2 Local algorithms

To avoid some technical issues we work most of the time, unless stated otherwise, with LCLs *without inputs*. That is, Σ_{in} contains at most one element. We also set $\Sigma := \Sigma_{out}$ and write LCLs as triplets and not quadruples, i.e., $\Pi = (\Sigma, t, \mathcal{P})$. Recall that $\mathbf{G}$ is a fixed class of graphs endowed with some vertex labelings and $\mathbf{G}_\bullet$ are rooted neighborhoods of graphs from $\mathbf{G}$.

Definition 0.1. *A* local algorithm *is a partial function $\mathcal{A}$ from $\mathbf{G}_\bullet$, possibly with a set of global parameters, to some set Σ. Given $H \in \mathbf{G}$, we define:*

1. *the* output of $\mathcal{A}$ on H after $r \in \mathbb{N}$ rounds *to be the function $\mathcal{A}(H, r) : V(H) \to \Sigma$ that is defined as*
$$\mathcal{A}(H, r)(v) = \mathcal{A}(\mathcal{B}_H(v, r)),$$
2. *the* uniform output of $\mathcal{A}$ on H *to be the function $\mathcal{A}(H) : V(H) \to \Sigma$ that is defined at vertex v as*
$$\mathcal{A}(H)(v) = \mathcal{A}(\mathcal{B}_H(v, r)),$$
where $r \in \mathbb{N}$ is minimal such that $\mathcal{B}_H(v, r) \in \mathrm{dom}(\mathcal{A})$, and undefined otherwise.

In order to be able to define anything non-trivial locally, we need to break symmetries:

- every vertex is assigned a unique label from the set $\{1, \ldots, \mathrm{poly}(|V(G)|)\}$,
- every vertex generates independently random real number from $[0, 1]$ according to the Lebesgue measure λ.

This leads to the notion of deterministic and randomized algorithms.

Definition 0.2. *A local algorithm $\mathcal{A}$ is:*

1. **deterministic** *if it assumes as an input (a) unique identifiers assigned to vertices from range polynomial in the size of the graph and (b) the size of the graph,*
2. **randomized** *if it assumes as an input (a) real numbers from the range $[0, 1]$ sampled at every vertex according to the Lebesgue measure and (b) the size of the graph,*
3. **uniform** *if it assumes as an input (a) real numbers from the range $[0, 1]$ sampled at every vertex according to the Lebesgue measure.*

Next we illustrate how we use this definition. Let $\mathcal{A}$ be a deterministic local algorithm. Given $H \in \mathbf{G}$ and $r \in \mathbb{N}$ (that will depend on $|V(H)|$), we take any assignment of unique identifiers α, that is α is an injective map

$$\alpha : V(H) \to \{1, \ldots, \mathrm{poly}(|V(H)|)\}.$$

Given this data, we run $\mathcal{A}$ on H_α for r rounds. This produces a function $\mathcal{A}(H_\alpha, r)$. In another words $\mathcal{A}$ operates not on graphs from $\mathbf{G}$ but on graphs from $\mathbf{G}$ with additional vertex labeling α. In the case of deterministic algorithms this labeling is any assignment of unique identifiers. We say that $\mathcal{A}$ *solves* an LCL Π, if the output is a Π-coloring regardless of the assignment of unique identifiers.

Define $\mathbf{G}_n$ to be the class of graphs from $\mathbf{G}$ of size exactly $n \in \mathbb{N}$.

Definition 0.3 (Complexity of LCLs)**.** *We say that the* deterministic complexity *of an LCL* Π *is* $O(f(n))$*, where* $f : \mathbb{N} \to \mathbb{N}$*, and write* $\mathsf{LOCAL}(O(f(n)))$*, if there is a deterministic local algorithm* $\mathcal{A}$ *and* $(r_n)_n \in O(f(n))$ *such that* $\mathcal{A}(H_\alpha, r_n)$ *is a* Π*-coloring for every* $H \in \mathbf{G}_n$ *and any injective map* $\alpha : V(H) \to \mathrm{poly}(n)$*.*

We say that the randomized complexity *of an LCL* Π *is* $O(f(n))$*, where* $f : \mathbb{N} \to \mathbb{N}$*, and write* $\mathsf{RLOCAL}(O(f(n)))$*, if there is a randomized local algorithm* $\mathcal{A}$ *and* $(r_n)_n \in O(f(n))$ *such that for every* $H \in \mathbf{G}$ *the labeling* $\mathcal{A}(H_\alpha, r_n)$ *is a* Π*-coloring with probability at least* $1 - 1/n$ *where* $\alpha : V(H) \to [0, 1]$ *is sampled according to the product measure.*

