FRT Tree Embedding

(building on Bartal's work)

1 Introduction

In many problems there is a metric in the input, e.g., Traveling Salesman Problem, Metric Labeling Problem and so on. Most of these problems are NP-hard for general metric spaces, and quite a few are polynomial-time solvable for some particular metric spaces. One metric space for which many problems become easy is the class of "tree metrics".

Definition 1. *A* metric space *is a pair* (V, d) *where* $d : V \times V \to R^{\geq 0}$*, so that*

- $d(i, i) = 0$ *for all* $i \in V$
- $d(i, j) = d(j, i)$ *for all* $i, j \in V$
- $d(i, j) \leq d(i, k) + d(k, j)$ *for all* $i, j, k \in V$ *(triangle inequality)*
- $d(i, j) > 0$ *for all* $i, j \in V$ *where* $i \neq j$

Definition 2. *A metric space* (V, d) *is a* tree metric *if the metric can be drawn as tree with weigths on the edges, such that the distance between any two points* i *and* j *is exactly equal to the weight of the path from* i *to* j*.*

In general, metric spaces are often represented as weighted graphs, where the distance between any two points i and j is then equal to the weight of the *shortest* path from i to j.

One method for getting an approximate solution to a problem is to approximate the metric space in the input by an easy metric that is similar to the original metric space, but for which the problem is easy. (In this lecture we will focus on tree metrics, but there are many other "easy" metric spaces.) We now have to make precise what we mean by "similar to the original metric".

Definition 3. *We say a metric space* (V, d) *is* embedded *in metric space* (V_2, d_2) *with* distortion $\gamma(\geq 1)$ *if there is a map* $f : V \to V_2$ *so that* $\frac{1}{\gamma} d_2(f(i), f(j)) \leq d(i, j) \leq \gamma d_2(f(i), f(j))$*.*

In general it is not possible to embed any metric in a tree metric with low distortion. Karp pioneered the idea of using a probability distribution on metric spaces to get around this problem. Bartal took this idea and created probabilistic tree embeddings, which until a few years ago where often referred to as "Bartal's trees". In 2003, Fakcharoenphol, Rao and Talwar gave a method to probabilistically embed any metric into tree metrics with (expected) distortion was $O(\log n)$ (where n is the number of points in the metric space). This distortion is (upto a constant factor) optimal, as Bartal also observed a lower bound on the distortion of probabilistic tree embeddings of $\Omega(\log n)$ for expander graphs.

We will now give the algorithm of FRT, and show that their method indeed gives an expected distortion of $O(\log n)$.

2 FRT Algorithm

Input: a general metric (V, d). We assume without loss of generality that $\min_{i \neq j} d(i, j) = 2$ (otherwise we scale the metric such that we have this property). Let M be smallest integer so that $2^M \geq \max_{i,j} d(i, j)$
Output: a tree metric (V_T, d_T) where $V \subset V_T$, so that $d(i, j) \leq d_T(i, j) \leq O(\log |V|) d(i, j)$.

1. Pick U uniformly at random in $(1, 2)$. Set $\ell \leftarrow M$. Pick a permutation π of V uniformly at random. Set $\mathcal{C} \leftarrow \{V\}$.
2. $C(1, \ell) \leftarrow \cap B(\pi_1, U2^\ell) := \{i \in V : d(i, \pi_1) \leq U2^\ell\}$,

   $C(2, \ell) \leftarrow B(\pi_2, U2^\ell) \setminus C(1, U2^\ell)$

   $C(3, \ell) \leftarrow B(\pi_3, U2^\ell) \setminus \big(C(1, U2^\ell) \cup C(2, U2^\ell)\big)$

   etcetera. In general,

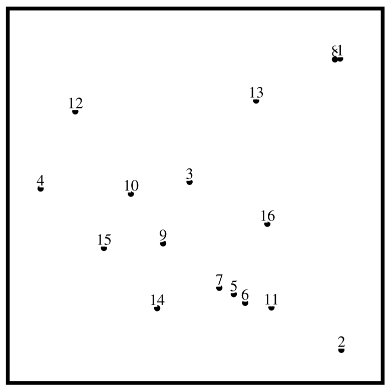
   $C(m, \ell) \leftarrow B(\pi_m, U2^\ell) \setminus \big(\bigcup_{k=1}^{m-1} C(k, \ell)\big)$.

