CHANGE DETECTION AND ESTIMATION IN LARGE SCALE SENSOR NETWORKS: LINEAR COMPLEXITY ALGORITHMS

Ting He, Shai Ben-David, and Lang Tong*
School of Electrical and Computer Engineering
Cornell University
Ithaca, NY 14853, USA
Email:{th255@,shai@ece.,ltong@ece.}cornell.edu

***Abstract*— We propose algorithms for nonparametric sample-based spacial change detection and estimation in large scale sensor networks. We collect random samples containing the location of sensors and their local decisions, and assume that the local decisions can be "stimulated" or "normal", reflecting the local strength of some stimulating agent. Then change in the location of the agent manifests itself by a change in the distribution of "stimulated" sensors. In this paper, we are aiming at developing a test that, given two collections of samples, can decide whether the distribution generating the samples has changed or not, and give an estimated changed area if a change is indeed detected. The focus of this paper is to reduce the complexity of the detection and estimation algorithm. We propose two fast algorithms with almost linear complexity and analyze their completeness, flexibility and robustness.**

***Keywords:* Non-parametric statistical methods, Change detection, Spacial change, Sensor Networks.**

I. INTRODUCTION

Given a large scale sensor network, where sensors are distributed to detect the local presence of some agents, we are interested in deciding whether the location of agents has changed by looking at random samples drawn from sensors. We assume that the agents have local property, i.e. their most intensive areas are clustered in space, forming some "heated areas" in the sensor field. One example is the presence of targets. Another example is sources stimulating "excited state" of sensors, with this stimulating effect decaying as distance grows. A sensor reports "RED" if it detects the agent or "GREEN" otherwise. Using the architecture of SENMA [6], and either letting sensors transmitting their location information together with decisions or polling sensors at specific locations, we can collect a set of samples containing local decisions and the locations where these decisions are made.

The problem of change detection and estimation in SENMA, illustrated in Figure 1, is to decide, from two consecutive records, whether there is a change of the underlying probability distribution (detection) and the distribution of the change (estimation) if changes occur. In practice, random access is used at the mobile access point when collecting packets from the sensor field(See Fig. 1), and sensors can make mistakes in their decisions, i.e. reporting "RED" with no agent present(*false alarm*) or "GREEN" when there is an agent present(*missing detection*). Hence neither is there guarantee that a measurement at a particular sensor in one data collection will appear in the next, nor can we say a certain measurement is true or not. The remedy is to take a large number of samples to compensate for the ill effect of different locations and false reports. However, two factors must be considered for change detection and estimation in a large scale sensor networks: the time required or the number of packets required for data collection and the complexity of the detection and estimation algorithm. In practice, minimizing the number of samples required for detection reduces the collection time of a mobile AP and, more importantly in terms of energy consumption, the number of transmissions from sensors. In a previous paper [1], we have provided a mathematical characterization of the required sample size for which, with high probability, the distance of any two probability distributions can be estimated accurately from empirical distance between samples generated by these distributions. The basic tool used here is the Vapnik-Chervonenkis Theory [2]. In this paper, we are mainly concerned with the complexity of the detection and estimation algorithm. We present variations of an algorithm in [1] and analyze their complexity, flexibility and levels of confidence.

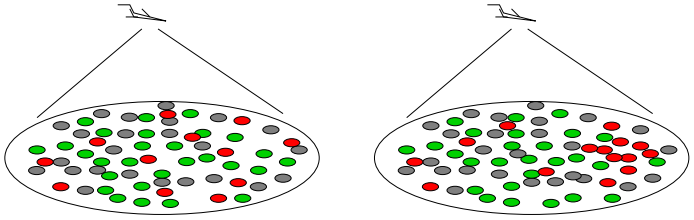

Fig. 1. Sensor Networks with Mobile Access. RED: sensors with detection. GREEN: sensors with no detection. Gray: Data not collected.

For many applications, there is no prior knowledge about the distribution of the sensor states. The change detection and estimation is therefore non-parametric in which we make no assumption about the specific form of the probability distribution of the binary random field. See the introduction of

This work is supported by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

[1] for a brief review of some previous work in non-parametric change detection problem.

