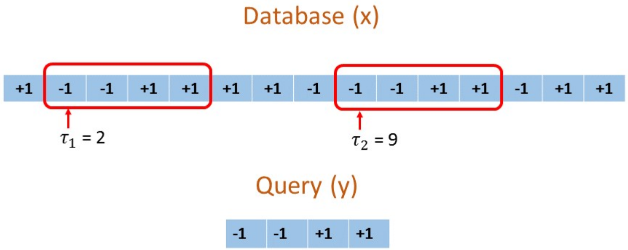
The Peeling Decoder : Theory and some Applications

Krishna R. Narayanan
Thanks to Henry Pfister, Avinash Vem, Nagaraj Thenkarai Janakiraman, Kannan Ramchandran, Jean-Francois Chamberland

Department of Electrical and Computer Engineering
Texas A&M University

Pattern Matching



- Database/String: $\underline{x} = [x[0], x[1], \cdots, x[N]]$ (length N)
- Query/Substring: $\underline{y} = [y[0], x[1], \cdots, x[M]]$ (length $M = N^\mu$)
- Determine all the L locations $\underline{\tau} = [\tau_1, \tau_2, \cdots \tau_L]$ with high probability where
  1. Exact Matching: $\underline{y}$ appears exactly in $\underline{x}$
     - $\underline{y} := \underline{x}[\tau : \tau + M - 1]$
  2. Approximate Matching: $\underline{y}$ is a noisy substring of $\underline{x}$
     - $\underline{y} := \underline{x}[\tau : \tau + M - 1] \odot \underline{b}$
     - $\underline{b}$ is a noise sequence with $d_H(\underline{y}, \underline{x}[\tau : \tau + M - 1]) \leq K$

Main Result

Theorem 1

Assume that a sketch of $\underline{x}$ of size $O(max(M, N/M)\log N)$ can be precomputed and stored. Then for the exact pattern matching and approximate pattern matching (with $K = \eta M,\ 0 \leq \eta \leq 1/6$) problems, with the number of matches L scaling as $O(N^{\lambda})$, our algorithm has

- *a sketching function for $\underline{y}$ that computes* $O(\max(M, \frac{N}{M})\log N) = O(N^{\max(\mu, 1-\mu)}\log N)$ *samples*
- *a computational complexity of*
  - $O(\max(N^{1-\mu}\log^2 N, N^{\mu+\lambda}\log N))$ *for* $\mu < 0.5$
  - $O(\max(N^{\mu}\log^2 N, N^{1-\mu+\lambda}\log N))$ *for* $\mu > 0.5$
- *a decoder that recovers all the L matching positions with a failure probability that approaches zero asymptotically*

Note: Particularly when $L = O(1)$ or $L < \frac{N}{M}$ (i.e. $\lambda < 1 - \mu$) our algorithm has a sub-linear time complexity.

Some Prior Work

Exact Matching

- **boyer1977fast**: First occurrence of the match (only τ_1)
  - Average complexity - $O(N^{1-\mu} \log N)$ (sublinear)
  - Worst case complexity - $O(N \log N)$

Approximate Matching

- **chang1994approximate**: Generalization of **boyer1977fast**
  - Average complexity - $O(NK/M \log N)$ (sub-linear only when $K \ll M$)
- **andoni2013shift**: $O(N/M^{0.359})$ (sub-linear even when $K = O(M)$)
  - Combinatorial in nature

Sparse Fourier Transform Approach

- **hassanieh2012faster**: Faster GPS receiver
  - Exploited sparsity in Correlation function R_{XY}
- **pawar2014robust**: Robust Sparse Fourier Transform
  - Sparse Graph code Approach
  - Computational complexity : $O(N \log N)$

Motivation

- **Cross-correlation** ($\underline{r}$):

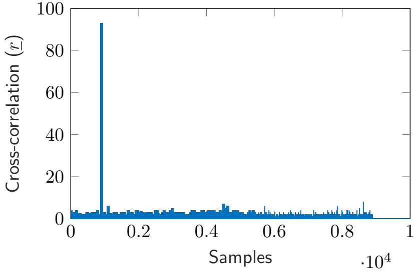
$$r[m] = (\underline{x} * \underline{y})[m] \triangleq \sum_{i=0}^{M-1} x[m+i]y[i], \quad 0 \leq m \leq N-1$$

- **Naive implementation**: $O(MN) = O(N^{1+\mu})$ (super-linear complexity)
- **Fourier Transform Approach**: $O(N \log N)$ complexity

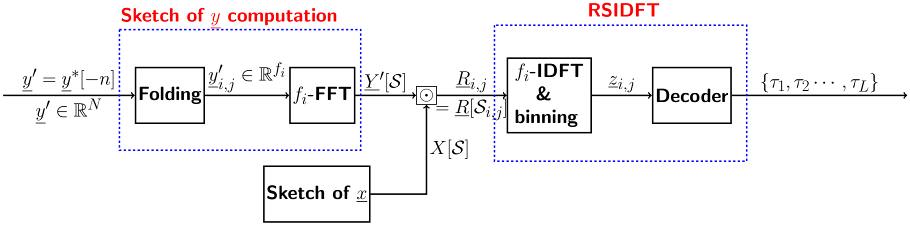
Key Observation

- $\underline{r}$ is Sparse with some noise.

$$r[m] = \begin{cases} M, & \text{if } m \in \mathcal{T} \\ n_m, & m \in [N] - \mathcal{T} \end{cases}$$



Sparse Fourier Transform Approach



$$\boxed{\underline{r} = \underbrace{\mathcal{F}_N^{-1}}_{1} \{\underbrace{\mathcal{F}_N\{\underline{x}\}}_{2} \odot \underbrace{\mathcal{F}_N\{\underline{y}'\}}_{3}\}}$$

1. *Sketch of $\underline{x}$:* Assume $\underline{X}[l] = \mathcal{F}\{\underline{x}\}$ is precomputed at positions $l \in \mathcal{S}$.
2. *Sketch of $\underline{y}$:*
   - Compute $\underline{Y}'[l] = \mathcal{F}\{\underline{y}'\}$ for $l \in \mathcal{S}$.
   - Only M non-zero values in $\underline{y}'$ - Efficient computation (folding and adding)
3. *Sparse $\mathcal{F}^{-1}$*:
   - Robust Sparse Inverse Fourier Transform (RSIDFT)
   - Efficient Implementation- sublinear time and sampling complexity

Questions?

Thank you!