   Now refine each set $S \in \mathcal{C}$ using these sets, by replacing S with $S \cap C(1, \ell)$, $S \cap C(2, \ell)$, etcetera.

   Add a point to V_T for all nonempty sets $S \in \mathcal{C}$, and add an edge from the set in the previous iteration that contains the set with weight $U2^{\ell+1}$.
3. $\ell \leftarrow \ell - 1$. If $\ell \geq 1$ then goto 2.

Note that π_m is not necessarily contained in $C(m, \ell)$, even if $C(m, \ell)$ is not empty.

On the next pages is an example of a run of the algorithm. The metric space that is being embedded is the Euclidean metric on the plane (i.e., the distance between two points is equal to what you would measure with a ruler) of this instance:



For ease of presentation π is chosen to be the identity permutation, and in each iteration the sets in $\mathcal{C}$ are visited in the order that they were created in the previous iteration.

Note that the sets are only refined in every iteration of the algorithm, and so the resulting metric will indeed be a tree metric.

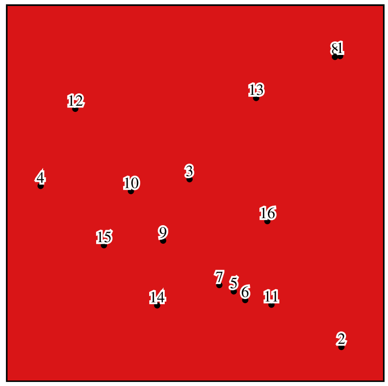
3 Analysis

We define the following random variables:

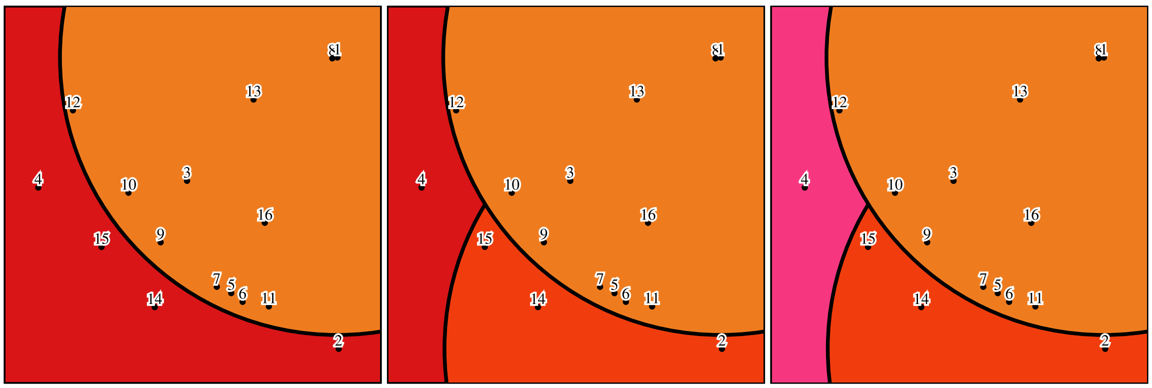
- $L(i, j)$ =value of ℓ in the first iteration (i.e., maximum ℓ) in which i and j are in different sets,
- $S(i, \ell)$ =set C in which i is contained in iteration ℓ,
- $K(i, \ell)$ =center of the set that contains i in iteration ℓ (so $B(K(i, \ell)) \supset S(i, \ell)$).

