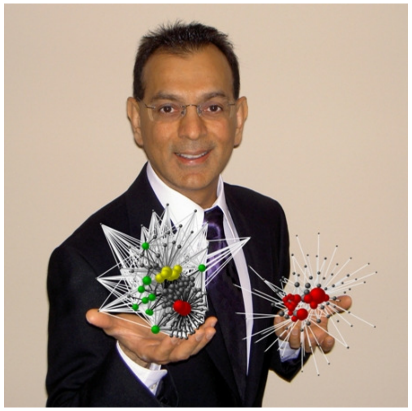
A Coding Theoretic Framework for Query Learning

Clayton Scott

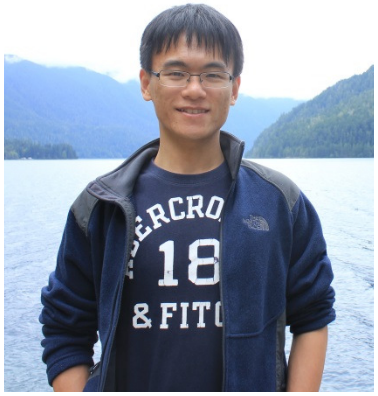
Electrical Engineering and Computer Science
University of Michigan

Collaborators

Gowtham Bellala



Suresh Bhavnani



Max Yi Ren

Panos Papalambros

Toxic Chemical Emergency

Hundreds of toxic chemical incidents per year (Kleindorfer et al., 2003)

Chemical Identification

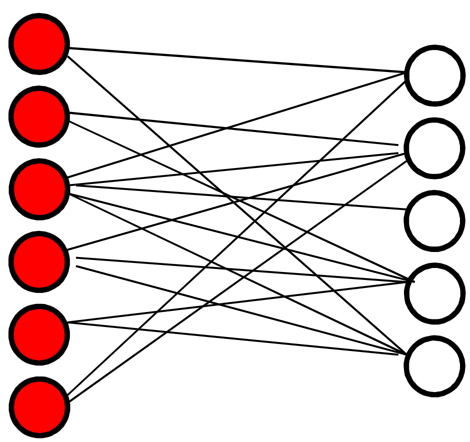
Needed to treat victims, decontaminate site, issue neighborhood warnings, etc.

Decision Support for Chemical Identification

WISER Database

<u>W</u>ireless <u>I</u>nformation <u>S</u>ystem for <u>E</u>mergency <u>R</u>esponders

- Maintained by NLM, panel of chemists/toxicologists
- Lists which symptoms are caused by which chemicals
- Represented as bipartite network, or binary table

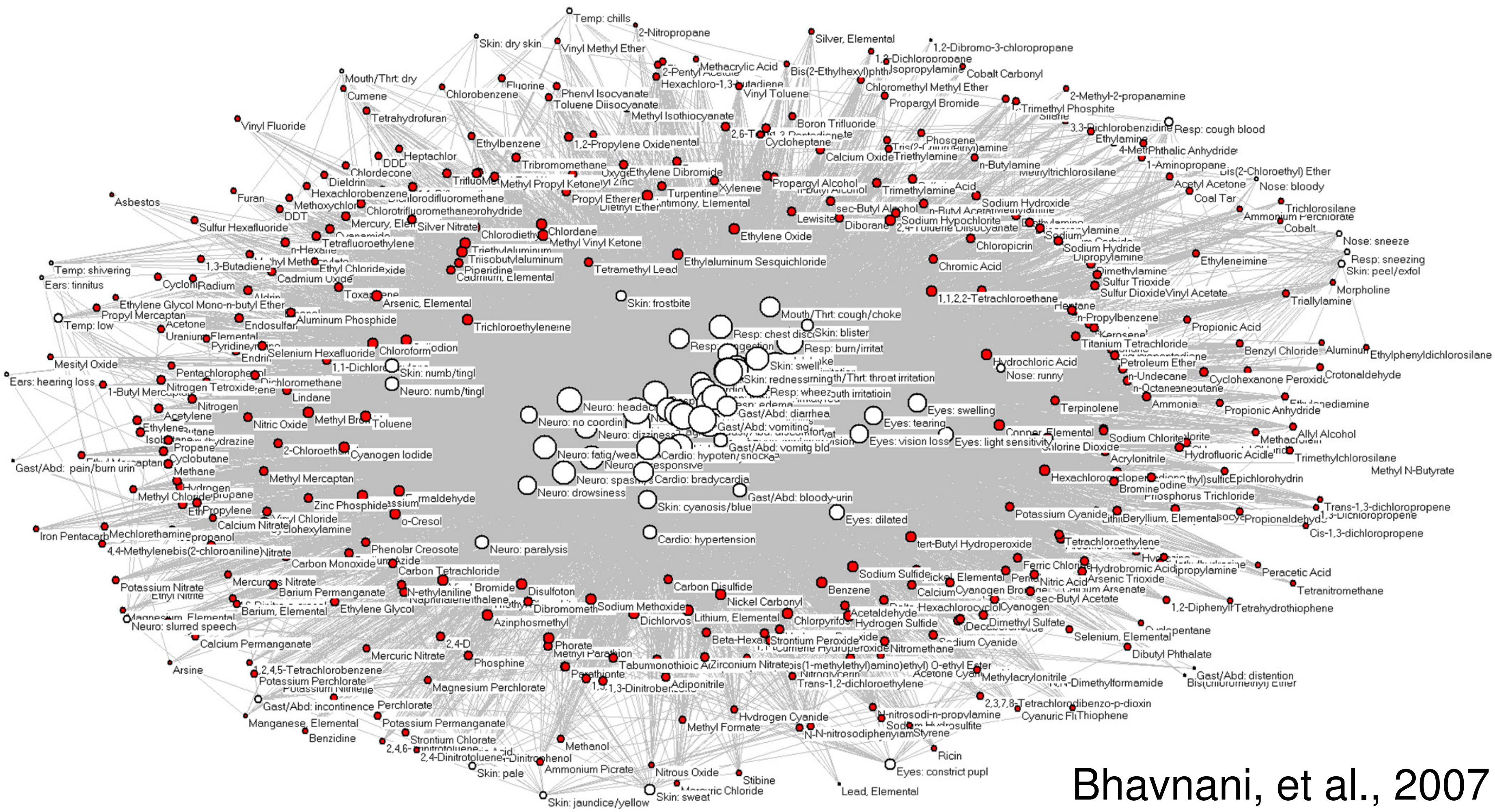


chemicals symptoms

$$\begin{array}{cc} & \text{symptoms} \\ \text{chemicals} & \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Network Layout of WISER Database

