

On the Universality of Round Elimination Fixed Points

Alkida Balliu · Gran Sasso Science Institute, TBD

Sebastian Brandt · CISA Helmholtz Center for Information Security, Germany

Ole Gabsdil · CISA Helmholtz Center for Information Security, Germany

Dennis Olivetti · Gran Sasso Science Institute, Italy

Jukka Suomela · Aalto University, Finland

Abstract. Recent work on distributed graph algorithms [e.g. STOC 2022, ITCS 2022, PODC 2020] has drawn attention to the following open question: *are round elimination fixed points a universal technique for proving lower bounds?* That is, given a locally checkable problem Π that requires at least $\Omega(\log n)$ rounds in the deterministic LOCAL model, can we always find a relaxation Π' of Π that is a *nontrivial fixed point* for the round elimination technique [see STOC 2016, PODC 2019]? If yes, then a key part of distributed computational complexity would be also *decidable*.

The key obstacle so far has been a certain family of homomorphism problems [ITCS 2022], which require $\Omega(\log n)$ rounds, but the only known proof is based on Marks' technique [J.AMS 2016].

We develop a new technique for constructing round elimination lower bounds systematically. Using so-called *tripotent inputs* we show that the aforementioned homomorphism problems indeed admit a lower bound proof that is based on round elimination fixed points. Hence we eliminate the only known obstacle for the universality of round elimination.

Yet we also present a new obstacle: we show that there are some problems *with inputs* that require $\Omega(\log n)$ rounds, yet there is no proof that is based on relaxations to nontrivial round elimination fixed points. Hence round elimination cannot be a universal technique for problems with inputs (but it might be universal for problems without inputs).

We also prove the first fully general lower bound theorem that is applicable to any problem, with or without inputs, that is a fixed point in round elimination. Prior results of this form were only able to handle certain very restricted inputs.

1 Introduction

Can we always systematically distinguish between easy and hard problems in distributed graph algorithms? We study this question in the context of locally checkable labeling problems (LCLs) [NS95] in trees; these are constraint satisfaction problems that can be conveniently specified by listing a finite set of valid labeled neighborhoods. Numerous problems familiar from the theory of distributed graph algorithms such as vertex and edge coloring, maximal independent set, maximal matching, sinkless orientation, and many other splitting and orientation problems are examples of LCLs in bounded-degree graphs.

1.1 Classification of LCL problems

As a result of a long research program [CV86, Nao91, Lin92, NS95, BFH⁺16, FG17, GHK18, BHK⁺18, CP19, CKP19, RG20, BBOS20, BBOS21, BCHM⁺21], the landscape of LCL problems is now well understood [Suo20], especially in the case of trees [Cha20, BBOS21, GRB22]. For our purposes the most interesting part is the following dichotomy; any LCL problem in trees falls in one of these complexity classes:

1. **Easy problems:** the round complexity is $O(\log^* n)$ in the deterministic and randomized LOCAL models [Lin92, Pel00] and $O(1)$ in the deterministic and randomized SLOCAL models [GKM17]. A canonical example of such a problem is *3-coloring of paths*, for which all of these complexities are tight [CV86, GPS88, Lin92].
2. **Intermediate (or harder) problems:** the round complexity is $\Omega(\log n)$ in deterministic LOCAL, $\Omega(\log \log n)$ in randomized LOCAL, $\Omega(\log \log n)$ in deterministic SLOCAL, and $\Omega(\log \log \log n)$ in randomized SLOCAL. A canonical example of such a problem is *sinkless orientation*, for which all of these complexities are tight [BFH⁺16, CKP19, GHK18, GS17, BKK⁺23].

But given some unknown problem Π , how can we tell whether it is easy or (at least) intermediate?

How to prove that a given problem is easy? One half of this is known; being easy is semi-decidable. More precisely, for any easy problem Π there exists a natural number k such that one can map a distance- k coloring to a valid solution of Π [CKP19, BHK⁺17]. For any fixed k , there are only finitely many candidate functions to check, so at least in principle one could systematically check $k = 1, 2, \dots$ and stop once a suitable function is found. However, the process will never terminate for harder problems.

How to prove that a given problem is intermediate-hard? What is missing is a systematic way to detect if a problem is at least intermediate-hard. There are many different proof techniques used to show that a problem is intermediate or harder, but especially when more elementary techniques have failed, by far the most successful proof technique has been round elimination, which we will discuss in more depth next.

1.2 Round elimination

Round elimination [Bra19, BFH⁺16] is a function \mathcal{Q} that maps an LCL problem Π to another problem $\mathcal{Q}(\Pi)$. The key property is this:

If Π is an easy but nontrivial problem, then $\mathcal{Q}(\Pi)$ is exactly one round easier to solve than Π .

So if $\mathcal{Q}(\Pi)$ is *not* one round easier than Π , and Π is *not* trivial (it requires at least one round to solve), the only possible explanation was that Π was at least intermediate-hard.

In general, it is hard to compare $\mathcal{Q}(\Pi)$ and Π . However, if we are lucky enough to find a *fixed point*, i.e., $\mathcal{Q}(\Pi) = \Pi$, then clearly $\mathcal{Q}(\Pi)$ cannot be one round easier than Π , and we can conclude that Π must be intermediate-hard (or trivial).

In this work we slightly extend (and abuse) the notion of fixed points to also cover cases where Π is a *relaxation* of $\mathcal{Q}(\Pi)$. We write $\mathcal{Q}(\Pi) \xrightarrow{0} \Pi$ to denote that a solution of $\mathcal{Q}(\Pi)$ can be turned in zero rounds into a solution of Π ; clearly in this case $\mathcal{Q}(\Pi)$ cannot be one round easier than Π , and hence Π has to be at least intermediate-hard.

Fixed points are rare, but for many intermediate problems Π the following proof idea does the trick: find a relaxation $\Pi \xrightarrow{0} \Pi'$, show that Π' is nontrivial, show that $\mathcal{Q}(\Pi') \xrightarrow{0} \Pi'$, conclude that Π' has to be at least intermediate-hard, and hence is the original problem Π .

1.3 Is round elimination universal?

If this scheme worked for *every* intermediate problem, then we would have a procedure for proving that a problem is at least intermediate-hard: simply start to enumerate the countable set of possible relaxations Π' of Π , and apply the \mathcal{Q} function to each of them, and stop if we find a nontrivial fixed point. If in parallel with this we also apply the procedure that searches for a proof that Π is easy, one of these procedures would always terminate. This would mean that distinguishing easy vs. intermediate-hard problems would be decidable, resolving a major open question of this field, see e.g. [BBC⁺22].

Inspired by this, the following open question was stated e.g. in [BBKO22, Open Problem 1] and [BO20, Open Problem 6.3]:

Question 1.1. Do all intermediate problems relax to a nontrivial round elimination fixed point?

So far there is only one known family of problems that is known to be intermediate-hard, yet it is not known if it admits a proof through a relaxation to a round elimination fixed point, namely the graph homomorphism problems introduced by [BCG⁺22]. The simplest example is the task that we will call *restricted 9-coloring*; the task is to find a homomorphism from the input graph G to the following 9-node graph H :



This problem is known to be intermediate-hard [BCG⁺22], and the proof uses Marks' technique [Mar16], familiar from the context of measurable combinatorics.

If the family of homomorphism problems is indeed outside the scope of round elimination fixed points, then it would mean that round elimination fixed points are not a universal technique.

Furthermore, it would mean that round elimination and Marks’ technique are orthogonal lower bound techniques, and one does not subsume the other. In [BCG⁺22] obtaining the same lower bounds for the homomorphism problems that Marks’ technique achieves is stated as an “exciting open problem.”

1.4 Our contributions

Contribution 1: homomorphism problems admit nontrivial round elimination fixed points (Section 5). We show that round elimination fixed points can be indeed used to show that the homomorphism problems studied in [BCG⁺22] are intermediate-hard. We show that for each of these problems Π , it is possible to find a relaxation $\Pi \xrightarrow{0} \Pi'$ such that $\mathcal{Q}(\Pi') \xrightarrow{0} \Pi'$.

In particular, we remove all known obstacles to Question 1.1; after this work, there is *no candidate problem* that would serve as a counterexample for the conjecture that round elimination fixed points are indeed a universal technique for proving that a problem is intermediate-hard. Furthermore, these were the only known examples in which Marks’ technique was previously necessary to prove lower bounds in the LOCAL model; our work is giving evidence that it might indeed be the case that Marks’ technique can be always replaced by round elimination in this context.

Contribution 2: problems with inputs do not admit nontrivial round elimination fixed points (Section 4). However, everything that we discussed above holds only for problems without inputs. We show that *problems with inputs* can be much more challenging. In particular, we prove that *sinkless and sourceless orientation given a sinkless orientation* (SSO-SO) is a problem that is intermediate-hard, yet it admits no relaxation to a round elimination fixed point.

Hence the answer to Question 1.1 in full generality is negative, but it could be still true for problems without inputs.

Contribution 3: auxiliary inputs can be eliminated (Section 3). We will next zoom into the role of inputs. One trick that was used already in the very first applications of round elimination [BFH⁺16] may at first look counterintuitive: we want to show that Π is intermediate-hard, and we make our life seemingly harder and try to argue that Π is intermediate-hard *even if we are given some friendly auxiliary input \mathcal{I}* . For example, to show that 3-vertex-coloring is intermediate-hard, one can show that 3-vertex-coloring is intermediate-hard even if we are given a 3-edge-coloring as input. Why this makes sense is that it may enable one to find a useful relaxation: a 3-vertex-coloring in 3-edge-colored graphs can be used to find a sinkless orientation, and sinkless orientations are a nontrivial round elimination fixed point that remains nontrivial even if we are given a 3-edge-coloring.

We show that such a roundabout approach is never needed. If we can find a relaxation to a nontrivial fixed point with the help of auxiliary inputs, we can also eliminate such inputs. We introduce the concept of *tripotent inputs* that allow us to turn fixed points with auxiliary inputs in a mechanical way into fixed points without inputs. We demonstrate the effectiveness of this tool to construct a pure input-free round elimination fixed point for the homomorphism problems (Contribution 1).

Tripotent inputs also open up a genuinely new way for discovering lower bounds: if we can *guess* a fixed point relaxation (say, using one of the well-known fixed points such as sinkless orientation), we can also work backwards to construct a suitable input, and then we can finally verify whether our guess was indeed correct. Hence instead of guessing a suitable input and finding a fixed point (as has been done in the literature so far), we can guess a fixed point and calculate the input.

Contribution 4: round elimination works with inputs (Section 6). Finally, we turn our attention to problems where inputs can play a crucial role; the input is not merely helpful information that the algorithm may choose to use to make the problem easier to solve (e.g. SSO-SO), but the input may restrict what are possible valid outputs (e.g. list coloring).

Prior proofs that show that round elimination fixed points imply that problems are intermediate-hard are not compatible with arbitrary inputs—indeed, it is not even entirely clear exactly what kind of inputs they can handle. We prove for the first time that we can use round elimination fixed points to derive lower bounds also for problems with general inputs.

Discussion. In summary, our work provides new evidence pointing in the following direction:

1. For intermediate problems without inputs, round elimination fixed points might be a universal technique, and if this is indeed true, classification of input-free problems to easy vs. intermediate would be decidable (at least in principle, if not in practice).
2. For intermediate problems with inputs, round elimination fixed points are not a universal technique, and automatic classification of such problems seems to require the development of new lower-bound proof techniques.
3. The distinction between problems with inputs vs. problems without inputs is clear-cut: auxiliary inputs that were only introduced as a proof technique in order to find appropriate fixed point relaxations can be always systematically eliminated.

1.5 Key new ideas and proof techniques

Tripotent inputs (Section 3). The key novel tool that we introduce in this work is what we call *tripotent inputs*. This is a function τ_P parameterized by an LCL problem P and it maps any given LCL problem Π to another problem $\tau_P(\Pi)$. For us the most interesting case is if $P = \mathcal{F}$ for some nontrivial round elimination fixed point \mathcal{F} .

The intuition is that $\tau_{\mathcal{F}}(\Pi)$ would be a good choice of input if we wanted to show that Π with some input can be relaxed to \mathcal{F} , a nontrivial fixed point. But the remarkable property of $\tau_{\mathcal{F}}$ is that as long as \mathcal{F} is nontrivial given $\tau_{\mathcal{F}}(\Pi)$, then $\Pi' = \tau_{\mathcal{F}}(\tau_{\mathcal{F}}(\Pi))$ will be a nontrivial fixed point relaxation of Π . Furthermore, as long as there is *some* input \mathcal{I} such that Π with \mathcal{I} can be relaxed to \mathcal{F} and \mathcal{F} remains nontrivial given \mathcal{I} , then \mathcal{F} is also nontrivial given $\tau_{\mathcal{F}}(\Pi)$.

Put together, tripotent inputs open up *two* new possibilities for proving hardness with the help of round elimination fixed points:

1. We can freely cheat with inputs (e.g. assume 3-edge-coloring) to discover that Π with some input \mathcal{I} can be relaxed to some fixed point \mathcal{F} that remains nontrivial even given \mathcal{I} . Then we know that \mathcal{F} is also nontrivial given $\tau_{\mathcal{F}}(\Pi)$. Then we can in a mechanical way compute $\Pi' = \tau_{\mathcal{F}}(\tau_{\mathcal{F}}(\Pi))$, and Π' is guaranteed to be a nontrivial input-free fixed-point relaxation of Π , that is, $\Pi \xrightarrow{0} \Pi'$ and $\mathcal{Q}(\Pi') \xrightarrow{0} \Pi'$ and Π' is nontrivial. We can fully eliminate the input!
2. We can also *guess* some fixed point candidate \mathcal{F} , compute $\tau_{\mathcal{F}}(\Pi)$ in a mechanical way, see if \mathcal{F} remains hard given $\tau_{\mathcal{F}}(\Pi)$, and if so, we can find a nontrivial input-free fixed point relaxation $\Pi' = \tau_{\mathcal{F}}(\tau_{\mathcal{F}}(\Pi))$ of Π .

We will make use of the first approach to argue that e.g. restricted 9-coloring problem and other homomorphism problems indeed admit an input-free round elimination fixed point. Here it is important to note that while Π , \mathcal{I} , and \mathcal{F} can be relatively simple problems with concise descriptions,

the description of Π' can be much longer, and in particular very hard to guess or systematically discover without the use of our new tool.

It turns out that tripotent inputs satisfy numerous convenient mathematical properties. Among others, as the name suggests, $\tau_{\mathcal{F}}$ is always a tripotent function in the sense that $\tau_{\mathcal{F}}(\Pi)$ and $\tau_{\mathcal{F}}(\tau_{\mathcal{F}}(\tau_{\mathcal{F}}(\Pi)))$ are equivalent problems. We expect that further study of this function will shed much more light on the round elimination technique.

Non-existence of round elimination fixed points (Section 4). In Section 4 we show that the SSO-SO problem (find a sinkless and sourceless orientation, given a sinkless orientation as input) does not admit any nontrivial fixed point relaxation, even though it is intermediate-hard.

To prove this result, let Π be the SSO problem. First, we analyze the structure of problems $\mathcal{Q}^k(\Pi)$ that are obtained by applying round elimination k times—we essentially obtain a closed-form characterization of all such problems.

Second, we consider a hypothetical fixed-point relaxation $\hat{\Pi}$ of Π . Since $\hat{\Pi}$ is a relaxation of Π , also $\mathcal{Q}^k(\hat{\Pi}) = \hat{\Pi}$ is a relaxation of $\mathcal{Q}^k(\Pi)$. We choose a k sufficiently large in comparison with the number of labels in $\hat{\Pi}$, and use the structural characterization of $\mathcal{Q}^k(\Pi)$ to identify a subproblem of $\hat{\Pi}$ that is easy to solve.

Building on this idea, we then show that given a sinkless orientation, we can find a feasible solution to $\mathcal{Q}^k(\hat{\Pi}) = \hat{\Pi}$; hence $\hat{\Pi}$ cannot be a relaxation of SSO that is a round elimination fixed point that remains nontrivial given a sinkless orientation.

Finally, we can then use the machinery developed for tripotent inputs to show that there is no such relaxation of SSO-SO, either.

ORCX problem (Section 5). To show that homomorphism problems from [BCG⁺22] admit a fixed-point relaxation, we introduce a family of graph problem that we call the *ORCX problem*.

To better understand the key idea, let us consider the special case of 3-regular graphs and the restricted 9-coloring problem introduced in Section 1.3. We can show that in 3-regular graphs, the ORCX problem has these properties:

1. Let Π be the restricted 9-coloring problem and let \mathcal{I} be the auxiliary input of 3-edge coloring. Then Π with input \mathcal{I} can be used to solve the ORCX problem.
2. The ORCX problem is a fixed point in round elimination.
3. The ORCX problem remains non-trivial even if we are given a 3-edge coloring.

Now we are ready to apply the machinery of tripotent inputs to turn ORCX together with 3-coloring into an input-free problem Π' that is a nontrivial fixed point relaxation of the 9-coloring problem.

This idea can be generalized to a broader family of homomorphism problems. In Section 5 we will study them in Δ -regular graphs and the input will be a Δ -edge-coloring, but the same construction of the ORCX problem applies also there.

Error probability analysis with inputs (Section 6). To show that round elimination fixed points imply an $\Omega(\log \log n)$ lower bound in the randomized LOCAL model, the key step is to analyze failure probabilities for a single step of round elimination.

While round elimination is a function that maps problems to problems, it also has algorithmic implications: given an algorithm \mathcal{A} that solves Π in T rounds, we can construct an algorithm \mathcal{A}' that solves $\Pi' = \mathcal{Q}(\Pi)$ in $T - 1$ rounds. However, the catch is that if \mathcal{A} is a randomized algorithm that fails with probability p , then \mathcal{A}' is a randomized algorithm that fails with probability $p' > p$.

While in prior work there are results that analyze how p' depends on p and the structural properties of Π , none of these results can handle inputs in full generality. In this work we present the first analysis of error probabilities that is able to take into account also inputs.

We will use our new analysis in Section 4 to show that the SSO-SO problem is indeed at least intermediate-hard, and in Section 5 to derive a lower bound for homomorphism problems.

2 Preliminaries

In this section, we provide some of the technical background from the literature and collect definitions and observations that we need to obtain our results.

Graph-theoretic basics. Throughout the paper, we will use standard graph notation, such as $G = (V, E)$ for a graph, $\deg(v)$ for the degree of a node v , etc. Moreover, we will make use of the following notion of a *half-edge*:

Definition 2.1 (Half-edge). Let $G = (V, E)$ be a graph. A *half-edge* h is a pair (v, e) such that e is incident to v . We say that the half-edge (v, e) is *incident to* v and *belongs to* e . We denote the set of half-edges of G by H , i.e., $H := \{(v, e) \mid v \in V, e \in E, v \text{ is an endpoint of } e\}$.

The LOCAL model. The LOCAL model is a synchronous message-passing model defined as follows. A network is represented as a graph $G = (V, E)$, where V is the set of machines and E is the set of communication links. At the beginning of the computation, each node v is aware of the size n of the graph, the maximum degree Δ of the graph, its own degree $\deg(v)$, and the input assigned to its incident half-edges. Moreover, in the case of deterministic algorithms, each node is assigned a unique ID in $\{1, \dots, n\}$, while in the case of randomized algorithms each node is assigned a random bit-string of infinite length. Then, the computation proceeds in rounds, and at each round each node can send a (possibly different) message to each of its neighbors. Then, as a function of the received messages, a node updates its state and proceeds to the next round. In a T -round algorithm, each node is required to output a solution for the considered problem within round T . In this model, the size of the messages, and the computational power of each node, are unbounded (messages do not even have to be of finite length: as in this paper we prove lower bounds, considering this unreasonable assumption only makes our results stronger). In the case of randomized algorithms, it is required that the produced solution is correct with probability at least $1 - 1/n$. For technical reasons, we also assume that each node v comes with a *port numbering*, that is, an arbitrary assignment of the numbers $1, \dots, \deg(v)$ to the half-edges incident to v , such that different half-edges get different numbers. Additionally, an arbitrary orientation is assigned to the edges, or in other words, also edges come with a port numbering, in this case from $\{1, 2\}$. The *port numbering* (PN) model is defined similarly as the LOCAL model, and the only difference is that there are no IDs assigned to the nodes. Since the size of the messages is not restricted, we can see any T -round algorithm as a function that maps a given radius- T neighborhood into an output.

Node-edge-checkable problems. Next, we introduce the notion of a node-edge-checkable problem [Bra19, BCHM⁺21]. These are problems that capture the vast majority of problems studied in the LOCAL model, including maximal independent set, maximal matching, and coloring problems. We define such problems in the case of regular graphs. The reason is that all known lower bounds that are obtained by the lower bound techniques relevant to our work—round elimination [BFH⁺16, Bra19] and Marks' technique [Mar16]—are obtained on regular graphs or regular

trees¹, and the same holds for the lower bound constructions we develop in this work. As such, we can restrict attention to Δ -regular graphs/trees and will provide our definitions for this setting. Consequently, for the remainder of the paper (except for Section 6), we will assume that some value Δ is fixed.

We note that our definitions generalize straightforwardly to the setting of arbitrary graphs though they might be much more cumbersome to write down in that setting. In fact, we will provide a generalized definition in Section 6, which will consider problems in graphs that are not necessarily regular, and problems where the validity of the output may depend on some given input.

Definition 2.2 (Node-edge-checkable problem). A *node-edge-checkable problem* Π is a tuple $(\Sigma_\Pi, \mathcal{N}_\Pi, \mathcal{E}_\Pi)$ where

1. Σ_Π is a finite set,
2. \mathcal{N}_Π is a set of cardinality- Δ multisets of elements from Σ_Π , and
3. \mathcal{E}_Π is a set of cardinality-2 multisets of elements from Σ_Π .

The set Σ_Π is called the (*output*) *label set* of Π and the elements of Σ_Π are called *output labels* (of Π). The set \mathcal{N}_Π is called the *node constraint* of Π and \mathcal{E}_Π is called the *edge constraint* of Π . A multiset of elements from Σ_Π is called a *configuration*; in particular, the elements of \mathcal{E}_Π and of \mathcal{N}_Π are configurations. A correct solution for Π on a Δ -regular graph is an assignment of labels from Σ_Π to the half-edges of the input graph such that, for each node v , the cardinality- Δ multiset of labels assigned to the half-edges incident to v is an element of \mathcal{N}_Π and, for each edge e , the cardinality-2 multisets of labels assigned to the half-edges belonging to e is an element of \mathcal{E}_Π . In the case of Δ -regular trees, the same requirements apply, but only on nodes of degree Δ , that is, leaves are unconstrained. When considering a distributed algorithm outputting a solution to some node-edge-checkable problem Π , we will assume that, for each half-edge $h = (v, e)$, the node deciding which output label is assigned to h is the incident node v . When considering a configuration that consists of the labels assigned to the half-edges incident to some node, resp. edge (or when considering a configuration in \mathcal{N}_Π , resp. in \mathcal{E}_Π), we may refer to it as a *node configuration*, resp. *edge configuration*.

We will write a configuration containing labels L_1, \dots, L_j as $L_1 \dots L_j$. Note that the contained labels do not have to be pairwise distinct, as a configuration is a multiset. Note further that, for the same reason, any displayed order of the labels represents the same configuration, e.g., $L_1 \dots L_j = L_j \dots L_1$.

Note that all the known lower bounds obtained via round elimination or via the Marks' technique hold on regular *trees*, and the lower bounds are already achieved on the “inner part” of the tree, i.e., which configurations are admissible on the leaf nodes does not affect the lower bounds (thereby ensuring that the presented standard definition is indeed all we need for our paper).

Condensed configurations. For writing down the configurations in a node or edge constraint, it is often convenient to use so-called condensed configurations, which provide a shorthand for capturing many configurations at once.

Definition 2.3 (Condensed configuration). Let Π be a node-edge-checkable problem with output label set Σ_Π . Formally, a *condensed configuration* is a configuration $S_1 \dots S_j$ (for some positive integer j) where each S_i is a collection of labels from Σ_Π . However, such a condensed configuration is to be interpreted as the collection of all configurations $L_1 \dots L_j$ of labels from Σ_Π such that

¹A regular tree is simply a tree where every node of degree $\neq 1$ has the same degree.

$L_i \in S_i$ for each $1 \leq i \leq j$. To support this interpretation notationally, we will write the sets S_i with square brackets, e.g., $[L_1 L_2] [L_1 L_3]$ represents the collection containing the configurations $L_1 L_1, L_1 L_2, L_1 L_3$, and $L_2 L_3$.

Example: sinkless orientation. We provide an example of a node-edge checkable problem. Consider the problem of orienting the edges of a 3-regular graph such that no node is a *sink*, i.e., no node has all edges oriented incoming. This problem is called sinkless orientation, and can be defined as a node-edge checkable problem $\Pi = (\Sigma_\Pi, \mathcal{N}_\Pi, \mathcal{E}_\Pi)$ as follows.

- $\Sigma_\Pi = \{I, O\}$, where I stands for *incoming* and O stands for *outgoing*.
- $\mathcal{E}_\Pi = \{I O\}$, that is, on the edges the only allowed configuration is $I O$. In other words, if a node claims that an edge is outgoing, then the other endpoint must claim that such an edge is incoming, and vice versa.
- \mathcal{N}_Π contains all the configurations given by the condensed configuration $[O] [I O] [I O]$, that is, $O I I$, $O O I$, and $O O O$. (Recall that the order of labels does not matter.) In other words, for each node, at least one incident half-edge must be labeled as outgoing.

Relaxations. In the following, we define the notion of a *relaxation* of a node-edge-checkable problem (introduced in [BO20] with a slightly different terminology). Intuitively, a relaxation of a problem Π is a problem $\hat{\Pi}$ such that $\hat{\Pi}$ can be solved in 0 rounds given a solution to Π . Formally, this can be expressed as follows, making use of the definition of a node-edge-checkable problem.

Definition 2.4 (Relaxation). Let $\Pi = (\Sigma_\Pi, \mathcal{N}_\Pi, \mathcal{E}_\Pi)$ and $\hat{\Pi} = (\Sigma_{\hat{\Pi}}, \mathcal{N}_{\hat{\Pi}}, \mathcal{E}_{\hat{\Pi}})$ be two node-edge-checkable problems. Let z denote the number of configurations in \mathcal{N}_Π and let $L_{i1} \dots L_{i\Delta}$, where $1 \leq i \leq z$, denote the z configurations. Then $\hat{\Pi}$ is called a *relaxation* of Π if there exists a mapping² $f: \{L_{ij} \mid 1 \leq i \leq z, 1 \leq j \leq \Delta\} \rightarrow \Sigma_{\hat{\Pi}}$ such that

1. for every configuration $L_{i1} \dots L_{i\Delta} \in \mathcal{N}_\Pi$, we have $f(L_{i1}) \dots f(L_{i\Delta}) \in \mathcal{N}_{\hat{\Pi}}$, and
2. for every configuration $L_{ij} L_{i'j'} \in \mathcal{E}_\Pi$, we have $f(L_{ij}) f(L_{i'j'}) \in \mathcal{E}_{\hat{\Pi}}$.

