Homework #4

(Due: Dec 10)

Task 1. [80 Points] Holes in the Wall

The *Hole in the Wall*[1] restaurant in our neighborhood has a wall full of nail holes. All its walls have holes, but that particular wall looks so ugly that the owner of the restaurant has decided to cover the holes using some form of inexpensive artwork. The artwork will consist only of a number of disks of the same size made from some inexpensive material. The disks will be placed carefully on the wall to cover all holes. Each disk will have a hole at its center which can be used to nail or screw the disk to the wall. Since the owner does not want to create additional holes in the wall to install the artwork, each disk must be nailed/screwed to the wall using one of the existing nail holes. The owner does not want to buy too many disks, and also wants to make sure that the total area of all disks used in the artwork is as small is possible because the cost of a disk is proportional to its area. Figure 1 shows a sample artwork.

Suppose there are $n > 1$ holes in the wall, and the artwork must not use more than K disks, where K is a positive integer constant specified by the owner of the restaurant. The wall is rectangular, and the location of a hole is given by a pair of real numbers (x, y), where x is the distance (in inches) of the hole from the left end of the wall and y is the distance (also in inches) from the bottom end (i.e., floor). Now given the value of K and the locations of all holes in the wall, how many disks of what common size (i.e., radius) one must use in order to cover all those holes so that the total area of all those disks is minimized?

Consider the algorithm Create-Artwork given in Figure 2 for computing the centers of at most k disks of a common radius that can be used to cover all holes in the given wall. By $d(\ g,\ h\ )$ we denote the Euclidean distance between two holes g and h. We define $d(\ g,\ H\ ) = \min_{h \in H}\{\ d(\ g,\ h\ )\ \}$, where H is a nonempty set of holes, and $d(\ g,\ H\ ) = \infty$ when H is an empty set.

Now answer the following questions.

(*a*) [**50 Points**] Prove that the total area of all k_{sol} disks in C_{sol} returned by Create-Artwork is within a factor of 4 of an optimal solution that also uses exactly k_{sol} disks.

(*b*) [**10 Points**] Show that the total area of all k_{sol} disks in C_{sol} returned by Create-Artwork is within a factor of 4 of an optimal solution that uses at most K disks.

(*c*) [**20 Points**] Show that the solution returned by Create-Artwork is still within a factor 4 of optimal even if the owner does not insist on installing the artwork using only existing nail holes.

[1] Hole-in-the-Wall: a small place (such as a bar or restaurant) that is not fancy or expensive (according to Merriam-Webster dictionary)

Figure 1: Sample artwork hiding holes in the wall.

CREATE-ARTWORK($K,\ h_1,\ h_2,\ \ldots, h_n$)

Input: Maximum number K of disks to use, and coordinates of $n > 1$ nail holes $h_1, h_2, \ldots, h_n$.
Output: A subset C_{sol} of $k_{\text{sol}} \leq K$ holes to use as disk centers, and the common radius r_{sol} of all disks.

1. $C_{\text{sol}} \leftarrow \emptyset,\ k_{\text{sol}} \leftarrow \infty,\ r_{\text{sol}} \leftarrow \infty$
2. $C \leftarrow \emptyset$
3. ***for*** $k \leftarrow 1$ ***to*** K ***do***
4. find the hole h_i with the maximum distance $d(\ h_i,\ C\ )$ from C (break ties arbitrarily)
5. $C \leftarrow C \cup \{\ h_i\ \}$
6. $r \leftarrow \max_{1 \leq j \leq n} d(\ h_j,\ C\ )$
7. ***if*** $k\left(\pi r^2\right) < k_{\text{sol}}\left(\pi r_{\text{sol}}^2\right)$ ***then***
8. $k_{\text{sol}} \leftarrow k$
9. $C_{\text{sol}} \leftarrow C$
10. $r_{\text{sol}} \leftarrow r$
11. ***return*** $\langle\ k_{\text{sol}}, C_{\text{sol}}, r_{\text{sol}}\ \rangle$

Figure 2: Finding disk coordinates for the artwork.

LOOPS(X, F)

Input: Inputs are an $n \times n$ matrix $X[0:(n-1), 0:(n-1)]$ of Boolean values, and an $(h+1) \times n$ matrix $F[0:h, 0:(n-1)]$ of real values.
Output: Updated matrix $F[0:h, 0:(n-1)]$ based on values in $X[0:(n-1), 0:(n-1)]$.

1. ***for*** $u \leftarrow 0$ ***to*** $n-1$ ***do***
2. ***for*** $i \leftarrow 0$ ***to*** $h-1$ ***do***
3. ***for*** $v \leftarrow u+1$ ***to*** $n-1$ ***do***
4. ***if*** $X[u,v] =$ TRUE ***then*** $F[i+1,v] \leftarrow F[i+1,v] + F[i,u]$

Figure 3: A looping code.

Task 2. [100 Points] Work and Span

You are given two sequential algorithms for solving the same problem in Figures 3 and 4. Your task is to parallelize them and analyze their work, span and parallelism.

(a) [**20 Points**] Explain how you would correctly parallelize the algorithm in Figure 3 using only parallel for loops. Compute its work, span and parallelism on as functions of n and h.

(b) [**80 Points**] Explain how you would correctly parallelize FUNC-A and FUNC-B in Figure 4 using parallel for loops, spawns and syncs. Compute their work, span and parallelism assuming both n and h to be powers of 2.