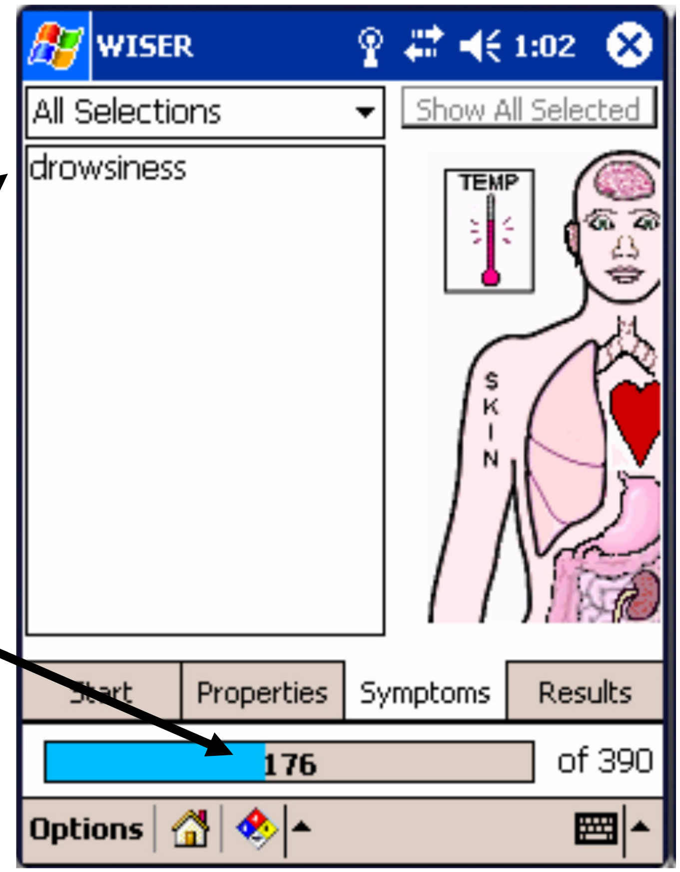
- ~ 300 chemicals, 80 symptoms
- Edge density ~ 0.4, symptoms tend to be **nonspecific**



Bhavnani, et al., 2007

WISER

Wireless Information System for Emergency Responders



User selects a symptom

Non-matching chemicals eliminated

Symptom nonspecificity
Unguided searching

→ Too many symptoms needed

Query Learning

Objects: $\Theta = \{\theta_1, \ldots, \theta_M\}$

Queries: $Q = \{q_1, \ldots, q_N\}$

	q_1	q_2	q_3	q_4	q_5	
θ_1	1	0	0	0	1	π_1
θ_2	0	1	0	1	0	π_2
θ_3	1	1	1	1	1	π_3
θ_4	0	1	0	1	1	π_4
θ_5	0	0	0	1	1	π_5
θ_6	1	1	0	0	0	π_6