We call such a mapping f a *relaxation function from Π to $\hat{\Pi}$* . Moreover, if $\hat{\Pi}$ is a relaxation of Π , then we write $\Pi \xrightarrow{0} \hat{\Pi}$, symbolizing the fact that a solution to $\hat{\Pi}$ can be obtained from a solution to Π by a 0-round algorithm. If $\Pi \xrightarrow{0} \hat{\Pi}$ does not hold, we write $\Pi \not\xrightarrow{0} \hat{\Pi}$.

This definition captures the aforementioned intuition in the following sense: The only information that a node v executing a 0-round algorithm has about a solution to Π is the output labels of that solution on the half-edges incident to v . Hence, in order to infer from said solution a (local) solution to $\hat{\Pi}$ (without making use of any further assumptions about the input graph), node v must change each of the labels on its incident half-edges to an output label for $\hat{\Pi}$ in a way that guarantees that the new assignment satisfies the node and edge constraints of the new problem $\hat{\Pi}$. As, for each edge $e = \{u, v\}$ incident to v , the only guarantee that v has about the label assigned to half-edge (u, e) in the solution to Π is that together with the label assigned to half-edge (v, e) it forms a configuration contained in \mathcal{E}_Π , we arrive at the definition given above.

²We slightly abuse notation here by considering the L_{ij} simultaneously as two different objects: entries in configurations and labels from Σ_Π . In particular, please note that the function f may map the same label L from Σ_Π to two different labels in $\Sigma_{\hat{\Pi}}$ if L occurs in two different configurations of \mathcal{N}_Π (or if L occurs more than once in the same configuration).

A second kind of relaxation we will make use of in our work is what we call *port-local relaxations*. Differently from the regular relaxations defined above in which *occurrences of labels in node configurations* are mapped (i.e., where the mapping takes entire node configurations into account), a port-local relaxation is a direct mapping from the output label set of a problem to an output label set of a second problem. Formally, we define port-local relaxations as follows.

Definition 2.5 (Port-local relaxation). Let $\Pi = (\Sigma_\Pi, \mathcal{N}_\Pi, \mathcal{E}_\Pi)$ and $\hat{\Pi} = (\Sigma_{\hat{\Pi}}, \mathcal{N}_{\hat{\Pi}}, \mathcal{E}_{\hat{\Pi}})$ be two node-edge-checkable problems. Then we say that $\hat{\Pi}$ is a *port-local relaxation* of Π (and, equivalently, that Π is *port-locally relaxable* to $\hat{\Pi}$) if there exists a function $f : \Sigma_\Pi \rightarrow \Sigma_{\hat{\Pi}}$ such that

1. for each $L_1 \dots L_\Delta \in \mathcal{N}_\Pi$, we have $f(L_1) \dots f(L_\Delta) \in \mathcal{N}_{\hat{\Pi}}$, and
2. for each $L_1 L_2 \in \mathcal{E}_\Pi$, we have $f(L_1) f(L_2) \in \mathcal{E}_{\hat{\Pi}}$.

From Definitions 2.4 and 2.5, it follows that a port-local relaxation is always a relaxation.

Observation 2.6. Let Π and $\hat{\Pi}$ be two node-edge-checkable problems such that $\hat{\Pi}$ is a port-local relaxation of Π . Then $\Pi \xrightarrow{0} \hat{\Pi}$.

Moreover, from the definition of a relaxation it follows immediately that a relaxation of a relaxation of some problem Π is a relaxation of Π (and analogously for port-local relaxations), which we capture in the following observation.

Observation 2.7. For any three node-edge-checkable problems Π, Π', Π'' , it holds that if $\Pi \xrightarrow{0} \Pi'$ and $\Pi' \xrightarrow{0} \Pi''$, then also $\Pi \xrightarrow{0} \Pi''$. Moreover, if Π' is a port-local relaxation of Π and Π'' a port-local relaxation of Π' , then Π'' is a port-local relaxation of Π .

Equivalent problems. Based on the notion of a relaxation, we now formalize the concept of *equivalent* problems, which we will encounter in several places throughout the paper.

Definition 2.8 (Equivalent problems). We call two node-edge-checkable problems Π, Π' *equivalent* if Π is a relaxation of Π' and Π' is a relaxation of Π .

A simple case of equivalent problems that we will encounter frequently are two problems Π, Π' where one is obtained from the other by renaming labels. As, in this case, these two problems are *identical* for essentially all purposes, for simplicity we will treat problems obtained from each other via renaming of labels as *the same* problem and write $\Pi = \Pi'$.

Round elimination. The lower bounds we obtain in our work make use of the round elimination technique, which we introduce in the following. The round elimination framework is based on two functions $\mathcal{R}(\cdot)$ and $\overline{\mathcal{R}}(\cdot)$ that take a node-edge-checkable problem as input and return a node-edge-checkable problem. For the definition of the two functions, we need the notion of a *maximal* configuration of sets.

Definition 2.9 (Maximal and dominated configurations). Let k be a positive integer, Σ a finite set of sets and \mathcal{C} a collection of cardinality- k multisets with elements from Σ (i.e., each such multiset consists of k sets). Then a configuration $S_1 \dots S_k \in \mathcal{C}$ is called *maximal (in \mathcal{C})* if there is no configuration $S'_1 \dots S'_k \in \mathcal{C}$ such that there exists a permutation $\rho : \{1, \dots, k\} \rightarrow \{1, \dots, k\}$ satisfying 1) $S_i \subseteq S'_{\rho(i)}$ for all $1 \leq i \leq k$ and 2) $S_i \subsetneq S'_{\rho(i)}$ for at least one $1 \leq i \leq k$. Otherwise, we say that $S_1 \dots S_k$ is non-maximal and that it is *dominated* by $S'_1 \dots S'_k$.

Now we are set to define $\mathcal{R}(\cdot)$ and $\overline{\mathcal{R}}(\cdot)$. Let Π be a node-edge-checkable problem. The node-edge-checkable problem $\mathcal{R}(\Pi)$ is defined as follows.

Define Σ' as the set of all nonempty subsets of Σ_Π , i.e., $\Sigma' := 2^{\Sigma_\Pi} \setminus \{\emptyset\}$. We will call the elements of Σ' labels but keep in mind that, formally, they are sets. For defining the edge constraint $\mathcal{E}_{\mathcal{R}(\Pi)}$ of $\mathcal{R}(\Pi)$, first define \mathcal{C}' as the collection of all configurations $S_1 \dots S_\Delta$ consisting of labels from Σ' (i.e., consisting of sets of labels from Σ_Π) such that, for each pair $(L_1, L_2) \in S_1 \times S_2$, we have $L_1 L_2 \in \mathcal{E}_\Pi$. Then, $\mathcal{E}_{\mathcal{R}(\Pi)}$ is simply defined as the set of all maximal configurations in \mathcal{C}' .

Next, define the output label set $\Sigma_{\mathcal{R}(\Pi)}$ of $\mathcal{R}(\Pi)$ as the subset of Σ' containing precisely those labels that appear in at least one configuration in $\mathcal{E}_{\mathcal{R}(\Pi)}$. Finally, define the node constraint $\mathcal{N}_{\mathcal{R}(\Pi)}$ of $\mathcal{R}(\Pi)$ as the set of all configurations $S_1 \dots S_\Delta$ consisting of labels from $\Sigma_{\mathcal{R}(\Pi)}$ such that there exists a tuple $(L_1, \dots, L_\Delta) \in S_1 \times \dots \times S_\Delta$ satisfying $L_1 \dots L_\Delta \in \mathcal{N}_\Pi$. This concludes the definition of $\mathcal{R}(\Pi)$.

The node-edge-checkable problem $\overline{\mathcal{R}}(\Pi)$ is defined dually (where the application of the universal and existential quantifiers is reversed): Define Σ'' as the set of all nonempty subsets of Σ_Π , i.e., $\Sigma'' := 2^{\Sigma_\Pi} \setminus \{\emptyset\}$. For defining the node constraint $\mathcal{N}_{\overline{\mathcal{R}}(\Pi)}$ of $\overline{\mathcal{R}}(\Pi)$, first define \mathcal{C}'' as the collection of all configurations $S_1 \dots S_\Delta$ consisting of labels from Σ'' such that, for each tuple $(L_1, \dots, L_\Delta) \in S_1 \times \dots \times S_\Delta$, we have $L_1 \dots L_\Delta \in \mathcal{N}_\Pi$. Then, $\mathcal{N}_{\overline{\mathcal{R}}(\Pi)}$ is simply defined as the set of all maximal configurations in \mathcal{C}'' .

Next, define the output label set $\Sigma_{\overline{\mathcal{R}}(\Pi)}$ of $\overline{\mathcal{R}}(\Pi)$ as the subset of Σ'' containing precisely those labels that appear in at least one configuration in $\mathcal{N}_{\overline{\mathcal{R}}(\Pi)}$. Finally, define the edge constraint $\mathcal{E}_{\overline{\mathcal{R}}(\Pi)}$ of $\overline{\mathcal{R}}(\Pi)$ as the set of all configurations $S_1 S_2$ consisting of labels from $\Sigma_{\overline{\mathcal{R}}(\Pi)}$ such that there exists a pair $(L_1, L_2) \in S_1 \times S_2$ satisfying $L_1 L_2 \in \mathcal{E}_\Pi$. This concludes the definition of $\overline{\mathcal{R}}(\Pi)$.

We note that, while the function $\overline{\mathcal{R}}(\cdot)$ can be applied to any node-edge-checkable problem, in this work we will only apply it to node-edge-checkable problems obtained by the application of the function $\mathcal{R}(\cdot)$, i.e., to problems of the form $\mathcal{R}(\Pi)$. Furthermore, to simplify notation, we define $\mathcal{Q}(\cdot) := \overline{\mathcal{R}}(\mathcal{R}(\cdot))$.

Moreover, when computing $\mathcal{E}_{\mathcal{R}(\Pi)}$, resp. $\mathcal{N}_{\overline{\mathcal{R}}(\Pi)}$, we will refer to configurations in the above collections \mathcal{C}' , resp. \mathcal{C}'' , as configurations that *satisfy the universal quantifier*.

An easy way to compute $\mathcal{N}_{\mathcal{R}(\Pi)}$ and $\mathcal{E}_{\overline{\mathcal{R}}(\Pi)}$. It has already been noticed in previous works [Bra19] that there is an easy way to compute $\mathcal{N}_{\mathcal{R}(\Pi)}$ and $\mathcal{E}_{\overline{\mathcal{R}}(\Pi)}$:

Observation 2.10. *The constraint $\mathcal{N}_{\mathcal{R}(\Pi)}$ (resp. $\mathcal{E}_{\overline{\mathcal{R}}(\Pi)}$) can be computed as follows: for each configuration $L_1 \dots L_\Delta \in \mathcal{N}_\Pi$ (resp. $L_1 L_2 \in \mathcal{E}_\Pi$), add to the constraint $\mathcal{N}_{\mathcal{R}(\Pi)}$ (resp. $\mathcal{E}_{\overline{\mathcal{R}}(\Pi)}$) all the configurations that can be picked from the condensed configuration obtained by replacing each L_i with the set containing all the labels in $\Sigma_{\mathcal{R}(\Pi)}$ (resp. $\Sigma_{\overline{\mathcal{R}}(\Pi)}$) that are supersets of L_i .*

An easy way to compute $\mathcal{E}_{\mathcal{R}(\Pi)}$ and $\mathcal{N}_{\overline{\mathcal{R}}(\Pi)}$. The hard part of computing $\mathcal{R}(\Pi)$ and $\overline{\mathcal{R}}(\Pi)$ is computing the constraints $\mathcal{E}_{\mathcal{R}(\Pi)}$ and $\mathcal{N}_{\overline{\mathcal{R}}(\Pi)}$. A recent work developed an algorithm that makes it easier to compute such constraints [BBK⁺24]. Such an algorithm is based on an iterative application of a simple operation, that we now describe.

Let $\mathcal{C} = S_1 \dots S_k$ and $\mathcal{C}' = S'_1 \dots S'_k$ be two configurations of sets of labels. Let $1 \leq u \leq k$, and let $\sigma: \{1, \dots, k\} \rightarrow \{1, \dots, k\}$ be a permutation. The *combination of \mathcal{C} and \mathcal{C}' w.r.t. u and σ* is defined as the configuration $\mathcal{C}'' = S''_1 \dots S''_k$ satisfying the following:

- $S''_j = S_j \cap S'_{\sigma(j)}$ for each $j \in \{1, \dots, k\} \setminus \{u\}$, and
- $S''_u = S_u \cup S'_{\sigma(u)}$.

We now describe an algorithm \mathcal{A} that takes as input a constraint \mathcal{C} described as a collection of condensed configurations and returns a new constraint \mathcal{C}' of configurations. If \mathcal{C} is not given as a set of condensed configurations, it is straightforward to convert it into a set of condensed configurations, that is, by replacing each label with the singleton set containing that label.

The algorithm \mathcal{A} works as follows. At first, \mathcal{C}' is initialized as $\mathcal{C}' := \mathcal{C}$. Then, remove non-maximal configurations from \mathcal{C}' . Choose two arbitrary (possibly the same) configurations C_1 and C_2 from \mathcal{C}' , and combine them w.r.t some arbitrarily chosen u and σ . Add the obtained configuration to \mathcal{C}' if it is not dominated by some configuration already present in \mathcal{C}' , and remove non-maximal configurations from \mathcal{C}' . Repeat this operation for all possible choices of C_1 , C_2 , u , and σ , where C_1 and C_2 can also be configurations that have been added to \mathcal{C}' in the previous steps, until no new configuration is added to \mathcal{C}' .

Observation 2.11 (Theorem 4.1 of [BBK⁺24]). *The result of applying \mathcal{A} on \mathcal{E}_Π is $\mathcal{E}_{\mathcal{R}(\Pi)}$, and the result of applying \mathcal{A} on \mathcal{N}_Π is $\mathcal{N}_{\overline{\mathcal{R}}(\Pi)}$.*

We say that two sets S and S' are *comparable* if $S \subseteq S'$ or $S' \subseteq S$. Otherwise, we say that they are non-comparable. In [BBCR⁺25], the authors made the following observation.

Observation 2.12. *Let \mathcal{A}' be defined similarly as \mathcal{A} , with the only difference that two configurations $C_1 = S_1 \dots S_k$ and $C_2 = S'_1 \dots S'_k$ are combined w.r.t. u and σ only if S_u and $S'_{\sigma(u)}$ are non-comparable. Then, the output of \mathcal{A}' is the same as the one of \mathcal{A} .*

Obtaining lower bounds via round elimination. The way the round elimination framework (and in particular the two functions $\mathcal{R}(\cdot)$ and $\overline{\mathcal{R}}(\cdot)$) are used to obtain lower bounds is based on the round elimination theorem [Bra19], which, very informally speaking, states that, for any node-edge-checkable problem Π , the complexity of $\mathcal{Q}(\Pi)$ is precisely one round less than the complexity of Π (or 0 in case also Π is 0-round-solvable).³ Applying this theorem recursively to the sequence $\Pi, \mathcal{Q}(\Pi), \mathcal{Q}^2(\Pi), \dots$ of problems then yields that the complexity of $\mathcal{Q}^i(\Pi)$ is i rounds less than the complexity of Π . Now, to obtain a lower bound of k rounds for the complexity of Π , we merely need to show that problem $\mathcal{Q}^{k-1}(\Pi)$ cannot be solved in 0 rounds (which is much easier to check than j -round solvability for $j > 0$).

We emphasize that this overview is highly informal and hides a lot of details. In particular, those hidden details guarantee that if none of the problems in the sequence $\Pi, \mathcal{Q}(\Pi), \mathcal{Q}^2(\Pi), \dots$ is 0-round solvable, then one obtains a deterministic lower bound of $\Omega(\log n)$ rounds and a randomized lower bound of $\Omega(\log \log n)$ rounds.⁴

Fixed points. One beautiful scenario in which the above approach yields the desired lower bounds without much effort is that Π satisfies $\mathcal{Q}(\Pi) = \Pi$, which implies that the aforementioned sequence reduces to Π, Π, Π, \dots and that therefore none of the problems in the sequence can be solved in 0 rounds, assuming that Π cannot be solved in 0 rounds. A problem Π satisfying $\mathcal{Q}(\Pi) = \Pi$ is called a *fixed point*. We slightly extend the definition of a fixed point in a way that guarantees the same lower bounds but, for technical reasons, may capture more problems.

Definition 2.13 (Fixed point). A *fixed point* is a node-edge-checkable problem Π satisfying $\mathcal{Q}(\Pi) \xrightarrow{0} \Pi$. We call a fixed point *nontrivial* if it cannot be solved in 0 rounds, and trivial otherwise.

³This comes with a lot of fine print; for a modern overview containing all the technical details, see, e.g., [BBKO22, Appendix A, arxiv version] or Section 6.

⁴Again, some details are hidden here; for a formal statement, see [BBKO22, Theorem 7.1], or Theorem 6.4 for a more general version.

Moreover, for two node-edge-checkable problems $\Pi, \hat{\Pi}$, we call $\hat{\Pi}$ a *(trivial/nontrivial) fixed point relaxation* of Π if $\hat{\Pi}$ is a (trivial/nontrivial) fixed point and $\Pi \xrightarrow{0} \hat{\Pi}$.

If Π is a nontrivial fixed point, then by combining the above argumentation with the fact that Π can be solved in 0 rounds given a solution to $\mathcal{Q}(\Pi)$, we indeed again obtain the deterministic and randomized lower bounds of $\Omega(\log n)$ and $\Omega(\log \log n)$, respectively.

Inputs. As discussed in Section 1, one interesting aspect that our work explores are settings with inputs. In this setting, each half-edge is labeled with an input label from some input label set Σ_{in} and each node is aware of the input labels on its incident half-edges at the beginning of an algorithm.

An important distinction in the setting with inputs is between problems where the correctness of the output depends on the input and problems where the correctness of the output is independent of the input (and the input is merely additional information that the algorithm can use). Apart from Section 6, where we will develop a generic lifting theorem that will allow for problems of the former kind (to capture problems like list coloring), throughout the paper we will focus on the latter kind as it is sufficient for our purposes.

One nice property of round elimination is that *in the deterministic port numbering model, i.e., when nodes do not have access to unique identifiers or random bits*, round elimination is known to work also in the setting with input (of the latter kind) as long as the input (or, more specifically, the considered input-labeled graph class) satisfies some independence property [Bra19]. This property, informally speaking, can be phrased as follows: for any sufficiently small T and any T -hop neighborhood N of any node v , “the input that v would see when extending its view one hop beyond N in any fixed direction does not reveal any information about the input v would see when extending its view one hop in any other direction”.⁵ In particular, this independence property is satisfied when the input labels assigned to the half-edges of the input graph constitute a correct solution to some fixed node-edge-checkable problem. As such, we will allow precisely those inputs: when studying the setting with inputs, a fixed node-edge-checkable problem \mathcal{I} specifies which input configurations can appear, namely, precisely those that satisfy the constraints of problem \mathcal{I} . In other words, the input graph comes with an arbitrary solution to \mathcal{I} , and this solution can be used by the nodes while performing their communication and computation to solve some given problem Π of interest.

As mentioned above, in the setting with such inputs, round elimination is only known to work in the port numbering model. In Section 6, we show that round elimination works in the input setting also in the LOCAL model.

To conclude the discussion of inputs, let us clarify what we mean when we say that round elimination “works” in the setting with inputs: the definition and use of the functions $\mathcal{R}(\cdot)$ and $\overline{\mathcal{R}}(\cdot)$ stays precisely the same; the only difference that the input makes is that a problem is considered to be 0-round solvable if it can be solved in 0 rounds *given the input*, i.e., in a potential 0-round algorithm, each node v may make use of the information about the input on its incident half-edges (which is all that v knows about the input).

Relaxations using inputs. A natural idea to make use of fixed points to obtain lower bounds is via relaxations: finding, for a given problem Π , a nontrivial fixed point \mathcal{F} satisfying $\Pi \xrightarrow{0} \mathcal{F}$ directly transfers the aforementioned $\Omega(\log n)$ - and $\Omega(\log \log n)$ -round lower bounds from \mathcal{F} to Π . Now, based on our result in Section 6 that round elimination works in the setting with inputs, we

⁵For the formal definition of this independence property, see the definition of *t-independence* in [Bra19].

can extend this idea to the setting with inputs: in particular, we obtain these lower bounds already whenever we can find just a single node-edge-checkable “input” problem \mathcal{I} such that

1. we can relax Π to some fixed point \mathcal{F} given (a solution to) \mathcal{I} as input, and
2. \mathcal{F} is nontrivial given input \mathcal{I} , i.e., $\mathcal{I} \xrightarrow{0} \mathcal{F}$.

Formally, for defining what we mean by a relaxation given some input, we introduce the notion of the product of two problems.

Definition 2.14 (Product of problems). Let Π, Π' be two node-edge-checkable problems. Then the *product* $\Pi \times \Pi' = (\Sigma_{\Pi \times \Pi'}, \mathcal{N}_{\Pi \times \Pi'}, \mathcal{E}_{\Pi \times \Pi'})$ of Π and Π' is defined by setting

$$\begin{aligned}\Sigma_{\Pi \times \Pi'} &:= \Sigma_{\Pi} \times \Sigma_{\Pi'}, \\ \mathcal{N}_{\Pi \times \Pi'} &:= \{(L_1, L'_1) \dots (L_{\Delta}, L'_{\Delta}) \mid L_1 \dots L_{\Delta} \in \mathcal{N}_{\Pi} \text{ and } L'_1 \dots L'_{\Delta} \in \mathcal{N}_{\Pi'}\}, \text{ and} \\ \mathcal{E}_{\Pi \times \Pi'} &:= \{(L_1, L'_1) (L_2, L'_2) \mid L_1 L_2 \in \mathcal{E}_{\Pi} \text{ and } L'_1 L'_2 \in \mathcal{E}_{\Pi'}\}.\end{aligned}$$

Now, we can formally restate point 1 from above as $\Pi \times \mathcal{I} \xrightarrow{0} \mathcal{F}$.

Similarly as for relaxations, in the setting with inputs, we can also strengthen the notion of a fixed point (in the sense that it captures more problems while providing the same complexity implications) by making use of the available input.

Definition 2.15 (Generalized fixed point). Assume that we are in the setting with inputs, where the input is given by some node-edge-checkable problem \mathcal{I} . Then we call a node-edge-checkable problem Π a *generalized fixed point* if $\mathcal{Q}(\Pi) \times \mathcal{I} \xrightarrow{0} \Pi$. Moreover, we call a generalized fixed point Π *nontrivial* if Π cannot be solved in 0 rounds given \mathcal{I} , i.e., if $\mathcal{I} \not\xrightarrow{0} \Pi$.

Given the definition of a generalized fixed point and using the insights from the above discussion, we can now formulate a strong version of the lower bound guarantees provided by fixed point relaxations in the setting with input \mathcal{I} : if \mathcal{F} is a nontrivial generalized fixed point and Π a node-edge-checkable problem satisfying $\Pi \times \mathcal{I} \xrightarrow{0} \mathcal{F}$, then Π requires $\Omega(\log n)$ rounds to be solved deterministically and $\Omega(\log \log n)$ rounds to be solved randomized. Note that the fact that these lower bounds hold in the setting with input also implies that the lower bounds hold in the (harder) setting without input.

3 The tripotent input

In this section we will investigate the technique of proving lower bounds for node-edge-checkable problems via fixed point relaxations with input. Suppose we are given a node-edge checkable problem Π for which we want to prove hardness. As we explained in Section 2, the hardness follows if we can find a nontrivial fixed point \mathcal{F} and an input \mathcal{I} (which are both node-edge-checkable problems themselves) such that

$$\Pi \times \mathcal{I} \xrightarrow{0} \mathcal{F}, \quad \mathcal{I} \not\xrightarrow{0} \mathcal{F}.$$

The first condition means that Π can be relaxed to \mathcal{F} in the setting with input \mathcal{I} , the second ensures that \mathcal{F} is nontrivial even given a solution to \mathcal{I} . For the moment, suppose that we already chose the fixed point \mathcal{F} we want to use. We are then left to find an input \mathcal{I} ensuring the above properties, where we focus on the first property $\Pi \times \mathcal{I} \xrightarrow{0} \mathcal{F}$. In the following, we provide a generic construction of such a suitable input problem. More than that, we will show that our construction in fact leads to *the easiest* node-edge-checkable problem \mathcal{I} satisfying this property. By “easiest”, we mean that any other input problem \mathcal{J} allowing a relaxation from Π to \mathcal{F} can itself be relaxed

to \mathcal{I} . Note that the existence of such an easiest input is highly nontrivial, but at the same time optimal for our purposes: since we want to also ensure the second property $\mathcal{I} \xrightarrow{0} \mathcal{F}$, the input \mathcal{I} should contain as little information as possible, i.e., be as easy as possible.

Later in this section we connect this new construction to the existing round elimination framework. Rather surprisingly, we will prove that under some conditions, the constructed inputs themselves are again fixed points. More than that, we will encounter cases in which we can find nontrivial fixed point relaxations (without input) for problems of interest using this construction.

3.1 Constructing an input problem

Suppose we are given two node-edge-checkable problems, Π and P , where Π is a problem of interest and P is some problem we chose in order to prove a lower bound on Π . One can think of P as being a fixed point, however for now this is not an assumption we make. Before we can define our generic input problem $\tau_P(\Pi)$, we need to introduce another problem, $\mathcal{R}^*(P)$, which is pretty similar to $\mathcal{R}(P)$. The only difference in the definition is that we also allow non-maximal edge configurations, Σ and \mathcal{N} stay the same. Formally, $\mathcal{R}^*(\cdot)$ is defined as follows:

$$\begin{aligned}\Sigma_{\mathcal{R}^*(P)} &:= 2^{\Sigma_P} \setminus \{\emptyset\}, \\ \mathcal{N}_{\mathcal{R}^*(P)} &:= \{S_1 \dots S_\Delta \mid \exists L_1 \in S_1, \dots, L_\Delta \in S_\Delta: L_1 \dots L_\Delta \in \mathcal{N}_P\}, \\ \mathcal{E}_{\mathcal{R}^*(P)} &:= \{S_1 S_2 \mid \forall L_1 \in S_1, L_2 \in S_2: L_1 L_2 \in \mathcal{E}_P\}.\end{aligned}$$

Although the definitions are similar, the reason we need $\mathcal{R}^*(P)$ has nothing to do with round elimination. Instead, we will later make use of its definition in Lemma 3.5, stating that $\mathcal{R}^*(P)$ behaves nicely w.r.t. port local relaxations.

We are now ready to define our generic input:

Definition 3.1. Let Π and P be node-edge-checkable problems. The *tripotent input* $\tau_P(\Pi)$ of Π w.r.t. P is defined as follows:

$$\begin{aligned}\Sigma_{\tau_P(\Pi)} &:= \{f \mid f: \Sigma_\Pi \rightarrow \Sigma_{\mathcal{R}^*(P)}\}, \\ \mathcal{N}_{\tau_P(\Pi)} &:= \{f_1 \dots f_\Delta \mid \forall L_1 \dots L_\Delta \in \mathcal{N}_\Pi: f_1(L_1) \dots f_\Delta(L_\Delta) \in \mathcal{N}_{\mathcal{R}^*(P)}\}, \\ \mathcal{E}_{\tau_P(\Pi)} &:= \{f_1 f_2 \mid \forall L_1 L_2 \in \mathcal{E}_\Pi: f_1(L_1) f_2(L_2) \in \mathcal{E}_{\mathcal{R}^*(P)}\}.\end{aligned}$$

We call τ the *tripotent input generator*.