Claim 1. *$d_T(i, j) \geq d(i, j)$ for all $i, j \in V$.*

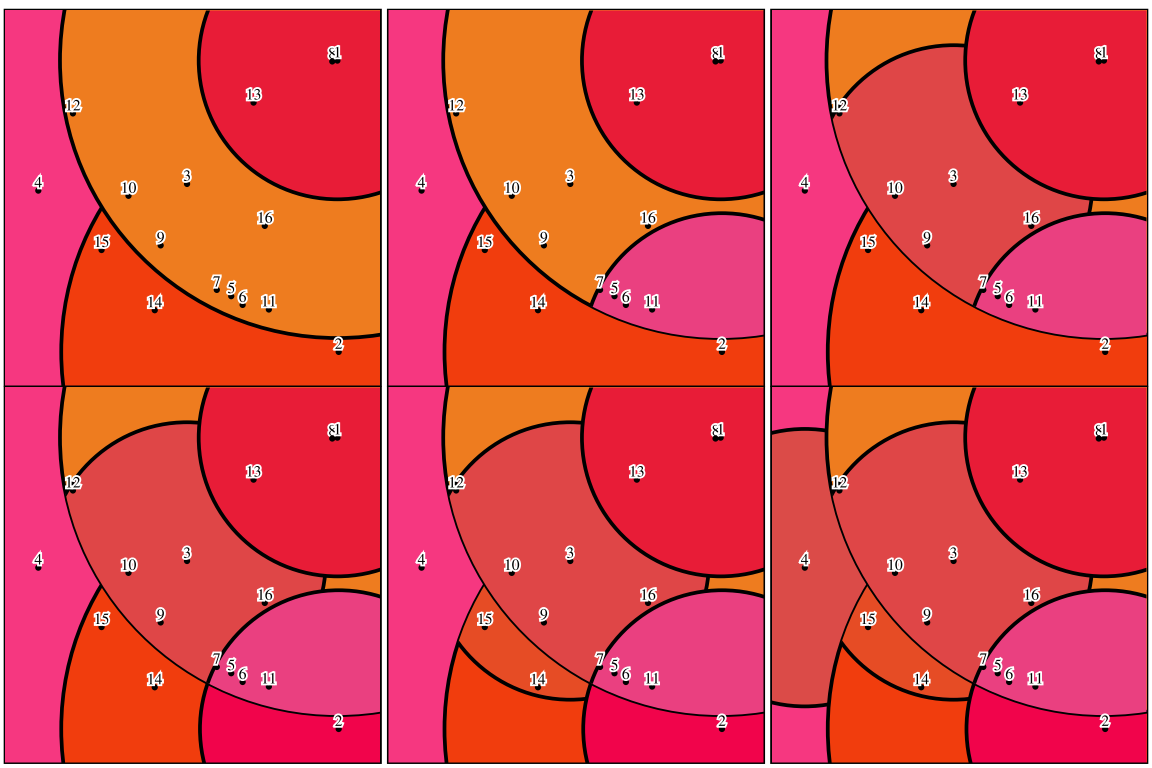
Proof. $d_T(i, j) \geq 2 \times U2^{L(i,j)+1} = U2^{L(i,j)+2}$ because the path in the resulting tree from i to j will contain two edges that are adjacent to the set $S(i, L(i, j) + 1) = S(j, L(i, j) + 1)$. Further, the triangle inequality gives us $d(i, j) \leq d(i, K(i, L(i, j)+1)) + d(j, K(i, L(i, j)+1)) = d(i, K(i, L(i, j)+1)) + d(j, K(j, L(i, j)+1)) \leq U2^{L(i,j)+1} + U2^{L(i,j)+1} \leq U2^{L(i,j)+2}$. So indeed we have $d_T(i, j) \geq d(i, j)$ for all $i, j \in V$, independent of the random choices of the algorithm. □



