Master2 Informatique
UE: Algorithmique Distribuées – 4TIN913U
Responsable : M. Gavoille
Date : 5 janvier 2022
Durée : 1h30
Documents (course and personnal notes) allowed

Answers can be written in French or in English. All "algorithms" involved in this exam are distributed algorithms in the LOCAL model.

Coloring Cycles with 3 Colors

Question 1. *Give the main features of the LOCAL model.*

Recall that an *oriented* cycle is a cycle graph where each vertex is aware of its successor (and thus of its predecessor). This knowledge can be used to speedup coloring algorithm.

In the lecture, we described a fast 3-coloring algorithm for oriented cycles with n vertices running in at most $f(n)+c$ rounds, where $f(n)$ is some suitable fonction of n, and c is some small constant. For simplicity, let us denote this algorithm by A.

Question 2. *Describe the principle of algorithm* A*, i.e., the main steps, and specify what is* $f(n)$ *and the constant* c*.*

The goal is to slightly improve algorithm A. Recall that, at some stage, algorithm A reduces the palette of colors (given as a range of integers) from $[0,6[= \{0,1,2,3,4,5\}$ to $[0,3[= \{0,1,2\}$.

Question 3. *Show that the colors* 4 *and* 5 *can be removed simultaneously from the initial palette* $[0,6[$ *in only one round. [Hint: Manage carefully the case where vertices of colors* 4 *and* 5 *are adjacent.] Does this round use the orientation of the cycle?*

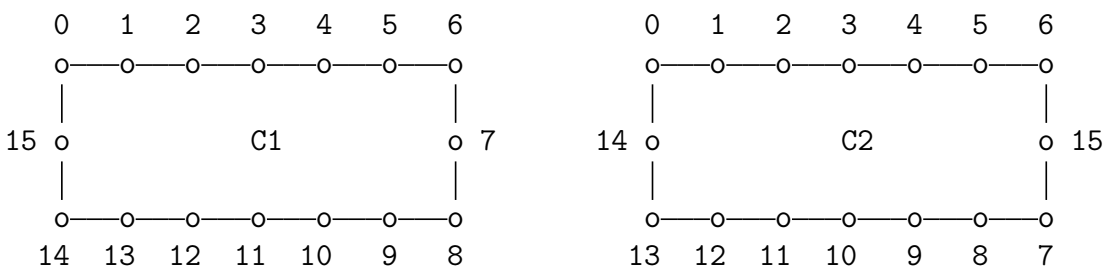
Let us call this improved algorithm, algorithm B.

Question 4. *What is the maximum number of rounds of* B *if* $n \leqslant 65536 = 2^{16}$*? Justify.*

Coloring Even Cycles with 2 Colors

In this part we are interesting in 2-coloring cycles with an even number of vertices. We will assume that all cycles are oriented, even if we did not say it explicitly.

Consider two cycles, C_1 and C_2, each with $n = 16$ vertices, and with the following ID distributions:



Question 5. *Design a 2-coloring algorithm specific for* C_1 *that runs in* 0 *round. Same question for* C_2*.*

In the lecture, we proved a lower bound of $(\frac{1}{2}\log^* n) - 1$ rounds for any 4-coloring algorithm for cycles with n vertices.

Question 6. *Noting that a 2-coloring is a particular 4-coloring, what is the minimum number of rounds implied by this above lower bound applied to 2-coloring for cycles of $n = 16$ vertices? Justify.*

Question 7. *Is the result of Question 6 in contradiction with the algorithms of Question 5? Justify.*

Question 8. *Design a fast 2-coloring algorithm that applies to both C_1 and C_2. Specify its number of rounds. [Hint: Detect if a vertex lives in C_1 or C_2.]*

In the lecture, we showed that any k-coloring algorithm running in t rounds can be seen as a mapping from t-views to $[0, k[$. Recall that a t-view of a vertex u is the sequence of all the IDs that u contains in its ball of radius t, i.e., the IDs that u can see after t rounds of communication. For a cycle with n vertices, this is a sequence of $2t+1$ unique integers taken from $[0, n[$. For instance, in C_1 the 3-view of vertex with ID 4 is the sequence $(1, 2, 3, \boxed{4}, 5, 6, 7)$ assuming a clockwise orientation.

Question 9. *Show that any 2-coloring algorithm for both C_1 and C_2 requires at least 4 rounds. [Hint: Consider the 3-views for vertices with IDs 3 and 10 for instance.]*

Question 10. *Show that, more generally, any 2-coloring algorithm for cycles of n vertices requires $\Omega(n)$ rounds. [Hint: Consider some cycles with $n \geqslant 4(t+1)$ vertices, and argue about the t-views of some specific vertices.]*

Fast Palette Reduction

The goal of this part is to design a $(\Delta + 1)$-coloring for any graph of maximum degree Δ. We have seen in the lecture such an algorithm running in $\log^* n + 2^{O(\Delta)}$ rounds. We will focus on the second phase of the algorithm, the palette reduction, that takes about $2^{O(\Delta)}$ rounds to decrease the palette from $[0, 3^\Delta[$ to $[0, \Delta]$.

