also reduces the decision time, although this may not be particularly relevant. For a fifty node network, the total number of bits is reduced from 509 to 310, and the average number of channel accesses from 509 to 155.

In all of the cases above, the increase in the probability of error caused by using the $0.6 * N$ stopping rule instead of the $0.8 * N$ rule is almost negligible. The number of parleys can be even further reduced, if one is willing to tolerate a slight increase in the probability of error; this can be accomplished by increasing $SF$ to values greater than 1.5. For comparison purposes, Table 3 shows the probability of error for a distributed detection system using the parallel architecture and a majority fusion rule. In this case, each sensor uses the same threshold and makes a binary decision, which is transmitted to the fusion center. While it is well known that using the same threshold is sub-optimal, the probability of error still approaches zero as the number of sensors approaches infinity [6, 7]. [1] Please note that for the finite number of sensors considered, the two bit parley algorithm clearly achieves better performance.

Now consider the more difficult case where $m = 2s = 0.2$,

[1] Alternatively, one could use the optimal thresholds for the parallel fusion case. However, determining them requires solving a system of coupled non-linear equations, which grows with the number of sensors. Still, the results will not be as good as the parley algorithm.

| Sensors N | Pr(e) | 1 bit | SF 1.1 | SF 1.25 | SF 1.5 |
|---|---|---|---|---|---|
| 5 | 0.2888 | 4.00 | 3.06 | 2.70 | 2.50 |
| 10 | 0.2152 | 5.61 | 4.01 | 3.59 | 3.43 |
| 20 | 0.1352 | 7.35 | 4.63 | 4.08 | 3.97 |
| 50 | 0.0404 | 10.18 | 5.22 | 4.27 | 4.16 |
| 100 | 0.0060 | 12.52 | 5.72 | 4.28 | 4.10 |
| 200 | 0.0002 | 15.07 | 6.39 | 4.42 | 4.10 |
| 5 | 0.2928 | 4.00 | 2.72 | 2.09 | 1.63 |
| 10 | 0.2176 | 5.61 | 3.63 | 2.79 | 2.22 |
| 20 | 0.1353 | 7.35 | 4.32 | 3.27 | 2.71 |
| 50 | 0.0405 | 10.18 | 5.10 | 3.65 | 3.09 |
| 100 | 0.0060 | 12.52 | 5.70 | 3.87 | 3.30 |
| 200 | 0.0002 | 15.07 | 6.39 | 4.22 | 3.56 |

Table 2: Comparison of the average number of parleys of the original algorithm [2] and the two bit parley algorithm. The top half is for the $0.8 * N$ stopping rule, while the bottom half is for the $0.6 * N$ stopping rule. $SF$ determines the quantization regions. $m = 2s = 0.5$. 50,000 trials per data point.

| Sensors N | Pr(e) m = 0.2 | Pr(e) m = 0.5 |
|---|---|---|
| 5 | 0.4255 | 0.3193 |
| 10 | 0.4026 | 0.2687 |
| 20 | 0.3670 | 0.1925 |
| 50 | 0.2914 | 0.0821 |
| 100 | 0.2151 | 0.0231 |
| 200 | 0.1314 | 0.0022 |

Table 3: Probability of error for a parallel fusion architecture using identical thresholds at all the sensors and a majority fusion rule. 50,000 trials per data point.

with the results shown in Table 4. Despite a very low signal-to-noise ratio, the use of the extended parley algorithm still leads to probabilities of error near those of the central processor. This is especially true as the number of sensors increases. Note that in these cases, there is more of a difference in error performance between the two early stopping rules for small numbers of sensors. Yet, even insisting that eighty percent of the sensors are in agreement, the number of iterations is still significantly reduced.

## Binary Symmetric Channel

In this subsection, we examine the affect of channel errors on the detection performance and the number of parleys. A binary symmetric channel model is used to potentially corrupt every transmitted bit. The crossover probability of the channel ranges from 0.0001 up to 0.05. Recently, Reardon and Kam [8] considered such communication errors for a parallel binary architecture.