II. Algorithms

In this section, we take one of the three algorithms in [1] and derive its two variations with marginal performance but greatly-reduced complexity. The algorithm we are based on is Algorithm 3: *Search in Axis-aligned Rectangles*. Using the same boundary search idea in Algorithm 3, we reduce its complexity to be almost linear by projecting samples onto one-dimension, and refining the results by multiple projections at a cost of only a constant factor increase in running time. As algorithms in [1], both of the algorithms are based on the fact that empirical probability distributions of i.i.d. samples uniformly converge to their actual distributions, no matter what kind of distributions they are [2]. Specifically, we rely on results in [1] to provide a guaranteed level of accuracy.

Practical implementations require that the distance between two empirical distributions be computed in a timely manner. Given limited computation resource, it is of interest to look at the largest sample size we can handle with our detection/estimation algorithms. We therefore need to reduce the complexity of our algorithms in terms of sample size to enable efficient processing of large samples. The key to complexity reduction is the projection of samples onto one-dimension. After the projection, the search is reduced to search on a line, and by reusing previous computation the search is done in a linear number of steps.

We consider the following detector:

Given two collection of samples S_1 and S_2, drawn i.i.d from probability distributions P_1 and P_2 respectively, and threshold $\epsilon \in (0,1)$, for simple binary hypothesis $\mathcal{H}_0 : P_1 = P_2$ vs. $\mathcal{H}_1 : P_1 \neq P_2$, the detector is defined as

$$\begin{aligned} \delta(S_1, S_2; \epsilon) &= 1 \text{ if } \textit{diffmax} > \epsilon \\ &= 0 \text{ otherwise} \end{aligned}$$

where *diffmax* is the maximum difference in the empirical probabilities computed by some algorithm.

Furthermore, if $\delta(S_1, S_2; \epsilon) = 1$, then the estimated changed area is the area where *diffmax* is achieved.

Now the problem is reduced to finding *diffmax*. Previous algorithms(see [1]) include Algorithm 1: *Exhaustive Search in Planar Disks*, Algorithm 2: *Search in Sample-centered Disks* and Algorithm 3: *Search in Axis-aligned Rectangles*, with complexities $O(M^4)$, $O(M^2 \log M)$ and $O(M^3)$, where $M = |S_1 \cup S_2|$. Based on the same boundary search idea of Algorithm 3, we have the following faster algorithms.

Algorithm 3.1*: Search in Axis-aligned Stripes*

This is a simplified version of Algorithm 3. The basic idea is to project samples onto x and y coordinates, and perform change detection/estimation on each coordinate. The search is based on the same boundary search idea of Algorithm 3, but by projection the complexity of the search is reduced to be linear.

Let $\mathcal{A}$ be the collection of vertical stripes, i.e., axis aligned rectangles with height equal to the field height. Similarly, let $\mathcal{B}$ be the collection of horizonal stripes. Given a collection of samples $S = S_1 \cup S_2$, it suffices to consider finite subsets $\mathcal{H}_3^x(S) \subset \mathcal{A}$ and $\mathcal{H}_3^y(S) \subset \mathcal{B}$ defined by

$$\begin{aligned} \mathcal{H}_3^x(S) &\triangleq \{V(x_i, x_j) : s_i = (x_i, y_i), \\ &\quad s_j = (x_j, y_j) \in S\} \quad (1) \\ \mathcal{H}_3^y(S) &\triangleq \{H(y_k, y_l) : s_k = (x_k, y_k), \\ &\quad s_l = (x_l, y_l) \in S\} \quad (2) \end{aligned}$$

where $V(x_i, x_j)$ is the vertical stripe with left and right boundary x_i and x_j, and $H(y_k, y_l)$ is the horizonal stripe with lower and upper boundary y_k and y_l. See Figure 2.