$$\sum_{i=1}^{M} \pi_i = 1$$

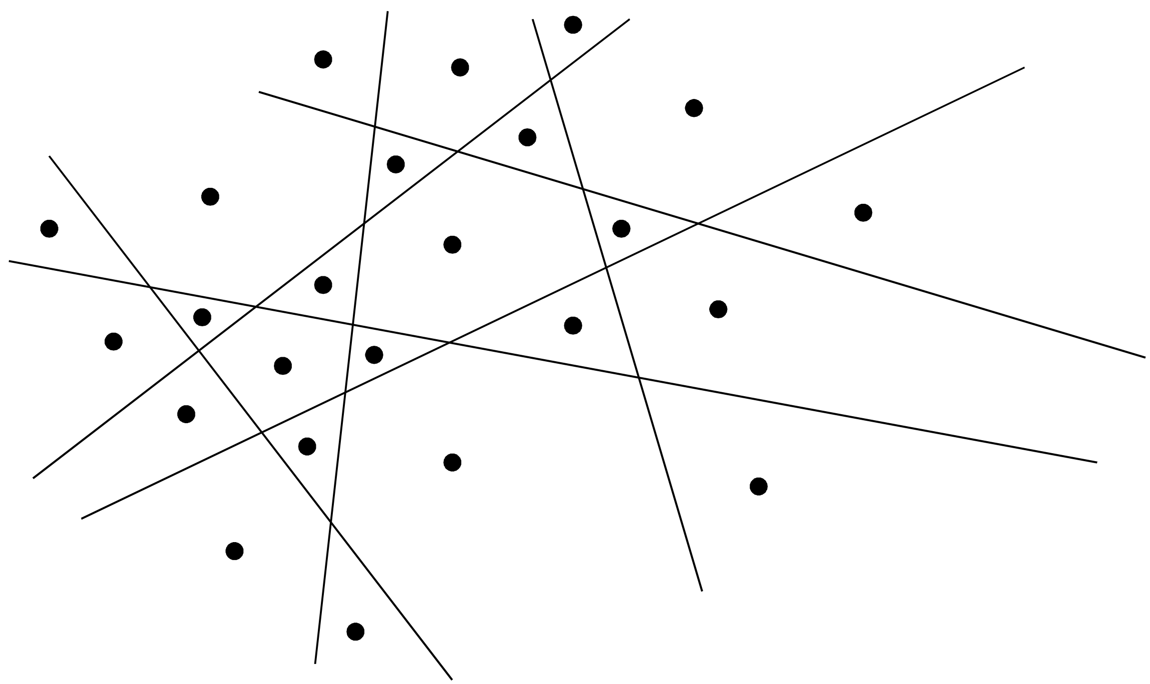
Problem statement

- object selected at random
- determine object with as few queries as possible

Other Applications of Query Learning

<u>Objects</u>
- chemicals
- network failures
- faults
- classifiers
- …

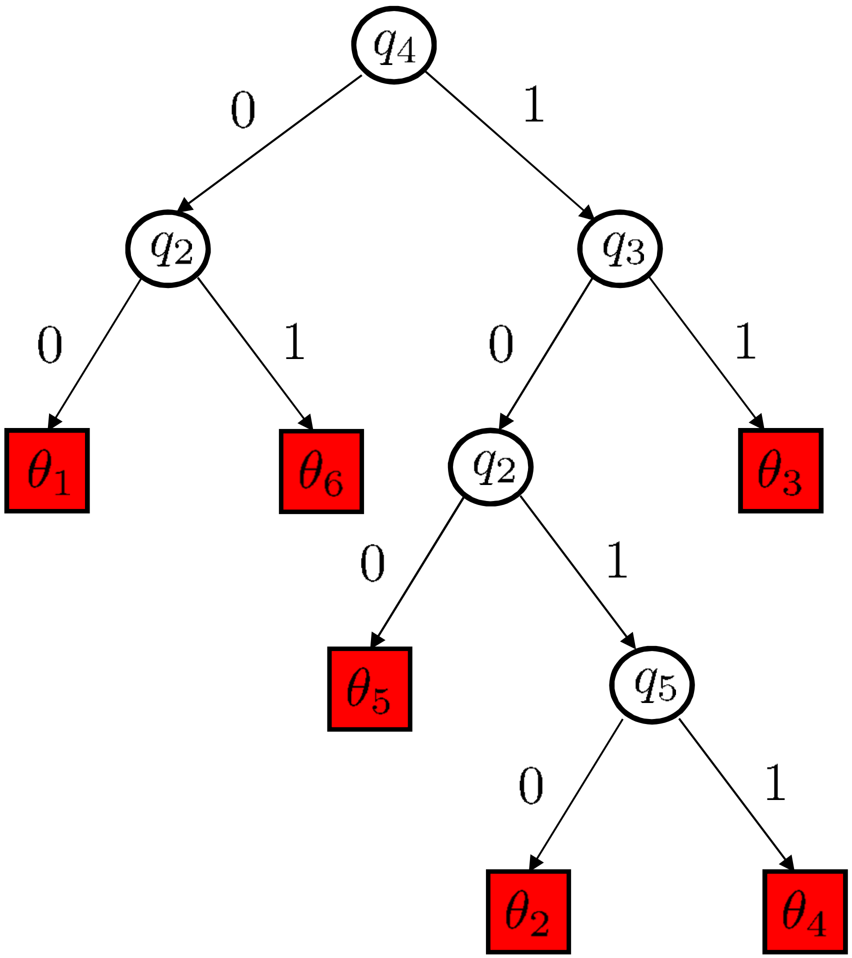
<u>Queries</u>
- symptoms
- network measurements
- alarms
- labels at specific points (active learning)
- …



Outline

- Connecting query learning to source coding
- Generalizations
  - ❑ Exponentially weighted costs
  - ❑ Group identification
- Application to
  - ❑ Query noise
  - ❑ Preference elicitation
- Not in this talk
  - ❑ Multiple objects present
  - ❑ Likelihoods, Bayesian networks
  - ❑ Network fault detection
  - ❑ Human factors, usability, etc.

Decision Trees



	q_1	q_2	q_3	q_4	q_5	
θ_1	1	0	0	0	1	π_1
θ_2	0	1	0	1	0	π_2
θ_3	1	1	1	1	1	π_3
θ_4	0	1	0	1	1	π_4
θ_5	0	0	0	1	1	π_5
θ_6	1	1	0	0	0	π_6

<u>Generalized binary search:</u>

- Greedy, top-down algorithm
- Select query that balances remaining objects

$$E[\#\text{ of queries}] = \pi_1 \cdot 2 + \pi_2 \cdot 4 + \pi_3 \cdot 2 + \pi_4 \cdot 4 + \pi_5 \cdot 3 + \pi_6 \cdot 2$$

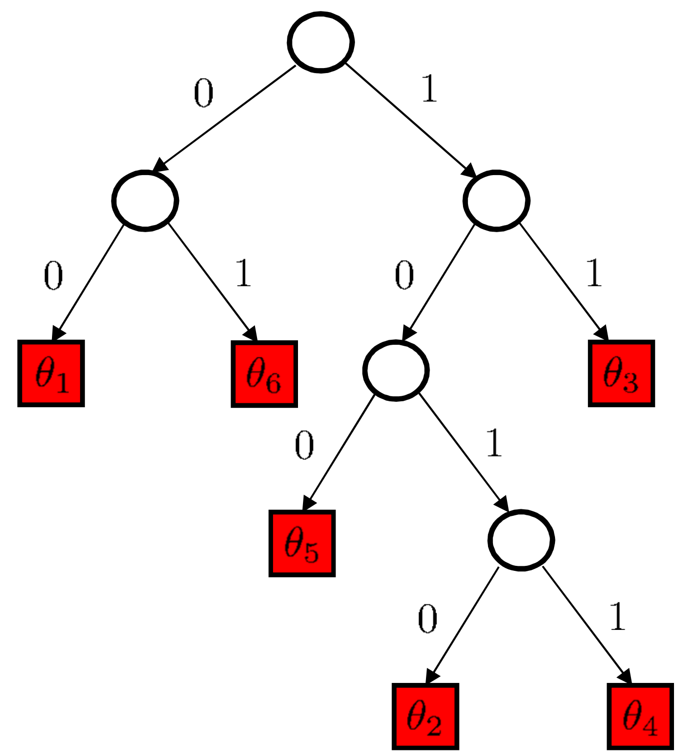
Source Coding

Fixed alphabet: $\{\theta_1,\ \theta_2,\ \theta_3,\ \theta_4,\ \theta_5,\ \theta_6\}$
Prior probabilities: $\pi_1, \pi_2, \pi_3, \pi_4, \pi_5, \pi_6$
Goal: efficient binary encoding

$$\theta_3\theta_2\theta_5\theta_3\theta_6\theta_2\theta_1\theta_3\theta_1\theta_3\theta_3\ldots \longrightarrow \boxed{\text{encoder}} \longrightarrow \underbrace{11}\underbrace{1010}\underbrace{100}\underbrace{11}\underbrace{01}\underbrace{1010}\underbrace{00}0\ldots$$