We say that the uniform complexity *of an LCL* Π *is* $O(f(\varepsilon))$*, where* $f : (0, 1] \to \mathbb{R}^+$*, and write* $\mathsf{ULOCAL}(O(f(\varepsilon)))$ *if there is a randomized local algorithm* $\mathcal{A}$ *and such that* $\mathcal{A}(H_\alpha)$ *is a* Π*-coloring with probability* 1 *for every* $H \in \mathbf{G}_n$ *and* $\alpha : V(H) \to [0, 1]$ *sampled according to the product measure, and*

$$\mathbf{P}(R_{\mathcal{A}} > f(\epsilon)) \leq \epsilon.$$

Meaning that, the supremum over all vertices and graphs in $\mathbf{G}$ *of the events that the vertex needs to look further than* $f(\epsilon)$ *is less than* ϵ*.*

Remark 0.4. *Often, the error probability for randomized algorithms are assumed to be polynomial in* n*, e.g.,* $1 - 1/n^3$ *etc.*

Note that in case of non-uniform local algorithms, we actually work with a sequence of local rules $(\mathcal{A}_n, r_n)$. We start with an easy observation.

Proposition 0.5 (Linial [Lin92], Holroyd–Schramm–Wilson [HSW17])**.** *Let* $\mathbf{G}$ *be the class of all graphs of degree bounded by* Δ *and* Π *be the proper vertex* $(\Delta + 1)$*-coloring problem. Then we have*

$$\Pi \in \mathsf{LOCAL}(O(\log^* n)),\ \mathsf{RLOCAL}(O(\log^* n)),\ \mathsf{ULOCAL}(O(\log^* 1/\varepsilon)).$$

Corollary 0.6. *Let* Π *be the* k-distance coloring *problem, that is, a coloring with at most* $(\Delta^k + 1)$*-many colors such that no distinct vertices of distance at most* k *have the same color. Then*

$$\Pi \in \mathsf{LOCAL}(O(\log^* n)),\ \mathsf{RLOCAL}(O(\log^* n)),\ \mathsf{ULOCAL}(O(\log^* 1/\varepsilon)).$$

Before the proof we discuss an important speed-up theorem. By a speed-up we mean the following type of result: whenever Π is in the class $\mathsf{LOCAL}(O(f(n)))$, then it is actually in the class $\mathsf{LOCAL}(O(g(n)))$ for some $g(n) \in o(f(n))$. Then we say that there is a speed up from f to g.

Proposition 0.7 (Deterministic speed up [CKP16])**.** *Assume that* $\mathbf{G}$ *is closed under induced subgraphs. Let* $\Pi \in \mathsf{LOCAL}(o(\log n))$*, or in general of complexity strictly lower than the growth of the graphs. Then* $\Pi \in \mathsf{LOCAL}(O(\log^* n))$*.*

On a high-level, the price that we have to pay to lie to our algorithm is $\log^* n$ and then we run a constant round local rule.

Proof. Let $\mathcal{A}$ be a deterministic local algorithm of complexity $(r_n) \in o(\log n)$ that solves LCL Π and let t be the locality of Π. Pick k large enough so that $\Delta^{2(r_k+t)} \ll k$. Given a graph H of size $n = |V(H)|$ (with unique identifiers), we first solve the $2(r_k + t)$-distance coloring problem in $O(\log^* n)$ rounds, by Corollary 0.6. Denote this coloring as c and write H_c for H with the additional structure c.

Now consider the output of $\mathcal{A}$ on G_c after r_k-rounds. As $\Delta^{2(r_k+t)} + 1 \ll k$ the algorithm only sees identifiers from the set $\{1, \ldots, k\}$ and as $\mathbf{G}$ is closed under induced subgraphs, $\mathcal{A}$ applied on a $(r_k + t)$-neighborhood of a vertex v cannot tell that it is not run on some $G \in \mathbf{G}_k$. This allows to conclude that $\mathcal{A}(H_c, r_k)$ is a Π-coloring. □

References

[CKP16] Yi-Jun Chang, Tswi Kopelowitz, and Seth Pettie. An exponential separation between randomized and deterministic complexity in the LOCAL model. In *Proc. 57th IEEE Symp. on Foundations of Computer Science (FOCS)*, 2016.

[HSW17] Alexander E. Holroyd, Oded Schramm, and David B. Wilson. Finitary coloring. *Ann. Probab.*, 45(5):2867–2898, 2017.

[Lin92] Nati Linial. Locality in distributed graph algorithms. *SIAM Journal on Computing*, 21(1):193–201, 1992.