The intuition behind this definition is that it provides an easy relaxation $\Pi \times \tau_P(\Pi) \xrightarrow{0} \mathcal{R}^*(P)$, meaning that Π can be relaxed to $\mathcal{R}^*(P)$ given $\tau_P(\Pi)$ as input. A half-edge that is assigned labels L and f by Π and $\tau_P(\Pi)$ respectively can choose its new label as $f(L) \in \Sigma_{\mathcal{R}^*(P)}$. The node and edge constraints of $\tau_P(\Pi)$ ensure that this indeed leads to a valid solution of $\mathcal{R}^*(P)$. In fact, we prove the following stronger statement, proving that $\tau_P(\Pi)$ can be chosen as input when one wants to solve P (instead of $\mathcal{R}^*(P)$) given a solution to Π .

Lemma 3.2. Two node-edge-checkable problems P and Π always satisfy $\Pi \times \tau_P(\Pi) \xrightarrow{0} P$.

Proof. We will prove $\Pi \times \tau_P(\Pi) \xrightarrow{0} \mathcal{R}^*(P) \xrightarrow{0} P$, our claim $\Pi \times \tau_P(\Pi) \xrightarrow{0} P$ then follows by Observation 2.7.

We start with the relaxation $\Pi \times \tau_P(\Pi) \xrightarrow{0} \mathcal{R}^*(P)$, which we perform port-locally as described above: each node replaces all its labels $(L, f) \in \Sigma_{\Pi \times \tau_P(\Pi)}$ by $f(L) \in \Sigma_{\mathcal{R}^*(P)}$. We need to prove that this is indeed a (port-local) relaxation, therefore consider an edge that was assigned labels

$(L_1, f_1) (L_2, f_2) \in \mathcal{E}_{\Pi \times \tau_P(\Pi)}$. By the definition of the product of two problems, this implies $L_1 L_2 \in \mathcal{E}_\Pi$ and $f_1 f_2 \in \mathcal{E}_{\tau_P(\Pi)}$. Definition 3.1 now immediately yields $f_1(L_1) f_2(L_2) \in \mathcal{E}_{\mathcal{R}^*(P)}$ which is what we needed to prove. The definition of \mathcal{N}_τ is analogous to \mathcal{E}_τ , thus one can use the exact same argument to conclude that $(L_1, f_1) \dots (L_\Delta, f_\Delta) \in \mathcal{N}_{\Pi \times \tau_P(\Pi)}$ implies $f_1(L_1) \dots f_\Delta(L_\Delta) \in \mathcal{N}_{\mathcal{R}^*(P)}$. This proves the above relaxation to be correct.

It is left to provide the second relaxation $\mathcal{R}^*(P) \xrightarrow{0} P$. Therefore, assume that a node is assigned labels $S_1 \dots S_\Delta$ by $\mathcal{R}^*(\Pi)$. By definition of $\mathcal{N}_{\mathcal{R}^*(\Pi)}$ that node can pick $L_1 \in S_1, \dots, L_\Delta \in S_\Delta$ such that $L_1 \dots L_\Delta \in \mathcal{N}_\Pi$. If each node proceeds like this, the definition of $\mathcal{E}_{\mathcal{R}^*(\Pi)}$ ensures that we end up with a solution of Π . \square

As we already mentioned, we will now prove that $\tau_P(\Pi)$ is not only a “valid input” for Π in the sense of Lemma 3.2 but is in fact the easiest such problem. Formally, we will prove that each input \mathcal{I} allowing a relaxation from Π to P can itself be relaxed to $\tau_P(\Pi)$, as stated in the following theorem:

Theorem 3.3. *Let Π, P and \mathcal{I} be node-edge-checkable problems. Then $\Pi \times \mathcal{I} \xrightarrow{0} P$ holds if and only if $\mathcal{I} \xrightarrow{0} \tau_P(\Pi)$.*

To prove Theorem 3.3, we first need to introduce the notion of a problem property related to port-local relaxations.

Definition 3.4. We say that a node-edge-checkable problem P *allows port-local relaxations* if for all node-edge-checkable problems Π satisfying $\Pi \xrightarrow{0} P$, Π is also port-locally relaxable to P .

Recall that port-local relaxations are a restriction of general relaxations. Hence it may seem weird to introduce the “allowing port-local relaxations”-property for node-edge-checkable problems we want to relax to. However, the following lemma shows that the $\mathcal{R}^*(\cdot)$ -operator constructs problems with exactly this property, which is why we needed $\mathcal{R}^*(\cdot)$ in the first place:

Lemma 3.5. *For any node-edge-checkable problem P , the problem $\mathcal{R}^*(P)$ allows port-local relaxations.*

Proof. Let Π be some node-edge checkable problem that can be relaxed to $\mathcal{R}^*(P)$, i.e., satisfying $\Pi \xrightarrow{0} \mathcal{R}^*(P)$. According to Definition 3.4, we need to prove that there exists a port-local relaxation from Π to $\mathcal{R}^*(P)$. Let f be the given relaxation $\Pi \xrightarrow{0} \mathcal{R}^*(P)$. f maps an $L \in \Sigma_\Pi$ to one of finitely many $S_1, \dots, S_k \in \Sigma_{\mathcal{R}^*(P)}$, depending on the other labels that node is assigned. Recalling that the S_1, \dots, S_k are actually sets of labels of P , we can define $g(l) = \bigcup_{i=1}^k S_i$. This can be done for every $L \in \Sigma_\Pi$ (where k of course depends on L), which yields a function $g: \Sigma_\Pi \rightarrow \Sigma_{\mathcal{R}^*(P)}$. We claim that g is a port-local relaxation in the sense of Definition 2.5.

For the first condition, we choose an arbitrary $L_1 \dots L_\Delta \in \mathcal{N}_\Pi$ which is mapped to $S_1 \dots S_\Delta$ by f . Since f is a correct relaxation, we know $S_1 \dots S_\Delta \in \mathcal{N}_{\mathcal{R}^*(P)}$. According to the definition of $\mathcal{N}_{\mathcal{R}^*(P)}$ this means that there exist $L_1 \in S_1, \dots, L_\Delta \in S_\Delta$ such that $L_1 \dots L_\Delta \in \mathcal{N}_P$. Note that by definition of g , we must have $S_i \subseteq g(L_i)$ and thus $L_i \in g(L_i)$ for all $i = 1, \dots, \Delta$. This implies $g(L_1) \dots g(L_\Delta) \in \mathcal{N}_{\mathcal{R}^*(P)}$.

For the second condition, we choose an arbitrary edge that is assigned labels L and L' by P . Let $S_1, \dots, S_k \in \Sigma_{\mathcal{R}^*(P)}$ be the sets f might map L to. Similarly, we define $S'_1, \dots, S'_{k'}$ for L' . Since g always maps L and L' to a valid edge configuration of $\mathcal{R}^*(P)$, we can apply the definition of $\mathcal{E}_{\mathcal{R}^*(P)}$ to all pairs $S_i, S'_{i'}$ where $1 \leq i \leq k$ and $1 \leq i' \leq k'$. This means we have $s s' \in \mathcal{E}_P$ for all $s \in S_i, s' \in S'_{i'}$. Since this holds for all pairs i, i' we also get $s s' \in \mathcal{E}_P$ for all $s \in g(L), s' \in g(L')$. Thus we have $g(L) g(L') \in \mathcal{E}_{\mathcal{R}^*(P)}$ which finishes the proof. \square

With this we are ready to prove Theorem 3.3.

Proof of Theorem 3.3. If we suppose $\mathcal{I} \xrightarrow{0} \tau_P(\Pi)$, we clearly have

$$\Pi \times \mathcal{I} \xrightarrow{0} \Pi \times \tau_P(\Pi) \xrightarrow{0} P,$$

where the last step follows from Lemma 3.2. We can combine these relaxations as explained in Observation 2.7 to obtain $\Pi \times \mathcal{I} \xrightarrow{0} P$.

For the other direction we are given a relaxation from $\Pi \times \mathcal{I}$ to P . Further, P is (port-locally) relaxable to $\mathcal{R}^*(P)$ by just mapping every label $l \in \Sigma_\Pi$ to $\{l\} \in \Sigma_{\mathcal{R}^*(\Pi)}$. Again using Observation 2.7, we can combine these relaxations $\Pi \times \mathcal{I} \xrightarrow{0} P$ and $P \xrightarrow{0} \mathcal{R}^*(P)$ to obtain $\Pi \times \mathcal{I} \xrightarrow{0} \mathcal{R}^*(P)$. Using Lemma 3.5, $\Pi \times \mathcal{I}$ must even be port-locally relaxable to $\mathcal{R}^*(P)$. Unpacking Definition 2.5 this means there exists an $f: \Sigma_{\Pi \times \mathcal{I}} \rightarrow \Sigma_{\mathcal{R}^*(P)}$ such that

$$\begin{aligned} \mathbf{L}_1 \dots \mathbf{L}_\Delta \in \mathcal{N}_\Pi, i_1 \dots i_\Delta \in \mathcal{N}_\mathcal{I} &\implies f(\mathbf{L}_1, i_1) \dots f(\mathbf{L}_\Delta, i_\Delta) \in \mathcal{N}_{\mathcal{R}^*(P)} \quad \text{and} \\ \mathbf{L}_1 \mathbf{L}_2 \in \mathcal{E}_\Pi, i_1 i_2 \in \mathcal{E}_\mathcal{I} &\implies f(\mathbf{L}_1, i_1) f(\mathbf{L}_2, i_2) \in \mathcal{E}_{\mathcal{R}^*(P)}. \end{aligned}$$

This allows us to define a port-local relaxation from \mathcal{I} to $\tau_P(\Pi)$ as follows:

$$\begin{aligned} g: \Sigma_\mathcal{I} &\rightarrow \Sigma_{\tau_P(\Pi)}, \\ i &\mapsto S(\cdot, i). \end{aligned}$$

We need to prove that g is indeed a port-local relaxation. Observe that for each $i_1 \dots i_\Delta \in \mathcal{N}_\mathcal{I}$ and $\mathbf{L}_1 \dots \mathbf{L}_\Delta \in \mathcal{N}_\Pi$ we have

$$g(i_1)(\mathbf{L}_1) \dots g(i_\Delta)(\mathbf{L}_\Delta) = f(\mathbf{L}_1, i_1) \dots f(\mathbf{L}_\Delta, i_\Delta) \in \mathcal{N}_{\mathcal{R}^*(P)}.$$

Thus by definition of $\tau_P(\Pi)$ for each $i_1 \dots i_\Delta \in \mathcal{N}_\mathcal{I}$ we have

$$g(i_1) \dots g(i_\Delta) \in \mathcal{N}_{\tau_P(\Pi)}.$$

The latter ensures that g maps each of \mathcal{I} 's valid node configurations to a valid node configuration of $\tau_P(\Pi)$. \mathcal{E} behaves exactly the same, proving that g is a port local relaxation from \mathcal{I} to $\tau_P(\Pi)$. \square

Note that the above proof actually constructs the (port-local) relaxation $\mathcal{I} \xrightarrow{0} \tau_P(\Pi)$, so whenever we use Theorem 3.3 to infer some relaxation of this type, the proof tells us how to perform that relaxation.

As a direct consequence, we can use Theorem 3.3 to see what happens if one repeatedly applies $\tau_P(\cdot)$ to a problem. We prove that $\tau_P(\tau_P(\Pi))$ is always a relaxation of Π and that τ_P is indeed tripotent, justifying its name.

Corollary 3.6. *For two node-edge-checkable problems Π and P the following statements hold:*

1. $\Pi \xrightarrow{0} \tau_P(\tau_P(\Pi))$
2. $\tau_P(\Pi)$ and $\tau_P(\tau_P(\tau_P(\Pi)))$ are equivalent.

Proof. We start with the first statement. By Lemma 3.2 we have $\tau_P(\Pi) \times \Pi \xrightarrow{0} P$. This allows us to apply Theorem 3.3, where Π is the input problem \mathcal{I} and $\tau_P(\Pi)$ takes the role of Π . That yields $\Pi \xrightarrow{0} \tau_P(\tau_P(\Pi))$ as claimed.

For the second statement, $\tau_P(\Pi) \xrightarrow{0} \tau_P(\tau_P(\tau_P(\Pi)))$ follows immediately from the first. The other direction's proof is quite similar. Observe that using the first statement, we get:

$$\Pi \times \tau_P(\tau_P(\tau_P(\Pi))) \xrightarrow{0} \tau_P(\tau_P(\Pi)) \times \tau_P(\tau_P(\tau_P(\Pi))) \xrightarrow{0} P$$

Here the latter step follows from Lemma 3.2, and we obtain $\Pi \times \tau_P(\tau_P(\tau_P(\Pi))) \xrightarrow{0} P$ by Observation 2.7. We can now apply Theorem 3.3 directly to $\Pi \times \tau_P(\tau_P(\tau_P(\Pi))) \xrightarrow{0} P$, where $\tau_P(\tau_P(\tau_P(\Pi)))$ takes the role of \mathcal{I} , in order to get $\tau_P(\tau_P(\tau_P(\Pi))) \xrightarrow{0} \tau_P(\Pi)$. This is everything we needed. \square

3.2 Inputs in the RE framework

So far we introduced the notion of a generic input $\tau_P(\Pi)$ for two node-edge-checkable problems Π and P and proved a couple of nice results coming solely from the definition of τ . However, for proving lower bounds, we also rely on the round elimination technique, which is not yet really connected to τ . Broadly speaking, the goal of this section is to prove statements about τ 's behavior in the round elimination framework.

For this purpose, we first need a lemma showing how relaxations behave w.r.t. round elimination. To this end, we need the notion of an *edge-based relaxation*, which can be seen as the dual to the standard relaxation notion where the role of nodes and edges has been reversed.

Definition 3.7 (Edge-based relaxation). Let $\Pi = (\Sigma_\Pi, \mathcal{N}_\Pi, \mathcal{E}_\Pi)$ and $\hat{\Pi} = (\Sigma_{\hat{\Pi}}, \mathcal{N}_{\hat{\Pi}}, \mathcal{E}_{\hat{\Pi}})$ be two node-edge-checkable problems. Let z denote the number of configurations in \mathcal{E}_Π and let $\mathbf{L}_{i1} \dots \mathbf{L}_{i2}$, where $1 \leq i \leq z$, denote the z configurations. Then $\hat{\Pi}$ is called an *edge-based relaxation* of Π if there exists a mapping $f: \{\mathbf{L}_{ij} \mid 1 \leq i \leq z, 1 \leq j \leq 2\} \rightarrow \Sigma_{\hat{\Pi}}$ such that

1. for every configuration $\mathbf{L}_{i1} \mathbf{L}_{i2} \in \mathcal{E}_\Pi$, we have $f(\mathbf{L}_{i1}) f(\mathbf{L}_{i2}) \in \mathcal{E}_{\hat{\Pi}}$, and
2. for every configuration $\mathbf{L}_{i_1 j_1} \dots \mathbf{L}_{i_\Delta j_\Delta} \in \mathcal{N}_\Pi$, we have $f(\mathbf{L}_{i_1 j_1}) \dots f(\mathbf{L}_{i_\Delta j_\Delta}) \in \mathcal{N}_{\hat{\Pi}}$.

We call such a mapping f an *edge-based relaxation function* from Π to $\hat{\Pi}$.

Now we are ready to prove the aforementioned statement, which shows, informally speaking, that the two notions of relaxing a problem and applying the function $\overline{\mathcal{R}}(\mathcal{R}(\cdot))$ commute.

Lemma 3.8 (RE commutativity). *Let Π and $\hat{\Pi}$ be two node-edge-checkable problems. Then all of these hold:*

1. *If $\hat{\Pi}$ is a relaxation of Π , then there exists an edge-based relaxation from $\mathcal{R}(\Pi)$ to $\mathcal{R}(\hat{\Pi})$.*
2. *If $\hat{\Pi}$ is an edge-based relaxation of Π , then there is a relaxation from $\overline{\mathcal{R}}(\Pi)$ to $\overline{\mathcal{R}}(\hat{\Pi})$.*
3. *If $\hat{\Pi}$ is a relaxation of Π , then $\mathcal{Q}^i(\hat{\Pi})$ is a relaxation of $\mathcal{Q}^i(\Pi)$ for all integers i .*

Proof. We start with Statement 1, therefore let f be a relaxation function from Π to $\hat{\Pi}$. Consider an arbitrary edge configuration $\mathbf{S}_1 \mathbf{S}_2 \in \mathcal{E}_{\mathcal{R}(\Pi)}$. Observe that $\mathbf{S}_1, \mathbf{S}_2$ are sets of labels from Σ_Π . Now, for each $j \in \{1, 2\}$, do the following.

Let $\mathbf{L}_{j1}, \dots, \mathbf{L}_{jk_j}$ denote the labels contained in \mathbf{S}_j . For each $1 \leq x \leq k_j$, the label \mathbf{L}_{jx} may occur in different configurations from \mathcal{N}_Π ; let \mathcal{L}_{jx} denote the set of labels that these different occurrences of \mathbf{L}_{jx} are mapped to by f . Set further $\mathcal{S}'_j := \bigcup_{1 \leq x \leq k_j} \mathcal{L}_{jx}$.

Observe that, by the definition of $\mathcal{R}(\cdot)$, the edge configuration $\mathbf{L}_{1x_1} \mathbf{L}_{2x_2}$ is contained in \mathcal{E}_Π for each $1 \leq x_1 \leq k_1$ and $1 \leq x_2 \leq k_2$. By the fact that f is a relaxation and the definition of \mathcal{S}'_j , it follows that $\mathbf{L}'_1 \mathbf{L}'_2 \in \mathcal{E}_{\hat{\Pi}}$ for each $\mathbf{L}'_1 \in \mathcal{S}'_1$ and $\mathbf{L}'_2 \in \mathcal{S}'_2$. This implies, again by the definition of $\mathcal{R}(\cdot)$, that there is an edge configuration $\hat{\mathbf{S}}_1 \hat{\mathbf{S}}_2 \in \mathcal{E}_{\mathcal{R}(\hat{\Pi})}$ such that $\mathcal{S}'_1 \subseteq \hat{\mathbf{S}}_1$ and $\mathcal{S}'_2 \subseteq \hat{\mathbf{S}}_2$.

Now, for the considered edge configuration $\mathbf{S}_1 \mathbf{S}_2 \in \mathcal{E}_{\mathcal{R}(\Pi)}$, set $g(\mathbf{S}_1) := \hat{\mathbf{S}}_1$ and $g(\mathbf{S}_2) := \hat{\mathbf{S}}_2$. As the considered edge configuration was chosen arbitrarily from $\mathcal{E}_{\mathcal{R}(\Pi)}$, this concludes the definition of g . From the construction of g , we immediately obtain that Condition 1 of Definition 3.7 is satisfied. In the following, we show that g also satisfies Condition 2.

Consider an arbitrary node configuration $\mathbf{S}_1 \dots \mathbf{S}_\Delta \in \mathcal{N}_{\mathcal{R}(\Pi)}$ and let $\hat{\mathbf{S}}_1, \dots, \hat{\mathbf{S}}_\Delta$ be labels that $\mathbf{S}_1, \dots, \mathbf{S}_\Delta$, respectively, are mapped to by g . By the definition of $\mathcal{R}(\cdot)$, we know that there exists some configuration $\mathbf{L}_1 \dots \mathbf{L}_\Delta \in \mathcal{N}_\Pi$ such that $\mathbf{L}_y \in \mathbf{S}_y$ for each $1 \leq y \leq \Delta$. Let $\mathbf{L}'_1, \dots, \mathbf{L}'_\Delta$, respectively, be the labels that the labels $\mathbf{L}_1, \dots, \mathbf{L}_\Delta$ of the configuration $\mathbf{L}_1 \dots \mathbf{L}_\Delta$ are mapped to by f . By the definition of f , this in particular implies that $\mathbf{L}'_1 \dots \mathbf{L}'_\Delta \in \mathcal{N}_{\hat{\Pi}}$. By the construction

of g , we know that there exist subsets S'_1, \dots, S'_Δ of $\hat{S}_1, \dots, \hat{S}_\Delta$, respectively, such that $L'_y \in S'_y$ for each $1 \leq y \leq \Delta$. By the definition of $\mathcal{R}(\cdot)$, it follows that $\hat{S}_1 \dots \hat{S}_\Delta \in \mathcal{N}_{\mathcal{R}(\hat{\Pi})}$, as desired. We conclude that g also satisfies Condition 2 of Definition 3.7. Hence, g is indeed an edge-based relaxation function, proving our first statement.

Statement 2 is dual to statement 1, where the roles of nodes and edges are swapped. Here we assume that g is an edge-based relaxation function from Π to $\hat{\Pi}$.

Consider an arbitrary node configuration $Z_1 \dots Z_\Delta \in \mathcal{N}_{\overline{\mathcal{R}}(\Pi)}$. For each $y \in \{1, \dots, \Delta\}$, do the following.

Let S_{y1}, \dots, S_{yk_y} denote the labels contained in Z_y . For each $1 \leq x \leq k_y$, the label S_{yx} may occur in different configurations from \mathcal{E}_Π ; let S_{yx} denote the set of labels that these different occurrences of S_{yx} are mapped to by g . Set further $Z'_y := \bigcup_{1 \leq x \leq k_y} S_{yx}$.

Observe that, by the definition of $\overline{\mathcal{R}}(\cdot)$, the edge configuration $S_{1x_1} \dots S_{\Delta x_\Delta}$ is contained in \mathcal{N}_Π for each $(x_1, \dots, x_\Delta) \in \{1, \dots, k_1\} \times \dots \times \{1, \dots, k_\Delta\}$. By the fact that g is an edge-based relaxation and the definition of Z'_y , it follows that $S'_1 \dots S'_\Delta \in \mathcal{N}_{\hat{\Pi}}$ for each $(S'_1, \dots, S'_\Delta) \in Z'_1 \times \dots \times Z'_\Delta$. This implies, again by the definition of $\overline{\mathcal{R}}(\cdot)$, that there is a node configuration $\hat{Z}_1 \dots \hat{Z}_\Delta \in \mathcal{N}_{\overline{\mathcal{R}}(\hat{\Pi})}$ such that Z'_1, \dots, Z'_Δ are subsets of $\hat{Z}_1, \dots, \hat{Z}_\Delta$, respectively.

Now, for the considered node configuration $Z_1 \dots Z_\Delta \in \mathcal{N}_{\overline{\mathcal{R}}(\Pi)}$, set $h(Z_y) := \hat{Z}_y$ for each $1 \leq y \leq \Delta$. As the considered node configuration was chosen arbitrarily from $\mathcal{N}_{\overline{\mathcal{R}}(\Pi)}$, this concludes the definition of h . From the construction of h , we immediately obtain that Condition 1 of Definition 2.4 is satisfied. In the following, we show that h also satisfies Condition 2.

Consider an arbitrary edge configuration $Z_1 Z_2 \in \mathcal{E}_{\overline{\mathcal{R}}(\Pi)}$ and let \hat{Z}_1, \hat{Z}_2 be labels that Z_1, Z_2 , respectively, are mapped to by h . By the definition of $\overline{\mathcal{R}}(\cdot)$, we know that there exists some configuration $S_1 S_2 \in \mathcal{E}_\Pi$ such that $S_j \in Z_j$ for each $1 \leq j \leq 2$. Let S'_1, S'_2 , respectively, be the labels that the labels S_1, S_2 of the configuration $S_1 S_2$ are mapped to by g . By the definition of g , this in particular implies that $S'_1 S'_2 \in \mathcal{E}_{\hat{\Pi}}$. By the construction of h , we know that there exist subsets Z'_1, Z'_2 of \hat{Z}_1, \hat{Z}_2 , respectively, such that $S'_j \in Z'_j$ for each $1 \leq j \leq 2$. By the definition of $\overline{\mathcal{R}}(\cdot)$, it follows that $\hat{Z}_1 \hat{Z}_2 \in \mathcal{E}_{\overline{\mathcal{R}}(\hat{\Pi})}$, as desired. We conclude that h also satisfies Condition 2 of Definition 2.4. Hence, h is indeed a relaxation function, which concludes the proof of the second part.

For Statement 3, it suffices to prove the statement for $i = 1$, it then holds for all i by induction. If Π is relaxable to $\hat{\Pi}$, then we know by part 1 that there exists an edge-based relaxation from $\mathcal{R}(\Pi)$ to $\mathcal{R}(\hat{\Pi})$. This allows to apply part 2, proving that $\overline{\mathcal{R}}(\mathcal{R}(\Pi)) = \mathcal{Q}(\Pi)$ is relaxable to $\overline{\mathcal{R}}(\mathcal{R}(\hat{\Pi})) = \mathcal{Q}(\hat{\Pi})$. \square

We are now ready to connect τ to the round elimination framework. To be precise, we will now prove that $\mathcal{Q}(\tau_P(\Pi))$ i.e., the problem obtained by performing one round elimination step on $\tau_P(\Pi)$, can always be relaxed to $\tau_{\mathcal{Q}(P)}(\Pi)$. This holds without any further assumptions on P and Π , however the application to the case where P is a fixed point is particularly important. Broadly speaking, P being a fixed point allows us to replace the appearing $\mathcal{Q}(P)$ with P , which yields a relaxation $\mathcal{Q}(\tau_P(\Pi)) \xrightarrow{0} \tau_P(\Pi)$. But this just means that $\tau_P(\Pi)$ itself is a fixed point, which we will formally see in Corollary 3.10.

We will now prove the statement $\mathcal{Q}(\tau_P(\Pi)) \xrightarrow{0} \tau_{\mathcal{Q}(P)}(\Pi)$ as mentioned above. The proof requires delving into the definitions of $\mathcal{R}(\cdot)$ and $\overline{\mathcal{R}}(\cdot)$ making it very technical, luckily it is the only time we need to unpack these definitions.

Lemma 3.9. *For any two node-edge-checkable problems P and Π we have*

$$\mathcal{Q}(\tau_P(\Pi)) \xrightarrow{0} \tau_{\mathcal{Q}(P)}(\Pi).$$

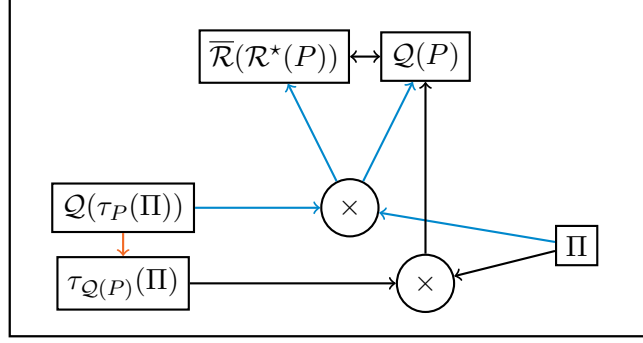


Figure 1: Diagram to visualize Lemma 3.9

Proof. To get a better overview, we visualize the present node-edge-checkable problems in a small diagram, see Figure 1. A “ \rightarrow ” means that one problem can be relaxed to another. Two problems pointing at a \times -node mean that their product can be relaxed to the problem the \times -node points to.