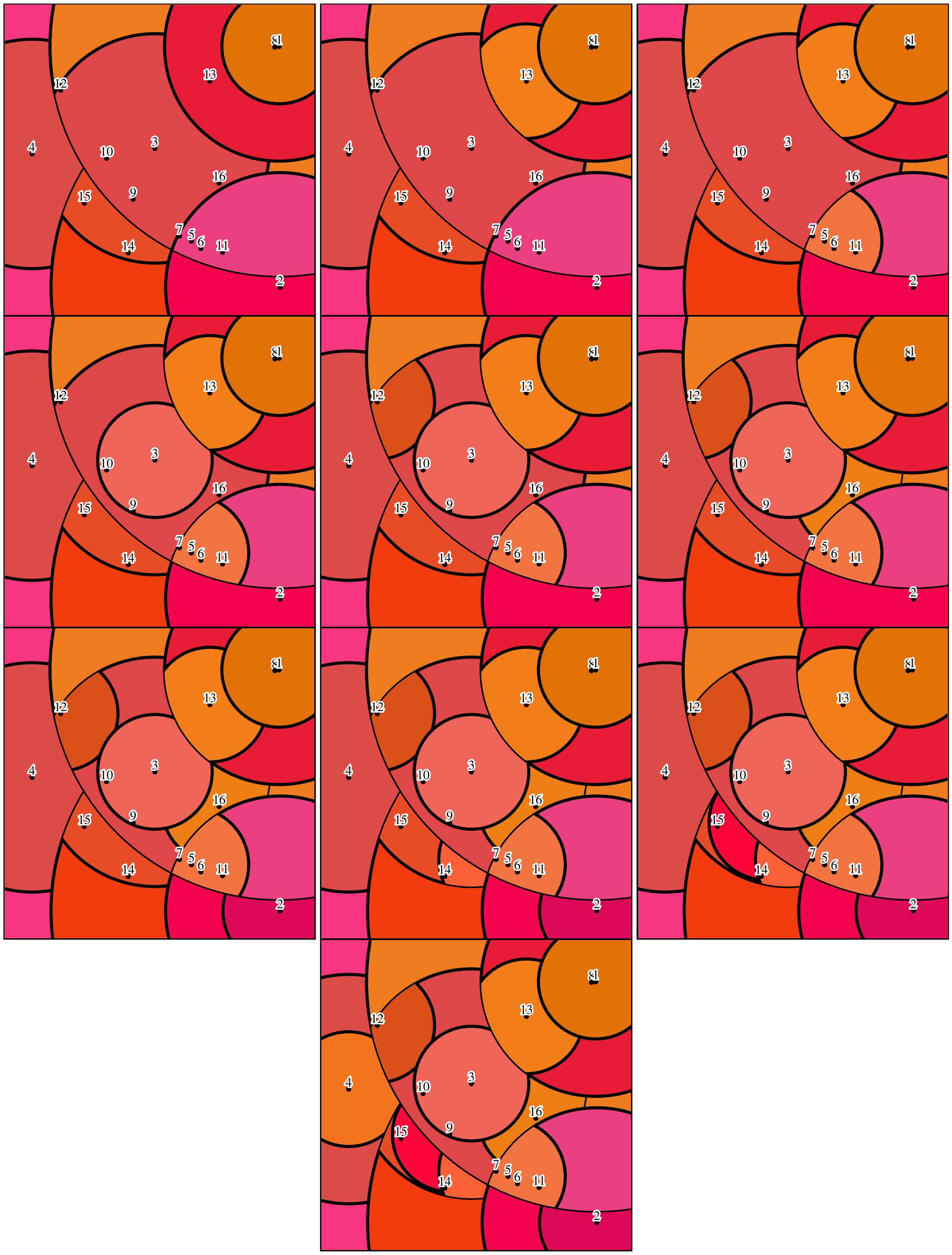
end of iteration $\ell = M$



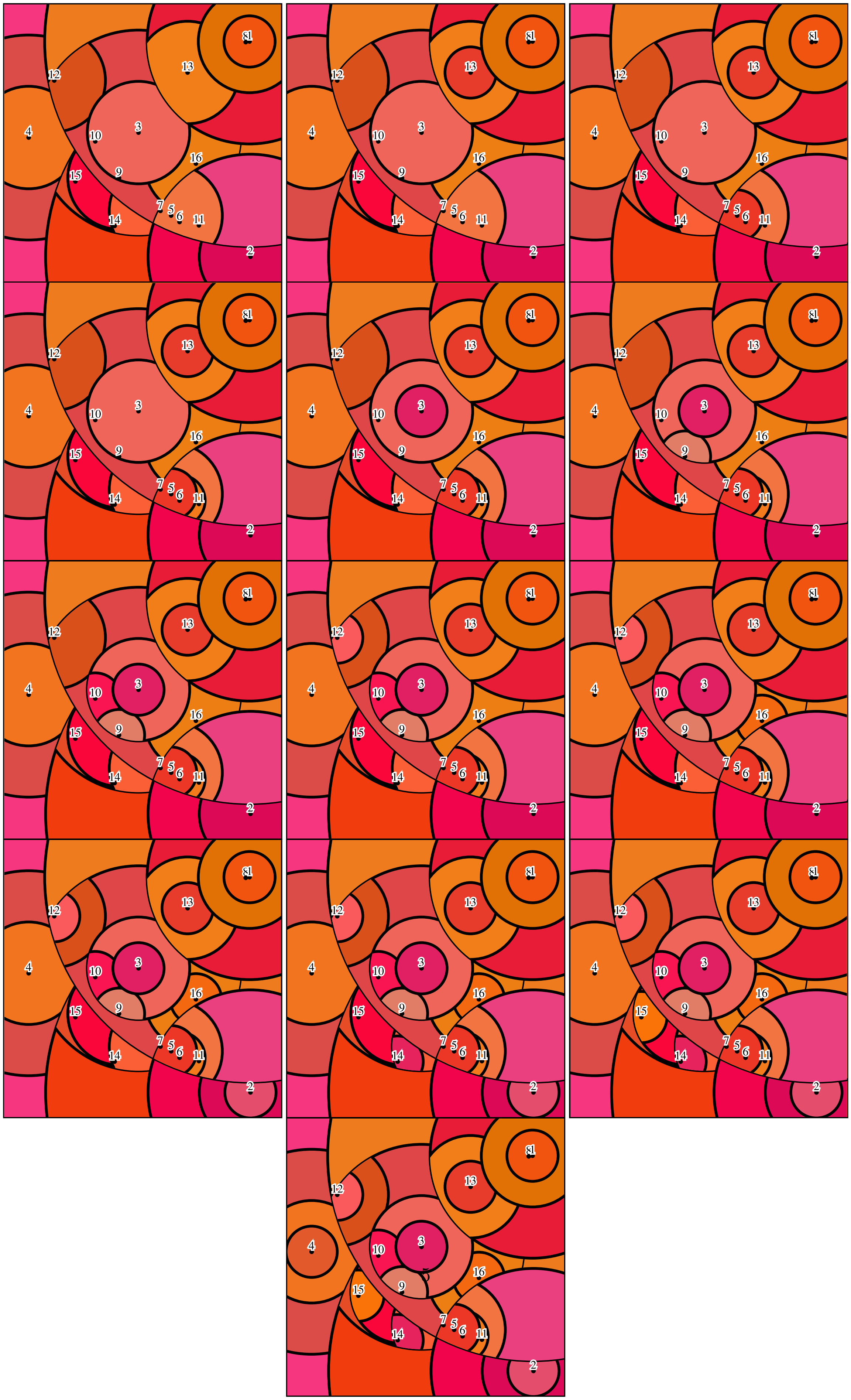
end of iteration $\ell = M - 1$



end of iteration $\ell = M - 2$



end of iteration $\ell = M - 3$



end of iteration $\ell = M - 4$

We can also upper bound $d_T(i,j)$ in terms of $L(i,j)$:

$$d_T(i,j) = 2(U2^{L(i,j)+1} + U2^{L(i,j)} + U2^{L(i,j)-1} + \ldots + U2^2),$$

as $U2^{L(i,j)+1} + U2^{L(i,j)} + U2^{L(i,j)-1} + \ldots + U2^2$ is the length of the path from both i and j to the node that represents $S(i, L(i,j)+1)$ that contains both i and j. So we have

$$d_T(i,j) \leq 2 \sum_{m=0}^{\infty} U2^{L(i,j)+1-m} = 8U2^{L(i,j)} \leq 16 \ 2^{L(i,j)}.$$

Denote by $D(i,j) =$ first according to π of $K(i, L(i,j))$ and $K(j, L(i,j))$; the center of the first set that is created that contains only one of i and j in the first iteration where i and j are not in the same set.

We will now show $E(d_T(i,j)) \leq O(\log |V|) d(i,j)$ for all $i, j \in V$, by conditioning on $D(i,j)$ first and showing that conditioned on $D(i,j)$, there are at most four possible values of $L(i,j)$. We obtain upper bounds on the probabilities for these events which will then give the result.

Suppose $D(i,j) = K(i, L(i,j))$, and let's call it this node c. Then $d(c,j) \geq U2^{L(i,j)}$. Also, by the definition of $L(i,j)$ (first iteration that have i and j separated) and the triangle inequality, we have $d(c,j) \leq d(c, K(c, L(i,j)+1)) + d(K(c, L(i,j)+1), j) \leq U2^{L(i,j)+1} + d(K(j, L(i,j)+1), j) \leq U2^{L(i,j)+2}$, because the sets that the algorithm creates are refinements so that $K(c, L(i,j)+1) = K(i, L(i,j)+1) = K(j, L(i,j)+1)$. So

$$2^{L(i,j)} < d(c,j) < 2^{L(i,j)+3}.$$

So for any given $D(i,j)$, i and j, there are at most two different possibilities for the $L(i,j)$. The same reasoning holds for $D(i,j) = K(j, L(i,j))$, which gives a grand total of at most four different values for $L(i,j)$ with positive probabilities, given $D(i,j)$.

Also, given $D(i,j) = c$, we can bound the probability that $L(i,j) = \ell$ (i and j are separated in iteration ℓ) by $|d(c,i) - d(c,j)|/2^{\ell} \leq d(i,j)/2^{\ell}$ by the triangle inequality ($d(c,j) \leq d(c,i) + d(i,j)$), because U is uniformly distributed in $(1,2)$.

So we have that

$$\begin{aligned} E(d_T(i,j) | D(i,j) = c) &\leq \sum_{\ell} E(d_T(i,j) | D(i,j) = c, L(i,j) = \ell) P(L(i,j) = \ell | D(i,j) = c) \\ &\leq \sum_{\ell : P(L(i,j)=\ell | D(i,j)=c) > 0} E(d_T(i,j) | D(i,j) = c, L(i,j) = \ell) d(i,j)/2^{\ell} \\ &\leq 4 \times 16 \ 2^{\ell} d(i,j)/2^{\ell} \\ &= 64 \ d(i,j). \end{aligned}$$

Finally we have to bound the probability $P(D(i,j) = c)$ for each node c. Imagine that the nodes are ordered in ascending $\min\{d(i,c), d(j,c)\}$ for a given i and j. The event $D(i,j) = c$ implies that there is no other node that is closer to either i or j in the permutation. So we can bound this probability by $1/\text{rank}(c)$. As we sum over all nodes we get all possible rankings. Therefore we get

$$E(d_T(i,j)) = \sum_{c} E d_T(i, j | D(i,j) = c) P(D(i,j) = c) \leq \sum_{r=1}^{n} 64 \ d(i,j) \frac{1}{r} = O(\log(|V|)) d(i,j).$$