

CS381 Assignment 6

Due November 29 2023 12:00 PM AoE

Most of the questions of this assignment have to do with the *Online Minimum Weight Metric Matching Problem*, ie the parking problem from class. In this problem, the algorithm oversees n parking spots, which are used to handle incoming visitors. Sequentially, visitors arrive and must be immediately matched to an available parking spot. The goal is to match these spots in such a way that visitors have to walk as little as possible to their intended location from where they park. The cost of an assignment is the sum of distances from each driver's assigned parking spot to their intended location. We rank the performance of algorithms by analyzing their competitive ratio, where the competitive ratio of an algorithm A is given by

$$\max_{\text{instances } I} \frac{|A(I)|}{|\text{OPT}(I)|},$$

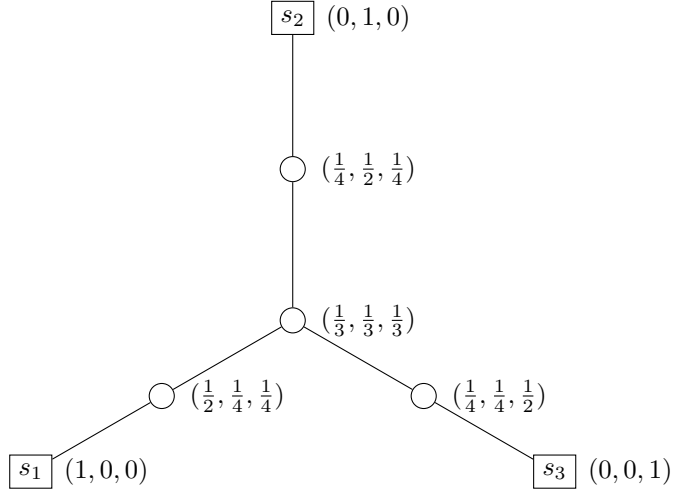
where $|A(I)|$ denotes the cost of A 's output on I and $|\text{OPT}(I)|$ denotes the cost of the optimal solution on I .

In the dynamic pricing version of the game, instead of assigning drivers to parking spots, the algorithm can instead place prices over the servers. In this model, we assume incoming drivers choose spots that minimize the sum of the price of the spot and the distance from the spot to their intended location. Sequentially, after each spot is taken, prices may be updated as desired by the algorithm.

1. Given the following randomized algorithm on the below tree, using the algorithm defined in class to determine a randomized pricing scheme that mimics the given algorithm. Note that this involves the following steps:
 - i) Decide on the order you are folding vertices in on each other
 - ii) When the graph is down to a singleton vertex, create a valid partitioning for that vertex.
 - iii) Vertex by vertex, unfold the tree and extend the distribution of partitioning schemes using the technique designed in class
 - iv) When the graph is fully unfolded, determine prices that mimic each individual partitioning scheme in the fully extended distribution.

This is a nontrivial amount of work which I'll expect fully shown for each step. While I'll allow uploads of **neat and clear** drawings, I'll give extra credit if you do the whole thing in TikZ (which I'll use below to give the original graph and algorithm to mimic)¹.

¹Also, check out TikZiT, a handy program for drawing what you want that automatically generates the code to match what you drew.



This is the same instance from class, but just as a reminder: the three available parking spots are s_1 , s_2 , and s_3 . Beside each vertex v is a vector of the form (x_1, x_2, x_3) where x_i is the probability the algorithm would assign a driver arriving at v to s_i .

2. Recall the HARMONIC algorithm for class. In class, we defined the HARMONIC algorithm to match on the line according to the following rule:

- (a) If a driver arrives at an available parking spot, assign it to that spot.
- (b) If a driver arrives to the left or right of all spots, then assign it to the closest spot.
- (c) If a driver arrives between two spots s_L and s_R , assign it to spot s_L with probability

$$\frac{\frac{1}{s_L}}{\frac{1}{s_L} + \frac{1}{s_R}} = \frac{s_R}{s_L + s_R}^2$$

Imagine we take the HARMONIC algorithm and extend it to trees. Specifically, imagine a driver that arrives between a bunch of available parking spots, as in figure 1. In figure 1, if a driver were to arrive at v , then the closest parking spots would be the red ones: every other parking spot would involve travelling through one of the red ones. So the red ones are the available parking spots that are reachable from v without travelling through another available parking spot: we'll call the collection of these parking spots S_v . Given a driver that arrives at v , the formal generalization of HARMONIC is to assign v to a spot s in S_v with probability

$$\frac{\frac{1}{d(v,s)}}{\sum_{t \text{ in } S_v} \frac{1}{d(v,t)}},$$

where $d(x, y)$ denotes the distance between x and y .

²While the expression on the right is what we used in class, the expression on the left is technically the definition that more aligns with HARMONIC in general. Those with a background in physics or statistics might recognize that it corresponds to the "Harmonic mean" of two values.

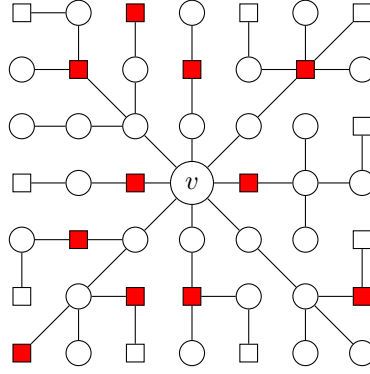
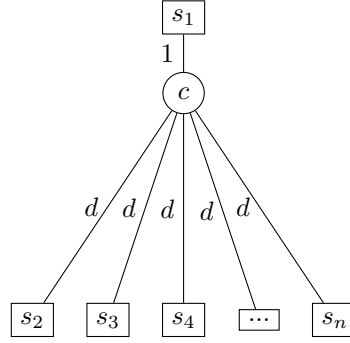


Figure 1: In this figure: the circles are vertices and the squares (regardless of color) are parking spots. The red spots are the parking spots which are reachable from v without passing through another parking spot.



Consider the graph above, along with the following sequence of drivers: $c, s_2, s_3, \dots, s_n$.

- i) What is the expected cost of HARMONIC on the very first driver as a function of d ?
- ii) What is the optimal solution on this input?
- iii) Use your answers to the above two questions to convince me that HARMONIC is not $O(\log \Delta)$ competitive on trees.