FUNC-A($X,\ F$)

(Inputs are an $n \times n$ matrix $X[0:(n-1), 0:(n-1)]$ of Boolean values, and an $(h+1) \times n$ matrix $F[0:h, 0:(n-1)]$ of real values. Both n and h are assumed to be powers of 2.)

1. ***if*** $n \leq c$ ***and*** $h \leq c$ ***then***
2. LOOPS($X,\ F$)
3. ***else***
4. ***if*** $h > n$ ***then***
5. FUNC-A($X,\ F_t$) $\left\{F_t \equiv F\left[0:\frac{h}{2}, 0:(n-1)\right]\right\}$
6. FUNC-A($X,\ F_b$) $\left\{F_b \equiv F\left[\frac{h}{2}:h, 0:(n-1)\right]\right\}$
7. ***else***
8. FUNC-A($X_{11},\ F_l$) $\left\{X_{11} \equiv X\left[0:\left(\frac{n}{2}-1\right), 0:\left(\frac{n}{2}-1\right)\right],\ F_l \equiv F\left[0:h, 0:\left(\frac{n}{2}-1\right)\right]\right\}$
9. FUNC-B($X_{12},\ F_l,\ F_r$) $\left\{X_{12} \equiv X\left[0:\left(\frac{n}{2}-1\right), \frac{n}{2}:(n-1)\right],\ F_r \equiv F\left[0:h, \frac{n}{2}:(n-1)\right]\right\}$
10. FUNC-A($X_{22},\ F_r$) $\left\{X_{22} \equiv X\left[\frac{n}{2}:(n-1), \frac{n}{2}:(n-1)\right]\right\}$

FUNC-B($X,\ F^{(s)},\ F^{(d)}$)

(Inputs are an $n \times n$ matrix $X[0:(n-1), 0:(n-1)]$ of Boolean values, and an $(h+1) \times n$ matrices $F^{(s)}[0:h, 0:(n-1)]$ and $F^{(d)}[0:h, 0:(n-1)]$ of real values. Both n and h are assumed to be powers of 2.)

1. ***if*** $n \leq c$ ***and*** $h \leq c$ ***then***
2. ***for*** $i \leftarrow 0$ ***to*** $h-1$ ***do***
3. ***for*** $u \leftarrow 0$ ***to*** $n-1$ ***do***
4. ***for*** $v \leftarrow 0$ ***to*** $n-1$ ***do***
5. ***if*** $X[u,v] =$ TRUE ***then*** $F^{(d)}[i+1,v] \leftarrow F^{(d)}[i+1,v] + F^{(s)}[i,u]$
6. ***else***
7. ***if*** $h > n$ ***then***
8. FUNC-B$\left(\ X,\ F_t^{(s)},\ F_t^{(d)}\ \right)$ $\left\{F_t^{(z)} \equiv F^{(z)}\left[0:\frac{h}{2}, 0:(n-1)\right]\ \textit{for}\ z \in \{\ s,\ t\ \}\right\}$
9. FUNC-B$\left(\ X,\ F_b^{(s)},\ F_b^{(d)}\ \right)$ $\left\{F_b^{(z)} \equiv F^{(z)}\left[\frac{h}{2}:h, 0:(n-1)\right]\ \textit{for}\ z \in \{\ s,\ t\ \}\right\}$
10. ***else***
11. FUNC-B$\left(\ X_{11},\ F_l^{(s)},\ F_l^{(d)}\ \right)$ $\left\{X_{11} \equiv X\left[0:\left(\frac{n}{2}-1\right), 0:\left(\frac{n}{2}-1\right)\right],\ F_l^{(z)} \equiv F^{(z)}\left[0:h, 0:\left(\frac{n}{2}-1\right)\right],\ z \in \{s,t\}\right\}$
12. FUNC-B$\left(\ X_{12},\ F_l^{(s)},\ F_r^{(d)}\ \right)$ $\left\{X_{12} \equiv X\left[0:\left(\frac{n}{2}-1\right), \frac{n}{2}:(n-1)\right],\ F_r^{(d)} \equiv F^{(d)}\left[0:h, \frac{n}{2}:(n-1)\right]\right\}$
13. FUNC-B$\left(\ X_{21},\ F_r^{(s)},\ F_l^{(d)}\ \right)$ $\left\{X_{21} \equiv X\left[\frac{n}{2}:(n-1), 0:\left(\frac{n}{2}-1\right)\right],\ F_r^{(s)} \equiv F^{(s)}\left[0:h, \frac{n}{2}:(n-1)\right]\right\}$
14. FUNC-B$\left(\ X_{22},\ F_r^{(s)},\ F_r^{(d)}\ \right)$ $\left\{X_{22} \equiv X\left[\frac{n}{2}:(n-1), \frac{n}{2}:(n-1)\right]\right\}$

Figure 4: Recursive divide-and-conquer algorithm for solving the problem solved by the looping code in Figure 3. Assuming that both n and h are powers of 2, function calls A($X,\ F$) and LOOPS($X,\ F$) produce the same results, where $X[0:(n-1), 0:(n-1)]$ is an $n \times n$ matrix of Boolean values, $F[0:h, 0:(n-1)]$ is an $(h+1) \times n$ matrix of real values, and c is a global positive integer constant.