Instant decoding $\Longrightarrow$ **prefix** code

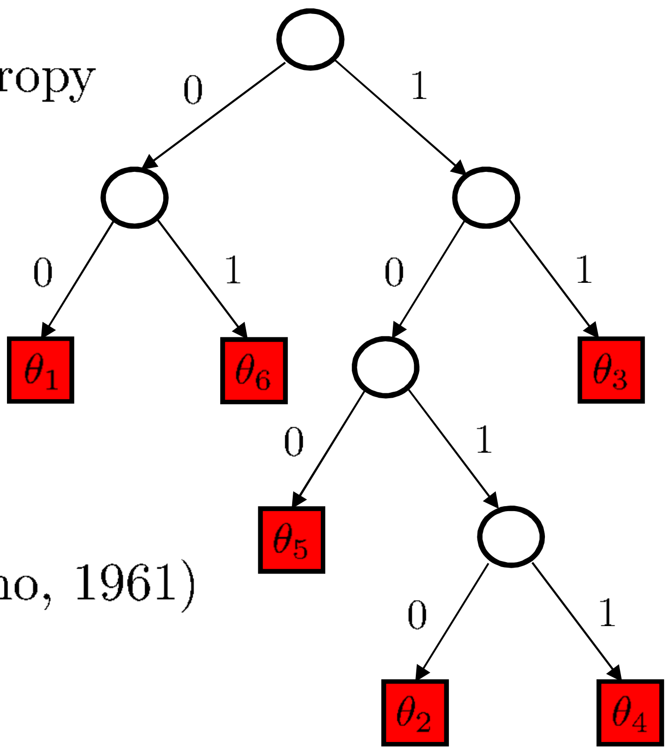
symbol	codeword	codelength ℓ_i
θ_1	00	2
θ_2	1010	4
θ_3	11	2
θ_4	1011	4
θ_5	100	3
θ_6	01	2



$$E[\text{codelength}] = \sum_i \pi_i \ell_i$$

Source Coding

- $E[\text{codelength}] \geq \underbrace{-\sum_i \pi_i \log_2 \pi_i}_{H_1(\{\pi_i\}),\ \text{Shannon entropy}}$



- Huffman coding (Huffman, 1952)
  - Optimal
  - Bottom-up
  - Doesn't generalize to query learning
- Shannon-Fano coding (Shannon, 1948; Fano, 1961)
  - Suboptimal
  - Top-down
  - Generalizes to query learning $\rightarrow$ GBS

Source Coding vs. Query Learning

- Same goal: minimize expected codelength / # of queries

  $\Longrightarrow E[\#\text{ of queries}] \geq H_1(\{\pi_i\})$

- Query learning does not allow arbitrary trees

	q_1	q_2	q_3	q_4	q_5
θ_1	1	0	0	0	1
θ_2	0	1	0	1	0
θ_3	1	1	1	1	1
θ_4	0	1	0	1	1
θ_5	0	0	0	1	1
θ_6	1	1	0	0	0

$\Longrightarrow$ only 5 possible splits
(versus 31 for source coding)

- In query learning, finding optimal tree is NP-complete

  $\Longrightarrow$ need suboptimal algorithms

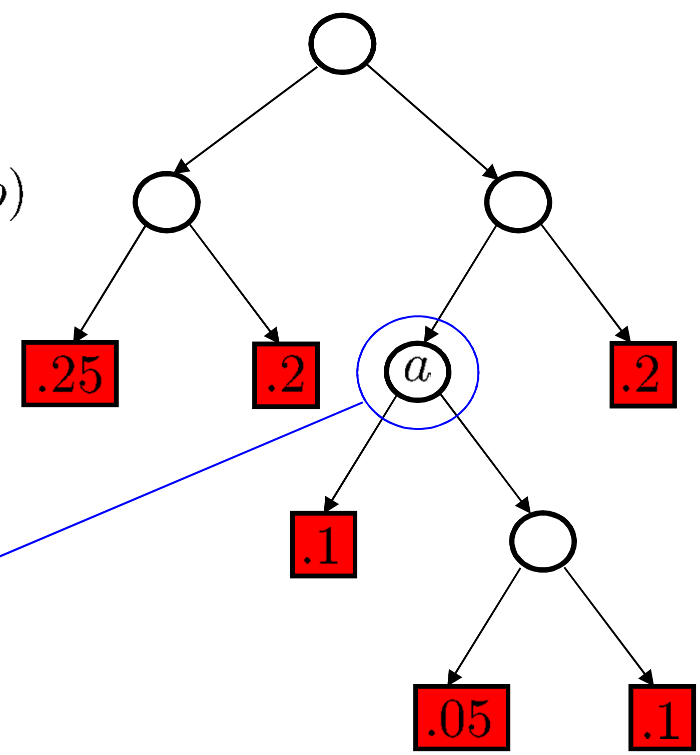
Exact formula for arbitrary tree/code

Theorem: For any decision tree T,

$$E[\#\text{ of queries}] = H_1(\{\pi_i\}) + \sum_{a \in \text{interior}(T)} \pi_a [1 - H(\rho_a)]$$

where

$$\begin{aligned}
\pi_a &:= \sum_{i \,:\, i \text{ reaches node } a} \pi_i \\
H(\rho) &:= -\rho \log_2 \rho - (1-\rho)\log_2(1-\rho) \\
\rho_a &:= \frac{\pi_{\text{leftchild}(a)}}{\pi_a}
\end{aligned}$$



$$\boxed{\rho_a = \frac{.1}{.1 + .05 + .1} = .4}$$

Query Learning as Greedy Optimization

$$E[\#\text{ of queries}] = H_1(\{\pi_i\}) + \sum_{a \in \text{interior}(T)} \pi_a [1 - H(\rho_a)]$$

