Advanced Algorithms: Homework 8

Due on April 10, 2024 at 11:59pm EST

Professor Dana Randall Spring 2024

As stated in the syllabus, unauthorized use of previous semester course materials is strictly prohibited in this course. EC problems are all-or-nothing. Furthermore, you must complete EC problems alone (i.e. without external resources or collaborators)

Sarthak Mohanty

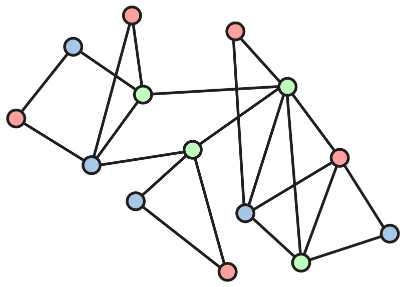
Exercise 1

(Erickson) Let $G = (V, E)$ be an undirected graph, each of whose vertices is colored either red, green, or blue. An edge in G is *boring* if its endpoints have the same color, and *interesting* if its endpoints have different colors. The *most interesting 3-coloring* is the 3-coloring with the maximum number of interesting edges, or equivalently, with the fewest boring edges. Computing the most interesting 3-coloring is NP-hard, because the standard 3-coloring problem is a special case.

(a) Let $\mathsf{wow}(G)$ denote the number of interesting edges in the most interesting 3-coloring of G. Suppose we assign each vertex in G a random color in $\{\text{red}, \text{green}, \text{blue}\}$. Prove that the expected number of interesting edges is at least $\frac{2}{3}\mathsf{wow}(G)$.

(b) Prove that with high probability, the expected number of interesting edges is at least $\frac{1}{2}\mathsf{wow}(G)$. (*Hint*: Use Chebyshev's inequality. But wait... How do we know that we **can** use Chebyshev's inequality?)

(c) Let $\mathsf{snooze}(G)$ denote the number of boring edges in the most interesting 3-coloring of a graph G. Prove that it is NP-hard to approximate $\mathsf{snooze}(G)$ within a factor of 2024^{2024}.


(a) A somewhat boring 3-coloring.

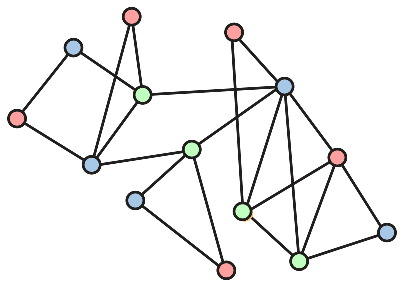

(b) The most interesting 3-coloring.[1]

Figure 1: Here $\mathsf{wow}(G) = 21$ and $\mathsf{snooze}(G) = 1$.

Exercise 2[2]

Define a *triangle matching* in an undirected graph G as a collection of vertex-disjoint triangles (cycles of length 3) in G. A triangle matching is *maximal* if it is not a proper subgraph of a larger triangle matching in the same graph.

(a) Let M and M' be two arbitrary maximal triangle matchings in the same underlying graph G. Prove that $|M| \le 3 \cdot |M'|$.

(b) Finding the *largest* triangle matching in a given graph is NP-hard. Describe and analyze a fast 3-approximation algorithm for this problem.

(c) Finding the *smallest maximal* triangle matching in a given graph is NP-hard. Describe and analyze a fast 3-approximation algorithm for this problem.

Exercise 3

Find a polynomial-time 4/3-approximation to instances of metric-TSP where distances are either 1 or 2.

Hint: The 2-matching problem (i.e. find a minimum weight subset of edges M such that each node is adjacent to exactly 2 edges in M) can be solved in polynomial time.

[1]Unless I missed something and this graph is actually 3-colorable.
[2]This could've been an exam problem!