At first we will explain why the black relaxations in Figure 1 hold. Observe that $\tau_{Q(P)}(\Pi) \times \Pi \xrightarrow{0} Q(P)$ follows directly from Lemma 3.2. Furthermore, $\mathcal{R}^*(P)$ can be relaxed to $\mathcal{R}(P)$ using an edge-based relaxation. The only thing each edge needs to do is to enlarge its labels (which are sets) so that it ends up with a maximal configuration of $\mathcal{R}^*(P)$, which by definition is contained in $\mathcal{R}(P)$. Since $\mathcal{R}(P)$ is a more restrictive version of $\mathcal{R}^*(P)$, it is clear that, conversely, $\mathcal{R}(P)$ can also be relaxed to $\mathcal{R}^*(P)$, using an edge-based relaxation that just does nothing. Thus we can apply part 2 of Lemma 3.8 to obtain that $\overline{\mathcal{R}}(\mathcal{R}^*(P))$ and $\overline{\mathcal{R}}(\mathcal{R}(P)) = Q(P)$ are equivalent in the sense of Definition 2.8. This concludes the black relaxations.

The orange relaxation $Q(\tau_P(\Pi)) \xrightarrow{0} \tau_{Q(P)}(\Pi)$ is the one we want to prove.

Now consider the blue relaxations and observe that $Q(\tau_P(\Pi)) \times \Pi \xrightarrow{0} Q(P)$ implies $Q(\tau_P(\Pi)) \xrightarrow{0} \tau_{Q(P)}(\Pi)$ using Theorem 3.3. Thus it suffices to prove $Q(\tau_P(\Pi)) \times \Pi \xrightarrow{0} Q(P)$. But since $Q(P)$ and $\overline{\mathcal{R}}(\mathcal{R}^*(P))$ are equivalent, we can instead finish the proof by showing the other blue relaxation

$$Q(\tau_P(\Pi)) \times \Pi \xrightarrow{0} \overline{\mathcal{R}}(\mathcal{R}^*(P))$$

which we will do in the following.

Therefore, we first define a function $T: \Sigma_{Q(\tau_P(\Pi))} \times \Sigma_{\Pi} \rightarrow \Sigma_{\overline{\mathcal{R}}(\mathcal{R}^*(P))}$ as follows: For

$$S = \{\{f_{11}, \dots, f_{1b_1}\}, \{f_{21}, \dots, f_{2b_2}\}, \dots, \{f_{a1}, \dots, f_{ab_a}\}\} \in \Sigma_{\overline{\mathcal{R}}(\mathcal{R}(\tau_P(\Pi)))}$$

and some $L \in \Sigma_{\Pi}$ we set

$$T(S, L) := \left\{ \bigcup_{k=1}^{b_1} f_{1k}(L), \bigcup_{k=1}^{b_2} f_{2k}(L), \dots, \bigcup_{k=1}^{b_a} f_{ak}(L) \right\}.$$

One can imagine each node applies this functions to all its label pairs, however that is not quite the relaxation yet. For now let $(S_1, L_1) \dots (S_{\Delta}, L_{\Delta}) \in \mathcal{N}_{Q(\tau_P(\Pi)) \times \Pi}$ be a node configuration. Suppose that each S_i is given by

$$S_i = \{\{f_{11}^i, \dots, f_{1b_1}^i\}, \dots, \{f_{a1}^i, \dots, f_{ab_a}^i\}\}.$$

We have $S_1 \dots S_{\Delta} \in \mathcal{N}_{\overline{\mathcal{R}}(\mathcal{R}(\tau_P(\Pi)))}$ and can thus apply the definitions of $\overline{\mathcal{R}}(\cdot)$ to obtain

$$\{f_{j1}^1, \dots, f_{j1b_1}^1\} \dots \{f_{j\Delta}^{\Delta}, \dots, f_{jb_{j\Delta}}^{\Delta}\} \in \mathcal{R}(\tau_P(\Pi))$$

for all indices $1 \leq j^i \leq a^i$. We now choose some arbitrary indices j^1, \dots, j^Δ . By definition of $\mathcal{R}(\cdot)$ we can w.l.o.g. assume $f_{j^1 1}^1 \dots f_{j^\Delta 1}^\Delta \in \mathcal{N}_{\tau_P(\Pi)}$ for these indices j^i . By definition of τ this implies $f_{j^1 1}^1(\mathbf{L}_1) \dots f_{j^\Delta 1}^\Delta(\mathbf{L}_\Delta) \in \mathcal{N}_{\mathcal{R}^*(P)}$. Following the definition of $\mathcal{R}^*(P)$ we can enlarge these sets to get

$$\bigcup_{k=1}^{b_{j^1}^1} f_{j^1 k}^1(\mathbf{L}_1) \dots \bigcup_{k=1}^{b_{j^\Delta}^\Delta} f_{j^\Delta k}^\Delta(\mathbf{L}_\Delta) \in \mathcal{N}_{\mathcal{R}^*(P)}.$$

Since the indices $1 \leq j^i \leq a^i$ were arbitrary, the latter relation holds for all possible j^i . We recall that $T(\mathbf{S}_i, \mathbf{L}_i)$ by definition is given by

$$T(\mathbf{S}_i, \mathbf{L}_i) = \left\{ \bigcup_{k=1}^{b_{j^i}^i} f_{j^i k}^i(\mathbf{L}_i); 1 \leq j^i \leq a^i \right\}.$$

Hence we can conclude:

$$\forall \mathbf{S}'_1 \in T(\mathbf{S}_1, \mathbf{L}_1), \dots, \forall \mathbf{S}'_\Delta \in T(\mathbf{S}_\Delta, \mathbf{L}_\Delta): \mathbf{S}'_1 \dots \mathbf{S}'_\Delta \in \mathcal{N}_{\mathcal{R}^*(P)}$$

Now by definition of $\overline{\mathcal{R}}(\cdot)$ there exist supersets $T'(\mathbf{S}_i, \mathbf{L}_i) \supseteq T(\mathbf{S}_i, \mathbf{L}_i)$ for all $1 \leq i \leq \Delta$ such that $T'(\mathbf{S}_1, \mathbf{L}_1) \dots T'(\mathbf{S}_\Delta, \mathbf{L}_\Delta) \in \overline{\mathcal{R}}(\mathcal{R}^*(P))$, which a node knowing the $T(\mathbf{S}_i, \mathbf{L}_i)$ can compute. For our relaxation, each node chooses these $T'(\mathbf{S}_i, \mathbf{L}_i)$ as its new labels. The definition of T' immediately proves condition 1 of Definition 2.4, so we are left to prove condition 2. Therefore, let $(\mathbf{S}_1, \mathbf{L}_1) (\mathbf{S}_2, \mathbf{L}_2) \in \mathcal{E}_{\mathcal{Q}(\tau_P(\Pi)) \times \Pi}$ be an arbitrary edge configuration. Similar to above, we set

$$\begin{aligned} \mathbf{S}_1 &= \{\{f_{11}, \dots, f_{1b_1}\}, \{f_{21}, \dots, f_{2b_2}\}, \dots, \{f_{a1}, \dots, f_{ab_a}\}\}, \\ \mathbf{S}_2 &= \{\{g_{11}, \dots, g_{1d_1}\}, \{g_{21}, \dots, g_{2d_2}\}, \dots, \{g_{c1}, \dots, g_{cd_c}\}\}. \end{aligned}$$

Since we have $\mathbf{S}_1 \mathbf{S}_2 \in \mathcal{E}_{\overline{\mathcal{R}}(\tau_P(\Pi))}$, we can use the definition of $\overline{\mathcal{R}}(\cdot)$ and w.l.o.g. assume

$$\{f_{11}, \dots, f_{1b_1}\} \{g_{11}, \dots, g_{1d_1}\} \in \mathcal{E}_{\mathcal{R}(\tau_P(\Pi))}.$$

We then get:

$$\begin{aligned} & \{f_{11}, \dots, f_{1b_1}\} \{g_{11}, \dots, g_{1d_1}\} \in \mathcal{E}_{\mathcal{R}(\tau_P(\Pi))} \\ \xrightarrow{\text{Def. } \mathcal{E}_{\mathcal{R}}} & \forall 1 \leq i \leq b_1 \forall 1 \leq j \leq d_1: f_{1i} g_{1j} \in \mathcal{E}_{\tau_P(\Pi)} \\ \xrightarrow{\text{Def. } \mathcal{E}_\tau} & \forall 1 \leq i \leq b_1 \forall 1 \leq j \leq d_1: f_{1i}(\mathbf{L}_1) g_{1j}(\mathbf{L}_2) \in \mathcal{E}_{\mathcal{R}^*(P)} \\ \xrightarrow{\text{Def. } \mathcal{E}_{\mathcal{R}^*}} & \forall 1 \leq i \leq b_1 \forall 1 \leq j \leq d_1 \forall p_1 \in f_{1i}(\mathbf{L}_1) \forall p_2 \in g_{1j}(\mathbf{L}_2): p_1 p_2 \in \mathcal{E}_P \\ \implies & \forall p_1 \in \bigcup_{i=1}^{b_1} f_{1i}(\mathbf{L}_1) \forall p_2 \in \bigcup_{j=1}^{d_1} g_{1j}(\mathbf{L}_2): p_1 p_2 \in \mathcal{E}_P \\ \xrightarrow{\text{Def. } \mathcal{E}_{\mathcal{R}^*}} & \bigcup_{i=1}^{b_1} f_{1i}(\mathbf{L}_1) \bigcup_{j=1}^{d_1} g_{1j}(\mathbf{L}_2) \in \mathcal{E}_{\mathcal{R}^*(P)}. \end{aligned}$$

Observe that by the definitions of T and T' we know have $\bigcup_{i=1}^{b_1} f_{1i}(\mathbf{L}_1) \in T(\mathbf{S}_1, \mathbf{L}_1) \subseteq T'(\mathbf{S}_1, \mathbf{L}_1)$ and $\bigcup_{j=1}^{d_1} g_{1j}(\mathbf{L}_2) \in T(\mathbf{S}_2, \mathbf{L}_2) \subseteq T'(\mathbf{S}_2, \mathbf{L}_2)$. So we finally obtain $T'(\mathbf{S}_1, \mathbf{L}_1) T'(\mathbf{S}_2, \mathbf{L}_2) \in \mathcal{E}_{\overline{\mathcal{R}}(\mathcal{R}^*(P))}$ which is what we needed to finish the proof. \square

Lemma 3.9 has no further assumptions on P and Π , however as we already mentioned, the case where P is a fixed point is particularly nice. Informally speaking, it allows us to replace $\mathcal{Q}(P)$ with P , which makes $\tau_P(\Pi)$ a fixed point. Indeed we have:

Corollary 3.10. *Let P, \mathcal{I} and \mathcal{F} be node-edge-checkable problems satisfying the following:*

1. $\mathcal{Q}(\mathcal{F}) \times \mathcal{I} \xrightarrow{0} \mathcal{F}$, meaning that \mathcal{F} is a generalized fixed point with input \mathcal{I} .
2. $P \xrightarrow{0} \mathcal{I}$.

Then $\tau_{\mathcal{F}}(P)$ is a fixed point, even in the setting without input.

Proof. We immediately get

$$\mathcal{Q}(\tau_{\mathcal{F}}(P)) \times P \xrightarrow[\text{Lemma 3.9}]{0} \tau_{\mathcal{Q}(\mathcal{F})}(P) \times P \xrightarrow[\text{Lemma 3.2}]{0} \mathcal{Q}(\mathcal{F}) \times P \xrightarrow{0} \mathcal{Q}(\mathcal{F}) \times \mathcal{I} \xrightarrow{0} \mathcal{F},$$

where the last two steps follow from the two assumptions. We can combine these relaxations using Observation 2.7 to obtain $\mathcal{Q}(\tau_{\mathcal{F}}(P)) \times P \xrightarrow{0} \mathcal{F}$. This allows us to apply Theorem 3.3 to get $\mathcal{Q}(\tau_{\mathcal{F}}(P)) \xrightarrow{0} \tau_{\mathcal{F}}(P)$, proving that $\tau_{\mathcal{F}}(P)$ is a fixed point. \square

What this means is that the fixed point property is carried over from \mathcal{F} to $\tau_{\mathcal{F}}(P)$. We will make use of this in the next section. Further, it is worth noting that while generating new fixed points using Corollary 3.10, we can get rid of a previously required input. Observe that while \mathcal{F} is a generalized fixed point requiring \mathcal{I} as input, $\tau_{\mathcal{F}}(P)$ is a fixed point in the setting without input. The input is, of course, hiding in the assumption $P \xrightarrow{0} \mathcal{I}$, however broadly speaking, this can easily be ensured by replacing P with $P \times \mathcal{I}$, which does not make P harder in the setting where \mathcal{I} is a given input.

3.3 Constructing fixed point relaxations

We have now collected everything necessary to prove a rather surprising result. Broadly speaking, it states that whenever a node-edge-checkable problem Π allows a relaxation with input to a nontrivial generalized fixed point, there also exists a nontrivial fixed point relaxation of Π .

Theorem 3.11. *For two node-edge-checkable problems Π and \mathcal{I} the following two statements are equivalent:*

1. *There exists a nontrivial generalized fixed point \mathcal{F} given \mathcal{I} which is a relaxation of Π in the setting with input \mathcal{I} :*

$$\Pi \times \mathcal{I} \xrightarrow{0} \mathcal{F}, \quad \mathcal{I} \not\xrightarrow{0} \mathcal{F}, \quad \mathcal{Q}(\mathcal{F}) \times \mathcal{I} \xrightarrow{0} \mathcal{F}.$$

2. *There exists a fixed point relaxation \mathcal{G} of Π in the setting without input, which is nontrivial even given the input \mathcal{I} :*

$$\Pi \xrightarrow{0} \mathcal{G}, \quad \mathcal{I} \not\xrightarrow{0} \mathcal{G}, \quad \mathcal{Q}(\mathcal{G}) \xrightarrow{0} \mathcal{G}.$$

Proof. Statement 2 clearly implies 1, as we can just choose $\mathcal{F} := \mathcal{G}$. For the other direction we assume that there exists a problem \mathcal{F} satisfying the properties listed in statement 1. We set $\mathcal{G} := \tau_{\mathcal{F}}(\mathcal{I})$ and prove that \mathcal{G} fulfills all required properties. First we note that

$$\Pi \times \mathcal{I} \xrightarrow{0} \mathcal{F} \xrightarrow[\text{Theorem 3.3}]{0} \Pi \xrightarrow{0} \mathcal{G},$$

where we used \mathcal{F} 's first property. Now assume $\mathcal{I} \xrightarrow{0} \mathcal{G}$. This would imply

$$\mathcal{I} \xrightarrow{0} \mathcal{I} \times \mathcal{G} = \mathcal{I} \times \tau_{\mathcal{F}}(\mathcal{I}) \xrightarrow{0} \mathcal{F}$$

by Lemma 3.2 contradicting $\mathcal{I} \not\xrightarrow{0} \mathcal{F}$. Thus we must have $\mathcal{I} \not\xrightarrow{0} \mathcal{G}$. Lastly, we have $\mathcal{Q}(\mathcal{G}) \xrightarrow{0} \mathcal{G}$ by Corollary 3.10, where we choose P to just be \mathcal{I} . \square

Here one should think about Π as some node-edge-checkable problem for which we want to prove hardness. \mathcal{I} is an input problem that may have been introduced artificially to find such a proof. If we were interested in Π in the setting given some input, \mathcal{I} can be chosen as some node-edge-checkable problem that is 0-round relaxable to this input.

With this interpretation in mind, part 1 of Theorem 3.11 means that the hardness of Π can be proven using a generalized fixed point relaxation with (artificial) input. Note that the input may be used in the relaxation of Π to the fixed point \mathcal{F} and in the relaxation from $\mathcal{Q}(\mathcal{F})$ to \mathcal{F} making \mathcal{F} a generalized fixed point. In contrast, part 2 states that the hardness of Π can be proven using a fixed point relaxation without input. Even more, we can “force” \mathcal{G} in strategy 2 to not be easier than our artificially introduced input.

Recall that the first is clearly a generalization of the latter, since we can just introduce a trivial input. However, Theorem 3.11 proves that these strategies are equally powerful in the sense that they allow to prove hardness for the same problems Π . This may be surprising, but note that it is actually a necessary condition for the correctness of the possible conjecture that there exists a nontrivial fixed point relaxation for each intermediate-hard node-edge-checkable problem.

We further note that given a node-edge-checkable problem Π we want to solve given a fix input \mathcal{I} , the part 1 of Theorem 3.11 is nothing else then finding a nontrivial generalized fixed point relaxation in the setting with this given input. It asks to find a node-edge-checkable problem \mathcal{F} , which is a relaxation of Π (here also adding the input \mathcal{I}), nontrivial given \mathcal{I} and a generalized fixed point in the setting with input \mathcal{I} .

Further, it is worth noticing that the proof of Theorem 3.11 is constructive. When given a node-edge-checkable problem Π and a nontrivial generalized fixed point relaxation \mathcal{F} under a certain input \mathcal{I} , a nontrivial fixed point relaxation of Π without input is given by $\tau_{\mathcal{F}}(\mathcal{I})$.

Lastly, we reconsider the setting where a fixed point candidate \mathcal{F} has already been chosen, and we are left to find a suitable input problem \mathcal{I} to ensure part 1 of Theorem 3.11. Since we need to ensure $\Pi \times \mathcal{I} \xrightarrow{0} \mathcal{F}$ and $\mathcal{I} \not\xrightarrow{0} \mathcal{F}$, a natural good candidate would be $\mathcal{I} = \tau_{\mathcal{F}}(\Pi)$, since it is the easiest problem satisfying $\Pi \times \mathcal{I} \xrightarrow{0} \mathcal{F}$ by Lemma 3.2 and Theorem 3.3. Following the proof of Theorem 3.11, the double-input $\tau_{\mathcal{F}}(\tau_{\mathcal{F}}(\Pi))$ would then be a nontrivial fixed point relaxation of Π in the setting without input. This suggests that whenever there exists a fixed point relaxation of Π given some input, we can choose that input as $\tau_{\mathcal{F}}(\Pi)$ and furthermore obtain that $\tau_{\mathcal{F}}(\tau_{\mathcal{F}}(\Pi))$ is a nontrivial fixed point relaxation of Π without input. The following corollary formally proves this statement.

Corollary 3.12. *For a node-edge-checkable problem Π and a fixed point \mathcal{F} , the following three statements are equivalent:*

1. *Problem $\tau_{\mathcal{F}}(\Pi)$ satisfies $\tau_{\mathcal{F}}(\Pi) \not\xrightarrow{0} \mathcal{F}$.*
2. *There exists a node-edge-checkable problem \mathcal{I} such that $\Pi \times \mathcal{I} \xrightarrow{0} \mathcal{F}$ and $\mathcal{I} \not\xrightarrow{0} \mathcal{F}$.*
3. *Problem $\tau_{\mathcal{F}}(\tau_{\mathcal{F}}(\Pi))$ is a nontrivial fixed point relaxation of Π .*

Proof. It is easy to see that 1 implies 2, since we can choose $\mathcal{I} := \tau_{\mathcal{F}}(\Pi)$ and obtain $\Pi \times \mathcal{I} \xrightarrow{0} \mathcal{F}$ directly from Lemma 3.2.

Conversely, $\Pi \times \mathcal{I} \xrightarrow{0} \mathcal{F}$ implies $\mathcal{I} \xrightarrow{0} \tau_{\mathcal{F}}(\Pi)$ by Theorem 3.3. Thus $\tau_{\mathcal{F}}(\Pi) \xrightarrow{0} \mathcal{F}$ would imply $\mathcal{I} \xrightarrow{0} \mathcal{F}$ by Observation 2.7, contradicting the assumption $\mathcal{I} \not\xrightarrow{0} \mathcal{F}$. Hence we must have $\tau_{\mathcal{F}}(\Pi) \not\xrightarrow{0} \mathcal{F}$, proving that 2 implies 1.

To prove 1 implies 3, observe that our assumptions $\mathcal{Q}(\mathcal{F}) \xrightarrow{0} \mathcal{F}$ and $\tau_{\mathcal{F}}(\Pi) \not\xrightarrow{0} \mathcal{F}$ together with Lemma 3.2 correspond to the assumptions in part 1 of Theorem 3.11, where $\tau_{\mathcal{F}}(\Pi)$ takes the role of \mathcal{I} . According to the proof of Theorem 3.11, $\tau_{\mathcal{F}}(\tau_{\mathcal{F}}(\Pi))$ is a nontrivial fixed point relaxation of Π .

Lastly, we prove the contraposition $\neg 1$ implies $\neg 3$ and hence assume $\tau_{\mathcal{F}}(\Pi) \xrightarrow{0} \mathcal{F}$. This immediately yields $\tau_{\mathcal{F}}(\Pi) \times P \xrightarrow{0} \mathcal{F}$ for the trivial node-edge-checkable problem P , and thus $P \xrightarrow{0} \tau_{\mathcal{F}}(\tau_{\mathcal{F}}(\Pi))$ by Theorem 3.3. It follows directly that $\tau_{\mathcal{F}}(\tau_{\mathcal{F}}(\Pi))$ is also trivial. \square

4 SSO with SO as input

In this section, our goal is to prove the following theorem.

Theorem 4.1. *Let Π denote the sinkless and sourceless orientation problem. Then, in the setting where a sinkless orientation is given as input, Π satisfies the following two properties.*

1. *Solving Π with a deterministic algorithm requires $\Omega(\log n)$ rounds and solving Π with a randomized algorithm requires $\Omega(\log \log n)$ rounds.*
2. *There is no nontrivial fixed point relaxation of Π . Moreover, there is no nontrivial generalized fixed point relaxation of Π .*

The sinkless and sourceless orientation problem (SSO) is the problem of orienting the edges such that no node is a sink and no node is a source, i.e., such that each node has at least one outgoing and at least one incoming edge. For simplicity, in the remainder of this section, we focus on the case of 3-regular SSO. This already suffices to provide our impossibility results (and generalizes straightforwardly to any $\Delta > 3$).

Analogously to how sinkless orientation can be phrased as a node-edge-checkable problem (see Section 2), SSO can be easily phrased as a node-edge-checkable problem by labeling half-edges with I (incoming) and O (outgoing), which naturally leads to the following node and edge constraints: the only (condensed) configuration in the node constraint of SSO is I O [I O], the only configuration in the edge constraint of SSO is I O.

Now we turn our attention towards proving Theorem 4.1. We start with characterizing the problems in the round elimination sequence $\Pi, \mathcal{Q}(\Pi), \mathcal{Q}(\mathcal{Q}(\Pi)), \dots$ for the case of Π being SSO.

Lemma 4.2. *Let Π be the problem of sinkless and sourceless orientation and k a nonnegative integer. Then the node and edge constraints of $\mathcal{Q}^k(\Pi)$ are:*

$$\begin{array}{ll}
\text{If } k = 0: & \mathcal{N}_{\mathcal{Q}^k(\Pi)}: \quad \text{B C [B C]} \\
& \mathcal{E}_{\mathcal{Q}^k(\Pi)}: \quad \text{B C}, \\
\text{If } k = 1: & \mathcal{N}_{\mathcal{Q}^k(\Pi)}: \quad \text{B C D} \\
& \mathcal{E}_{\mathcal{Q}^k(\Pi)}: \quad [\text{B D}] [\text{C D}], \\
\text{If } k \geq 2: & \mathcal{N}_{\mathcal{Q}^k(\Pi)}: \quad \text{B C D}_k \\
& \quad \text{A}_i \text{ D}_i \text{ D}_k \text{ for each } 1 \leq i \leq k-1 \\
& \mathcal{E}_{\mathcal{Q}^k(\Pi)}: \quad [\text{B D}_1 \text{ D}_2 \dots \text{D}_k] [\text{C D}_1 \text{ D}_2 \dots \text{D}_k] \\
& \quad \text{A}_i \text{ D}_j \text{ for each } 1 \leq i < j \leq k.
\end{array}$$

Proof. We prove the lemma for the cases $k = 0$, $k = 1$, and $k \geq 2$ in that order. The case $k = 0$ follows directly from the definition of sinkless and sourceless orientation by renaming the labels.

Now consider the case $k = 1$. It is straightforward to verify that, by applying $\mathcal{R}(\cdot)$ to Π , we obtain

$$\begin{aligned}\mathcal{E}_{\mathcal{R}(\Pi)} &= \{\{B\} \{C\}\}, \\ \Sigma_{\mathcal{R}(\Pi)} &= \{\{B\}, \{C\}\}, \text{ and} \\ \mathcal{N}_{\mathcal{R}(\Pi)} &= \{\{B\} \{B\} \{C\}, \{B\} \{C\} \{C\}\}.\end{aligned}$$

Again, it is straightforward to apply $\overline{\mathcal{R}}(\cdot)$ to $\mathcal{R}(\Pi)$ (where, for computing $\mathcal{E}_{\overline{\mathcal{R}}(\mathcal{R}(\Pi))}$, we can make use of Observation 2.10), and we obtain

$$\begin{aligned}\mathcal{N}_{\overline{\mathcal{R}}(\mathcal{R}(\Pi))} &= \{\{\{B\}\} \{\{C\}\} \{\{B\}, \{C\}\}\}, \\ \Sigma_{\overline{\mathcal{R}}(\mathcal{R}(\Pi))} &= \{\{\{B\}\}, \{\{C\}\}, \{\{B\}, \{C\}\}\}, \text{ and} \\ \mathcal{E}_{\overline{\mathcal{R}}(\mathcal{R}(\Pi))} &= \{\{\{B\}\} \{\{C\}\}, \{\{B\}\} \{\{B\}, \{C\}\}, \{\{C\}\} \{\{B\}, \{C\}\}, \{\{B\}, \{C\}\} \{\{B\}, \{C\}\}\}.\end{aligned}$$

By performing the renaming

$$\begin{aligned}\{\{B\}\} &\rightarrow B \\ \{\{C\}\} &\rightarrow C \\ \{\{B\}, \{C\}\} &\rightarrow D,\end{aligned}$$

we obtain

$$\begin{aligned}\mathcal{N}_{\overline{\mathcal{R}}(\mathcal{R}(\Pi))} &= \{B \ C \ D\} \text{ and} \\ \mathcal{E}_{\overline{\mathcal{R}}(\mathcal{R}(\Pi))} &= \{B \ C, \ B \ D, \ C \ D, \ D \ D\},\end{aligned}$$

as desired.

Finally consider the case $k \geq 2$. We will prove this case via induction. For the base case, consider that $k = 2$. We claim that by applying $\mathcal{R}(\cdot)$ to $\mathcal{Q}(\Pi)$ (which we already established), we obtain

$$\begin{aligned}\mathcal{E}_{\mathcal{R}(\mathcal{Q}(\Pi))} &= \{\{B, D\} \{C, D\}, \{D\} \{B, C, D\}\}, \\ \Sigma_{\mathcal{R}(\mathcal{Q}(\Pi))} &= \{\{D\}, \{B, D\}, \{C, D\}, \{B, C, D\}\}, \text{ and} \\ \mathcal{N}_{\mathcal{R}(\mathcal{Q}(\Pi))} &= \{[\{B, D\} \{B, C, D\}] [\{C, D\} \{B, C, D\}] [\{D\} \{B, D\} \{C, D\} \{B, C, D\}]\}.\end{aligned}$$

It is straightforward to verify that all configurations listed in the claimed $\mathcal{E}_{\mathcal{R}(\mathcal{Q}(\Pi))}$ satisfy the universal quantifier. Moreover, it is clear that no listed configuration dominates another (different) listed configuration and that each configuration obtainable by taking a configuration from $\mathcal{E}_{\mathcal{Q}(\Pi)}$ and replacing each label in the configuration by the singleton set containing the label is dominated by a listed configuration. Since $\{B, D\}$ and $\{C, D\}$ are the only incomparable sets appearing in the claimed $\mathcal{E}_{\mathcal{Q}(\Pi)}$ and the configuration $\{B, D\} \cup \{C, D\} \{C, D\} \cap \{B, D\}$ is identical to $\{D\} \{B, C, D\}$ (which is listed above), we obtain by Observation 2.11 and Observation 2.12 that $\mathcal{E}_{\mathcal{R}(\mathcal{Q}(\Pi))}$ is indeed as given above. Now the correctness of the claim follows straightforwardly by applying Observation 2.10.