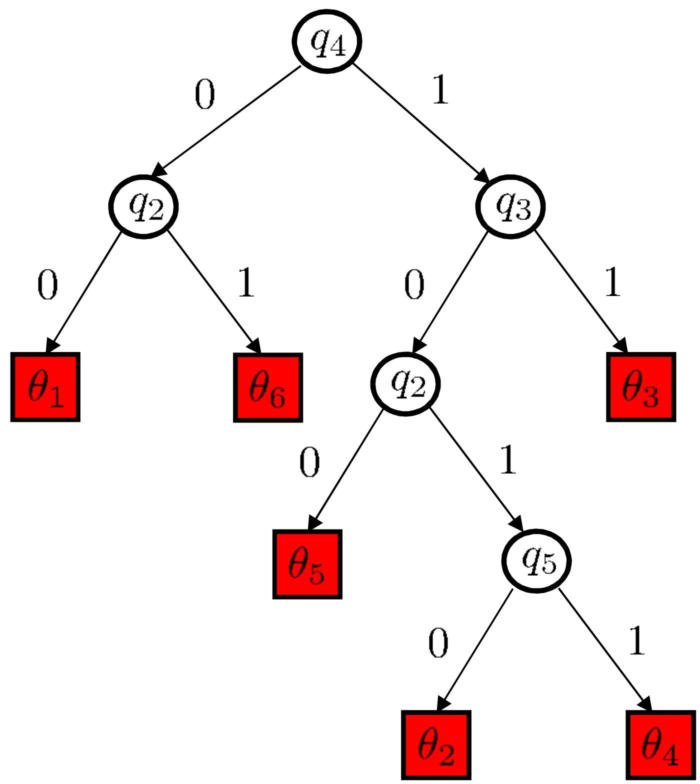
Top-down, greedy optimization

$\Longrightarrow$ maximize $H(\rho_a)$

$\Longrightarrow$ minimize $|\rho_a - \frac{1}{2}|$

$\Longrightarrow$ generalized binary search

	q_1	q_2	q_3	q_4	q_5	π_i
θ_1	1	0	0	0	1	.25
θ_2	0	1	0	1	0	.05
θ_3	1	1	1	1	1	.3
θ_4	0	1	0	1	1	.1
θ_5	0	0	0	1	1	.1
θ_6	1	1	0	0	0	.2



Exponentially Weighted Costs

For $\lambda \geq 1$, minimize

$$\log_\lambda \left(\sum_{i=1}^{M} \pi_i \lambda^{d_i} \right)$$

where $d_i =$ depth of θ_i

- $\lambda \to 1 \Longrightarrow$ average depth
- $\lambda \to \infty \Longrightarrow$ maximum depth (worst case)
- Source coding (arbitrary trees allowed) $\Longrightarrow$ efficient optimal algorithm
- Query learning $\Longrightarrow$ no efficient optimal algorithm

Rényi Entropy

Lower bound (Campbell, 1956): For any $\lambda > 1$ and any tree

$$\log_\lambda \left(\sum_{i=1}^{M} \pi_i \lambda^{d_i} \right) \geq H_\alpha(\{\pi_i\})$$

where

$$H_\alpha(\{\pi_i\}) = \frac{1}{1-\alpha} \log_2 \left(\sum_{i=1}^{M} \pi_i^\alpha \right)$$

and $\alpha = \frac{1}{1+\log_2 \lambda}$

Exact Formula for Exponential Costs

Theorem: For any fixed $\lambda \geq 1$, and any tree T,

$$\sum_{i=1}^{M} \pi_i \lambda^{d_i} = \lambda^{H_\alpha(\{\pi_i\})} + \sum_{a \in \mathrm{int}(T)} \pi_a \left[(\lambda - 1)\lambda^{d_a} - D_a^\alpha + \rho_a D_{\mathrm{left}(a)}^\alpha + (1 - \rho_a) D_{\mathrm{right}(a)}^\alpha \right]$$

