1 Universal Traversal Sequence

1.1 Reminder

In the previous lecture we’ve seen that $USTCON \in L$. Shortly, given an undirected graph $G$ of degree $D$ and two vertices $s$ and $t$, we’ve used a known explicit construction of degree $d$ expander’s family $\{H_i\}_i$ to define $G_0 = G$ and

$$G_i = G_{i-1} \circledast H_{i-1},$$

for every $i \leq m_0 = O(\log n)$. Then, for $i > m_0$, we defined the following family of expanders

$$H'_i = (H_{m_0 - 1 + 2^{i-m_0}})^{2^{i-m_0}},$$

and the graphs

$$G_i = G_{i-1} \circledast H'_{i-1}.$$

Finally, we looked at $G_m$ for $m = m_0 + \log \log n + O(1)$ and claimed that the connected components of $G$ and $G_m$ are identical, and also that if $s$ is connected to $t$ in $G$ then $s$ is a neighbor of $t$ in $G_m$. Finally, we argued that we can iterate over all neighbors of $s$ in $G_m$ in logarithmic space.

Note that for every vertex $v$ in $G_m$, every edge-label of $v$ is of the form

$$(\sigma, i_0, \ldots, i_{m_0}, i_{m_0+1}, \ldots, i_m),$$

where $\sigma \in [D], i_0, \ldots, i_{m_0} \in [d]$ and so on. We’ve also seen that computing

$$\text{Rot}_{G_m}(v, (\sigma, i_0, \ldots, i_{m_0}, i_{m_0+1}, \ldots, i_m))$$

can be though of as an in-order walk on a trinary-tree, where each node represents a computation, and the computations on the leaves correspond to a sequence of walking instructions on $G$ that takes us from $s$ to $t$.

1.2 Universal Traversal Sequence

Definition 1. A labelled graph is locally-invertible if

$$\text{Rot}_G(v, i) = (v[i], \phi(i)),$$

for some permutation $\phi$.

Observation 2. If $G$ is $\phi$-locally-invertible then the generated sequence of instructions that takes us from $s$ to $t$ does not depend on $s$. 


The following claims follow by similar proofs to those we saw in the last lecture:

\[ \text{Rot} \text{ ends as edge-labels in } G \text{H as before, and note that now it corresponds to picking an edge of } G \]

Using the above definition of the rotation map for digraphs, we can define \( \text{Rot} \)

\[ \text{Claim 6. Let } F \text{ be the family of undirected } D\text{-regular labelled graphs which are } \phi\text{-locally invertible.} \]

\[ \text{Then there exists a logspace construction of UTS for } F. \]

**Proof.** From the observation we see that for every graph \( G \in F \), every vertex \( v \), and every edge-label \( i = (\sigma, i_0, \ldots, i_m, i_{m+1}, \ldots, i_n) \), the sequence of instructions that are generated by computing \( \text{Rot}_{G_m}(v, i) \) is independent of \( v \). Hence, we can simply write \( \text{Rot}_{G_m}(i) \). Moreover, note that the output of \( \text{Rot}_{G_m}(i) \) is some edge-label \( \tilde{i}' \), and \( \text{Rot}_{G_m}^m(\tilde{i}')) = \tilde{i} \).

This implies the following algorithm: iterate over all possible edge-labels \( \tilde{i} \), and for each one compute \( \text{Rot}_{G_m}^m(\tilde{i}) \), and while computing, print to the output tape the corresponding sequence of instructions generated by the computation of \( \text{Rot}_{G_m} \). After computing \( \text{Rot}_{G_m}^m(\tilde{i}) \) the work-tape has changed to some other edge-label \( \tilde{i}' \), for which we compute \( \text{Rot}_{G_m}^m(\tilde{i}') \) and print to the output tape the corresponding sequence of instructions. Now the work-tape is once again \( \tilde{i} \) and we move to the next edge-label.