Let us rename the labels in $\Sigma_{\mathcal{R}(\mathcal{Q}(\Pi))}$ to make the obtained problem $\mathcal{R}(\mathcal{Q}(\Pi))$ more readable. By performing the renaming

$$\begin{aligned}\{D\} &\rightarrow D \\ \{B, D\} &\rightarrow B \\ \{C, D\} &\rightarrow C \\ \{B, C, D\} &\rightarrow A,\end{aligned}$$

we obtain

$$\begin{aligned}\mathcal{E}_{\mathcal{R}(\mathcal{Q}(\Pi))} &= \{B \ C, \quad A \ D\} \text{ and} \\ \mathcal{N}_{\mathcal{R}(\mathcal{Q}(\Pi))} &= \{[A \ B] \ [A \ C] \ [A \ B \ C \ D]\}.\end{aligned}$$

We claim that, by applying $\overline{\mathcal{R}}(\cdot)$ to $\mathcal{R}(\mathcal{Q}(\Pi))$, we obtain

$$\begin{aligned}\mathcal{N}_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}(\Pi)))} &= \{\{A, B\} \ \{A, C\} \ \{A, B, C, D\}, \quad \{A\} \ \{A, B, C\} \ \{A, B, C, D\}\}, \\ \Sigma_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}(\Pi)))} &= \{\{A\}, \{A, B\}, \{A, C\}, \{A, B, C\}, \{A, B, C, D\}\}, \text{ and} \\ \mathcal{E}_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}(\Pi)))} &= \{[\{A, B\} \ \{A, B, C\} \ \{A, B, C, D\}] \ [\{A, C\} \ \{A, B, C\} \ \{A, B, C, D\}], \\ &\quad [\{A\} \ \{A, B\} \ \{A, C\} \ \{A, B, C\} \ \{A, B, C, D\}] \ \{A, B, C, D\}\}.\end{aligned}$$

To prove the claim, observe that $\{A, B\}$ and $\{A, C\}$ are the only incomparable sets appearing in the claimed $\mathcal{N}_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}(\Pi)))}$. Since the configuration

$$\{A, B\} \cup \{A, C\} \ \{A, C\} \cap \{A, B\} \ \{A, B, C, D\} \cap \{A, B, C, D\}$$

is identical to $\{A\} \ \{A, B, C\} \ \{A, B, C, D\}$ and the configuration

$$\{A, B\} \cup \{A, C\} \ \{A, C\} \cap \{A, B, C, D\} \ \{A, B, C, D\} \cap \{A, B\}$$

is dominated by $\{A, B\} \ \{A, C\} \ \{A, B, C, D\}$, analogously to the argumentation in the case of $\mathcal{E}_{\mathcal{R}(\mathcal{Q}(\Pi))}$ we obtain that $\mathcal{N}_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}(\Pi)))}$ is indeed as given above. Again, the claim follows by applying Observation 2.10.

By performing the renaming

$$\begin{aligned}\{A\} &\rightarrow A_1 \\ \{A, B\} &\rightarrow B \\ \{A, C\} &\rightarrow C \\ \{A, B, C\} &\rightarrow D_1 \\ \{A, B, C, D\} &\rightarrow D_2,\end{aligned}$$

we obtain

$$\begin{aligned}\mathcal{N}_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}(\Pi)))} &= \{B \ C \ D_2, A_1 \ D_1 \ D_2\} \text{ and} \\ \mathcal{E}_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}(\Pi)))} &= \{[B \ D_1 \ D_2] \ [C \ D_1 \ D_2], \quad [A_1 \ B \ C \ D_1 \ D_2] \ D_2\}.\end{aligned}$$

Observing that

$$\{[B \ D_1 \ D_2] \ [C \ D_1 \ D_2], \quad [A_1 \ B \ C \ D_1 \ D_2] \ D_2\} = \{[B \ D_1 \ D_2] \ [C \ D_1 \ D_2], \quad A_1 \ D_2\}$$

concludes the base case.

For the induction step, fix any $k \geq 2$ and assume the induction hypothesis, i.e., that $\mathcal{N}_{\mathcal{Q}^{k-1}(\Pi)}$ and $\mathcal{E}_{\mathcal{Q}^{k-1}(\Pi)}$ are as given in the lemma. We claim that, by applying $\mathcal{R}(\cdot)$ to $\mathcal{Q}^{k-1}(\Pi)$, we obtain that the edge constraint $\mathcal{E}_{\mathcal{R}(\mathcal{Q}^{k-1}(\Pi))}$ of $\mathcal{R}(\mathcal{Q}^{k-1}(\Pi))$ is given by the following configurations:

$$\begin{aligned} & \{B, D_1, \dots, D_{k-1}\} \{C, D_1, \dots, D_{k-1}\}, \\ & \{B, C, D_1, \dots, D_{k-1}\} \{D_1, \dots, D_{k-1}\}, \\ & \{A_1, \dots, A_i, B, C, D_1, \dots, D_{k-1}\} \{D_{i+1}, \dots, D_{k-1}\} \text{ for each } 1 \leq i \leq k-2. \end{aligned}$$

Analogously to before (i.e., making use of Observations 2.11 and 2.12), the claim follows from the facts that $\{B, D_1, \dots, D_{k-1}\}$ and $\{C, D_1, \dots, D_{k-1}\}$ are the only incomparable listed sets and

$$\{B, D_1, \dots, D_{k-1}\} \cup \{C, D_1, \dots, D_{k-1}\} \{C, D_1, \dots, D_{k-1}\} \cap \{B, D_1, \dots, D_{k-1}\}$$

is identical to (the listed) $\{B, C, D_1, \dots, D_{k-1}\} \{D_1, \dots, D_{k-1}\}$.

The claim implies that the output label set for $\mathcal{R}(\mathcal{Q}^{k-1}(\Pi))$ is

$$\begin{aligned} \Sigma_{\mathcal{R}(\mathcal{Q}^{k-1}(\Pi))} = & \{\{B, D_1, \dots, D_{k-1}\}, \{C, D_1, \dots, D_{k-1}\}, \{B, C, D_1, \dots, D_{k-1}\}\} \\ & \cup \{\{A_1, \dots, A_i, B, C, D_1, \dots, D_{k-1}\} \mid 1 \leq i \leq k-2\} \\ & \cup \{\{D_i, \dots, D_{k-1}\} \mid 1 \leq i \leq k-1\}. \end{aligned}$$

By performing the renaming

$$\begin{aligned} & \{D_i, \dots, D_{k-1}\} \rightarrow D_i \text{ for each } 1 \leq i \leq k-1 \\ & \{B, D_1, \dots, D_{k-1}\} \rightarrow B \\ & \{C, D_1, \dots, D_{k-1}\} \rightarrow C \\ & \{B, C, D_1, \dots, D_{k-1}\} \rightarrow A_0 \\ & \{A_1, \dots, A_i, B, C, D_1, \dots, D_{k-1}\} \rightarrow A_i \text{ for each } 1 \leq i \leq k-2, \end{aligned}$$

we can rewrite the edge constraint and output label set for $\mathcal{R}(\mathcal{Q}^{k-1}(\Pi))$ as follows:

$$\begin{aligned} \mathcal{E}_{\mathcal{R}(\mathcal{Q}^{k-1}(\Pi))}: & \quad B \ C \\ & \quad A_i \ D_{i+1} \text{ for each } 0 \leq i \leq k-2, \\ \Sigma_{\mathcal{R}(\mathcal{Q}^{k-1}(\Pi))}: & \quad \{B, C\} \cup \{A_i \mid 0 \leq i \leq k-2\} \cup \{D_i \mid 1 \leq i \leq k-1\}. \end{aligned}$$

Using the aforementioned renaming, we obtain, by Observation 2.10, that the node constraint $\mathcal{N}_{\mathcal{R}(\mathcal{Q}^{k-1}(\Pi))}$ of $\mathcal{R}(\mathcal{Q}^{k-1}(\Pi))$ is given by the following condensed configurations:

$$\begin{aligned} & [A_0 \ \dots \ A_{k-2} \ B] [A_0 \ \dots \ A_{k-2} \ C] [A_0 \ \dots \ A_{k-2} \ B \ C \ D_1 \ \dots \ D_{k-1}], \\ & [A_i \ \dots \ A_{k-2}] [A_0 \ \dots \ A_{k-2} \ B \ C \ D_1 \ \dots \ D_i] [A_0 \ \dots \ A_{k-2} \ B \ C \ D_1 \ \dots \ D_{k-1}] \quad \forall 1 \leq i \leq k-2. \end{aligned}$$

Finally, we claim that, by applying $\overline{\mathcal{R}}(\cdot)$ to $\mathcal{R}(\mathcal{Q}^{k-1}(\Pi))$, we obtain that the node constraint $\mathcal{N}_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}^{k-1}(\Pi)))}$ of $\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}^{k-1}(\Pi)))$ is given by the following configurations:

$$\begin{aligned} & \{A_0, \dots, A_{k-2}, B\} \{A_0, \dots, A_{k-2}, C\} \{A_0, \dots, A_{k-2}, B, C, D_1, \dots, D_{k-1}\}, \\ & \{A_0, \dots, A_{k-2}\} \{A_0, \dots, A_{k-2}, B, C\} \{A_0, \dots, A_{k-2}, B, C, D_1, \dots, D_{k-1}\}, \\ & \{A_i, \dots, A_{k-2}\} \{A_0, \dots, A_{k-2}, B, C, D_1, \dots, D_i\} \{A_0, \dots, A_{k-2}, B, C, D_1, \dots, D_{k-1}\} \quad \forall 1 \leq i \leq k-2. \end{aligned}$$

Analogously to before, the claim follows from the facts that $\{A_0, \dots, A_{k-2}, B\}$ and $\{A_0, \dots, A_{k-2}, C\}$ are the only incomparable listed sets, the configuration

$$\begin{aligned} & \{A_0, \dots, A_{k-2}, B\} \cup \{A_0, \dots, A_{k-2}, C\} \\ & \{A_0, \dots, A_{k-2}, C\} \cap \{A_0, \dots, A_{k-2}, B\} \\ & \{A_0, \dots, A_{k-2}, B, C, D_1, \dots, D_{k-1}\} \cap \{A_0, \dots, A_{k-2}, B, C, D_1, \dots, D_{k-1}\} \end{aligned}$$

is identical to $\{A_0, \dots, A_{k-2}\} \{A_0, \dots, A_{k-2}, B, C\} \{A_0, \dots, A_{k-2}, B, C, D_1, \dots, D_{k-1}\}$ (which is listed above), and the configuration

$$\begin{aligned} & \{A_0, \dots, A_{k-2}, B\} \cup \{A_0, \dots, A_{k-2}, C\} \\ & \{A_0, \dots, A_{k-2}, C\} \cap \{A_0, \dots, A_{k-2}, B, C, D_1, \dots, D_{k-1}\} \\ & \{A_0, \dots, A_{k-2}, B, C, D_1, \dots, D_{k-1}\} \cap \{A_0, \dots, A_{k-2}, B\} \end{aligned}$$

is dominated by $\{A_0, \dots, A_{k-2}, B\} \{A_0, \dots, A_{k-2}, C\} \{A_0, \dots, A_{k-2}, B, C, D_1, \dots, D_{k-1}\}$ (which is listed above).

The claim implies that the output label set for $\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}^{k-1}(\Pi)))$ is

$$\begin{aligned} \Sigma_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}^{k-1}(\Pi)))} &= \{\{A_0, \dots, A_{k-2}, B\}, \{A_0, \dots, A_{k-2}, C\}, \{A_0, \dots, A_{k-2}, B, C\}\} \\ &\cup \{\{A_i, \dots, A_{k-2}\} \mid 0 \leq i \leq k-2\} \\ &\cup \{\{A_0, \dots, A_{k-2}, B, C, D_1, \dots, D_i\} \mid 1 \leq i \leq k-1\}. \end{aligned}$$

By performing the renaming

$$\begin{aligned} & \{A_i, \dots, A_{k-2}\} \rightarrow A_{i+1} \text{ for each } 0 \leq i \leq k-2 \\ & \{A_0, \dots, A_{k-2}, B\} \rightarrow B \\ & \{A_0, \dots, A_{k-2}, C\} \rightarrow C \\ & \{A_0, \dots, A_{k-2}, B, C\} \rightarrow D_1 \\ & \{A_0, \dots, A_{k-2}, B, C, D_1, \dots, D_i\} \rightarrow D_{i+1} \text{ for each } 1 \leq i \leq k-1, \end{aligned}$$

we can rewrite the node constraint and output label set for $\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}^{k-1}(\Pi)))$ as follows:

$$\begin{aligned} \mathcal{N}_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}^{k-1}(\Pi)))}: & \quad B \ C \ D_k \\ & \quad A_i \ D_i \ D_k \text{ for each } 1 \leq i \leq k-1, \\ \Sigma_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}^{k-1}(\Pi)))}: & \quad \{B, C\} \cup \{A_i \mid 1 \leq i \leq k-1\} \cup \{D_i \mid 1 \leq i \leq k\}. \end{aligned}$$

Note that in this last renaming, indices have been “shifted by 1”. Again making use of this renaming (and Observation 2.10), we obtain that the edge constraint $\mathcal{E}_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}^{k-1}(\Pi)))}$ of $\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}^{k-1}(\Pi)))$ is given by the following configurations:

$$\begin{aligned} & [B \ D_1 \ \dots \ D_k] [C \ D_1 \ \dots \ D_k] \\ & [A_1 \ \dots \ A_i \ B \ C \ D_1 \ \dots \ D_k] [D_{i+1} \ \dots \ D_k] \text{ for each } 1 \leq i \leq k-1 \end{aligned}$$

Observing that this characterization of $\mathcal{E}_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}^{k-1}(\Pi)))}$ can be equivalently written as

$$\begin{aligned} \mathcal{E}_{\overline{\mathcal{R}}(\mathcal{R}(\mathcal{Q}^{k-1}(\Pi)))}: & \quad [B \ D_1 \ \dots \ D_k] [C \ D_1 \ \dots \ D_k] \\ & \quad A_1 \ D_j \text{ for each } 1 \leq i < j \leq k \end{aligned}$$

concludes the induction and the entire proof. \square

Now we are ready to prove Theorem 4.1.

Proof of Theorem 4.1. We start by showing Property 1. By applying Theorem 6.4 with $k \in \Theta(\log n)$, resp. $k \in \Theta(\log \log n)$, we see that it suffices to show for any $\ell \geq 0$ that $\mathcal{Q}^\ell(\Pi)$ is not solvable in 0 rounds in the deterministic PN model, even if a solution for sinkless orientation is given as input. (Note that we can choose the parameter L in Theorem 6.4 to be from $\Theta(\log n)$, resp. $\Theta(\log \log n)$, since the characterization of the $\mathcal{Q}^\ell(\Pi)$ given in Lemma 4.2 guarantees that the number of labels used to describe $\mathcal{Q}^\ell(\Pi)$ is in $O(\ell)$.)

To this end, consider first the case that $\ell \geq 2$, and assume for a contradiction that there is a 0-round PN-model algorithm \mathcal{A} that solves $\mathcal{Q}^\ell(\Pi)$ given a solution to sinkless orientation as input. Consider a node with input configuration $\mathbf{O} \mid \mid$, i.e., two of the half-edges incident to the node are oriented towards the node and the last half-edge is oriented away from the node. We consider two cases.

First consider the case that \mathcal{A} chooses the node configuration $\mathbf{A}_i \mathbf{D}_i \mathbf{D}_\ell \in \mathcal{N}_{\mathcal{Q}^\ell(\Pi)}$ for some $1 \leq i \leq \ell$ for such a node. If \mathcal{A} outputs the label \mathbf{A}_i on the outgoing half-edge and therefore outputs label \mathbf{D}_i on one of the incoming half-edges, then on some input graphs, \mathcal{A} will produce the edge configuration $\mathbf{A}_i \mathbf{D}_i$, which is not contained in $\mathcal{E}_{\mathcal{Q}^\ell(\Pi)}$, yielding a contradiction. Hence, \mathcal{A} necessarily outputs label \mathbf{A}_i on an incoming half-edge. Now consider a node with three outgoing half-edges. No matter which node configuration \mathcal{A} chooses for such a node, there is some output label from $\{\mathbf{A}_1, \dots, \mathbf{A}_{\ell-1}, \mathbf{B}, \mathbf{C}\}$ that \mathcal{A} outputs on an outgoing half-edge. Since none of the labels contained in this set yields an edge configuration in $\mathcal{E}_{\mathcal{Q}^\ell(\Pi)}$ when combined with \mathbf{A}_i , we conclude that \mathcal{A} will produce an incorrect output on some input graphs, yielding a contradiction.

Now consider the complementary case, i.e., that \mathcal{A} chooses the node configuration $\mathbf{B} \mathbf{C} \mathbf{D}_\ell$ for a node with precisely one incident outgoing half-edge. Then \mathcal{A} outputs one of \mathbf{B} and \mathbf{C} on an incoming half-edge. Due to symmetry, we can assume w.l.o.g. that \mathcal{A} outputs \mathbf{B} on an incoming half-edge. Now consider again an edge with three outgoing half-edges. No matter which node configuration \mathcal{A} chooses for such a node, there is some output label from $\{\mathbf{A}_1, \dots, \mathbf{A}_{\ell-1}, \mathbf{B}\}$ that \mathcal{A} outputs on an outgoing half-edge. Since none of the labels contained in this set yields an edge configuration in $\mathcal{E}_{\mathcal{Q}^\ell(\Pi)}$ when combined with \mathbf{B} , we conclude that \mathcal{A} will produce an incorrect output on some input graphs, yielding a contradiction.

We remark that, while we omitted an explicit specification of the ports of the incoming, resp. outgoing, edges in the above considerations, it is straightforward to select such a specification. Moreover, for the case that $\ell \in \{0, 1\}$, we obtain a contradiction analogously to the second case in the above considerations. This concludes the proof that Π satisfies Property 1.

Now we turn to showing Property 2. First we prove that there is no nontrivial fixed point relaxation of Π . We will do so by showing that each fixed point relaxation of Π can be solved in 0 rounds in our setting, i.e., given a sinkless orientation as input. Let $\hat{\Pi}$ be an arbitrary fixed point relaxation of Π . Let y be an integer satisfying $y \geq |\Sigma_{\hat{\Pi}}| + 2$. Consider the problem $\mathcal{Q}^y(\Pi)$ obtained from Π after y recursive applications of $\overline{\mathcal{R}}(\cdot)$.

As $\hat{\Pi}$ is a relaxation of Π , we obtain by Lemma 3.8 that $\mathcal{Q}^y(\hat{\Pi})$ is a relaxation of $\mathcal{Q}^y(\Pi)$. Since $\mathcal{Q}^y(\hat{\Pi}) = \hat{\Pi}$ (due to $\hat{\Pi}$ being a fixed point), it follows that $\hat{\Pi}$ is a relaxation of $\mathcal{Q}^y(\Pi)$. Moreover, by Lemma 4.2, the definition of y implies that $\mathcal{N}_{\mathcal{Q}^y(\hat{\Pi})}$ contains configuration $\mathbf{A}_i \mathbf{D}_i \mathbf{D}_y$ for each $1 \leq i \leq |\Sigma_{\hat{\Pi}}| + 1$.

Recall the definition of a relaxation function given in Definition 2.4, and let f be a relaxation function from $\mathcal{Q}^y(\Pi)$ to $\hat{\Pi}$. By the pigeonhole principle, we obtain that $\mathcal{N}_{\mathcal{Q}^y(\hat{\Pi})}$ must contain two configurations $\mathbf{A}_i \mathbf{D}_i \mathbf{D}_y$ and $\mathbf{A}_j \mathbf{D}_j \mathbf{D}_y$ satisfying $i < j < |\Sigma_{\hat{\Pi}}| + 2$ and $f(\mathbf{A}_i) = f(\mathbf{A}_j)$. Moreover, the definition of f ensures that $f(\mathbf{A}_j) f(\mathbf{D}_j) f(\mathbf{D}_y)$ is contained in $\hat{\Pi}$. We claim that if we select any two (not necessarily distinct) labels in this configuration, the configuration consisting of the

selected two labels is contained in $\mathcal{E}_{\hat{\Pi}}$, except possibly if we select $f(A_j)$ twice. In the following, we prove this claim.

By Lemma 4.2 and the fact that $j < |\Sigma_{\hat{\Pi}}| + 2 \leq y$, we know that the four configurations $D_j D_j$, $D_j D_y$, $D_y D_y$, and $A_j D_y$ are all contained in $\mathcal{E}_{\mathcal{Q}^y(\Pi)}$. Hence, by the fact that f is a relaxation function from Π to $\hat{\Pi}$, we obtain that the configurations $f(D_j) f(D_j)$, $f(D_j) f(D_y)$, $f(D_y) f(D_y)$, and $f(A_j) f(D_y)$ are all contained in $\mathcal{E}_{\hat{\Pi}}$. Moreover, since $i < j$, Lemma 4.2 also implies that $A_i D_j$ is contained in $\mathcal{E}_{\mathcal{Q}^y(\Pi)}$, which in turn implies that $f(A_i) f(D_j)$ is contained in $\mathcal{E}_{\hat{\Pi}}$. As $f(A_i) = f(A_j)$, it follows that also $f(A_j) f(D_j)$ is contained in $\mathcal{E}_{\hat{\Pi}}$, which concludes the proof of the claim.

The proved claim now offers a simple way to solve $\hat{\Pi}$ in 0 rounds given a sinkless orientation as input: each node v simply selects one of its incident half-edges for which the corresponding edge is oriented away from v in the given sinkless orientation, outputs $f(A_j)$ on the selected half-edge, outputs $f(D_j)$ on an arbitrarily chosen half-edge out of the remaining two, and outputs $f(D_y)$ on the last remaining half-edge. It remains to show that the output returned by this 0-round algorithm is indeed a correct solution for $\hat{\Pi}$. From the design of the algorithm it directly follows that, for each node v , the configuration consisting of the output labels on the half-edges incident to v is $f(A_j) f(D_j) f(D_y)$ and therefore contained in $\mathcal{N}_{\hat{\Pi}}$. For the correctness on the edges, observe that the design of the algorithm ensures that there is no edge e for which the output label on *both* half-edges is A_j (since e is outgoing for only one of its endpoints in the given sinkless orientation). Now, the claim that we proved above guarantees that, for each edge e , the configuration consisting of the output labels on the half-edges belonging to e is contained in $\mathcal{E}_{\hat{\Pi}}$, which concludes the proof for the statement that there is no nontrivial fixed point relaxation of Π .

Now, applying Theorem 3.11 yields that there is no generalized fixed point \mathcal{F} (with sinkless orientation as input), as that would imply the existence of a non-generalized fixed point relaxation that is nontrivial in the setting with sinkless orientation as input. This implies the second part of Property 2. \square

5 Fixed point relaxations for homomorphism problems

In [BCG⁺22], the authors show an interesting connection between two fields that, at a first glance, seem unrelated: distributed computing and descriptive combinatorics. A class of problems that has been widely studied in the context of descriptive combinatorics is the so-called Borel class. In [BCG⁺22], the authors show that, if a problem does not belong to the class Borel, then such a problem requires $\Omega(\log n)$ deterministic rounds and $\Omega(\log \log n)$ randomized rounds in the LOCAL model. In the literature of the field of descriptive combinatorics, there is a technique that can be used to show that a problem is not in the Borel class [Mar16]. The authors of [BCG⁺22] use such a technique to derive lower bounds in the LOCAL model for a so-called homomorphism problem. They leave as an open question the problem of understanding whether the same lower bounds can be achieved via the round elimination technique, and in particular they ask whether there is a nontrivial fixed point relaxation for the problems they achieve lower bounds for. In this section, we solve this open question by providing a positive answer. We start by defining the class of problems for which [BCG⁺22] obtains $\Omega(\log n)$ -round deterministic and $\Omega(\log \log n)$ -round randomized lower bounds via the aforementioned technique from descriptive combinatorics, which, in the distributed context, is often referred to as *Marks' technique*.

5.1 A subclass of homomorphism problems

Let \mathcal{P} denote the class of problems [BCG⁺22] obtain the aforementioned lower bounds for. The problems in \mathcal{P} come from a class of problems called *homomorphism problems*. In a homomorphism problem the task is to compute a homomorphism from the input graph to a given fixed graph H . This task can be reformulated as a coloring problem: color the nodes of the input graph such that each color is a node of H and any two neighboring nodes u, v in the input graph must output colors c_u, c_v such that c_u and c_v are adjacent nodes in H . To phrase a homomorphism problem as a node-edge-checkable problem, we simply require outputting the same color at each half-edge incident to the same node and that the two outputs on the half-edges belonging to the same edge satisfy the aforementioned coloring property.

The class \mathcal{P} studied in [BCG⁺22] is the class of all homomorphism problems where graph H satisfies a certain reasonably natural property, called Δ -($*$). For a formal definition of Δ -($*$), see [BCG⁺22, Definition 4.8, arXiv version]. The authors show further that, for any⁶ $\Delta \geq 3$, a graph H satisfies Δ -($*$) if and only if there is a homomorphism from H to a specific graph H_Δ [BCG⁺22, Proposition 4.10, arXiv version]. Observe that the existence of a homomorphism from a graph H to H_Δ implies that there is a relaxation from the homomorphism problem with target graph H to the homomorphism problem with target graph H_Δ as any homomorphism from the input graph G to H can be concatenated with the homomorphism from H to H_Δ to yield a homomorphism from G to H_Δ (and this argumentation translates straightforwardly to the node-edge-checkability formalism). Hence, to obtain the statement that each problem from \mathcal{P} has a nontrivial fixed point relaxation, it suffices to show that, for each $\Delta \geq 3$ the homomorphism problem with target graph H_Δ has a nontrivial fixed point relaxation.

Assume for the remainder of this section that $\Delta \geq 3$ is fixed. In the following, we state the homomorphism problem with target graph H_Δ , introduced in [BCG⁺22, Section 4.1, arXiv version], in the node-edge-checkability formalism. We will refer to this problem as Π^Δ .

As alluded to before, we can regard Π^Δ as a coloring problem where only certain color combinations are allowed for neighboring nodes. The colors of Π^Δ are all pairs $(y, z) \in \{1, \dots, \Delta\} \times \{1, \dots, \Delta\}$, i.e., $\Sigma_{\Pi^\Delta} := \{1, \dots, \Delta\} \times \{1, \dots, \Delta\}$. The node constraint \mathcal{N}_{Π^Δ} contains precisely the configurations $i \dots i$ for all colors $i \in \Sigma$. The edge constraint \mathcal{E}_{Π^Δ} contains all configurations $(y, z) (y', z')$ satisfying

1. $y \neq 1 \neq z, y \neq y',$ and $z \neq z'$ (or, dually, $y' \neq 1 \neq z', y \neq y',$ and $z \neq z'$), or
2. $y = y' = 1, z \neq 1 \neq z',$ and $z \neq z',$ or
3. $z = z' = 1, y \neq 1 \neq y',$ and $y \neq y'.$

As an example, we explicitly provide the node and edge constraint of Π^Δ for $\Delta = 3$ below.