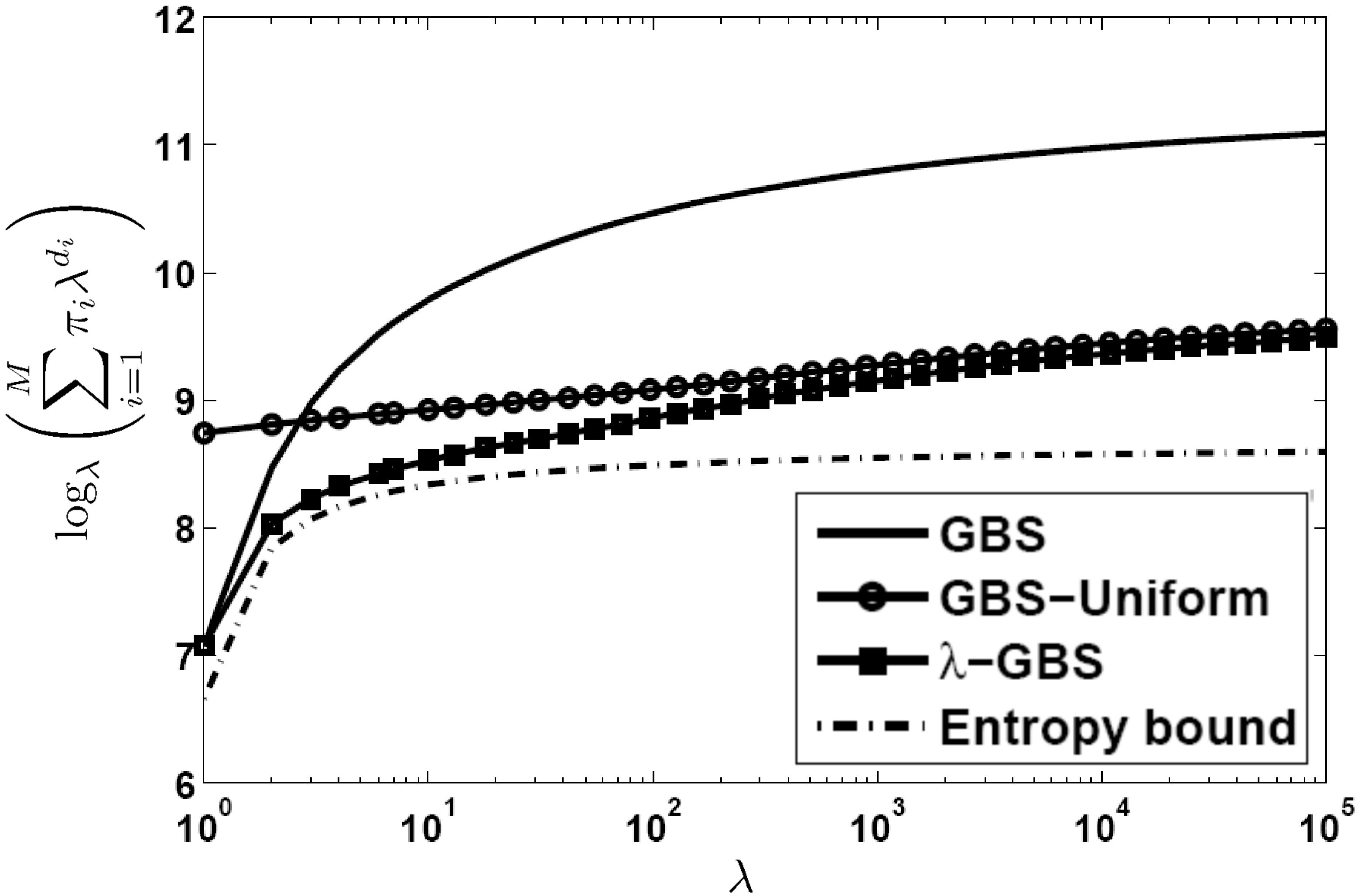
where

$$D_a^\alpha := \left[\sum_{i \,:\, i \text{ reaches node } a} \left(\frac{\pi_i}{\pi_a} \right)^\alpha \right]^{1/\alpha}$$

Greedy, top-down algorithm: λ-GBS

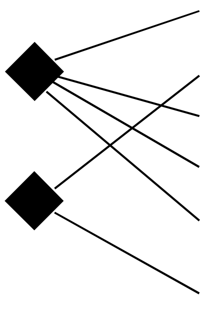
Results: WISER Database

300 chemicals, 80 symptoms, $\pi_i \propto i^{-1}$



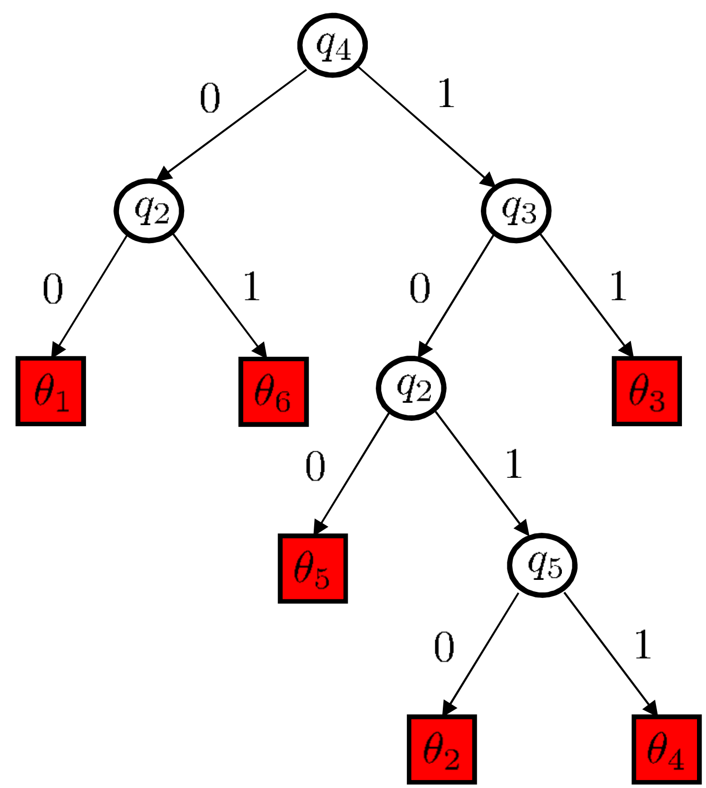
Group Identification

Example: Identify class to which chemical belongs (pesticide, poison, etc.)

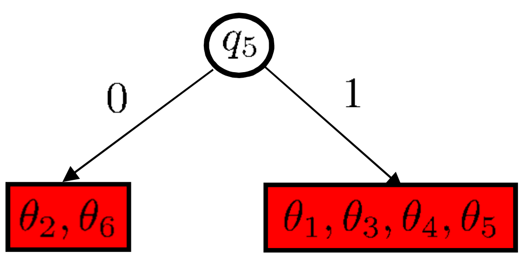


	q_1	q_2	q_3	q_4	q_5	π_i
θ_1	1	0	0	0	1	.25
θ_2	0	1	0	1	0	.05
θ_3	1	1	1	1	1	.3
θ_4	0	1	0	1	1	.1
θ_5	0	0	0	1	1	.1
θ_6	1	1	0	0	0	.2

