Code Jam to I/O for Women 2018 - Code Jam to I/O for Women

Analysis: Burger Optimization

Burger Optimization: Analysis

Small dataset

This dataset can be solved using complete search. Since **K** ≤ 8, there are only at most 8! = 40320 different orderings of ingredients.

Therefore, we can try all **K**! different orderings of ingredients and find which ordering of ingredients has the minimum sum of squared differences between each ingredient's optimal and actual distance-to-bun values. Since we can calculate the sum of the squared differences between each ingredient's optimal and actual distance-to-bun values for one particular ordering of ingredients in O(**K**) time, this solution runs in O(**K**! × **K**) time.

Large dataset

We need an observation to solve this dataset. Suppose we have two ingredients A and B to be placed at two positions X and Y. Let ingredients A and B have optimal distance-to-bun values of D_A and D_B, respectively, and let positions X and Y have actual distance-to-bun values of D_X and D_Y, respectively. Suppose $D_A \le D_B$ and $D_X \le D_Y$. We claim that it is no less optimal to put ingredient A at X and to put ingredient B at Y.

Why? Let us assume that $D_B = D_A + K_1$ and $D_Y = D_X + K_2$ for some non-negative integers K_1 and K_2. Therefore, the cost of putting ingredient A at X and putting ingredient B at Y is $(D_A - D_X)^2 + (D_B - D_Y)^2$, and the cost of putting ingredient A at Y and putting ingredient B at X is $(D_A - D_Y)^2 + (D_B - D_X)^2$.

We can derive that

$$[(D_A - D_X)^2] + [(D_B - D_Y)^2]$$
$$= [D_A^2 - 2D_AD_X + D_X^2] + [D_B^2 - 2D_BD_Y + D_Y^2]$$
$$= [D_A^2 - 2D_AD_X + D_X^2] + [D_B^2 - 2(D_A + K_1)(D_X + K_2) + D_Y^2]$$
$$= [D_A^2 - 2D_AD_X + D_X^2] + [D_B^2 - 2D_AD_X - 2D_AK_2 - 2K_1D_X - 2K_1K_2 + D_Y^2]$$
$$= [D_A^2 - 2D_AD_X - 2D_AK_2 + D_Y^2] + [D_B^2 - 2D_AD_X - 2K_1D_X + D_X^2] - [2K_1K_2]$$
$$= [D_A^2 - 2D_A(D_X + K_2) + D_Y^2] + [D_B^2 - 2(D_A + K_1)D_X + D_X^2] - [2K_1K_2]$$
$$= [D_A^2 - 2D_AD_Y + D_Y^2] + [D_B^2 - 2D_BD_X + D_X^2] - [2K_1K_2]$$
$$= [(D_A - D_Y)^2] + [(D_B - D_X)^2] - [2K_1K_2]$$

Therefore, $(D_A - D_X)^2 + (D_B - D_Y)^2 \le (D_A - D_Y)^2 + (D_B - D_X)^2$, since $2K_1K_2$ is a non-negative integer. Therefore, we have proved that it is no less optimal to put ingredient A at X and to put ingredient B at Y. In other words, we want to put an ingredient which has a smaller optimal distance-to-bun value in a position which has a smaller actual distance-to-bun value.

Based on the above observation, the solution to this problem becomes simple. Let D' be the list of actual distance-to-bun values for the ingredients sorted in non-decreasing order (e.g. if **K** is 7, then D' = {0, 0, 1, 1, 2, 2, 3}), and let D be the list of optimal distance-to-bun values for the ingredients sorted in non-decreasing order. Using the observation above, it is optimal to put the i-th ingredient on the list of D to a position which has an actual distance-to-bun value of D'[i]. Therefore, the answer is simply Σ $(D'[i] - D[i])^2$.

We can sort the ingredients in O(**K** log (**K**)) time and calculate the sum of the squared differences between each ingredient's optimal and actual distance-to-bun values in O(**K**) time. Therefore, this solution runs in O(**K** log (**K**)) time.

(For a similar approach, you might enjoy reading about the rearrangement inequality).