Zero error Function computation in Sensor Networks

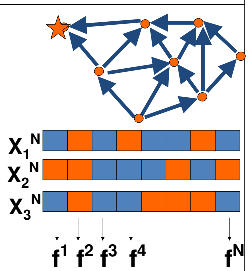
Hemant Kowshik and P. R. Kumar
CSL and Department of ECE, University of Illinois

Introduction

- In wireless sensor network applications, nodes are not typically interested in raw data
- Instead, nodes are interested in a relevant *function* of the measurements
- Examples
  - MAXIMUM in a fire-alarm system
  - MINIMUM
  - MEAN for weather monitoring
  - MEDIAN, MODE, Boolean functions

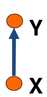
Problem formulation

- Nodes are connected in a given directed graph topology
- Suppose a single collector node needs to compute the function block with **zero error**
- Nodes accumulate blocks of measurements and can achieve savings by **block computation**



- **Objective:** To find optimal communication and in-network computation strategies

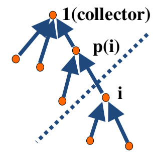
The Case of Two nodes



- Two nodes with measurements X,Y
- X communicates to Y only
- Y wants to compute f(X,Y) with zero-error
- Minimum worst-case/average case number of bits?

Worst-case	Average-case
Node X *separates* x_1 and x_2 if $\exists y$ such that $f(x_1, y) \neq f(x_2, y)$	Node X *separates* x_1 and x_2 if $\exists y$ s.t. $f(x_1, y) \neq f(x_2, y)$ and $p(x_1, y)p(x_2, y) > 0$
Separation rule creates equivalence classes	Separation rule creates equivalence classes if p(x,y)>0
Min. number of bits = $\log_2$(no. of equiv. classes)	Huffman code is optimal

Functions on trees: Worst-case



- Directed tree (V,E) with root node 1
- Each edge is a cut-edge
- Codebook on i→p(i) is

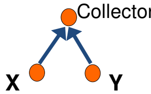
$$\{g : X_{V\setminus T(i)} \to D \mid \text{there exists } x^*_{T(i)} \text{ s.t. } f(x^*_{T(i)}, x_{V\setminus T(i)}) = g(x_{V\setminus T(i)})\ \forall x_{V\setminus T(i)}\}$$

- **Decoder:** Assign nominal values to variables in $X_{T(i)}$ based on codewords received on incoming edges
- **Encoder:** Substitute the nominal values of variables in $X_{T(i)}$ to obtain a function
- **Theorem:** This coding scheme achieves zero-error function computation at root, and is optimal for every edge

Functions on trees: Average-case

- If $p(x_1, x_2, \ldots, x_n)>0$ for all $(x_1, x_2, \ldots, x_n)$, then can find optimal codes on each edge as before
- However if there are zeros in the probability matrix, we can tradeoff between links
- Example: Distributed compression will result in error

X \ Y	y_1	y_2
x_1	a	--
x_2	--	b

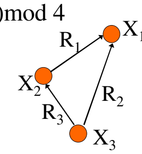


(0,1) and (1,0) are feasible

- **Theorem:** If $p(x_1, x_2, \ldots, x_n)>0$ for all $(x_1, x_2, \ldots, x_n)$, then the optimal code on each edge is the Huffman code for the two node problem corresponding to that edge

Functions on DAGs: Issues

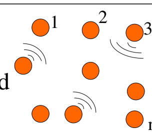
- Rate = Number of bits per computation
- No unique rate point that is jointly optimal for all edges
  $X_1, X_2, X_3 \in \{0,1,2,3\}$; $f = (X_1+X_2+X_3) \bmod 4$
- (R_1,R_2,R_3) = (2,0,2), (2,2,0) and (2,1,1)
- are all Pareto optimal



- Rate points which satisfy cut capacity constraints give outer bound to rate region.
- **Theorem:** Rates achieved by only activating some tree of edges in the DAG are extreme points of the rate region
- Are these the only extreme points?

For the parity, MAX/MIN functions, cut set bound is tight

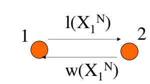
Collocated Networks



- Broadcast scenario: every node hears transmissions from every other node
- In earlier work, Giridhar-Kumar showed that the complexity of computing *type-threshold* functions is *O(log n)*
- Suppose X_i ~ Bern(p), i.i.d measurements
- **Theorem:** For type-threshold functions, the average case complexity is O(1)
- Example: Consider the function AND$(X_1, X_2, \ldots X_n)$
  - Expected no. of bits = H(p) + pH(p) + … $p^{(n-1)}$H(p) $\leq$ H(p)/1-p

Interactive computation

- Nodes 1 and 2, both want to compute AND(X_1,X_2)
- Trivial protocol has complexity 2 bits
- Alternate protocol:
  - Suppose node 1 releases prefix-free codeword $l(X_1^N)$ = Nlog$_2$3- $w(X_1^N)$
    This satisfies Kraft inequality
  - Node 2 replies only for the instances where $X_1 = 1$
  - Worst-case number of bits = $\log_2 3$!



- Lower bound can be shown by observing that any protocol must have at least three rectangular partitions

X_1 \ X_2	0	1
0	0	0
1	0	1

Boolean symmetric functions

- Extend the above approach to larger class of functions
- For a Boolean symmetric function of $X_1, X_2, \ldots X_n$, it is enough to know number of 1s
- Threshold type functions: Are there <k ones
- Exact complexity is $\log_2 \binom{n+1}{k}$ bits
- Delta functions: Are there exactly k ones
- Exact complexity is $\log_2 \left[\binom{n+1}{k} + \binom{n}{k+1}\right]$ bits
- Lower bound is shown by fooling set lower bounds from communication complexity theory