Table 5 shows the results for clusters of 5, 10, and 20 sensors [2]. The original one bit algorithm is able to maintain a relatively constant probability of error, but at the cost of increasingly larger number of parleys. When the crossover

[2]The results for zero BER use a different random number seed from those in Table 2.

| Sensors N | Pr(e) | 1 bit | SF 1.1 | SF 1.25 | SF 1.5 |
|---|---|---|---|---|---|
| 5 | 0.4125 | 4.19 | 2.82 | 2.54 | 2.39 |
| 10 | 0.3763 | 5.99 | 3.86 | 3.70 | 3.61 |
| 20 | 0.3271 | 7.92 | 4.61 | 4.54 | 4.51 |
| 50 | 0.2421 | 10.81 | 5.29 | 5.23 | 5.22 |
| 100 | 0.1633 | 13.21 | 5.62 | 5.53 | 5.52 |
| 200 | 0.0790 | 15.58 | 5.76 | 5.61 | 5.59 |
| 5 | 0.4223 | 4.19 | 2.11 | 1.42 | 1.06 |
| 10 | 0.3784 | 5.99 | 2.87 | 2.04 | 1.46 |
| 20 | 0.3309 | 7.92 | 3.49 | 2.74 | 2.08 |
| 50 | 0.2426 | 10.81 | 4.18 | 3.54 | 3.06 |
| 100 | 0.1633 | 13.21 | 4.64 | 3.97 | 3.58 |
| 200 | 0.0790 | 15.58 | 5.03 | 4.19 | 3.78 |

Table 4: Comparison of the average number of parleys of the original algorithm [2] and the two bit parley algorithm. The top half is for the $0.8 * N$ stopping rule, while the bottom half is for the $0.6 * N$ stopping rule. $m = 2s = 0.2$. 50,000 trials per data point.

probability exceeds 0.01, there is a sharp increase in the average number. The two bit parley algorithm is also capable of maintaining a relatively constant error probability, while preventing the exponential growth in the number of parleys. While one could argue that more trials would be useful, especially for the lower bit error rates, the more interesting cases are the high error rate ones.

| | $N = 5$ | | $N = 10$ | | $N = 20$ | |
|---|---|---|---|---|---|---|
| BER | Pr(e) | Num. | Pr(e) | Num. | Pr(e) | Num. |
| 0 | 0.288 | 4.00 | 0.213 | 5.61 | 0.131 | 7.36 |
| 0.0001 | 0.288 | 4.03 | 0.213 | 5.67 | 0.133 | 7.52 |
| 0.001 | 0.289 | 4.26 | 0.212 | 6.19 | 0.131 | 8.49 |
| 0.01 | 0.291 | 5.74 | 0.217 | 9.62 | 0.135 | 16.07 |
| 0.05 | 0.294 | 9.24 | 0.224 | 21.03 | 0.139 | 42.17 |
| 0 | 0.288 | 2.50 | 0.216 | 3.42 | 0.132 | 3.96 |
| 0.0001 | 0.288 | 2.50 | 0.216 | 3.43 | 0.131 | 3.99 |
| 0.001 | 0.289 | 2.53 | 0.217 | 3.52 | 0.133 | 4.17 |
| 0.01 | 0.290 | 2.72 | 0.220 | 4.09 | 0.132 | 5.40 |
| 0.05 | 0.299 | 3.12 | 0.226 | 5.91 | 0.144 | 10.23 |

Table 5: Affect of channel bit error rate on the probability of error and the average number of parleys. The top half is for the one bit algorithm while the bottom half is for the two bit algorithm with $SF = 1.5$ and the $0.8 * N$ stopping rule. $m = 2s = 0.5$. 100,000 trials per data point.

## Neyman-Pearson Criterion

Since one often does not know the *a priori* probabilities of the two hypotheses and choosing useful costs can be difficult, a Neyman-Pearson criterion may be more suitable. Here we briefly show that the two bit parley algorithm still reduces the number of iterations compared to the original parley algorithm, but with essentially no loss in performance. Figure 3 shows receiver operating curves for clusters of 5 and 10 sensors with $m = 2s = 0.5$. The analytical curves are the same as in Figure 2, and they are included to allow an easy comparison.