Question 11. *Give an accurate bound on the number of rounds it takes to perform this palette reduction?*

In order to perform a faster palette reduction, we will remove several colors in a single rounds, as done in Question 3. Assume that after the first phase, we have a coloring c with palette $[0, k[$ and that each vertex see at most d different colors in its neighborwood. In other words, $d = \max_u |c(N(u))|$. For simplicity we assume that $k = sd + s$ for some integer $s \geqslant 2$.

The goal is to reduce the intial palette from $[0, k[= [0, sd + s[$ to $[0, sd[$ by removing the s largest colors of $[0, k[$. For this, every vertex u with a color in $[sd, sd+s[$ applies in parallel a FirstFree($X \cup I_i$), where $X = c(N(u))$ and I_i is some subinterval of $[0, sd[$ depending on the color $i = c(u)$ of u and whose aim is to avoid collisions. Indeed, two neighbors of different colors in $[sd, sd + s[$ will apply in parallel FirstFree.

To make more concrete this technique, consider the following example where $k = 9$, $d = 2$ and $s = 3$. The initial palette is $[0, k[= [0, sd + s[= [0, 3 \cdot 2 + 3[= [0, 9[$ and the goal is to reduce it to $[0, 6[$ by removing the colors $6, 7, 8$ in a single round. These latter colors are called *special*. Consider a vertex u with a special color. The round of communication consists in exchanging the color of u with its neighbors. Let $X = c(N(u))$ be the set of colors of u's neighbors. If X does not contain any special color, then u can recolor with FirstFree(X) since none of its neighbors are concerned with a recoloring. Otherwise, split the palette $[0, k[= [0, sd + s[$ into s subintervals of length d followed by the s special colors. Namely, split $[0, 9[$ into $[0, 1] \cup [2, 3] \cup [4, 5] \cup [6, 7, 8]$. Then, the recoloring is as follows: if $c(u) = 6$, then u tries to recolor in $[0, 1]$ using a FirstFree(X); if $c(u) = 7$, then u tries to recolor in $[2, 3]$ using a FirstFree($X \cup [0, 2[$); and if $c(u) = 8$, then u tries to recolor in $[4, 5]$ using a FirstFree($X \cup [0, 4[$). In other words, u tries to recolor in the subinterval corresponding

to the rank of its special color. The rank of the special color of u is precisely $c(u) - sd$. Note that intervals of length d suffice. Indeed, if two neighbors are of special colors, then they can see only $d-1$ non-special colors each or less (instead of d). And thus FIRSTFREE will necessarily find a free color in an interval of length $(d-1)+1 = d$.

Question 12. *Formalized the above algorithm that reduces the palette from* $[0, sd+s[$ *to* $[0, sd[$ *in one round.*

In the remaining, we fix $d = \Delta$. We will apply iteratively the Fast Palette Reduction (FPR) as in Question 12, possibly with a different parameter s, until we get a palette of 2Δ colors or less. From that point the FPR does not apply anymore and a $O(\Delta)$-round classical palette reduction is needed to get the wanted $(\Delta+1)$-coloring.

More precisely, let $s_0, s_1, \ldots, s_t$ be some reals such that: (1) The initial palette is $[0, k[= [0, s_0\Delta[$; (2) At step $i \in \{1, \ldots, t\}$, whenever FPR is applied, the palette reduces from $[0, s_{i-1}\Delta[= [0, s_i\Delta + s_i[$ to $[0, s_i\Delta[$.

In the remaining, we will negligate the fact that some s_i are not necessarily integers[1].

Question 13. *Express* s_i *as fonction of* i, Δ *and* k.

Question 14. *Find the smallest number* t *of FPR steps so that the final palette is* $[0, 2\Delta[$ *or smaller. Overall, what is the complexity of the resulting new* $(\Delta+1)$*-coloring algorithm? [Hint: Assume known the inequality* $1/(x+1) < \ln(1+1/x) < 1/x$ *for every* $x > 0$*.]*

In a celebrate paper of 1992, Nathan Linial showed how to find a $O(\Delta^2)$-coloring in $O(\log^* n)$ rounds.

Question 15. *Using this Linial's precoloring, and combining with Question 14, design an even faster* $(\Delta+1)$*-coloring algorithm. What is its complexity?*

FIN.

[1] Otherwise we will have to consider $\lfloor s_i \rfloor$ instead of s_i, which is awkward and does not affect anway the asymptotic results.