CS 830: Randomness and Complexity 4/5/05

Lecture 21: Saks-Zhou Rounding (con't)

Instructor: Jin-Yi Cai Scribe: Aparna Das, Giordano Fusco

1 Overview

We continue presenting the Saks-Zhou result that $BPL \subseteq DTISP[n^{O(\log^{0.5} n)}, O(\log^{1.5} n)]$. We end with a brief introduction to highlight the main idea of the CCM paper.

2 Saks-Zhou Rounding Sequence

As we saw last time, the Saks-Zhou algorithm saves space by reusing hash functions. Specifically, instead of using r hash functions, they use r_1 hash function $\vec{h} = (h_1, h_2, \ldots, h_{r_1})$ cyclically r_2 times, where $r_1 r_2 = r$. Between each cycle a randomized rounding scheme is applied to the matrix $M = \mathcal{M}(Q_{\vec{h}})$ to make the automata independent of $\vec{h}$. Recall that the rounding scheme guaranteed that:

$$\forall z, s \in [0,1] \text{ and all but two values of } \delta \in \{0,1\}^D, \text{ if } |z - s| < 2^{-(t+D)}, \text{ then } R_{\delta,t}(z) = R_{\delta,t}(s) \quad (1)$$

We will take $D = O(\log d)$.

The rounded sequence is defined as follows:

Definition 2.1. *Let*

- *M be a (d,m)-matrix over $[0,1]$,*
- *t, D, r_1, k be positive integers with $t \leq m$,*
- *$\vec{\delta} = (\delta_1, \delta_2, \ldots, \delta_k) \in (\{0,1\}^D)^k$,*

Define the rounded sequence *$R(M, \vec{\delta})$ as*

$$N_0 \xrightarrow{\mathcal{P}} \widetilde{N}_1 \xrightarrow{\mathcal{R}_{\delta_1,t}} N_1 \xrightarrow{\mathcal{P}} \widetilde{N}_2 \xrightarrow{\mathcal{R}_{\delta_1,t}} N_2 \longrightarrow \cdots\cdots \longrightarrow N_{r_2-1} \xrightarrow{\mathcal{P}} \widetilde{N}_{r_2} \xrightarrow{\mathcal{R}_{\delta_1,t}} N_{r_2}$$

where $N_0 = M$, $\widetilde{N}_i = N_{i-1}^{2^{r_1}}$, and $N_i = R_{\delta_i,t}(\widetilde{N}_i)$. The $\mathcal{P}$ operator denotes powering a matrix by 2^{r_1}.

3 Saks-Zhou Rounding is not too bad

The following lemma says that for most δ, rounding does not effect the final result by much.

LEMMA 3.1. *Let M, t, D, r_1, k and $\vec{\delta}$ be defined as above. Then the matrix, N_k defined in the rounded sequence defined in $R(M, \delta)$ satisfies,*

$$||N_k - M^{2^{r_1 k}}|| \leq 2^{-t+\log d + r_1 k}$$

Note: $k = r_2$ where $r_1 r_2 = r = O(\log n)$.

Proof.

We will prove inductively that:

$$||N_k - M^{2^{r_1 k}}|| \leq 2^{-t+\log d}(1 + 2^{r_1} + 2^{2r_1} + \ldots + 2^{(k-1)r_1})$$

Since $(1 + 2^{r_1} + 2^{2r_1} + \ldots + 2^{(k-1)r_1}) = \frac{2^{kr_1}-1}{2^{r_1}-1} < 2^{kr_1}$, the lemma will follow.

Base case ($k = 1$):

Applying definitions we have that $\widetilde{N}_1 = M^{2^{r_1}}$ and $N_1 = R_{\delta,t}(\widetilde{N}_1)$. Since the maximum round off error of one step by (1) is $2^{-(t+\log d)} < 2^{-t+\log d}$ the base case is verified.

Inductive Case: Assume the lemma holds for k and prove for $k + 1$.