GBS



Best tree

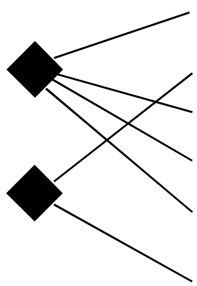


$E[\#\text{ of queries}] = 1$

$E[\#\text{ of queries}] = 2.4$

Group Identification

labels $y_i \in \{1, 2, \ldots, K\}$



	q_1	q_2	q_3	q_4	q_5	π_i	y_i
θ_1	1	0	0	0	1	.25	1
θ_2	0	1	0	1	0	.05	2
θ_3	1	1	1	1	1	.3	1
θ_4	0	1	0	1	1	.1	1
θ_5	0	0	0	1	1	.1	1
θ_6	1	1	0	0	0	.2	2

$\tilde{\pi}_1 = .75$

$\tilde{\pi}_2 = .25$

Theorem: For any tree T,

$$E[\#\text{ of queries}] = H_1(\{\tilde{\pi}_k\}) + \sum_{a \in \text{interior}(T)} \pi_a \left[1 - H(\rho_a) + \sum_{k=1}^{K} \frac{\pi_a^k}{\pi_a} H(\rho_a^k) \right]$$

Group-GBS

Greedy algorithm: At each successive node, choose query to minimize

$$1 - H(\rho_a) + \sum_{k=1}^{K} \frac{\pi_a^k}{\pi_a} H(\rho_a^k)$$

This prefers queries such that

- $\rho_a \approx \frac{1}{2} \Longrightarrow$ balanced trees
- $\rho_a^k \approx 0$ or 1 for each $k \Longrightarrow$ preserve groups

Group Identification Results

- WISER database (300 chemicals, 80 symptoms)
- 16 chemical classes (pesticide, poison, corrosive acid, etc.)
- uniform prior on chemicals

Entropy lower bound	3.07
Group-GBS	7.79
GBS	7.95
Random Search	16.33

- WISER-like database with better "concordance" within classes

Entropy lower bound	3.07
Group-GBS	3.50
GBS	7.51
Random Search	16.12

Performance Guarantee

Alternate greedy algorithm: At each successive node, choose query to maximize

$$\pi_{l(a)}\pi_{r(a)} - \sum_{k=1}^{K} \frac{\pi_a^k}{\pi_a} \pi_{l(a)}^k \pi_{r(a)}^k$$

Similar intuition as previous criterion.

Theorem: Let $\widehat{T}$ denote the tree based on the above splitting criterion, and let T^* be the tree with minimum expected depth. Then

$$\mathbb{E}[\text{depth}(\widehat{T})] \leq \left(2\ln\left(\frac{1}{\sqrt{3}\pi_{min}}\right) + 1\right) \mathbb{E}[\text{depth}(T^*)]$$

where $\pi_{min} = \min_i \pi_i$.

Query Noise

- Suppose θ_2 is the true object

	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	q_9
θ_1	1	0	1	1	0	1	1	1	0
θ_2	0	1	1	0	1	0	0	1	0
θ_3	0	1	0	1	0	0	1	1	1
θ_4	1	1	0	0	1	1	1	0	1

- Ideal query responses:

θ_2	0	1	1	0	1	0	0	1	0

- Noisy query responses:

θ_2	0	1	0	0	1	1	0	1	0

Query Noise

Nearest neighbor decoding: If

$$\min_{i \neq j} \quad d_{\text{Hamming}}(\theta_i, \theta_j) \geq \epsilon$$

can recover

$$\delta = \left\lfloor \frac{\epsilon - 1}{2} \right\rfloor$$

query errors

	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	q_9
θ_1	1	0	1	1	0	1	1	1	0
θ_2	0	1	1	0	1	0	0	1	0
θ_3	0	1	0	1	0	0	1	1	1
θ_4	1	1	0	0	1	1	1	0	1

$$\epsilon = 5 \Longrightarrow \delta = 2$$

Query Noise

θ_1	1	0	1	1	0	1	1	1	0
θ_2	0	1	1	0	1	0	0	1	0
θ_3	0	1	0	1	0	0	1	1	1
θ_4	1	1	0	0	1	1	1	0	1

Group 1

1	0	1	1	0	1	1	1	0
0	0	1	1	0	1	1	1	0
1	1	1	1	0	1	1	1	0
1	0	0	1	0	1	1	1	0
1	0	1	0	0	1	1	1	0
1	0	1	1	1	1	1	1	0
1	0	1	1	0	0	1	1	0
1	0	1	1	0	1	0	1	0
	$\vdots$							

$\{\theta : d_{\text{Hamming}}(\theta, \theta_1) \leq \delta\}$

Group 2

0	1	1	0	1	0	0	1	0
1	1	1	0	1	0	0	1	0
0	0	1	0	1	0	0	1	0
0	1	0	0	1	0	0	1	0
0	1	1	1	1	0	0	1	0
0	1	1	0	0	0	0	1	0
0	1	1	0	1	1	0	1	0
0	1	1	0	1	0	1	1	0
	$\vdots$							

$\{\theta : d_{\text{Hamming}}(\theta, \theta_2) \leq \delta\}$

Group 3

0	1	0	1	0	0	1	1	1
1	1	0	1	0	0	1	1	1
0	0	0	1	0	0	1	1	1
0	1	1	1	0	0	1	1	1
0	1	0	0	0	0	1	1	1
0	1	0	1	1	0	1	1	1
0	1	0	1	0	1	1	1	1
0	1	0	1	0	0	0	1	1
	$\vdots$							