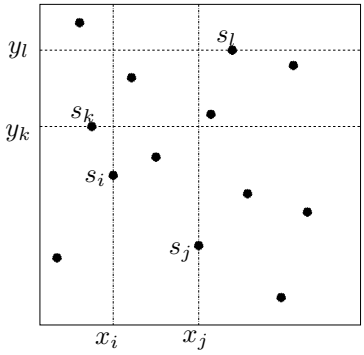

Fig. 2. Members of $\mathcal{H}_3^x$ and $\mathcal{H}_3^y$

Recall the definition of completeness in [1], we have the following claim.

Claim 0.1: Let $\mathcal{A}, \mathcal{B}$ be as defined above. Given S_1, S_2, the finite subset $\mathcal{H}_3^x(S_1 \cup S_2) \cup \mathcal{H}_3^y(S_1 \cup S_2)$ defined in 1 and 2 is complete for $\mathcal{A} \cup \mathcal{B}$ w.r.t. $S_1 \cup S_2$.

Given $S = S_1 \cup S_2$, Algorithm 3.1 performs the following exhaustive search in $\mathcal{H}_3^x(S) \cup \mathcal{H}_3^y(S)$:

$$\max_{A \in \mathcal{H}_3^x \cup \mathcal{H}_3^y} \left| \frac{|S_1 \cap A|}{|S_1|} - \frac{|S_2 \cap A|}{|S_2|} \right|.$$

where S_1, S_2 are sets of the previous and current red samples.

The algorithm includes (i) projecting the red samples onto x and y coordinates; (ii) sorting the projected samples into increasing order; (iii) in x coordinate, scanning from left to right, at the ith sample with x coordinate x_i, computing $F_x(i)$,

the difference in the empirical probability of stripe $V(0, x_i)$ between S_1 and S_2, which is given by

$$F_x(i) = F_x(i-1) + \frac{1}{|S_1|} \text{ if } s_i \in S_1 \quad (3)$$

$$= F_x(i-1) - \frac{1}{|S_2|} \text{ if } s_i \in S_2 \quad (4)$$

computing $F_y(\cdot)$ in y coordinate similarly; (iv) finding the peak and bottom point of $F_x(\cdot)$ achieved at m_1 and m_2, and the peak and bottom of $F_y(\cdot)$ at n_1 and n_2. Then the maximum difference *diffmax* $\stackrel{\Delta}{=} \max_{A \in \mathcal{H}_3^x \cup \mathcal{H}_3^y} \left| \frac{|S_1 \cap A|}{|S_1|} - \frac{|S_2 \cap A|}{|S_2|} \right|$ is given by

$$\max(F_x(m_1) - F_x(m_2), F_y(n_1) - F_y(n_2))$$

and the axis aligned stripe bounded by $x = x_{m_1}$ and $x = x_{m_2}$ if $F_x(m_1) - F_x(m_2) > F_y(n_1) - F_y(n_2)$, or $y = y_{n_1}$ and $y = y_{n_2}$ if $F_x(m_1) - F_x(m_2) \leq F_y(n_1) - F_y(n_2)$, gives an estimation of the changed area.

The complexity of Algorithm 3.1 for sample size $M = |S_1 \cup S_2|$ is $O(M \log M)$[1]. Thus by projection we reduce the complexity to make Algorithm 3.1 an almost-linear algorithm. Like Algorithm 3, Algorithm 3.1 is also not amendable to relativized discrepancy defined in [1].

Algorithm 3.2: Search in Random Stripes

This is a variation of Algorithm 3.1. It is based on the same projection and boundary search idea as in Algorithm 3.1. The difference is when performing the projection, we project samples onto a random direction instead of the fixed direction of x or y axis. The rest of the algorithm is the same as Algorithm 3.1.