Since $\widetilde{N}_{k+1} = N_k^{2^{r_i}}$ and $N_{k+1} = R_{\delta_{k+1},t}(\widetilde{N}_{k+1})$ by the triangle inequality we have that:

$$\begin{aligned} ||N_{k+1} - M^{2^{r_i(k+1)}}|| &= ||N_{k+1} - \widetilde{N}_{k+1} + N_k^{2^{r_1}} - M^{2^{r_1(k+1)}}|| \\ &\leq ||N_{k+1} - \widetilde{N}_{k+1}|| + ||N_k^{2^{r_1}} - M^{2^{r_1(k+1)}}|| \end{aligned}$$

We claim that the following equation holds (proof provided below):

$$||N_k^{2^{r_1}} - M^{2^{r_1(k+1)}}|| \leq 2^{-t+\log d}(2^{r_1} + 2^{2r_1} + \ldots + 2^{r_1 k}) \tag{2}$$

Using (2), and that $||N_{k+1} - \widetilde{N}_{k+1}|| \leq 2^{-t+\log d}$ by the rounding property, (1) we have that

$$\begin{aligned} ||N_{k+1} - M^{2^{r_1(k+1)}}|| &\leq ||N_{k+1} - \widetilde{N}_{k+1}|| + ||N_k^{2^{r_1}} - M^{2^{r_1(k+1)}}|| \\ &\leq 2^{-t+\log d} + 2^{-t+\log d}(2^{r_1} + 2^{2r_1} + \ldots + 2^{r_1 k}) \\ &= 2^{-t+\log d}(1 + 2^{r_1} + 2^{2r_1} + \ldots + 2^{r_1 k}) \end{aligned}$$

Now we show that equation (2) holds.

For ease of notation, rename the variables in the equation as $||N_k^{2^{r_1}} - M^{2^{r_1(k+1)}}|| = ||A^l - B^l||$, where $A = N_k$, $B = M^{2^{r_1 k}}$ and $l = 2^{r_1}$. We have the following:

$$\begin{aligned} A^l - B^l &= A^l - A^{l-1}B + A^{l-1}B - A^{l-2}B^2 + \ldots \\ &= \sum_{i=0}^{l-1}(A^{l-i}B^i - A^{l-i-1}B^{i+1}) \\ &= \sum_{i=0}^{l-1}(A^{l-i-1}(A - B)B^i) \end{aligned}$$

Using the fact that norm of a sum (or a product) is $\leq$ to the sum (or product) of the norm, we get that:

$$\begin{aligned} ||A^l - B^l|| &\leq \sum_{i=0}^{l-1} ||A||^{l-i-1} ||A-B|| \cdot ||B||^i \\ &= l\,||A-B|| \\ &= 2^{-t+\log d}(2^{r_1} + 2^{2r_1} + \ldots + 2^{r_1 k}) \end{aligned}$$

The second line follows from the fact A, B are sub-stochastic which implies that $||A||, ||B|| = 1$. Substituting back for variables A, B and l, we get the result.

□

Recall that the goal for simulating BPL was to approximate M^{2^r}. So, since $d = r = O(\log n)$, we can take $t = O(\log n)$, with a large constant so that error from rounding is small and we still get that $N_k \sim M^{2^r}$.

4 Saks-Zhou Result

In this section we examine Saks-Zhou Sequence in greater detail and see how "safe" $\vec{\delta} = (\delta_1, \delta_2, \ldots, \delta_k)$ can be found for which the Saks-Zhou rounding works well and complete the Saks-Zhou analysis.

Definition 4.1. *Let $M, t, D, r_1, k, \vec{\delta}$ be defined as above. Let $\vec{h} = (h_1, h_2, \ldots, h_{r_1}) \in \mathcal{H}_m^{r_1}$ be a sequence of hash functions from $\mathcal{H}_m$. We define the Saks-Zhou sequence $SZ(M, \vec{h}, \vec{\delta})$ with parameters t, r_1 and k as the following sequence of matrices and automata:*