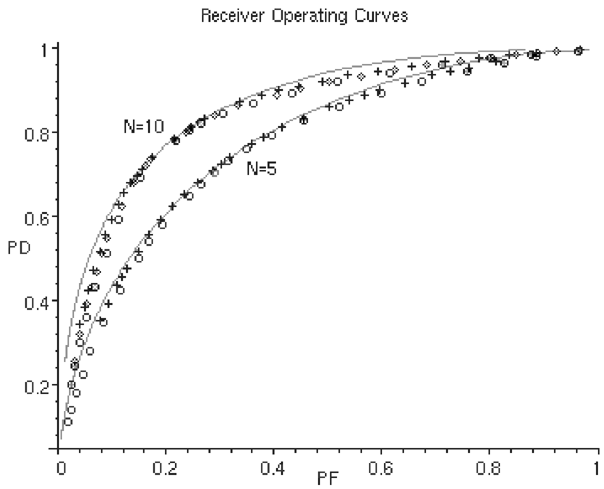


Figure 3: Comparison of the ROCs. The one bit algorithm is indicated by crosses; the two bit algorithm with the $0.8 * N$ rule is indicated by diamonds and with the $0.6 * N$ rule by circles. $SF = 1.5$. $m = 0.5$. 50,000 trials per data point.

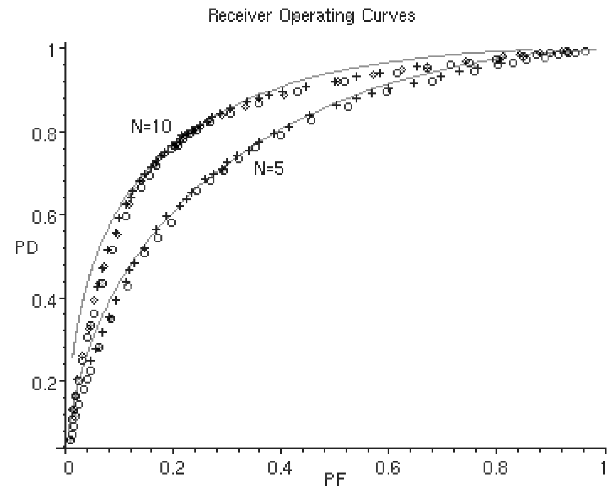


Figure 4: Comparison of the ROCs. The one bit algorithm is indicated by crosses; the two bit algorithm with the $0.8 * N$ rule is indicated by diamonds and with the $0.6 * N$ rule by circles. $SF = 1.5$. $m = 0.5$. 100,000 trials per data point. Binary Symmetric Channel with crossover probability $p = 0.001$.

First consider the 5 sensor case. The data for the original parley algorithm and the two bit parley algorithm are nearly overlapping, regardless of whether the $0.8 * N$ or the $0.6 * N$ criterion is used. For the this reason, only the latter is shown.

The results are similar for the 10 sensor case. However as seen in Figure 3, there is a bit of degradation for the $0.6 * N$ early stopping criterion. Using the $0.8 * N$ criterion, still with $SF = 1.5$, improves the two bit ROC so that it is almost the same as the one bit curve. Yet, both of them show a little degradation compared to the central processor as the threshold, $\lambda$, moves away from a value of one. For the two bit algorithm, the choice of the two quantization thresholds, $\lambda_{i,m,L}$ and $\lambda_{i,m,H}$ based on the scale factor, is part of the cause of this phenomenon. Even with this slightly suboptimal processing, the overall results are quite good. Figure 4 shows the receiver operating curves when the data transmitted among the sensors uses a binary symmetric channel model with a crossover probability of $p = 0.001$. Note that the performance is effectively unchanged from that of an ideal channel.

Figure 5 shows the average number of parleys required as a function of the central threshold, $\lambda$, for communication among the sensors over an ideal channel. As one can see, the maximum number of parleys occurs when the threshold is equal to one. This is a suitable choice when the *a priori* probabilities are nearly equal and the costs are similar, *i.e.* the Bayesian case. Please note that in all cases, the two bit algorithm leads to substantial reductions. Figure 6 shows the average number of parleys for communication over the binary symmetric channel with $p = 0.001$. There is a modest increase in the average number of parleys for the one bit algorithm, while the increase is almost negligible for the two bit algorithm. These results are also consistent with those shown in Table 5.

## VI. Conclusions and Future Study

An extension to Swaszek's and Willet's algorithm can substantially reduce the average number of parleys, while maintaining the probability of error almost equal to that of a central processor. The proposed algorithm uses soft information, and it is reasonably robust to communication channel errors. Presently, work is continuing with improving the Neyman-Pearson results for values of $\lambda$ further from one. Less heuristic early stopping rules are being developed and evaluated. Sequential extensions are also being studied.