⁶Recall that for all considered lower bound techniques, it suffices to study regular graphs. Here Δ refers to the maximum degree of the (Δ -regular) input instance, not to the maximum degree of H or H_Δ .

\mathcal{N}_{Π^Δ} :	\mathcal{E}_{Π^Δ} :
(1, 1) (1, 1) (1, 1)	[(1, 1)] [(2, 2) (2, 3) (3, 2) (3, 3)]
(1, 2) (1, 2) (1, 2)	[(1, 2)] [(1, 3) (2, 3) (3, 3)]
(1, 3) (1, 3) (1, 3)	[(1, 3)] [(1, 2) (2, 2) (3, 2)]
(2, 1) (2, 1) (2, 1)	[(2, 1)] [(3, 1) (3, 2) (3, 3)]
(2, 2) (2, 2) (2, 2)	[(2, 2)] [(1, 1) (1, 3) (3, 1) (3, 3)]
(2, 3) (2, 3) (2, 3)	[(2, 3)] [(1, 1) (1, 2) (3, 1) (3, 2)]
(3, 1) (3, 1) (3, 1)	[(3, 1)] [(2, 1) (2, 2) (2, 3)]
(3, 2) (3, 2) (3, 2)	[(3, 2)] [(1, 1) (1, 3) (2, 1) (2, 3)]
(3, 3) (3, 3) (3, 3)	[(3, 3)] [(1, 1) (1, 2) (2, 1) (2, 2)]

We devote the rest of this section to proving the following theorem and deriving lower bounds from it.

Theorem 5.1. *There exists a nontrivial fixed point relaxation for Π^Δ .*

As discussed above, Theorem 5.1 implies the following corollary that covers all problems for which lower bounds via Marks' technique have been known but (so far) no lower bounds via round elimination.

Corollary 5.2. *Every problem in \mathcal{P} has a nontrivial fixed point relaxation.*

To prove Theorem 5.1, we make use of Corollary 3.12 where we choose Π to be Π^Δ . Then Part 3 of Corollary 3.12 states that there exists a nontrivial fixed point relaxation of Π^Δ , and hence it suffices to prove part 2 of Corollary 3.12 which we will do in the following.

To this end, we introduce a problem Π^{orcX} , which we will prove to be a round elimination fixed point. Moreover, we will prove that Π^{orcX} is a relaxation of Π^Δ , given a Δ -edge coloring as input. Lastly we need to show that Π^{orcX} is nontrivial given a Δ -edge coloring.

The ORCX problem. In the following we define the aforementioned problem Π^{orcX} . We set the output label set of Π^{orcX} to be $\Sigma_{\Pi^{\text{orcX}}} := \Sigma_1 \cup \Sigma_2$, where $\Sigma_1 := \{\text{O}, \text{R}, \text{C}, \text{X}\}$ and $\Sigma_2 := \{\text{o}, \text{r}, \text{c}, \text{x}\}$. The node constraint $\mathcal{N}_{\Pi^{\text{orcX}}}$ of Π^{orcX} contains precisely those configurations $\text{L}_1 \dots \text{L}_\Delta$ that satisfy that

1. there is precisely one index $1 \leq k \leq \Delta$ such that $\text{L}_k \in \Sigma_1$ and
2. there are two indices $k, k' \in \{1, \dots, \Delta\}$ satisfying $k \neq k'$ such that $\text{L}_{k''} \in \{\text{O}, \text{o}\}$ for each $k'' \in \{1, \dots, \Delta\} \setminus \{k, k'\}$ and
 - (a) either $\text{L}_k \in \{\text{X}, \text{x}\}$ and $\text{L}_{k'} \in \{\text{O}, \text{o}\}$,
 - (b) or $\text{L}_k \in \{\text{R}, \text{r}\}$ and $\text{L}_{k'} \in \{\text{C}, \text{c}\}$.

The edge constraint $\mathcal{E}_{\Pi^{\text{orcX}}}$ of Π^{orcX} contains precisely the following configurations.

[O]	[O R C X]
[O R]	[O R]
[O C]	[O C]
[o]	[o r c x]
[o r]	[o c]

5.2 A 0-round transformation

We now show that problem Π^{orc} can be reduced to Π^Δ if a Δ -edge coloring (with colors $1, \dots, \Delta$) is given as input.

Lemma 5.3. *Given a Δ -edge coloring and a solution to Π^Δ , it is possible to solve Π^{orc} in 0 rounds.*

Proof. Each node v which is labeled (i, j) (i.e., whose incident half-edges are labeled (i, j)) in the solution to Π^Δ produces a solution for Π^{orc} as follows. First, node v marks the incident edge of color i as “my row” and the incident edge of color j as “my column”. Then, on the incident half-edge of color 1, node v outputs the following label:

- X, if the edge is marked both as “my row” and as “my column”. Observe that this output is produced if v is labeled $(1, 1)$.
- R, if the edge is marked as “my row” only. Observe that this output is produced if v is labeled $(1, j)$ for some $j > 1$.
- C, if the edge is marked as “my column” only. Observe that this output is produced if v is labeled $(i, 1)$ for some $i > 1$.
- O, if the edge is not marked.

Then, on any incident half-edge of color $z \in \{2, \dots, \Delta\}$, node v outputs the following label:

- x, if the edge is marked both as “my row” and as “my column”. Observe that this output is produced if v is labeled (z, z) .
- r, if the edge is marked as “my row” only. Observe that this output is produced if v is labeled (z, z') where $z' \neq z$.
- c, if the edge is marked as “my column” only. Observe that this output is produced if v is labeled (z', z) , where $z' \neq z$.
- o, if the edge is not marked.

Since v outputs a label from Σ_1 on the incident half-edge of color 1 but on no other incident half-edge, the node configuration produced by v satisfies Property 1 of the definition of $\mathcal{N}_{\Pi^{\text{orc}}}$. Since v marks exactly one incident edge as “my row” and exactly one incident edge as “my column”, there are $\Delta - 2$ incident half-edges on which v outputs a label from $\{O, o\}$. Moreover, if v marks two different incident edges, then of the two half-edges where v does not output a label from $\{O, o\}$ one receives a label from $\{R, r\}$, and the other a label from $\{C, c\}$, while if v marks the same edge twice, then one of the two “remaining” label comes from $\{X, x\}$, and the other from $\{O, o\}$. Hence, the node configuration produced by v satisfies also Property 2 of the definition of $\mathcal{N}_{\Pi^{\text{orc}}}$.

We now argue that all produced edge configurations are in $\mathcal{E}_{\Pi^{\text{orc}}}$. Let u and v be two neighboring nodes. Assume that u has color (i, j) and v color (i', j') . Let z be the color of the edge connecting u and v . If $z = 1$, the algorithm produces an edge configuration that is contained in [ORCX] [ORCX]. We show that the produced configuration is not in $\{R\ C, R\ X, C\ X, X\ X\}$, which are all the configurations from [ORCX] [ORCX] not contained in $\mathcal{E}_{\Pi^{\text{orc}}}$:

- X X cannot be obtained, since X is only produced by a node labeled $(1, 1)$, but only one node among u and v can have this color.

- $R X$ cannot be obtained, since R is only produced by nodes labeled $(1, j'')$ for some $j'' > 1$, which cannot be neighbors of $(1, 1)$ by the definition of $\mathcal{E}_{\Pi\Delta}$.
- $C X$ cannot be obtained, since C is only produced by nodes labeled $(i'', 1)$ for some $i'' > 1$, which cannot be neighbors of $(1, 1)$ by the definition of $\mathcal{E}_{\Pi\Delta}$.
- $R C$ cannot be obtained, since nodes labeled $(1, j'')$ for some $j'' > 1$ (which are the only nodes that can produce R) cannot be neighbors of nodes labeled $(i'', 1)$ for some $i'' > 1$ (which are the only nodes that can produce C), by the definition of $\mathcal{E}_{\Pi\Delta}$.

Let $z \in \{2, 3\}$. On edges colored z , the algorithm produces an edge configuration that is contained in $[orcx][orcx]$. We show that the produced configuration is not in $\{r r, c c, r x, c x, x x\}$:

- $x x$ cannot be obtained, since x is only produced by a node labeled (z, z) , but only one node among u and v can have this color.
- $r x$ cannot be obtained, since r is only produced by nodes with a label in $\{(z, z') \mid z' \neq z\}$, which cannot be neighbors of (z, z) by the definition of $\mathcal{E}_{\Pi\Delta}$.
- $r r$ cannot be obtained, since r is only produced by nodes with a label in $\{(z, z') \mid z' \neq z\}$, and two such nodes cannot be neighbors by the definition of $\mathcal{E}_{\Pi\Delta}$.
- $c x$ cannot be obtained, since c is only produced by nodes with a label in $\{(z', z) \mid z' \neq z\}$, which cannot be neighbors of (z, z) by the definition of $\mathcal{E}_{\Pi\Delta}$.
- $c c$ cannot be obtained, since c is only produced by nodes with a label in $\{(z', z) \mid z' \neq z\}$, and two such nodes cannot be neighbors by the definition of $\mathcal{E}_{\Pi\Delta}$. \square

5.3 The ORCX problem is a fixed point

In the following we prove that Π^{orcX} is a fixed point under the round elimination framework. More precisely, we prove the following theorem.

Theorem 5.4. $\overline{\mathcal{R}}(\mathcal{R}(\Pi^{orcX})) = \Pi^{orcX}$.

Computing the constraint $\mathcal{E}'_{\Pi^{orcX}}$. We start by computing the set of maximal configurations that satisfy the universal quantifier w.r.t. $\mathcal{E}_{\Pi^{orcX}}$.

Lemma 5.5. *Let $\mathcal{E}'_{\Pi^{orcX}} = \mathcal{E}_{\mathcal{R}(\Pi^{orcX})}$ be the set of maximal configurations that satisfy the universal quantifier w.r.t. $\mathcal{E}_{\Pi^{orcX}}$. Then, $\mathcal{E}'_{\Pi^{orcX}}$ contains exactly the following configurations:*

- $\{O\} \{O, R, C, X\}$
- $\{O, R\} \{O, R\}$
- $\{O, C\} \{O, C\}$
- $\{o\} \{o, r, c, x\}$
- $\{o, r\} \{o, c\}$.

Proof. By the definition of $\mathcal{E}_{\Pi^{orcX}}$, it is clear that all listed configurations satisfy the universal quantifier. Moreover, it is clear that there is no listed configuration that dominates another (different) listed configuration and that each configuration obtainable by taking a configuration from $\mathcal{E}_{\Pi^{orcX}}$ and replacing each label in the configuration by the singleton set containing the label is dominated by a listed configuration. Hence, what remains to show is that there is no maximal configuration

that satisfies the universal quantifier that is not listed above. In order to show this, by Observation 2.11, it is enough to show that, by combining any two configurations of the list we obtain a new configuration that is dominated by some configuration in the list.

Let $\mathcal{C} = L_1 L_2$ and $\mathcal{C}' = L'_1 L'_2$ be two arbitrary (possibly the same) configurations from the list. Let σ be a permutation of $\{1, 2\}$, and let $u \in \{1, 2\}$. Let \mathcal{C}'' be the combination of \mathcal{C} and \mathcal{C}' w.r.t. u and σ . We first observe that, in order for \mathcal{C}'' to be a valid configuration, we must not obtain any empty set, and hence either both \mathcal{C} and \mathcal{C}' contain only labels from $\{O, R, C, X\}$ or they both contain only labels from $\{o, r, c, x\}$. Moreover, by Observation 2.12, it must hold that the union is taken on non-comparable sets.

Hence, in the former case, up to symmetries, the only case to consider is when $\mathcal{C} = \{O, R\} \{O, R\}$, $\mathcal{C}' = \{O, C\} \{O, C\}$, $u = 2$, and σ is the identity function. In this case, we get that $\mathcal{C}'' = \{O\} \{O, R, C\}$, which is dominated by $\{O\} \{O, R, C, X\}$.

In the latter case, up to symmetries, the only case to consider is when $\mathcal{C} = \mathcal{C}' = \{o, r\} \{o, c\}$, $u = 2$, $\sigma(1) = 2$, and $\sigma(2) = 1$. In this case, we get that $\mathcal{C}'' = \{o\} \{o, r, c\}$, which is dominated by $\{o\} \{o, r, c, x\}$. \square

For the sake of readability, we rename the obtained sets of labels as follows.

- $\boxed{O} := \{O\}$
- $\boxed{R} := \{R, O\}$
- $\boxed{C} := \{C, O\}$
- $\boxed{X} := \{O, R, C, X\}$
- $\boxed{o} := \{o\}$
- $\boxed{r} := \{r, o\}$
- $\boxed{c} := \{c, o\}$
- $\boxed{x} := \{o, r, c, x\}$

Under such a renaming, we obtain that $\mathcal{E}'_{\Pi^{\text{orcX}}}$ contains exactly the following configurations.

- $\boxed{O} \boxed{X}$
- $\boxed{R} \boxed{R}$
- $\boxed{C} \boxed{C}$
- $\boxed{o} \boxed{x}$
- $\boxed{r} \boxed{c}$

In the following, let $\Sigma'_{\Pi^{\text{orcX}}} = \{\boxed{O}, \boxed{R}, \boxed{C}, \boxed{X}, \boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\}$.

Computing the constraint $\mathcal{N}'_{\Pi^{\text{orcX}}}$. We now compute $\mathcal{N}'_{\Pi^{\text{orcX}}}$, that is, we apply the existential quantifier.

Lemma 5.6. *The constraint $\mathcal{N}'_{\Pi^{\text{orcX}}} = \mathcal{N}_{\mathcal{R}(\Pi^{\text{orcX}})}$ contains precisely those configurations $L_1 \dots L_\Delta$ that satisfy that*

1. *there is precisely one index $1 \leq k \leq \Delta$ such that $L_k \in \{\boxed{O}, \boxed{R}, \boxed{C}, \boxed{X}\}$ and*
2. *there are two indices $k, k' \in \{1, \dots, \Delta\}$ satisfying $k \neq k'$ such that*
 - (a) *either $L_k \in \{\boxed{X}, \boxed{x}\}$*
 - (b) *or $L_k \in \{\boxed{R}, \boxed{X}, \boxed{r}, \boxed{x}\}$ and $L_{k'} \in \{\boxed{C}, \boxed{X}, \boxed{c}, \boxed{x}\}$.*

Proof. The lemma follows directly by Observation 2.10. \square

Computing the constraint $\mathcal{N}_{\Pi^{\text{orcX}}}''$. We now compute the set of maximal configurations that satisfy the universal quantifier w.r.t. $\mathcal{N}_{\Pi^{\text{orcX}}}'$.

Lemma 5.7. *Let $\mathcal{N}_{\Pi^{\text{orcX}}}'' = \mathcal{N}_{\overline{\mathcal{R}}(\mathcal{R}(\Pi^{\text{orcX}}))}$ be the set of maximal configurations that satisfy the universal quantifier w.r.t. $\mathcal{N}_{\Pi^{\text{orcX}}}'$. Then, $\mathcal{N}_{\Pi^{\text{orcX}}}''$ contains precisely those configurations $L_1 \dots L_\Delta$ that satisfy that*

1. *there is precisely one $1 \leq k \leq \Delta$ such that $L_k \in \{\{\overline{\text{O}}, \text{R}, \text{C}, \text{X}\}, \{\text{R}, \text{X}\}, \{\text{C}, \text{X}\}, \{\text{X}\}\}$ and*
2. *there are two indices $k, k' \in \{1, \dots, \Delta\}$ satisfying $k \neq k'$ such that $L_{k''}$ is contained in $\{\{\overline{\text{O}}, \text{R}, \text{C}, \text{X}\}, \{\underline{\text{O}}, \text{r}, \text{c}, \text{x}\}\}$ for each $k'' \in \{1, \dots, \Delta\} \setminus \{k, k'\}$ and*
 - (a) *either $L_k \in \{\{\text{X}\}, \{\underline{\text{x}}\}\}$ and $L_{k'} \in \{\{\overline{\text{O}}, \text{R}, \text{C}, \text{X}\}, \{\underline{\text{O}}, \text{r}, \text{c}, \text{x}\}\}$,*
 - (b) *or $L_k \in \{\{\text{R}, \text{X}\}, \{\text{r}, \text{x}\}\}$ and $L_{k'} \in \{\{\text{C}, \text{X}\}, \{\text{c}, \text{x}\}\}$.*

Proof. By the definition of $\mathcal{N}_{\Pi^{\text{orcX}}}''$, it is clear that all listed configurations (i.e., all configurations characterized in the lemma) satisfy the universal quantifier. Moreover, it is clear that there is no listed configuration that dominates another (different) listed configuration and that each configuration obtainable by taking a configuration from $\mathcal{N}_{\Pi^{\text{orcX}}}'$ and replacing each label in the configuration by the singleton set containing the label is dominated by a listed configuration. Hence, what remains to show is that there is no maximal configuration that satisfies the universal quantifier that is not listed above. In order to show this, by Observation 2.11, it is enough to show that, by combining any two configurations of the list we obtain a new configuration that is dominated by some configuration in the list.

Let $\mathcal{C} = L_1 \dots L_\Delta$ and $\mathcal{C}' = L'_1 \dots L'_\Delta$ be two arbitrary (possibly the same) configurations from the list. Let σ be a permutation of $\{1, \dots, \Delta\}$, and let $u \in \{1, \dots, \Delta\}$. Let \mathcal{C}'' be the combination of \mathcal{C} and \mathcal{C}' w.r.t. u and σ . We first observe that, in order for \mathcal{C}'' to be a valid configuration, we must not obtain any empty set, and hence L_i and $L_{\sigma(i)}$ must either be both subsets of $\{\overline{\text{O}}, \text{R}, \text{C}, \text{X}\}$ or both subsets of $\{\underline{\text{O}}, \text{r}, \text{c}, \text{x}\}$, for all $i \neq u$. Due to Property 1 in the lemma, this behavior holds also for $i = u$.

Moreover, by Observation 2.12, it must hold that L_u and $L_{\sigma(u)}$ are non-comparable sets. We thus get that, up to symmetries, $L_u = \{\text{R}, \text{X}\}$ and $L'_{\sigma(u)} = \{\text{C}, \text{X}\}$ or that $L_u = \{\text{r}, \text{x}\}$ and $L'_{\sigma(u)} = \{\text{c}, \text{x}\}$.

In the former case, up to symmetries, we must have

$$\begin{aligned} \mathcal{C} &= \{\text{R}, \text{X}\} \{\text{c}, \text{x}\} \{\underline{\text{O}}, \text{r}, \text{c}, \text{x}\} \dots \{\underline{\text{O}}, \text{r}, \text{c}, \text{x}\} \text{ and} \\ \mathcal{C}' &= \{\text{C}, \text{X}\} \{\text{r}, \text{x}\} \{\underline{\text{O}}, \text{r}, \text{c}, \text{x}\} \dots \{\underline{\text{O}}, \text{r}, \text{c}, \text{x}\}. \end{aligned}$$

Depending on the choice of σ we obtain that either

$$\mathcal{C}'' = \{\overline{\text{O}}, \text{R}, \text{C}, \text{X}\} \{\text{X}\} \{\underline{\text{O}}, \text{r}, \text{c}, \text{x}\} \dots \{\underline{\text{O}}, \text{r}, \text{c}, \text{x}\},$$

which is in the list, or

$$\mathcal{C}'' = \{\overline{\text{O}}, \text{R}, \text{C}, \text{X}\} \{\text{c}, \text{x}\} \{\text{r}, \text{x}\} \{\underline{\text{O}}, \text{r}, \text{c}, \text{x}\} \dots \{\underline{\text{O}}, \text{r}, \text{c}, \text{x}\},$$

which is also in the list.

In the latter of the two cases discussed above, we have to consider four cases:

- We have

$$\mathcal{C} = \mathcal{C}' = \{\boxed{O}, \boxed{R}, \boxed{C}, \boxed{X}\} \{\boxed{r}, \boxed{x}\} \{\boxed{c}, \boxed{x}\} \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \dots \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\}.$$

In this case, the resulting configuration \mathcal{C}'' is dominated by

$$\begin{aligned} & \{\boxed{O}, \boxed{R}, \boxed{C}, \boxed{X}\} \{\boxed{x}\} \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \dots \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \text{ or} \\ & \{\boxed{O}, \boxed{R}, \boxed{C}, \boxed{X}\} \{\boxed{r}, \boxed{x}\} \{\boxed{c}, \boxed{x}\} \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \dots \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\}. \end{aligned}$$

- We have

$$\begin{aligned} \mathcal{C} &= \{\boxed{R}, \boxed{X}\} \{\boxed{c}, \boxed{x}\} \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \dots \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \text{ and} \\ \mathcal{C}' &= \{\boxed{C}, \boxed{X}\} \{\boxed{r}, \boxed{x}\} \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \dots \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\}. \end{aligned}$$

In this case, \mathcal{C}'' is dominated by $\{\boxed{X}\} \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \dots \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\}.$

- We have

$$\begin{aligned} \mathcal{C} &= \{\boxed{R}, \boxed{X}\} \{\boxed{c}, \boxed{x}\} \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \dots \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \text{ and} \\ \mathcal{C}' &= \{\boxed{O}, \boxed{R}, \boxed{C}, \boxed{X}\} \{\boxed{r}, \boxed{x}\} \{\boxed{c}, \boxed{x}\} \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \dots \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\}. \end{aligned}$$

In this case, \mathcal{C}'' is dominated by \mathcal{C} .

- We have

$$\begin{aligned} \mathcal{C} &= \{\boxed{C}, \boxed{X}\} \{\boxed{r}, \boxed{x}\} \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \dots \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \text{ and} \\ \mathcal{C}' &= \{\boxed{O}, \boxed{R}, \boxed{C}, \boxed{X}\} \{\boxed{r}, \boxed{x}\} \{\boxed{c}, \boxed{x}\} \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\} \dots \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\}. \end{aligned}$$

This case is symmetric to the previous one, i.e., again \mathcal{C}'' is dominated by \mathcal{C} . □

For the sake of readability, we rename the obtained sets of labels as follows:

- $\boxed{O} := \{\boxed{O}, \boxed{R}, \boxed{C}, \boxed{X}\}$
- $\boxed{R} := \{\boxed{R}, \boxed{X}\}$
- $\boxed{C} := \{\boxed{C}, \boxed{X}\}$
- $\boxed{X} := \{\boxed{X}\}$
- $\boxed{o} := \{\boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\}$
- $\boxed{r} := \{\boxed{r}, \boxed{x}\}$
- $\boxed{c} := \{\boxed{c}, \boxed{x}\}$
- $\boxed{x} := \{\boxed{x}\}$

In the following, let $\Sigma''_{\Pi^{\text{orcX}}} = \{\boxed{O}, \boxed{R}, \boxed{C}, \boxed{X}, \boxed{o}, \boxed{r}, \boxed{c}, \boxed{x}\}.$

Computing the constraint $\mathcal{E}''_{\Pi^{\text{orcX}}}$. We now compute $\mathcal{E}''_{\Pi^{\text{orcX}}}$, that is we apply the existential quantifier.

Lemma 5.8. *The constraint $\mathcal{E}''_{\Pi^{\text{orcX}}} = \mathcal{E}_{\overline{\mathcal{R}}(\mathcal{R}(\Pi^{\text{orcX}}))}$ contains exactly the following condensed configurations.*

- $[\boxed{O}] [\boxed{O} \boxed{R} \boxed{C} \boxed{X}]$

- $\begin{bmatrix} \text{R} \end{bmatrix} \begin{bmatrix} \text{O} \end{bmatrix} \begin{bmatrix} \text{R} \end{bmatrix} \begin{bmatrix} \text{O} \end{bmatrix}$
- $\begin{bmatrix} \text{C} \end{bmatrix} \begin{bmatrix} \text{O} \end{bmatrix} \begin{bmatrix} \text{C} \end{bmatrix} \begin{bmatrix} \text{O} \end{bmatrix}$
- $\begin{bmatrix} \text{O} \end{bmatrix} \begin{bmatrix} \text{O} \end{bmatrix} \begin{bmatrix} \text{r} \end{bmatrix} \begin{bmatrix} \text{c} \end{bmatrix} \begin{bmatrix} \text{x} \end{bmatrix}$
- $\begin{bmatrix} \text{r} \end{bmatrix} \begin{bmatrix} \text{O} \end{bmatrix} \begin{bmatrix} \text{c} \end{bmatrix} \begin{bmatrix} \text{O} \end{bmatrix}$

Proof. The lemma directly follows from Observation 2.10. \square

Proof of Theorem 5.4. In Lemma 5.5 and Lemma 5.6 we computed $\mathcal{R}(\Pi^{\text{orcX}})$, while in Lemma 5.7 and Lemma 5.8 we computed $\overline{\mathcal{R}}(\mathcal{R}(\Pi^{\text{orcX}}))$. Theorem 5.4 follows by the fact that the equivalence of $\mathcal{N}_{\Pi^{\text{orcX}}}''$ with $\mathcal{N}_{\Pi^{\text{orcX}}}$ and the equivalence of $\mathcal{E}_{\Pi^{\text{orcX}}}''$ with $\mathcal{E}_{\Pi^{\text{orcX}}}$ is witnessed by the renaming $\begin{bmatrix} \ell \end{bmatrix} \mapsto \ell$. \square

5.4 The ORCX problem is non-trivial

We previously showed that Π^{orcX} is a fixed point under the round elimination framework. We now show that it is a nontrivial fixed point, even when a Δ -edge coloring is given as input.

Lemma 5.9. *The problem Π^{orcX} cannot be solved in 0 rounds in the PN model, even if a Δ -edge coloring is given.*

Proof. Consider a graph in which, for each node v , the i -th port of v is assigned to the edge of color i incident to v . In such a graph, any 0-round deterministic PN algorithm must pick a node configuration $L_1 \dots L_\Delta$ from $\mathcal{N}_{\Pi^{\text{orcX}}}$ and output it on all nodes, such that label L_i is output on the edge of color $\sigma(i)$, for some permutation σ . Observe that, if a node v assigns label L to an incident edge $\{v, u\}$, then u is also assigning L to the edge $\{v, u\}$. Hence, for this algorithm to be correct, all the edge configurations in $\{L_1 L_1, \dots, L_\Delta L_\Delta\}$ must be contained in $\mathcal{E}_{\Pi^{\text{orcX}}}$.

Observe that all node configurations containing any of the labels r, c, x cannot be used by the algorithm, as they would create an edge configuration from $\{r r, c c, x x\}$, neither of which is contained in $\mathcal{E}_{\Pi^{\text{orcX}}}$. The only remaining node configuration is $X o \dots o$. However, using such a configuration would create the edge configuration XX , which is not contained in $\mathcal{E}_{\Pi^{\text{orcX}}}$. \square

5.5 Putting things together

We now combine the statements proved above to prove Theorem 5.1 along the lines discussed before.