$\{\theta : d_{\text{Hamming}}(\theta, \theta_3) \leq \delta\}$

Group 4

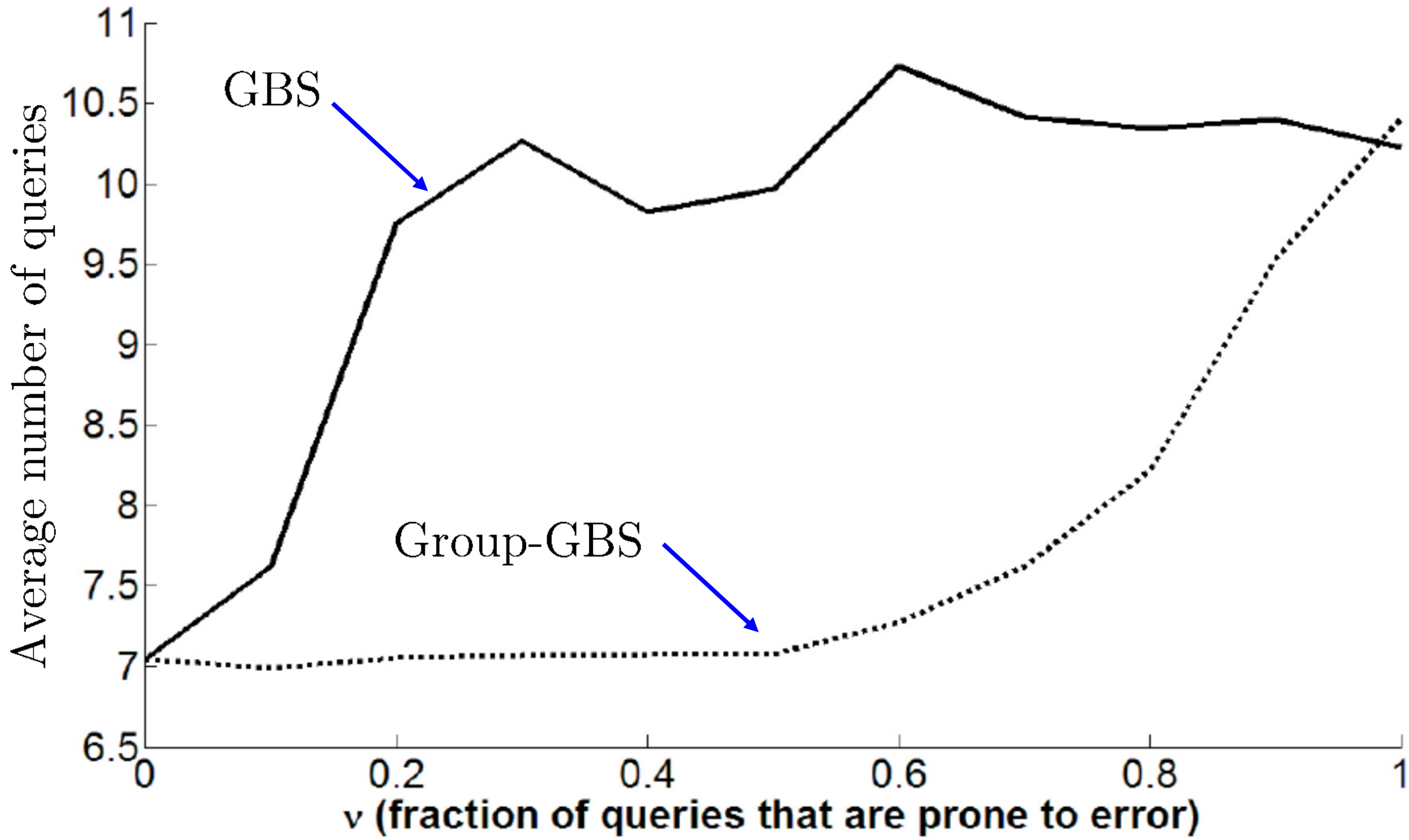
1	1	0	0	1	1	1	0	1
0	1	0	0	1	1	1	0	1
1	0	0	0	1	1	1	0	1
1	1	1	0	1	1	1	0	1
1	1	0	1	1	1	1	0	1
1	1	0	0	0	1	1	0	1
1	1	0	0	1	0	1	0	1
1	1	0	0	1	1	0	0	1
	$\vdots$							

$\{\theta : d_{\text{Hamming}}(\theta, \theta_4) \leq \delta\}$

Now apply Group-GBS

Query Noise Results

- Modified WISER database ($\epsilon = 5, \delta = 2$)



Preference Elicitation

- Given several designs for a product
  - E.g., laptop computers: each design has different combinations of features: memory, weight, cost, size, battery life, etc.
- Choose the most preferred design for a population of users
- Survey the population; Each survey consists of a sequence of pairwise comparisons
  - “Do you prefer Design A or Design B?”
- Goal: Construct survey to determine most preferred design using the minimal number of queries

Preference Elicitation as Group Identification

- Products $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_K \in \mathbb{R}^d$
- Each user has a ranking of products, e.g.,

$$\theta = \boldsymbol{x}_3 \prec \boldsymbol{x}_7 \prec \boldsymbol{x}_2 \prec \cdots$$

- Set of possible rankings

$$\Theta = \{\theta_1, \theta_2, \ldots, \theta_M\}$$

- Group rankings by most preferred design

$$\Theta_i^k := \{\theta \in \Theta \,|\, \boldsymbol{x}_k \text{ most preferred}\}$$

Pairwise Comparisons are Queries

	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	q_9
θ_1	1	0	1	1	0	1	1	1	0
θ_2	0	1	1	0	1	0	0	1	0
θ_3	0	1	0	1	0	0	1	1	1
θ_4	1	1	0	0	1	1	1	0	1
$\vdots$					$\vdots$				

This ranking is consistent with this query

This ranking is not consistent with this query

Some Interesting Features

- Initial probability distribution π_i on rankings: uniform
- Users surveyed sequentially, so can update distribution on rankings after each survey
- Estimating probability π_a, where a is a node in the tree, is nontrivial
- Assume *partworth model*: Each user is represented by a vector $\boldsymbol{w} \in \mathbb{R}^D$, and pairwise comparison between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ depends on the sign of

  $$\boldsymbol{w}^T(\boldsymbol{x}_i - \boldsymbol{x}_j).$$

  This enables geometric approximations of π_a quantities (details omitted but it's an SVM type algorithm)

Conclusion

- Query learning = constrained source coding
- Exact formulas for performance → greedy algorithms
- Group identification → preference elicitation