Note that the above can be implemented in logarithmic space, and that if the sequence of instructions that corresponds to \( \tilde{i} \) goes from \( v \) to \( u \), then the sequence of instructions that corresponds to \( \tilde{i}' \) goes from \( u \) to \( v \). This implies that the whole sequence, when starting at some vertex \( v \) of \( G \), repeatedly goes (on \( G! \)) from \( v \) to some neighbor of \( v \) in \( G_m \), and then back to \( v \). Since every vertex in the connected component of \( v \) in \( G \) is a neighbor of \( v \) in \( G_m \) it follows that the whole sequence visits every vertex in the connected component of \( v \), as required.

\[ \square \]

### 1.3 Generalization

In the following generalization we will look at \( D\)-regular digraphs which are **consistently labelled**.

**Definition 5.** A labelled \( D\)-regular graph is **consistently labelled** if for every \( v \in V \) and every \( i \in [D] \) there exists exactly one neighbor \( w \) s.t. \( w[i] = v \).

**Claim 6. Let \( G \) be a \( D\)-regular digraph. Then**

1. \( ||G|| \leq 1 \).
2. The all 1’s vector is an eigenvector with eigenvalue 1.
3. Let \( V^± \) be the orthogonal subspace to the span of the all 1’s vector. Then \( V^± \) is invariant under \( G \).

For such a \( D\)-regular digraph we define the rotation map \( \text{Rot} : V \times [D] \to V \times [D] \) by \( \text{Rot}(v, i) = (v[i], i) \). Note that if \( G \) is consistently labelled then \( \text{Rot}_G \) is a permutation.

Using the above definition of the rotation map for digraphs, we can define \( G \otimes H \) in the same way as before, and note that now it corresponds to picking and edge of \( H \) at random and using both ends as edge-labels in \( G \). Formally, for \( v \in V, \sigma \in [D] \) and \( i \in [d] \) we have

\[ \text{Rot}_{G \otimes H}(v, \sigma, i) = (v'' \sigma, \sigma, i) \]

where \( \text{Rot}_G(v, \sigma) = (v', \sigma), \text{Rot}_H(\sigma, i) = (\sigma', i) \) and \( \text{Rot}_G(v', \sigma') = (v'', \sigma') \).

The following claims follow by similar proofs to those we saw in the last lecture:
Claim 7. If $G$ is a connected $D$-regular digraph then $\lambda(G) \geq 1/n^4$.

Claim 8. If $G$ is a connected $D$-regular digraph then $\lambda(G_m) \geq 1 - 1/10n$.

Corollary 9. If $s$ is connected to $t$ in $G$ then $s$ is a neighbor of $t$ in $G_m$.

2 Universal Exploration Sequence

Let $G$ be a $D$-regular undirected graph. We’ve seen that one way of walking on the graph is keeping in memory only the current vertex $v$ where we stand at, and given an instruction $\sigma \in [D]$ simply walk to the $\sigma$ neighbor of $v$.

Another way of walking on the graph is keeping in memory, in addition to the vertex $v$, also $v$’s label of the last edge $(u,v)$ that we’ve just traversed. If this label is $\tau$ and we are given an instruction $\sigma \in [D]$, then we simply traverse the edge whose label is $\tau + \sigma \mod D$. This kind of walk is called exploration sequence.

Definition 10. Let $F$ be a family of $D$-regular undirected labelled graphs. We say that $\sigma = (\sigma_1,\ldots,\sigma_T) \in [D]^T$ is a universal exploration sequence (UES) for $F$ if for every $G \in F$ and starting edge $e$, the walk obtained by $\sigma$ visits all the edges of the graph.

Claim 11. The exists a logspace construction of UES.

We will prove the above claim in HW. One way to prove it is using the construction of UTS for regular locally-invertible graphs that we’ve seen. Another way is that given an undirected $D$-regular graph $G$, we can construct a graph $L(G)$ whose vertices are the (directed) edges $(i,j)$ (i.e. for every undirected edge $\{i,j\}$ in $G$ there are two vertices $(i,j)$ and $(j,i)$), and a vertex $(i,j)$ is connected to $(j,k)$ iff $\{i,j\}$ and $\{j,k\}$ are edges of $G$. Note that every labelling of the neighbors in $G$ induces a labelling on the neighbors in $L(G)$, and we claim that $L(G)$ is consistently labelled.