Proof of Theorem 5.1. By combining Theorem 5.4 (that shows that Π^{orcX} is a fixed point) and Lemma 5.9 (that shows that Π^{orcX} is nontrivial even under the assumed input), we obtain that Π^{orcX} is a nontrivial fixed point, even if a Δ -edge coloring is given. By Lemma 5.3, in Δ -edge colored graphs, it is possible to convert a solution for Π^Δ into a solution for Π^{orcX} , in 0 rounds. Thus, given a Δ -edge coloring, Π^{orcX} is a relaxation of Π^Δ . It follows that, in the setting where a Δ -coloring is given as input, there exists a nontrivial fixed point relaxation of Π^Δ . Now Corollary 3.12 implies that there also exists a nontrivial fixed point relaxation of Π^Δ in the setting without input. \square

By Corollary 6.5, we obtain the desired lower bounds for any problem from \mathcal{P} .

Corollary 5.10. *Each problem $\Pi \in \mathcal{P}$ requires $\Omega(\log n)$ rounds to be solved by a deterministic algorithm and $\Omega(\log \log n)$ rounds to be solved by a randomized algorithm. In particular, for any $\Delta \geq 3$, problem Π^Δ requires $\Omega(\log n)$, resp. $\Omega(\log \log n)$, rounds to be solved deterministically, resp. randomized.*

6 Lifting with inputs

In order to obtain a lower bound result using the round elimination technique, one has to first produce a sequence of problems, $\Pi_0, \Pi_1, \Pi_2, \dots, \Pi_T$, such that Π_0 is the problem for which we want to prove a lower bound, Π_T is nontrivial, and for all i it holds that Π_{i+1} is a relaxation of $\overline{\mathcal{R}}(\mathcal{R}(\Pi_i))$. Once such a sequence is obtained, a lower bound in the LOCAL model is automatically obtained by applying a so-called lifting theorem [BBKO22]. The strongest lifting theorem that is currently known works in regular graphs in which either no input is provided or the input comes from some specific class. However, the input that we assumed in Section 4 does not belong to this class. Thus, in this section, we prove a stronger lifting theorem. We actually prove a lifting theorem that is stronger than what we need for the purposes of this paper: the new lifting theorem that we prove holds also in non-regular graphs and also in the case where the input can be an arbitrary problem expressed in the black-white formalism.

6.1 Preliminaries

In this section, we consider a class of problems that is strictly larger than the one defined in Section 2. For this reason, we define such a class of problems, and we then define the functions \mathcal{R} and $\overline{\mathcal{R}}$ for this setting.

Problems with input in the black-white formalism. In more detail, we define problems on bipartite graphs that are not necessarily regular and that have constraints that can depend on some given input. Note that this setting contains the problems defined in Section 2: a graph can be seen as a bipartite 2-colored graph, where nodes are white nodes and edges are black nodes of degree 2.

Definition 6.1 (Problems with input in the black-white formalism). A problem Π with input in the black-white formalism is a tuple $(\Sigma_{\text{in}}, \Sigma_{\text{out}}, \mathcal{N}, \mathcal{E}, g)$, where:

- Σ_{in} and Σ_{out} are finite sets representing, respectively, the input and output labels of Π .
- $\mathcal{N} = (\mathcal{N}^1, \dots, \mathcal{N}^\Delta)$ consists of Δ collections \mathcal{N}^i of cardinality- i multisets $\{\ell_1, \dots, \ell_i\}$ with ℓ_1, \dots, ℓ_i from Σ_{out} . \mathcal{N} is the *white constraint* of Π . \mathcal{N}^i represents the constraint of white nodes of degree i .
- $\mathcal{E} = (\mathcal{E}^1, \dots, \mathcal{E}^\delta)$ consists of Δ collections \mathcal{E}^i of cardinality- i multisets $\{\ell_1, \dots, \ell_i\}$ with ℓ_1, \dots, ℓ_i from Σ_{out} . \mathcal{E} is the *black constraint* of Π . \mathcal{E}^i represents the constraint of black nodes of degree i .
- $g : \Sigma_{\text{in}} \rightarrow 2^{\Sigma_{\text{out}}}$ is a function that assigns to each label from Σ_{in} a subset of labels from Σ_{out} .

The *degree* of Π is defined as the maximum between Δ and δ . Let $G = (U, V, E)$ be a 2-colored bipartite graph. Let U be the set of white nodes and V be the set of black nodes. Assume that white nodes have maximum degree Δ and black nodes have maximum degree δ , and that to each edge e is assigned a label $f_{\text{in}}(e)$ from Σ_{in} . A correct solution for Π on G is given by an edge labeling $f_{\text{out}} : E(G) \rightarrow \Sigma_{\text{out}}$ such that the following holds:

- For each white node u , the multiset consisting of the labels assigned by f_{out} to the edges incident to u is contained in $\mathcal{N}^{\deg(u)}$.
- For each black node v , the multiset consisting of the labels assigned by f_{out} to the edges incident to v is contained in $\mathcal{E}^{\deg(v)}$.
- For every edge e , the label $f_{\text{out}}(e)$ is contained in the label set $g(f_{\text{in}}(e))$.

White and black algorithms. In order to apply round elimination for problems in the black-white formalism, we need to introduce the concept of white and black algorithms. A white algorithm for Π running on $G = (U, V, E)$ is an algorithm that is executed by white nodes, and in which black nodes act as passive relayers of messages. In such a setting, the output is only produced by white nodes. The output of a white node u defines $f_{\text{out}}(e)$ for each edge e incident to u . The runtime complexity of an algorithm is defined such that there is a natural mapping between the runtime and the distance at which a node sees on G . For this reason, if two white nodes u and u' are incident to the same black node v , then we define the number of rounds that it takes for u and u' to communicate as 2. A black algorithm is defined in the same way, but by reversing the roles of white and black nodes.

Round elimination. We now define $\mathcal{R}(\Pi)$ and $\overline{\mathcal{R}}(\Pi)$ for problems Π with input in the black-white formalism. Note that round elimination in the case of problems with input has already been introduced in [GRB22], and we now report its trivial generalization to the case in which the black nodes have a degree that is not necessarily 2.

Definition 6.2 (The function \mathcal{R}). Let $\Pi = (\Sigma_{\text{in}}^{\Pi}, \Sigma_{\text{out}}^{\Pi}, \mathcal{N}_{\Pi} = (\mathcal{N}_{\Pi}^1, \dots, \mathcal{N}_{\Pi}^{\Delta}), \mathcal{E}_{\Pi} = (\mathcal{E}_{\Pi}^1, \dots, \mathcal{E}_{\Pi}^{\delta}), g_{\Pi})$ be a problem with input in the black-white formalism. Then, we define the problem $\mathcal{R}(\Pi) = (\Sigma_{\text{in}}^{\mathcal{R}(\Pi)}, \Sigma_{\text{out}}^{\mathcal{R}(\Pi)}, \mathcal{N}_{\mathcal{R}(\Pi)}, \mathcal{E}_{\mathcal{R}(\Pi)}, g_{\mathcal{R}(\Pi)})$ as follows:

- $\Sigma_{\text{in}}^{\mathcal{R}(\Pi)} := \Sigma_{\text{in}}^{\Pi}$.
- $\mathcal{E}_{\mathcal{R}(\Pi)}$ is defined as $(\mathcal{E}_{\mathcal{R}(\Pi)}^1, \dots, \mathcal{E}_{\mathcal{R}(\Pi)}^{\delta})$, where $\mathcal{E}_{\mathcal{R}(\Pi)}^i$ contains all the multisets $\{\mathbf{L}_1, \dots, \mathbf{L}_i\}$, where $\mathbf{L}_i \subseteq 2^{\Sigma_{\text{out}}^{\Pi}}$, such that:
 - For any choice $(\ell_1, \dots, \ell_i) \in \mathbf{L}_1 \times \dots \times \mathbf{L}_i$, the multiset $\{\ell_1, \dots, \ell_i\}$ is in \mathcal{E}_{Π}^i ;
 - By adding any label in $\Sigma_{\text{out}}^{\Pi}$ to any set \mathbf{L}_i , the above condition is not satisfied anymore.
- $\Sigma_{\text{out}}^{\mathcal{R}(\Pi)} := \bigcup_{i=1}^{\delta} \bigcup_{\mathcal{C} \in \mathcal{E}_{\mathcal{R}(\Pi)}^i} \mathcal{C}$. That is, the set of output labels of $\mathcal{R}(\Pi)$ is the set of sets that appear in at least one configuration contained in $\mathcal{E}_{\mathcal{R}(\Pi)}$.
- $\mathcal{N}_{\mathcal{R}(\Pi)}$ is defined as $(\mathcal{N}_{\mathcal{R}(\Pi)}^1, \dots, \mathcal{N}_{\mathcal{R}(\Pi)}^{\Delta})$, where $\mathcal{N}_{\mathcal{R}(\Pi)}^i$ contains all the multisets $\{\mathbf{L}_1, \dots, \mathbf{L}_i\}$, where $\mathbf{L}_i \in \Sigma_{\text{out}}^{\mathcal{R}(\Pi)}$, such that there exists a choice $(\ell_1, \dots, \ell_i) \in \mathbf{L}_1 \times \dots \times \mathbf{L}_i$ satisfying that the multiset $\{\ell_1, \dots, \ell_i\}$ is in \mathcal{N}_{Π}^i ;
- For each $\ell \in \Sigma_{\text{in}}^{\Pi}$, $g_{\mathcal{R}(\Pi)}$ is defined as $g_{\mathcal{R}(\Pi)}(\ell) := 2^{g_{\Pi}(\ell)} \cap \Sigma_{\text{out}}^{\mathcal{R}(\Pi)}$. In other words, a label of $\Sigma_{\text{out}}^{\mathcal{R}(\Pi)}$, which is a subset of $\Sigma_{\text{out}}^{\Pi}$, is allowed on input ℓ if and only if all its elements are allowed with ℓ according to the function g_{Π} .

Definition 6.3 (The function $\overline{\mathcal{R}}$). The function $\overline{\mathcal{R}}(\Pi)$ is defined analogously as \mathcal{R} , except that the roles of \mathcal{N} and \mathcal{E} are reversed. In more detail, let $\Pi = (\Sigma_{\text{in}}^{\Pi}, \Sigma_{\text{out}}^{\Pi}, \mathcal{N}_{\Pi}, \mathcal{E}_{\Pi}, g_{\Pi})$, and let $\Pi' = (\Sigma_{\text{in}}^{\Pi}, \Sigma_{\text{out}}^{\Pi}, \mathcal{E}_{\Pi}, \mathcal{N}_{\Pi}, g_{\Pi})$. Let $\mathcal{R}(\Pi') = (\Sigma_{\text{in}}^{\mathcal{R}(\Pi')}, \Sigma_{\text{out}}^{\mathcal{R}(\Pi')}, \mathcal{E}_{\mathcal{R}(\Pi')}, \mathcal{N}_{\mathcal{R}(\Pi')}, g_{\mathcal{R}(\Pi')})$. The problem $\overline{\mathcal{R}}(\Pi)$ is defined as $(\Sigma_{\text{in}}^{\overline{\mathcal{R}}(\Pi)}, \Sigma_{\text{out}}^{\overline{\mathcal{R}}(\Pi)}, \mathcal{N}_{\overline{\mathcal{R}}(\Pi)}, \mathcal{E}_{\overline{\mathcal{R}}(\Pi)}, g_{\overline{\mathcal{R}}(\Pi)})$.

Relaxations. The definition of relaxations, provided in Section 2, directly extend to the setting considered in this section. In particular, a problem Π' is a white (resp. black) relaxation of a problem Π if there exists a white (resp. black) 0-round algorithm that solves Π' given a solution for Π , where the 0-round algorithm can exploit the given input.

6.2 The extended lifting theorem

Theorem 6.4. *Let $\Pi_0, \Pi_1, \dots, \Pi_k$ be a sequence of problems with input in the black-white formalism, where input labels come from Σ_{in} . Let Π_{in} be a problem (without inputs) in the black-white formalism where output labels come from Σ_{in} . Assume that, for all $0 \leq i < k$, and for an integer L , the following holds:*

- *There exists a problem Π'_i that is a black relaxation of $\mathcal{R}(\Pi_i)$;*
- *Π_{i+1} is a white relaxation of $\overline{\mathcal{R}}(\Pi'_i)$;*
- *The number of input and output labels of Π_i , and the ones of Π'_i , are upper bounded by L .*

Also, assume that Π_k has at most L input and output labels and is not 0-round solvable in the deterministic PN model, even if a solution for Π_{in} is given as input. Let Δ be the degree of Π_0 . Then, for all n , on trees of n nodes and maximum degree at most Δ , Π_0 requires at least

$$\min\{k - 1, \frac{1}{16}(\log_{\Delta} n - \log_{\Delta} \log L - 624)\}$$

rounds in the deterministic LOCAL model and at least

$$\min\{k - 1, \frac{1}{16}(\log_{\Delta} \log n - \log_{\Delta} \log L - 5)\}$$

rounds in the randomized LOCAL model, even if a solution for Π_{in} is given as input.

Before proving the theorem, we discuss how to use such a theorem to obtain lower bounds starting from sequences of problems. The following corollary shows that a nontrivial fixed point directly implies a lower bound.

Corollary 6.5. *Let Π be a problem in the black-white formalism satisfying $\overline{\mathcal{R}}(\mathcal{R}(\Pi)) = \Pi$, and where inputs come from Σ_{in} . Let Π_{in} be a problem (without inputs) in the black-white formalism where output labels come from Σ_{in} . Assume that Π is not 0-round solvable in the deterministic PN model, even if a solution for Π_{in} is given as input. Then, on trees of n nodes and maximum degree at most $\Delta = O(1)$, Π requires $\Omega(\log n)$ rounds in the deterministic LOCAL model and $\Omega(\log \log n)$ rounds in the randomized LOCAL model, even if a solution for Π_{in} is given as input.*

Proof. Since Π is a nontrivial fixed point, for any t we can construct a sequence of length t satisfying the requirements of Theorem 6.4. Hence, the lower bounds of Theorem 6.4 are determined by the second terms in the min operator. Let L be the maximum between the number of input and output labels of Π , and let Δ be the degree of Π . Observe that L may depend on Δ but cannot depend on n . Since Δ is a constant that solely depends on Π , the claim follows. \square

A similar statement holds for infinite sequences, assuming that the number of labels does not grow too fast. The following corollary considers the case in which the number of labels grows linearly, but it can be easily generalized to other cases.

Corollary 6.6. *Let Π_0, Π_1, \dots be an infinite sequence of problems with input in the black-white formalism, where input labels come from Σ_{in} . Let Π_{in} be a problem (without inputs) in the black-white formalism where output labels come from Σ_{in} . Assume that, for all $i \geq 0$, and for some constant c , the following holds:*

- *there exists a problem Π'_i that is a black relaxation of $\mathcal{R}(\Pi_i)$,*
- *Π_{i+1} is a white relaxation of $\overline{\mathcal{R}}(\Pi'_i)$,*
- *the number of input and output labels of Π_i , and the ones of Π'_i , are upper bounded by $c \cdot (i+1)$, and*

- Π_i is not 0-round solvable in the deterministic PN model, even if a solution for Π_{in} is given as input.

Let Δ be the degree of Π_0 . Then, for all n , on trees of n nodes and maximum degree at most Δ , Π_0 requires $\Omega(\log n)$ rounds in the deterministic LOCAL model and $\Omega(\log \log n)$ rounds in the randomized LOCAL model, even if a solution for Π_{in} is given as input.

Proof. For all i , we apply Theorem 6.4 on the prefix sequence of length $i + 1$. We obtain that, for all i and for all n , Π_0 requires at least $\min\{i - 1, \frac{1}{16}(\log_{\Delta} n - \log_{\Delta} \log(c(i + 1))) - 624\}$ rounds in the deterministic LOCAL model and at least $\min\{i - 1, \frac{1}{16}(\log_{\Delta} \log n - \log_{\Delta} \log(c(i + 1))) - 5\}$ rounds in the randomized LOCAL model, even if a solution for Π_{in} is given as input. Hence, for each n , we obtain infinite different lower bounds, parameterized by i . For each fixed n , we can obtain an asymptotically optimal lower bound by considering the case in which $i = \log n$, obtaining a lower bound of $\Omega(\log n)$ for deterministic algorithms and $\Omega(\log \log n)$ for randomized ones. (Note that, similarly as in Corollary 6.5, Δ is a constant that depends solely on Π_0 .) \square

In some cases, we just know that an infinite sequence exists, but we have no bound on the number of labels. In such a case, we directly obtain a lower bound of $\Omega(\log^* n)$. A similar statement holds for infinite sequences, assuming that the number of labels does not grow too fast. The following corollary considers the case in which the number of labels grows linearly, but it can be easily generalized to other cases.

Corollary 6.7. *Let Π_0, Π_1, \dots be an infinite sequence of problems with input in the black-white formalism, where input labels come from Σ_{in} . Let Π_{in} be a problem (without inputs) in the black-white formalism where output labels come from Σ_{in} . Assume that, for all $i \geq 0$, the following holds:*

- *there exists a problem Π'_i that is a black relaxation of $\mathcal{R}(\Pi_i)$,*
- *Π_{i+1} is a white relaxation of $\overline{\mathcal{R}}(\Pi'_i)$, and*
- *Π_i is not 0-round solvable in the deterministic PN model, even if a solution for Π_{in} is given as input.*

Let Δ be the degree of Π_0 . Then, for all n , on trees of n nodes and maximum degree at most Δ , Π_0 requires $\Omega(\log^* n)$ rounds for both randomized and deterministic LOCAL algorithms, even if a solution for Π_{in} is given as input.

Proof. By the definition of \mathcal{R} and $\overline{\mathcal{R}}$, for all i , the number of labels of Π_{i+1} is at most doubly exponentially larger than the number of labels of Π_i . Hence, an upper bound on the number of labels of problem Π_i is roughly a power tower of height $2i$.

Similarly as in the proof of Corollary 6.6, we apply Theorem 6.4 on each prefix sequence of length $i + 1$, and for each n we then optimize the lower bound by considering the case $i = \frac{1}{2} \log^* n - O(1)$. \square

We now consider the case in which we want to prove a lower bound as a function of Δ . In this case, we usually define a problem \mathcal{P} (e.g., MIS) as a family of problems Π^j (e.g., MIS ^{j}), one for each degree j , and we create a round elimination sequence for each such problem. In the case of inputs, we additionally assume that for each j , in graphs of degree j , we receive as input a solution for Π_{in}^j . The following corollary captures exactly this case.

Corollary 6.8. *Let $\mathcal{P} = (\Pi^j \mid j > 0)$ be a family of problems, where Π^j is a problem of degree j in the black-white formalism, with inputs from $\Sigma_{\text{in}}^{\Pi^j}$. Let $\mathcal{P}_{\text{in}} = (\Pi_{\text{in}}^j \mid j > 0)$ be a family of problems (without inputs) in the black-white formalism where output labels come from $\Sigma_{\text{in}}^{\Pi^j}$. Suppose that,*

for each j , there exists a sequence of problems starting from Π^j satisfying the requirements of Theorem 6.4 w.r.t. input Π_{in}^j and of length $f(j)$, for some f . Moreover, suppose that, for each j , the number of labels of Π^j is at most 2^{j^c} for some universal constant c . Then, \mathcal{P} requires

$$\Omega(\min\{f(\Delta), \log_{\Delta} n\})$$

rounds for deterministic algorithms and

$$\Omega(\min\{f(\Delta), \log_{\Delta} \log n\})$$

for randomized ones, in n -node trees of maximum degree Δ , even if a solution for \mathcal{P}_{in} is given as input.

Proof. The claim follows by applying Theorem 6.4 on each sequence independently. \square

6.3 Lifting theorem in a nutshell

We devote the rest of this section for proving Theorem 6.4. Our proof uses ingredients that are already present in the literature [BBH⁺19, BBO20, BO20, BBE⁺20, BBKO22], and the proofs that we provide in this section are essentially an extension of the proofs present in [BBKO22, Appendix A in the arXiv version]. In the following, we summarize the high-level ideas behind the lifting theorem.

In order to lift a result obtained with round elimination to the LOCAL model, we need to follow the steps listed below.

1. Let Π' be a relaxation of $\overline{\mathcal{R}}(\mathcal{R}(\Pi))$. Prove that, if there exists a randomized k -round algorithm for Π with failure probability p (which could depend on n), then there exists a randomized $(k-1)$ -round algorithm for Π' with some failure probability p' that is not too large compared to p .
2. Prove that, if a problem cannot be solved in 0 rounds in the deterministic PN model, then any 0-rounds randomized PN algorithm for the same problem must fail with some large probability.
3. Apply Item 1 recursively. The number of times that we can apply this recursive procedure before obtaining a failure probability that is larger than the one in Item 2 is a lower bound on the runtime for solving the problem in the randomized PN model. This directly gives a lower bound in the randomized LOCAL model as well.
4. Prove that a randomized lower bound in the LOCAL model implies an even stronger deterministic lower bound.

6.4 Evolution of the failure probability in a single step of round elimination

We first observe that, existing results in the literature, imply that Item 1 holds also in the case in which the graph is not necessarily regular and in the presence of inputs.

Lemma 6.9 (Theorem 3.4 of [GRB22]). *Let \mathcal{F} be the set containing all forests of maximum degree at most Δ . For any set N of positive integers, let \mathcal{F}_N denote the class of forests with a number of nodes that is contained in N . Let Π be a node-edge-checkable LCL problem and \mathcal{A} a randomized algorithm solving Π on \mathcal{F} with runtime $T(n)$ and local failure probability at most $p \leq 1$. Let N be the set of all positive integers n satisfying $T(n) + 2 \leq \log_{\Delta} n$. Then, there exists a randomized*

algorithm \mathcal{A}' solving $\overline{\mathcal{R}}(\mathcal{R}(\Pi))$ on \mathcal{F}_N with runtime $\max\{0, T(n) - 1\}$ and local failure probability at most $Sp^{1/(3\Delta+3)}$, where

$$S = \left(10\Delta(|\Sigma_{\text{in}}| + \max\{|\Sigma_{\text{out}}^\Pi|, |\Sigma_{\text{out}}^{\mathcal{R}(\Pi)}|\})\right)^{4\Delta^{T(n)+1}}.$$

In other words, the above lemma says that, if there exists a randomized k -rounds algorithm for Π that has some failure probability p , then we can construct a $(k-1)$ -rounds algorithm that solves $\overline{\mathcal{R}}(\mathcal{R}(\Pi))$ (and hence also a relaxation of it) with failure probability that is not too large compared to p .

While this lemma holds for graphs, in the following we show that it is straightforward to extend it to the case of 2-colored bipartite graphs and problems expressed in the black-white formalism.

In the following, let Π be a problem in the black-white formalism, let Δ be its degree, let Σ_{in} be the set of input labels of Π , let Σ_{out}^Π be the set of output labels of Π , and let Π_{in} be a problem (without inputs) in the black-white formalism where output labels come from Σ_{in} . Let \mathcal{A} be a randomized white algorithm solving Π on \mathcal{F} with runtime at most $T(n)$ and local failure probability at most $p \leq 1$, when a solution for Π_{in} is given as input⁷.

To simplify the notation, in the following, we state a version of Lemma 6.9 for problems in the black-white formalism, for a fixed choice of n . Fix a value of n , assume that $T(n)$ is even, and let $2T = T(n)$. Assume that $2T + 2 \leq \log_\Delta n$. Moreover, let $\mathcal{F}_{n,\Delta}$ be the set of forests of maximum degree at most Δ and containing n nodes.

Lemma 6.10. *Let Π' be a black relaxation of $\mathcal{R}(\Pi)$, and let $\Sigma_{\text{out}}^{\Pi'}$ be the set of output labels of Π' . Then, there exists a randomized white algorithm \mathcal{A}'' solving $\overline{\mathcal{R}}(\Pi')$ on $\mathcal{F}_{n,\Delta}$ when a solution for Π_{in} is given as input, with runtime $\max\{0, 2T - 2\}$ and local failure probability at most*

$$p'' \leq 2\Delta(s + |\Sigma_{\text{out}}^{\Pi'}|)(p')^{1/(\Delta+1)},$$

where $s = (3|\Sigma_{\text{in}}|)^{2\Delta^{2T+2}}$, and $p' \leq 2\Delta(s + |\Sigma_{\text{out}}^\Pi|)p^{1/(\Delta+1)}$.

Proof. While the original proof is phrased on graphs, for the ease of the presentation we now express it in the case of bipartite 2-colored graphs in which black nodes (which correspond to edges) have degree exactly 2. The proof of [GRB22] works as follows.

Given a white algorithm \mathcal{A} for Π , they first use \mathcal{A} to construct a black algorithm \mathcal{A}' for $\mathcal{R}(\Pi)$ that has complexity $2T - 1$. Observe that \mathcal{A}' solves Π' as well. Then, they use \mathcal{A}' to construct a white algorithm \mathcal{A}'' for $\overline{\mathcal{R}}(\Pi')$ that has complexity $2T - 2$. This construction is done such that the failure probability of \mathcal{A}'' is not too large compared to the one of \mathcal{A} .

The black algorithm \mathcal{A}' running on a black node v is constructed as follows: gather the $(2T-1)$ -radius neighborhood of v , imagine all possible extensions of the gathered neighborhood into a $(2T+1)$ -radius neighborhood one, and for each of them compute the output of \mathcal{A} that the white neighbors of v give on the edges incident to v . The assumption $2T+2 \leq \log_\Delta n$ is used to guarantee that \mathcal{A} does not see more than n nodes, and hence it works correctly. The output of \mathcal{A}' on such edges is the set of labels that \mathcal{A} outputs on a fraction of the extensions that is at least K , for some parameter $0 < K < 1$. The authors of [GRB22], in [GRB22, Lemma 3.7, arXiv version], show that, if the failure probability of \mathcal{A} is p , then the failure probability of \mathcal{A}' is at most $2\Delta(s + |\Sigma_{\text{out}}^\Pi|)p^{1/3}$, where $s = (3|\Sigma_{\text{in}}|)^{2\Delta^{2T+1}}$. This bound is obtained as follows:

⁷In the original paper, it is assumed that algorithm \mathcal{A} is able to solve Π for any input that comes from Σ_{in} . However, it is easy to check that the same proof holds in the case in which it is assumed that the input is a solution for a fixed problem Π_{in} .

- On white nodes, the failure probability is upper bounded by $p + |\Sigma_{\text{out}}^\Pi| \Delta K + ps\Delta/K$.
- On black nodes, the failure probability is upper bounded by ps/K^2 .

Then, the bound is obtained by picking $K = p^{1/3}$. In the above formulas, s is the number of possible extensions, and the exponent 2 on K on the second bound comes from the fact that a black node has 2 white neighbors.

By going through the proof of these lemmas, it is easy to see that they work also in the case in which black nodes have degree at most Δ , by using the following new bound on the failure probability of \mathcal{A}' on black nodes:

$$\frac{p \cdot s}{K\Delta}, \text{ where } s = (3|\Sigma_{\text{in}}|)^{2\Delta^{2T+2}}.$$

Indeed, a black node has at most Δ white neighbors, and the possible extensions of a $(2T-1)$ -radius neighborhood into a $(2T+1)$ -radius neighborhood are at most s :

- There are at most Δ^{2T} black nodes at distance $2T$ from the center, and at most Δ^{2T+1} white nodes at distance $2T$ from the center. Hence, there are at most $(\Delta+1)\Delta^{2T} \leq 2\Delta^{2T+1}$ nodes in the extension.
- There are at most $2\Delta^{2T+2}$ edges incident to the nodes in the extension.
- In order to describe an extension, for each edge, we need to express its label, which comes from a domain of size $|\Sigma_{\text{in}}|$, and one of the following three possibilities:
 - The edge does not exist.
 - The edge exists, and connects the node to another node that is nearer to the center.
 - The edge exists, and connects the node to another node that is further away.
- Hence, up to isomorphism, there are at most $s = (3|\Sigma_{\text{in}}|)^{2\Delta^{2T+2}}$ possible extensions.