## References

[1] G. J. Pottie, "Wireless sensor networks," *IEEE Information Theory Workshop*, pp. 139-140, June. 1998.

[2] P F. Swaszek and P. Willett, "Parley as an approach to distributed detection," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 31. no. 1, pp. 447-457, Jan. 1995.

[3] P. K. Varshney, *Distributed Detection and Data Fusion*, Springer, New York, 1997.

[4] M. Longo, T. D. Lookabaugh, and R. M. Gray, "Quantization for decentralized hypothesis testing under communication constraints," *IEEE Trans. on Info. Theory*, vol. 36, pp. 241-255, Mar. 1990.

[5] H. L. Van Trees, *Detection, Estimation, and Modulation Theory, vol. 1*, John Wiley & Sons, New York, 1968.

[6] J. N. Tsitsiklis, "Decentralized detection by a large number of sensors," *Mathematics of Control, Signals, and Systems*, vol. 1, pp. 167-182, 1988.

[7] Q. Zhang, P. K. Varshney, and R. D. Wesel, "Optimal distributed binary hypothesis testing with independent identical sensors," *Proc. Conf. on Information Sciences and Systems (CISS)*, Princeton, NJ, pp. WA7b-1 to WA7b-6, Mar. 2000.

[8] B. Reardon and M. Kam, "Parallel binary decision fusion under communication errors," *Proc. CISS*, Princeton, NJ, pp. WA7b-7 to WA7b-13, Mar. 2000.