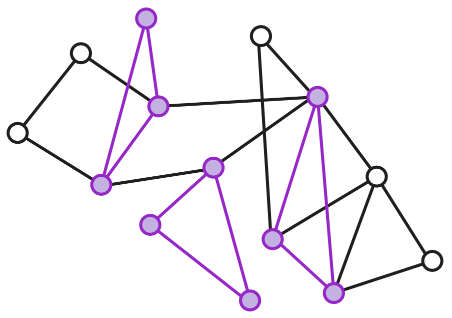


Figure 2: A maximal (but not maximum!) triangle matching

Exercise 4 (EC)

You (hopefully) have heard of the **Maximum Independent Set (MIS)** problem from previous courses, where the goal was to find an maximum-cardinality independent set V' of vertices in some graph G. (If not, refer to the Appendix for more information)

In the **Maximum Independent Set on Unit Disk Graph ($\mathsf{MIS_{UDG}}$)** problem, we are given a set S of unit-diameter disks in the plane. The goal is to find a maximum-cardinality subset $S' \subseteq S$ of disks, such that no two disks in S' overlap. (We say that disks that only touch on the boundary do not overlap). As with MIS, this problem is also NP-Complete.

(a) *(8 pts)* Consider the following greedy algorithm: start with $S' = \emptyset$ and process the disks in S one-by-one, in any order. For each such disk C, if $S' \cup \{C\}$ is a feasible solution, then add C to S'. What is the approximation factor of this algorithm? (Give the best upper bound you can.)

(b) *(4 pts)* Let $0 < \varepsilon < 1$ be some constant. Assume that we are given a grid whose lines are spaced $1/\varepsilon$ units apart. Assume further that no disk in the input set S intersects the grid lines. Give a polynomial time algorithm for solving the problem exactly in this case.

(c) *(8 pts)* Derive and analyze a PTAS for the $\mathsf{MIS_{UDG}}$ problem.

Hint: Consider a grid with *random offset*: Take a grid of lines spaced $\frac{1}{2}$ apart, such that the origin is at the intersection of a horizontal and vertical grid line. Pick a shift/offset L uniformly at random from $[0, \frac{1}{2})$, and shift the grid vertically and horizontally by an distance L. (Equivalently, consider the grid of spacing $\frac{1}{2}$ such that the point (L, L) is at the intersection of two grid lines.) What is the probability that a disk is intersected by a grid line?

This problem has applications in VLSI chip manufacturing (Again, refer to the Appendix for more information.)

Appendix

Maximum Independent Set

An **independent set** in a graph is a subset of the vertices with no edges between them. Finding the largest independent set in an arbitrary graph is famously NP-hard. But in some special classes of graphs, we can find largest independent sets quickly. In particular, when the input graph is a tree with n vertices, we can actually compute the largest independent set in $O(n)$ time using **dynamic programming** (Hint!), as follows:

For any node v in T , let $MIS(v)$ denote the size of the largest independent set in the subtree rooted at v. Any independent set in this subtree that excludes v itself is the union of independent sets in the subtrees rooted at the children of v. On the other hand, any independent set that includes v necessarily excludes all of v's children, and therefore includes independent sets in the subtrees rooted at v's grandchildren. Thus, the table MIS obeys the following recurrence, which we can compute using a post-order traversal on the tree (here the set $C(v)$ contains the children of v):

$$MIS(v) = \max\left\{ \sum_{w \in C(v)} MIS(w),\ 1 + \sum_{w \in C(v)} \sum_{x \in C(w)} MIS(x) \right\}$$

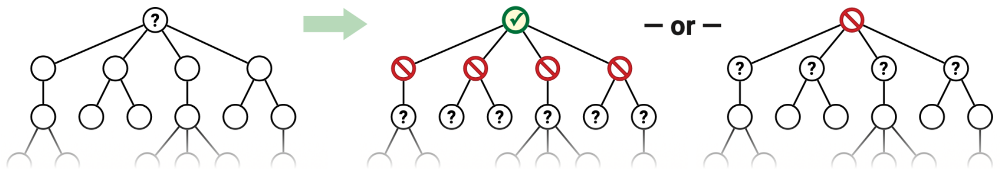


Figure 3: Computing maximum independent set in a tree.