Algorithm 3.2 has the same complexity and flexibility as Algorithm 3.1. It is also complete for the class of stripes along the particular direction as chosen by Algorithm 3.2 in the sense that it exhausts all such stripes with at least one sample point on each boundary. Its advantage is that it is robust with respect to changing patterns. This is in the sense that Algorithm 3.2 will perform equally well under all kinds of changing patterns(the directions along which changes occur) while Algorithm 3.1 can be affected significantly by the particular changing pattern that is taken. For example, Algorithm 3.1 is vulnerable to the pattern where changes always occur along a tilted line of 45° or 135°, because in that case the increasing and decreasing parts of the change will largely get cancelled when projected onto axes. We can also project onto multiple random directions to increase the accuracy of the algorithm at the cost of a constant factor increase in the complexity. For example, in the simulation we use two random directions for fair comparison with Algorithm 3.1 since the latter use two directions, i.e. x and y axes.

[1]The complexity $O(M \log M)$ is determined by the critical step of sorting. Strictly speaking, this is more than linear complexity. However, for sample size of practical interest, the factor $\log M$ can be incorporated into the constant factor and the algorithm scales like a linear algorithm.

III. Simulation

A. Simulation Setup

In the simulation, we assume that the sensor field is a 50×50 m^2 square, and sensors are randomly distributed in the field. Assume the distribution of operating sensors does not change. Then it suffices to consider only "RED" samples. We adopt a simple sampling strategy that is continuing sampling until enough "RED" samples are collected. The change simulated here is a change in the location of the most intensive area of the stimulating agent. For the stimulating agent, we adopt a simplified model that says a sensor reports "RED" with probability p if it falls inside distance r from the center of the agent and q if it falls outside this distance, where $p > q$. We let $p < 1, q > 0$ to model the randomness in the decision of each sensor.

Given the highest false alarm allowed (*size* of the detector), we first perform Monte Carlo runs to decide the detection threshold, and then do detection and estimation using this threshold. Hence the nonparametric property of the simulation is preserved. Difference between algorithms becomes in which class(e.g. disks, stripes, rectangles) we are trying to fit the changed area and how to find the best match in the class.

In our simulation, we let $p = 0.8$, $q = 0.2$, $r = \frac{50}{6}$ m. The size of the detector is $\alpha = 0.05$.

B. Detection Threshold and Missing Detection Probability

In the simulation, we first fix a range of sample size we are interested in, and then do Monte Carlo runs at each sample size with fixed agent location (no change) to find out the distribution of *diffmax*(the maximum change in empirical probability found by our algorithms) under $\mathcal{H}_0$. The detection threshold for that sample size is determined as the smallest value such that the frequency that *diffmax* exceeds this value is below the detector's size α.

For evaluation purpose, we compare the detection thresholds and missing detection probabilities for Algorithm 3.1 and Algorithm 3.2. Specifically, we plot the detection threshold obtained by simulation as described above as a function of the red sample size for red sample size between 594 and 10^4, and compare it to the theoretical threshold computed from the upper bound in [1]. In computing the theoretical threshold, we find the smallest ϵ such that $8(2n)^2 e^{-n\epsilon^2/32} \leq \alpha$ and choose ϵ as the theoretical threshold. See Fig. 3.