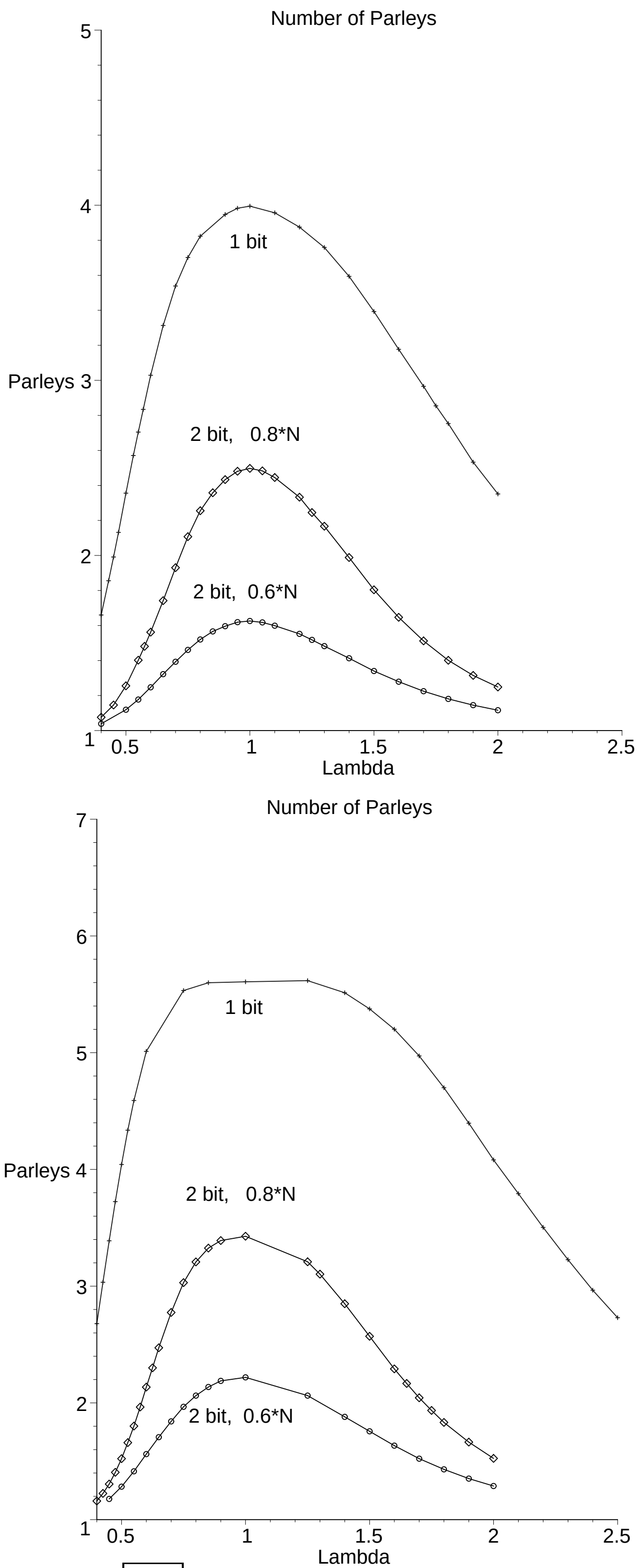


Figure 5: $\boxed{\frac{(a)}{(b)}}$ Average number of parleys required for consensus vs. $\lambda$. These values of $\lambda$ correspond to those used to obtain the ROCs. (a) $N = 5$. (b) $N = 10$. Ideal channel.

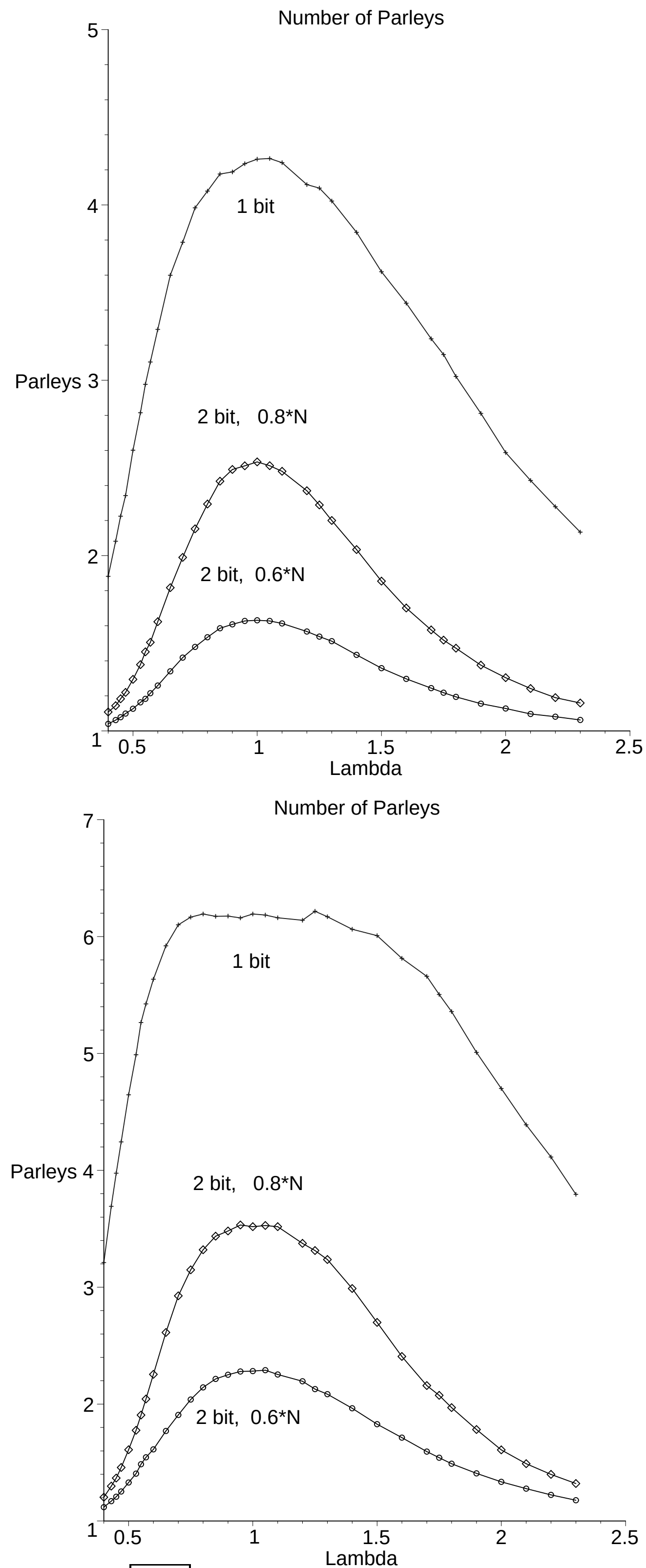


Figure 6: $\boxed{\frac{(a)}{(b)}}$ Average number of parleys required for consensus vs. $\lambda$. These values of $\lambda$ correspond to those used to obtain the ROCs. (a) $N = 5$. (b) $N = 10$. BSC with $p = 0.001$.