Configurability in VLSI Chip Manufacturing[3]

We begin with a motivating question: Why is it so hard to create large chips? One reason is because of precision. It is very hard to avoid defects in the creation of these chips. Thus, a construct called **configurability** was introduced to VLSI chip design.

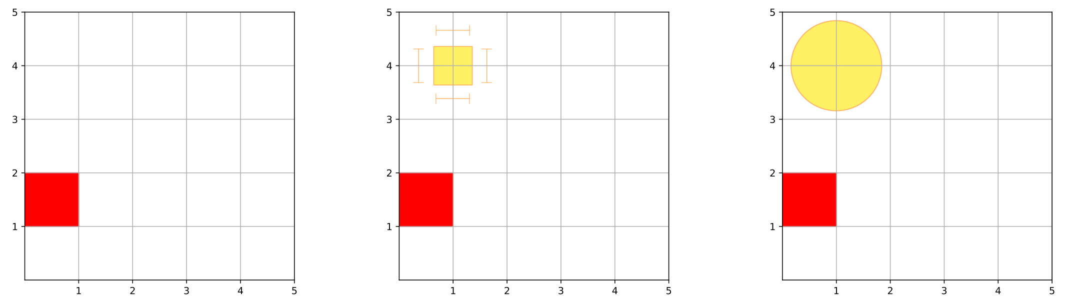
In manufacturing, chips are regions of a larger unit called a "wafer." Ordinarily after fabrication, a wafer is tested and faulty chips are marked; then the wafer is diced, the faulty chips are discarded, and the functional chips are packaged. However, since the entire chip is being manufactured at once, this is wasteful because even if only a small portion of the chip is defective, it must be entirely discarded.

To use configurability, this process is changed somewhat. Special circuitry is added that enables adjacent chips to be connected. Then, after the wafer is tested, adjacent functional chips are connected into a group and the entire group is used as a single enlarged chip. In this way, even if a smaller chips becomes defective, the adjacent chips can be rewired so that they can still be used in the larger chip.

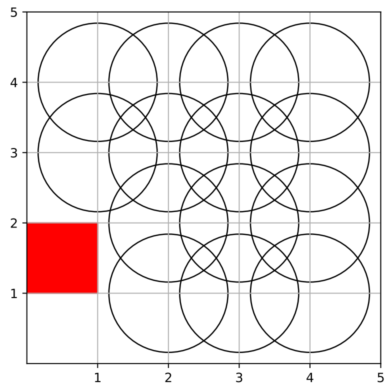
A natural problem that arises from this setup is packing a maximal number of working chips on a wafer. As an example, consider the Cartesian grid representing a wafer, shown below. Wherever there is a 2×2 grid of non-defective chips, these grid cells can be wired together. This potential grouping of chips can be visualized as a unit disk which overlaps with the four grid cells that are being wired together. For example,

[3]I have limited knowledge in this field, so forgive me if I abuse some terminology.

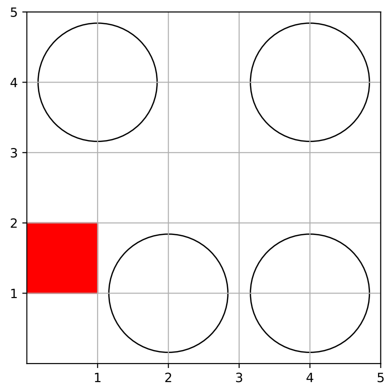
in the figure below, the four grid cells in the upper left corner could be wired together, so we would put a unit disk at (1, 4) with radius 1.



Continuing this process for every possible wiring, we obtain the collection of disks shown in Figure 4a. Overlapping unit disks represents two wirings which conflict with each other because they use the same chips. In this scenario, the $\mathsf{MIS}_{\mathsf{UDG}}$ represents the maximum number of non-conflicting wirings, as shown in Figure 4b.



(a) Unit disks representing chip wirings.



(b) The $\mathsf{MIS}_{\mathsf{UDG}}$ of unit disks.