$$\begin{tikzcd}
M_0 \arrow[d, "\mathcal{Q}"'] & \widetilde{M}_1 \arrow[r, "\mathcal{R}_{\delta_1,t}"] & M_1 \arrow[d, "\mathcal{Q}"'] & \widetilde{M}_2 \arrow[r, "\mathcal{R}_{\delta_2,t}"] & M_2 \arrow[d, "\mathcal{Q}"'] & \cdots\cdots \arrow[r] & M_{r_2-1} \arrow[d, "\mathcal{Q}"'] & \widetilde{M}_{r_2} \arrow[r, "\mathcal{R}_{\delta_2,t}"] & M_{r_2} \\
Q_0 \arrow[r, "\mathcal{N}_{\vec{h}}"] & \widetilde{Q}_1 \arrow[u, "\mathcal{M}"] & Q_1 \arrow[r, "\mathcal{N}_{\vec{h}}"] & \widetilde{Q}_2 \arrow[u, "\mathcal{M}"] & Q_2 \arrow[r] & \cdots\cdots & Q_{r_2-1} \arrow[r, "\mathcal{N}_{\vec{h}}"] & \widetilde{Q}_{r2} \arrow[u, "\mathcal{M}"] &
\end{tikzcd}$$

where $M_0 = M$, and for $1 \leq i \leq k$:

- $Q_{i-1} = \mathcal{Q}(M_{i-1})$
- $\widetilde{Q}_{i-1} = \mathcal{N}_{h_1,h_2,\ldots,h_{r_1}}(Q_{i-1})$
- $\widetilde{M}_1 = \mathcal{M}(\widetilde{Q}_i)$
- $M_i = \mathcal{R}_{\delta_1,t}(\widetilde{M}_i)$

Now we define two levels of *safety* for δ:

Definition 4.2. *Let M and N be $d \times d$-matrices over $[0,1]$, t and D positive integers. We say that $\delta \in \{0,1\}^D$ is* t-safe *if for all $d \times d$-matrix Z over $[0,1]$ satisfying $||Z - M|| < 2^{-(t+D)}$ we have*

$$\mathcal{R}_{\delta,t}(Z) = \mathcal{R}_{\delta,t}(M)$$

We say that $\delta \in \{0,1\}^D$ is t-weakly-safe *if the same hold when Z is $||Z - M|| < 2^{-(t+D+1)}$*

By (1) for any $z \in [0,1]$ there are at most 2 values of δ, which are not safe. This implies that for every M there are at most 2^{d^2} many δ, out of 2^D, that are not safe for M. For a large enough choice of $D = O(\log d)$, there will be many safe δ and we can exhaustively search for a t-safe δ for M in space $O(D + \log d + \log t)$.

If δ_i is t-safe for each $\widetilde{N}_i$ then $\vec{\delta}$ is safe for the sequence rounded sequence $R(M, \vec{\delta})$. When $\vec{\delta}$ is safe for $R(M, \vec{\delta})$, for most $\vec{h}$, $M_i = N_i$ for $0 \le i \le r_2$. To see this, note that if we find a $\vec{h}$ which is ϵ-pseudorandom for Q_0, then with high probability $\vec{h}$ will be ϵ-pseudorandom for $Q_1, Q_2, \ldots, Q_r$, since these automata are defined in terms of δ and M_0 and not in terms of $\vec{h}$. Thus $M_{r_2} = N_{r_2} \sim M^{2^{r_1 r_2}}$.

In summary the Sak-Zhou algorithm works as follows:

1. After computing the Q_0 from M_0 we find a $\vec{h} = (h_1, \ldots h_{r_1})$ that is ϵ-psuedorandom for Q_0.
2. Finding the $\vec{h}$ requires $(r_1 \log n)$ space since we find $\vec{h}$ all at once and not one at a time.
3. Then we compute $\widetilde{M}_1$. At this point we have that $\widetilde{N}_1 \approx \widetilde{M}_1$.
4. Now we find a safe δ for $\widetilde{M}_1$ and apply the Saks-Zhou rounding to obtain that $N_1 = M_1$.

   The following diagram shows these steps

$$\begin{tikzcd}
N_0 \arrow[r, "\mathcal{P}"] \arrow[d, equal] & \widetilde{N}_1 \arrow[r, "\mathcal{R}_{\delta_1,t}"] \arrow[d, phantom, "\wr"] & N_1 \arrow[d, equal] \\
M_0 \arrow[d, "\mathcal{Q}"'] & \widetilde{M}_1 \arrow[r, "\mathcal{R}_{\delta_1,t}"] & M_1 \\
Q_0 \arrow[r, "\mathcal{N}_{\vec{h}}"] & \widetilde{Q}_1 \arrow[u, "\mathcal{M}"] &
\end{tikzcd}$$

5. At this point we repeat the steps above r_2 times. In each cycle we find a δ_i that is safe for $\widetilde{M}_i$, but we always use the same $\vec{h}$ as we did in the first round.