3 Some Words on Reingold’s Proof that $USTCON \in L$

Now we will shortly describe Reingold’s proof that $USTCON \in L$ which we will also see in HW. Let $G$ be a (wlog) $D^2$-regular undirected graph with self-loops on every vertex. Let $H$ be a fixed $[D^4,D,1/4]$-graph. We define $G_0 = G$ and

$$G_{t+1} = G_t^2 @ H.$$ 

Note that squaring improves the gap but also increases the degree, while the zig-zag product reduces the degree back to $D^2$ but also slightly decreases the gap (and also, as a side effect, increases the number of vertices). Since the gap of $G_0$ is non-negligible, it can be shown that for $m = O(\log n)$ we have $\text{gap}(G_m) \geq 1/18$. Note that $G_m$ is a constant degree graph with polynomial-number of vertices, and that every node $s_m$ in the cloud that corresponds to $s$ in $G_m$ is connected to any node $t_m$ in the cloud that corresponds to $t$ in $G_m$ iff $s$ is connected to $t$ in $G$. Hence all that remains is to try all paths of length $O(\log n)$ in $G_m$ from some $s_m$ to some $t_m$, and we can show that this can be implemented in logarithmic space.
4 Extractors

Definition 12. Let $X$ be a distribution on $\{0,1\}^n$. We say that $X$ is a $k$-source if for every $a \in \text{Supp}(X)$, $\Pr[X = a] \leq 2^{-k}$. Equivalently, $X$ is a $k$-source if $H_\infty(X) \geq k$ where $H_\infty(X) := \log \frac{1}{\max_a \Pr[X = a]}$.

Some examples:

1. If $X$ is the uniform distribution on $\{0,1\}^n$ then $X$ is an $n$-source, and we have $H_\infty(X) = n$.
2. If $X$ is 0 with probability $1/2$ and otherwise uniform on $\{0,1\}^n \setminus \{0^n\}$ then $H_\infty(X) = 1$.

Claim 13. Let $f : \{0,1\}^n \to \{0,1\}^s$ and let $X$ be the uniform distribution over $\{0,1\}^n$. Then for every $\epsilon > 0$,

$$\Pr_{X}[H_\infty(X|f(X)) \leq n - s - \log(1/\epsilon)] \leq \epsilon.$$ 

Intuitively, the above claim says that if $f$ compresses $n$ bits to $s$ bits, then with high probability knowing $f(X)$ reduces only about $s$ bits of entropy from $X$.

We would like to have a function $Ext : \{0,1\}^n \to \{0,1\}^m$ s.t. given a $k$-source $X$, $Ext(X)$ will be close to $U_m$ (we can think of $Ext$ as a “hash function”). Note that such a function does not exist: Assume that we only want one random bit (i.e. $m = 1$) from an $(n - 1)$-source, and let $Ext : \{0,1\}^n \to \{0,1\}$. Assume wlog that 0 has at least $2^{n-1}$ preimages in $Ext$, and define $X$ to be the random distribution over $Ext^{-1}(0)$. Then $X$ is an $(n - 1)$-source, but $Ext(X) \equiv 0$.

Hence we use a weaker definition:

Definition 14. A function $Ext : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called an $(k,\epsilon)$-extractor if for every $k$-source $X$ we have

$$|Ext(X,U_d) - U_m|_1 \leq \epsilon.$$ 

An intuitive way of thinking of it is that $U_d$ chooses at random a function $h$ from a family of “hash functions” $H$ and applies it on $X$ (i.e. $Ext(X,h) = h(x)$). We know that every function has a distribution $X$ for which it fails, but for a specific distribution most of the functions in $H$ are good.