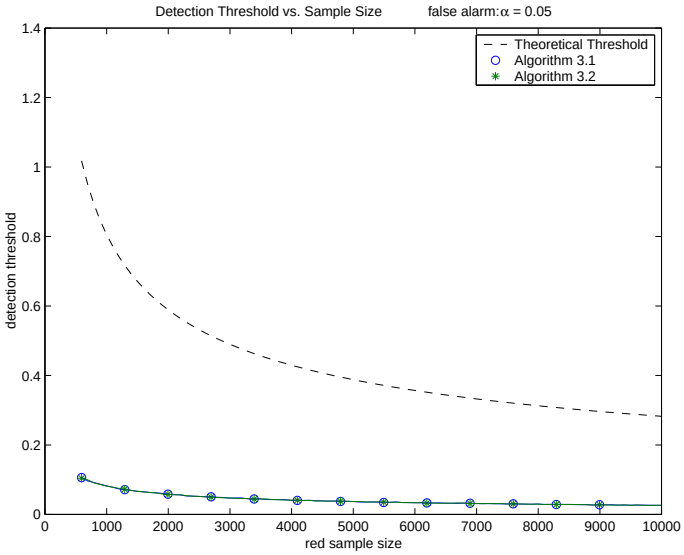

Fig. 3. Detection threshold as a function of the red sample size

As expected, both simulated and theoretical threshold is a decreasing function of the sample size, which reflects a trade-off between sampling time, energy consumed and data processing expense and detection precision. Also as expected, the theoretical threshold is a loose upper bound of the simulated threshold. This is because of the nonparametric nature of the theoretical upper bound. This bound is claimed to hold under arbitrary distributions by the Vapnik-Chervonenkis Theory. Therefore for a given distribution, this bound is loose.

Then using the threshold obtained from simulation, for randomly changing locations, we see from Fig. 4 how the missing detection probability of these algorithms decreases as sample size grows.

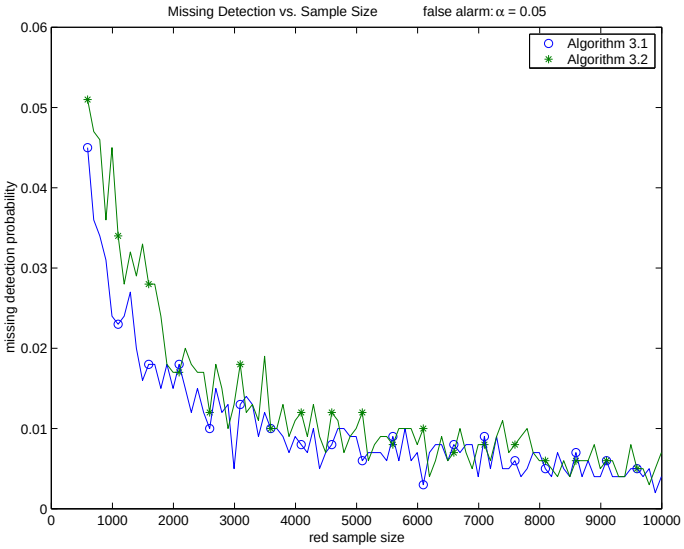

Fig. 4. Missing detection probability as a function of the red sample size

It is worth noting that here Algorithm 3.1 performs slightly better because of the particular set up we are using in the simulation. For randomly changing locations, Algorithm 3.1 ensures good utilization of samples by searching along orthogonal directions, while Algorithm 3.2 performs equally well under all changing directions by searching along random directions.

REFERENCES

[1] S. Ben-David, T. He, and L. Tong, "Non-Parametric Approach to Change Detection and Estimation in Large Scale Sensor Networks," in Proc. 2004 CISS, (Princeton, NJ), Mar. 2004.

[2] V.N. Vapnik and A. Ya. Chervonenkis "On the uniform convergence of relative frequency of events to their probabilities" in Theory of Probability and its Applications, Vol. 16, pp 264-280, 1971.

[3] M. Talagrand, "Majorazing maesures: the generic chaining", in Annals of Probability, vol. 24, pp. 1049-1103, 1996.

[4] T. Batu, L. Fortnow, R. Rubinfeld, W.D. Smith, and P. White "Testing that distributions are close", in IEEE Symposium on foundations of Computer Science, pp. 259-269, 2000.

[5] M. Anthony and J. Shawe-Taylor, "A result of Vapnik with applications", in Discrete and Applied Mathematics, vol. 47(2), pp. 207-217, 1993.

[6] L. Tong, Q. Zhao, and S. Adireddy, "Sensor Networks with Mobile Agents," in Proc. IEEE 2003 MILCOM, (Boston, MA), Oct. 2003.

[7] B. Brodsky and B. Darkovsky, *Non-Parametric Methods in Change-Point Problems*, Kluwer Academic, The Netherlands, 1993.

[8] J. Shao, *Mathematical Statistics*, Springer, 1999.

[9] C. Budianu, S. Ben-David and L. Tong, "Estimation of the Number of Operating Sensors in Large-Scale Sensor Networks", submitted to IEEE Transactions on Signal Processing, 2004.