This diagram shows the correspondence of the two sequences

$$\begin{tikzcd}
N_0 \arrow[r, "\mathcal{P}"] \arrow[d, equal] & \widetilde{N}_1 \arrow[r, "\mathcal{R}_{\delta_1,t}"] \arrow[d, phantom, "\wr"] & N_1 \arrow[r, "\mathcal{P}"] \arrow[d, equal] & \widetilde{N}_2 \arrow[r, "\mathcal{R}_{\delta_1,t}"] \arrow[d, phantom, "\wr"] & N_2 \arrow[r] \arrow[d, equal] & \cdots\cdots \arrow[r] & N_{r_2-1} \arrow[r, "\mathcal{P}"] \arrow[d, equal] & \widetilde{N}_{r_2} \arrow[r, "\mathcal{R}_{\delta_1,t}"] \arrow[d, phantom, "\wr"] & N_{r_2} \arrow[d, equal] \\
M_0 \arrow[d, "\mathcal{Q}"'] & \widetilde{M}_1 \arrow[r, "\mathcal{R}_{\delta_1,t}"] & M_1 \arrow[d, "\mathcal{Q}"'] & \widetilde{M}_2 \arrow[r, "\mathcal{R}_{\delta_2,t}"] & M_2 \arrow[d, "\mathcal{Q}"'] & \cdots\cdots \arrow[r] & M_{r_2-1} \arrow[d, "\mathcal{Q}"'] & \widetilde{M}_{r_2} \arrow[r, "\mathcal{R}_{\delta_2,t}"] & M_{r_2} \\
Q_0 \arrow[r, "\mathcal{N}_{\vec{h}}"] & \widetilde{Q}_1 \arrow[u, "\mathcal{M}"] & Q_1 \arrow[r, "\mathcal{N}_{\vec{h}}"] & \widetilde{Q}_2 \arrow[u, "\mathcal{M}"] & Q_2 \arrow[r] & \cdots\cdots & Q_{r_2-1} \arrow[r, "\mathcal{N}_{\vec{h}}"] & \widetilde{Q}_{r2} \arrow[u, "\mathcal{M}"] &
\end{tikzcd}$$

The Saks-Zhou algorithm runs through all possible $\vec{h} = (h_1, \ldots, h_r)$ and $\vec{\delta} = (\delta_1, \ldots \delta_{r_2})$ to find good choices. This requires $O(\max\{r_1, r_2\} \log d)$ space and $d^{r_1 + r_2}$ time. Since $d = n^c$ and taking $r_1 = r_2 = \sqrt{\log d}$, this gives $BPL = DTISP[n^{O(\log^{0.5} n)}, O(\log^{1.5} n)]$.

5 CCM Overview

The goal in CCM is to kill the factor of $d^{O(r_1)}$ in the running time of Saks-Zhou algorithm. This extra factor comes trying all possible sequences of r_1 hash functions. Instead here the idea is to find the hash functions h_i in $\vec{h}$ one at a time (like the Nissan way). In order to do this a new sequence of hash functions $\vec{h}$ is found in each of the r_2 cycles. Due to space bounds it is infeasible to store all sequences of hash function used. However discarding $\vec{h}$, means we lose access to matrices and automata at lower levels of the recursion. For example suppose that after computing $\widetilde{M}_1$, M_1 and Q_1, using $\vec{h}$, we compute a new $\vec{h}$ which is ϵ-psuedorandom for Q_0 and Q_1. If we discard $\vec{h}$, we no longer have access to $\widetilde{M}_1 = \mathcal{M}$, which is what the current automaton Q_1 is defined from. If at this point the new automaton $\mathcal{Q}(M_1') \neq Q_1$, where M_1' is computed using h', then no progress would be made (i.e we would need to find a new $\vec{h}$ again).
The important fact which save us is that δ_1 is weakly-safe for $\widetilde{N}_1$. So while $\widetilde{M}_1' \neq \widetilde{M}_1$, the rounding operator $R_{\delta_1,t}$ guarantees that $M_1 = M_1'$ so $Q_1 = Q_1'$.