By picking $K = p^{\frac{1}{\Delta+1}}$, we obtain that the failure probability of \mathcal{A}' is at most

$$p' \leq 2\Delta(s + |\Sigma_{\text{out}}^\Pi|)p^{1/(\Delta+1)},$$

where $s = (3|\Sigma_{\text{in}}|)^{2\Delta^{2T+2}}$.

The algorithm \mathcal{A}'' is constructed analogously (by considering neighborhoods of radius $2T-2$ and extensions of radius $2T$), and its failure probability is at most

$$p'' \leq 2\Delta(s + |\Sigma_{\text{out}}^{\Pi'}|)(p')^{1/(\Delta+1)},$$

where p' is an upper bound on the failure probability of \mathcal{A}' . Hence, the claim follows. \square

6.5 Zero-round solvability

We now show that Item 2 holds also in the case of (possibly) non-regular graphs and for problems with input in the black-white formalism. In other words, we show that if a problem Π cannot be solved in 0 rounds in the deterministic PN model, then any randomized 0-rounds PN algorithm for Π must fail with some large probability.

Lemma 6.11. *Let Π be a problem with input that cannot be solved in 0 rounds with a deterministic white algorithm in the PN model, even if a solution for Π_{in} is given. Then, any randomized 0-round algorithm solving Π must fail with probability at least $\frac{1}{L^{\Delta^2}}$, even if a solution for Π_{in} is given.*

Proof. In 0 rounds, an algorithm running on node v only sees the input assigned to v , and the ports assigned to v (which includes the degree of v). Hence, we can see any 0-round algorithm as a function mapping each possible input assigned to white nodes and a permutation of a subset of $\{1, \dots, \Delta\}$ into a probability assignment to the white output configurations of Π .

Since Δ is an upper bound on the degree of the nodes, and L is an upper bound on the number of output labels, for each input and port assignment there are at most L^Δ possible output configurations. Hence, there exists a configuration \mathcal{C} to which the algorithm assigns probability at least $\frac{1}{L^\Delta}$.

Since Π is not solvable in 0 rounds with a white algorithm in the deterministic PN model, then for each (deterministic) mapping from input and port assignment to outputs there exists a way to connect (at most Δ) white nodes to a single black node, and assign inputs and ports to these nodes, such that, by applying the considered mapping, we obtain an invalid configuration on the black node.

In the case of randomized algorithms, for each input and port assignment on a white node, there exists a configuration that is given with probability at least $\frac{1}{L^\Delta}$. Hence, it is possible to connect white nodes to a single black node such that the black node obtains an invalid configuration with probability at least $(\frac{1}{L^\Delta})^\Delta = \frac{1}{L^{\Delta^2}}$. \square

6.6 Evolution of the failure probability in multiple steps of round elimination

Lemma 6.12. *Let $\Pi_0, \Pi_1, \dots, \Pi_T$ be a sequence of problems satisfying the conditions of Theorem 6.4. Let Δ be the degree of Π_0 . Let n be an integer, and assume that $2T + 2 \leq \log_\Delta n$. Let \mathcal{A} be a randomized $2T$ -round white algorithm for Π_0 on $\mathcal{F}_{n, \Delta}$ with local failure probability at most p (which may depend on n). Then there exists a randomized $(2T - 2j)$ -round white algorithm \mathcal{A}' for Π_j with local failure probability at most $(2\Delta(s + L))^2 p^{1/(\Delta+1)^{2j}}$, for all $0 < j \leq T$, where $s = (3|\Sigma_{\text{in}}|)^{2\Delta^{2T+2}}$.*

Proof. By assumption, $|\Sigma_{\text{out}}^\Pi|$ and $|\Sigma_{\text{out}}^{\Pi'}|$ are upper bounded by L . Hence, by Lemma 6.10, we obtain:

$$\begin{aligned} p'' &\leq 2\Delta(s + |\Sigma_{\text{out}}^{\Pi'}|)(p')^{1/(\Delta+1)} \\ &\leq 2\Delta(s + L)(p')^{1/(\Delta+1)}. \end{aligned}$$

Similarly,

$$\begin{aligned} p' &\leq 2\Delta(s + |\Sigma_{\text{out}}^\Pi|)p^{1/(\Delta+1)} \\ &\leq 2\Delta(s + L)p^{1/(\Delta+1)}. \end{aligned}$$

Summarizing, we get that:

$$\begin{aligned} p'' &\leq 2\Delta(s + L)(p')^{\frac{1}{\Delta+1}}, \text{ where} \\ p' &\leq 2\Delta(s + L)p^{\frac{1}{\Delta+1}}. \end{aligned}$$

By recursively applying Lemma 6.10, we get the following:

$$p_j \leq 2\Delta(s + L)(p_{j-1})^{\frac{1}{\Delta+1}},$$

where $p_0 = p$ and p_{2j} , are, respectively, the local failure probability bounds for Π_0 and Π_j . Technically, for different values of j , we would have different values for s , since for each step we have a

different runtime. However, the value of s that we are using upper bounds all of them. We prove by induction that, for all $j \geq 0$,

$$p_j \leq (2\Delta(s + L))^2 p^{\frac{1}{(\Delta+1)^j}}.$$

For the base case where $j = 0$, we get that $p \leq (2\Delta(s + L))^2 p$, which clearly holds. Let us assume that the claim holds for j , and let us prove it for $j + 1$. We obtain the following, where the second inequality holds by the inductive hypothesis:

$$\begin{aligned} p_{j+1} &\leq 2\Delta(s + L)(p_j)^{\frac{1}{\Delta+1}} \leq 2\Delta(s + L) \left((2\Delta(s + L))^2 p^{\frac{1}{(\Delta+1)^j}} \right)^{\frac{1}{\Delta+1}} \\ &\leq (2\Delta(s + L))^{1+\frac{2}{\Delta+1}} p^{\frac{1}{(\Delta+1)^{j+1}}} \leq (2\Delta(s + L))^2 p^{\frac{1}{(\Delta+1)^{j+1}}}. \end{aligned} \quad \square$$

We now handle Item 3, by showing that any randomized algorithm that runs in strictly less than $2T$ rounds must fail with large probability.

Lemma 6.13. *Let $\Pi_0, \Pi_1, \dots, \Pi_T$ be a sequence of problems satisfying the conditions of Theorem 6.4. Let Δ be the degree of Π_0 . Let n be an integer, and assume that $2T + 2 \leq \log_\Delta n$. Any algorithm for Π_0 running in strictly less than $2T$ rounds on $\mathcal{F}_{n,\Delta}$ must fail with probability at least $1/L^{\Delta^{16T}}$, even if a solution for Π_{in} is given as input.*

Proof. For any $T' < T$, by applying Lemma 6.12 on the subsequence containing the first $T' + 1$ problems, we get that a white algorithm solving Π_0 in $2T' < 2T$ rounds with local failure probability at most p implies an algorithm solving $\Pi_{T'}$ in 0 rounds with local failure probability at most $(2\Delta(s + L))^2 p^{1/(\Delta+1)^{2T'}}$, where $s = (3|\Sigma_{\text{in}}|)^{2\Delta^{2T'+2}} \leq (3L)^{2\Delta^{2T'+2}}$. Then, since by assumption $\Pi_{T'}$ is not 0-round solvable in the deterministic PN model⁸, by applying Lemma 6.11 we get that

$$(2\Delta(s + L))^2 p^{1/(\Delta+1)^{2T'}} \geq \frac{1}{L^{\Delta^2}},$$

which implies the following:

$$\begin{aligned} p &\geq \left(\frac{1}{L^{\Delta^2} (2\Delta(s + L))^2} \right)^{(\Delta+1)^{2T'}} \geq \left(\frac{1}{L^{\Delta^2} (2\Delta((3L)^{2\Delta^{2T'+2}} + L))^2} \right)^{(\Delta+1)^{2T'}} \\ &\geq \left(\frac{1}{L^{\Delta^2} 4\Delta^2 (4L)^{4\Delta^{2T'+2}}} \right)^{(\Delta+1)^{2T'}} \geq \left(\frac{1}{\Delta^4 L^{\Delta^{2T'+8}}} \right)^{(\Delta+1)^{2T'}} \\ &\geq \left(\frac{1}{L^{\Delta^{2T'+8} + 4\log \Delta}} \right)^{(\Delta+1)^{2T'}} \geq \left(\frac{1}{L^{\Delta^{2T'+11}}} \right)^{(\Delta+1)^{2T'}} \\ &\geq \frac{1}{L^{\Delta^{2T'+11} \cdot (\Delta+1)^{2T'}}} \geq \frac{1}{L^{\Delta^{2T'+11} \cdot \Delta^{2T'+1}}} \\ &\geq \frac{1}{L^{\Delta^{4T'+12}}} \geq \frac{1}{L^{\Delta^{16T'}}} \geq \frac{1}{L^{\Delta^{16T}}} \end{aligned} \quad \square$$

6.7 A lower bound for randomized LOCAL

We now prove that a sequence satisfying the conditions of Theorem 6.4 implies a lower bound for the LOCAL model on *forests*. We will later show that such a lower bound implies a lower bound for *trees* as well.

⁸Observe that, if Π_T is not 0-rounds solvable in the deterministic PN model, then also $\Pi_{T'}$ is not.

Lemma 6.14. *Let $\Pi_0, \Pi_1, \dots, \Pi_T$ be a sequence of problems satisfying the conditions of Theorem 6.4. Let Δ be the degree of Π_0 , and let n be an integer. Any randomized algorithm running on $\mathcal{F}_{n,\Delta}$ in strictly less than*

$$\min\{2T, \frac{1}{8}(\log_\Delta \log n - \log_\Delta \log L - 3)\}$$

rounds must fail with probability $> 1/n$, even if a solution for Π_{in} is given as input.

Proof. Let $2t = 2\lfloor \min\{2T, \frac{1}{8}(\log_\Delta \log n - \log_\Delta \log L - 2)\}/2 \rfloor$. Observe that $2t + 2 \leq \log_\Delta n$. We apply Lemma 6.13 on the subsequence Π_0, \dots, Π_t . We obtain that any algorithm for Π_0 running in strictly less than $2t$ rounds on $\mathcal{F}_{n,\Delta}$ must fail with probability at least $1/L^{\Delta^{16t}}$. Observe that the following holds.

$$\begin{aligned} 2t &< \frac{1}{8}(\log_\Delta \log n - \log_\Delta \log L) \\ \implies 16t \log \Delta &< \log \log n - \log \log L \\ \implies 16t \log \Delta + \log \log L &< \log \log n \\ \implies \Delta^{16t} \log L &< \log n \\ \implies L^{\Delta^{16t}} &< n \\ \implies \frac{1}{L^{\Delta^{16t}}} &> 1/n. \end{aligned}$$

Hence, any algorithm for Π_0 running in strictly less than $2t$ rounds on $\mathcal{F}_{n,\Delta}$ must fail with probability strictly larger than $1/n$. \square

Lemma 6.14 proves a lower bound for forests. We show that such a lower bound implies a lower bound for trees as well. For this purpose, we exploit a lemma already proved in [GRB22]⁹.

Lemma 6.15 (Lemma 3.3 of [GRB22], rephrased). *Let Π be a problem in the black-white formalism that has deterministic (resp. randomized) complexity $T(n)$ on trees. Then, the deterministic (resp. randomized) complexity of Π on forests is at most $2T(n^2) + 2$.*

We are now ready to prove a lower bound for trees, that is, we prove the randomized lower bound of Theorem 6.4. Let $\mathcal{T}_{n,\Delta}$ be the class of trees of n nodes of maximum degree at most Δ .

Lemma 6.16. *Let $\Pi_0, \Pi_1, \dots, \Pi_T$ be a sequence of problems satisfying the conditions of Theorem 6.4. Let Δ be the degree of Π_0 , and let n be an integer. Any randomized algorithm running on $\mathcal{T}_{n,\Delta}$ in strictly less than*

$$\min\{T - 1, \frac{1}{16}(\log_\Delta \log n - \log_\Delta \log L - 5)\}$$

rounds must fail with probability $> 1/n$.

Proof. Let Δ be the degree of Π . Suppose for a contradiction that there exists an algorithm for $\mathcal{T}_{n,\Delta}$ with runtime strictly less than

$$\frac{1}{2}(\min\{2T, \frac{1}{8}(\log_\Delta \log \sqrt{n} - \log_\Delta \log L - 3)\} - 2).$$

Then, by Lemma 6.15, this implies an algorithm for $\mathcal{F}_{n,\Delta}$ with runtime strictly less than

$$\min\{2T, \frac{1}{8}(\log_\Delta \log n - \log_\Delta \log L - 3)\}$$

⁹In [GRB22], this lemma is stated for complexities in $o(\log^* n)$, since it was enough for the purposes of that paper. However it is easy to check that such an assumption is not used in the proof.

rounds, which is a contradiction with Lemma 6.14. Hence, any algorithm for $\mathcal{T}_{n,\Delta}$ requires at least

$$\begin{aligned} & \frac{1}{2}(\min\{2T, \frac{1}{8}(\log_{\Delta} \log \sqrt{n} - \log_{\Delta} \log L - 3)\} - 2) \\ &= \min\{T, \frac{1}{16}(\log_{\Delta} \log \sqrt{n} - \log_{\Delta} \log L - 3)\} - 1 \\ &\geq \min\{T, \frac{1}{16}(\log_{\Delta} \log n - \log_{\Delta} \log L - 4)\} - 1 \\ &\geq \min\{T - 1, \frac{1}{16}(\log_{\Delta} \log n - \log_{\Delta} \log L - 5)\} \end{aligned}$$

rounds. □

6.8 A lower bound for deterministic LOCAL

We now prove the deterministic lower bound of Theorem 6.4.

Lemma 6.17. *Let $\Pi_0, \Pi_1, \dots, \Pi_T$ be a sequence of problems satisfying the conditions of Theorem 6.4. Let Δ be the degree of Π_0 , and let n be an integer. Any deterministic algorithm running on $\mathcal{T}_{n,\Delta}$ requires at least*

$$\min\{T - 1, \frac{1}{16}(\log_{\Delta} \log n - \log_{\Delta} \log L - 624)\}/13$$

rounds.

Proof. Suppose for a contradiction that there exist n and T satisfying the following:

- There exists a sequence of problems $\Pi_0, \Pi_1, \dots, \Pi_T$ satisfying the conditions of Theorem 6.4.
- There exists a deterministic algorithm \mathcal{A} running on $\mathcal{T}_{n,\Delta}$ in strictly less than

$$t = \min\{T - 1, \frac{1}{16}(\log_{\Delta} n - \log_{\Delta} \log L - 624)\}/13$$

rounds.

We show that, under such assumptions, we can construct an algorithm \mathcal{A}' with runtime strictly less than $\min\{T - 1, \frac{1}{16}(\log_{\Delta} n - \log_{\Delta} \log L - 624)\}$ on graphs in $\mathcal{T}_{2^n, \Delta}$. By renaming $n = \log N$, this implies an algorithm with runtime strictly less than $\min\{T - 1, \frac{1}{16}(\log_{\Delta} \log N - \log_{\Delta} \log L - 624)\}$ on graphs in $\mathcal{T}_{N, \Delta}$. Since any deterministic algorithm trivially implies the existence of a randomized algorithm with the same complexity, this contradicts Lemma 6.16.

In order to construct \mathcal{A}' , we use a standard indistinguishability argument. In more detail, in [CKP16, Theorem 6], it is shown that, under some conditions, we can run an algorithm, designed to work on graphs of size n in time t , on graphs of different size in the same runtime t , such that the algorithm still works correctly. The conditions are the following:

- The IDs in each $(t + 1)$ -radius neighborhood are unique and from $\{1, 2, \dots, n\}$;
- Each $(t + 1)$ -radius neighborhood contains at most n nodes.

As we will show later, the second condition is guaranteed by the assumed bound on the runtime of \mathcal{A} . In order to guarantee the first condition, it is sufficient to compute a coloring and use it as a new ID assignment. For this purpose, we use Linial's algorithm, which can convert a given k coloring into a $5\Delta^2 \log k$ coloring in 1 deterministic round [Lin92, Corollary 4.1]. Observe that, at the beginning of the algorithm, we can use the original IDs of the nodes, which come from $\{1, \dots, N\}$, as a coloring. Hence, at first, algorithm \mathcal{A}' runs 3 rounds of Linial's algorithm on G^{2t+2} (the graph obtained by putting an edge between all nodes that are at distance at most $2t + 2$). This

requires at most $6t + 6 \leq 12t$ rounds. Let $\bar{\Delta} = \Delta^{2t+2}$ be an upper bound of the degree of G^{2t+2} . We obtain a coloring with a palette of size at most

$$50\bar{\Delta}^2(\log \log \log N + \log \bar{\Delta}) = 50\Delta^{2(2t+2)}(\log \log n + \log \Delta^{2t+2}).$$

Since $t \leq \frac{1}{16}(\log_{\Delta} n - 624)/13 = \frac{1}{16 \cdot 13} \log_{\Delta} n - 3 \leq \frac{1}{8} \log_{\Delta} n - 3$, we obtain the following:

- Each $(t+1)$ -radius neighborhood contains at most $\Delta^{t+1} < n$ nodes;
- The number of colors is at most

$$\begin{aligned} & 50\Delta^{2(2t+2)}(\log \log n + \log \Delta^{2t+2}) \\ & \leq 50\Delta^{2(2(\frac{1}{8} \log_{\Delta} n - 3) + 2)}(\log \log n + \log \Delta^{2(\frac{1}{8} \log_{\Delta} n - 3) + 2}) \\ & \leq \frac{50}{\Delta^8} \sqrt{n}(\log \log n + (1/4) \log n - 4 \log \Delta) \\ & \leq \frac{100}{\Delta^8} \sqrt{n} \log n < n. \end{aligned}$$

After computing the coloring, \mathcal{A}' runs the original algorithm \mathcal{A} , and as argued, it works correctly. In total, \mathcal{A}' requires strictly less than $12t + t = 13t$ rounds. Hence the claim follows. \square

References

- [BBC⁺22] Alkida Balliu, Sebastian Brandt, Yi-Jun Chang, Dennis Olivetti, Jan Studený, and Jukka Suomela. Efficient classification of locally checkable problems in regular trees. In Christian Scheideler, editor, *36th International Symposium on Distributed Computing, DISC 2022, October 25-27, 2022, Augusta, Georgia, USA*, volume 246 of *LIPICs*, pages 8:1–8:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.DISC.2022.8.
- [BBCR⁺25] Alkida Balliu, Sebastian Brandt, Xavier Coiteux-Roy, Francesco D’Amore, Massimo Equi, François Le Gall, Henrik Lievonen, Augusto Modanese, Dennis Olivetti, Marc-Olivier Renou, Jukka Suomela, Lucas Tendick, and Isadora Veeren. Distributed quantum advantage for local problems. In Michal Koucký and Nikhil Bansal, editors, *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC 2025, Prague, Czechia, June 23-27, 2025*, pages 451–462. ACM, 2025. doi:10.1145/3717823.3718233.
- [BBE⁺20] Alkida Balliu, Sebastian Brandt, Yuval Efron, Juho Hirvonen, Yannic Maus, Dennis Olivetti, and Jukka Suomela. Classification of distributed binary labeling problems. In Hagit Attiya, editor, *34th International Symposium on Distributed Computing, DISC 2020, October 12-16, 2020, Virtual Conference*, volume 179 of *LIPICs*, pages 17:1–17:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICS.DISC.2020.17.
- [BBH⁺19] Alkida Balliu, Sebastian Brandt, Juho Hirvonen, Dennis Olivetti, Mikael Rabie, and Jukka Suomela. Lower bounds for maximal matchings and maximal independent sets. In David Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 481–497. IEEE Computer Society, 2019. doi:10.1109/FOCS.2019.00037.

- [BBK⁺24] Alkida Balliu, Sebastian Brandt, Fabian Kuhn, Dennis Olivetti, and Joonatan Saarhelo. Towards fully automatic distributed lower bounds. *CoRR*, abs/2410.20224, 2024. [arXiv:2410.20224](https://arxiv.org/abs/2410.20224), [doi:10.48550/ARXIV.2410.20224](https://doi.org/10.48550/ARXIV.2410.20224).
- [BBKO22] Alkida Balliu, Sebastian Brandt, Fabian Kuhn, and Dennis Olivetti. Distributed Δ -coloring plays hide-and-seek. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 464–477. ACM, 2022. [doi:10.1145/3519935.3520027](https://doi.org/10.1145/3519935.3520027).
- [BBO20] Alkida Balliu, Sebastian Brandt, and Dennis Olivetti. Distributed lower bounds for ruling sets. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 365–376. IEEE, 2020. [doi:10.1109/FOCS46700.2020.00042](https://doi.org/10.1109/FOCS46700.2020.00042).
- [BBOS20] Alkida Balliu, Sebastian Brandt, Dennis Olivetti, and Jukka Suomela. How much does randomness help with locally checkable problems? In Yuval Emek and Christian Cachin, editors, *PODC '20: ACM Symposium on Principles of Distributed Computing, Virtual Event, Italy, August 3-7, 2020*, pages 299–308. ACM, 2020. [doi:10.1145/3382734.3405715](https://doi.org/10.1145/3382734.3405715).
- [BBOS21] Alkida Balliu, Sebastian Brandt, Dennis Olivetti, and Jukka Suomela. Almost global problems in the LOCAL model. *Distributed Computing*, 34(4):259–281, 2021. [doi:10.1007/S00446-020-00375-2](https://doi.org/10.1007/S00446-020-00375-2).
- [BCG⁺22] Sebastian Brandt, Yi-Jun Chang, Jan Grebík, Christoph Grunau, Václav Rozhon, and Zoltán Vidnyánszky. Local problems on trees from the perspectives of distributed algorithms, finitary factors, and descriptive combinatorics. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPICs*, pages 29:1–29:26. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. [doi:10.4230/LIPICS.ITCS.2022.29](https://doi.org/10.4230/LIPICS.ITCS.2022.29).
- [BCHM⁺21] Alkida Balliu, Keren Censor-Hillel, Yannic Maus, Dennis Olivetti, and Jukka Suomela. Locally checkable labelings with small messages. In Seth Gilbert, editor, *35th International Symposium on Distributed Computing, DISC 2021, October 4-8, 2021, Freiburg, Germany (Virtual Conference)*, volume 209 of *LIPICs*, pages 8:1–8:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. [doi:10.4230/LIPICS.DISC.2021.8](https://doi.org/10.4230/LIPICS.DISC.2021.8).
- [BFH⁺16] Sebastian Brandt, Orr Fischer, Juho Hirvonen, Barbara Keller, Tuomo Lempiäinen, Joel Rybicki, Jukka Suomela, and Jara Uitto. A lower bound for the distributed Lovász local lemma. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 479–488. ACM, 2016. [doi:10.1145/2897518.2897570](https://doi.org/10.1145/2897518.2897570).
- [BHK⁺17] Sebastian Brandt, Juho Hirvonen, Janne H. Korhonen, Tuomo Lempiäinen, Patric R. J. Östergård, Christopher Purcell, Joel Rybicki, Jukka Suomela, and Przemysław Uznanski. LCL problems on grids. In Elad Michael Schiller and Alexander A. Schwarzmann, editors, *Proceedings of the ACM Symposium on Principles of Distributed Computing, PODC 2017, Washington, DC, USA, July 25-27, 2017*, pages 101–110. ACM, 2017. [doi:10.1145/3087801.3087833](https://doi.org/10.1145/3087801.3087833).

- [BHK⁺18] Alkida Balliu, Juho Hirvonen, Janne H. Korhonen, Tuomo Lempiäinen, Dennis Olivetti, and Jukka Suomela. New classes of distributed time complexity. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 1307–1318. ACM, 2018. doi:[10.1145/3188745.3188860](https://doi.org/10.1145/3188745.3188860).
- [BKK⁺23] Alkida Balliu, Janne H. Korhonen, Fabian Kuhn, Henrik Lievonen, Dennis Olivetti, Shreyas Pai, Ami Paz, Joel Rybicki, Stefan Schmid, Jan Studený, Jukka Suomela, and Jara Uitto. Sinkless orientation made simple. In Telikepalli Kavitha and Kurt Mehlhorn, editors, *2023 Symposium on Simplicity in Algorithms, SOSA 2023, Florence, Italy, January 23-25, 2023*, pages 175–191. SIAM, 2023. doi:[10.1137/1.9781611977585.CH17](https://doi.org/10.1137/1.9781611977585.CH17).
- [BO20] Sebastian Brandt and Dennis Olivetti. Truly tight-in- Δ bounds for bipartite maximal matching and variants. In Yuval Emek and Christian Cachin, editors, *PODC '20: ACM Symposium on Principles of Distributed Computing, Virtual Event, Italy, August 3-7, 2020*, pages 69–78. ACM, 2020. doi:[10.1145/3382734.3405745](https://doi.org/10.1145/3382734.3405745).
- [Bra19] Sebastian Brandt. An automatic speedup theorem for distributed problems. In Peter Robinson and Faith Ellen, editors, *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019, Toronto, ON, Canada, July 29 - August 2, 2019*, pages 379–388. ACM, 2019. doi:[10.1145/3293611.3331611](https://doi.org/10.1145/3293611.3331611).
- [Cha20] Yi-Jun Chang. The complexity landscape of distributed locally checkable problems on trees. In Hagit Attiya, editor, *34th International Symposium on Distributed Computing, DISC 2020, October 12-16, 2020, Virtual Conference*, volume 179 of *LIPICs*, pages 18:1–18:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:[10.4230/LIPICS.DISC.2020.18](https://doi.org/10.4230/LIPICS.DISC.2020.18).
- [CKP16] Yi-Jun Chang, Tsvi Kopelowitz, and Seth Pettie. An exponential separation between randomized and deterministic complexity in the LOCAL model. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 615–624. IEEE Computer Society, 2016. doi:[10.1109/FOCS.2016.72](https://doi.org/10.1109/FOCS.2016.72).
- [CKP19] Yi-Jun Chang, Tsvi Kopelowitz, and Seth Pettie. An exponential separation between randomized and deterministic complexity in the LOCAL model. *SIAM Journal on Computing*, 48(1):122–143, 2019. doi:[10.1137/17M117537](https://doi.org/10.1137/17M117537).
- [CP19] Yi-Jun Chang and Seth Pettie. A time hierarchy theorem for the LOCAL model. *SIAM Journal on Computing*, 48(1):33–69, 2019. doi:[10.1137/17M1157957](https://doi.org/10.1137/17M1157957).
- [CV86] Richard Cole and Uzi Vishkin. Deterministic coin tossing with applications to optimal parallel list ranking. *Information and Control*, 70(1):32–53, 1986. doi:[10.1016/S0019-9958\(86\)80023-7](https://doi.org/10.1016/S0019-9958(86)80023-7).
- [FG17] Manuela Fischer and Mohsen Ghaffari. Sublogarithmic distributed algorithms for Lovász local lemma, and the complexity hierarchy. In Andréa W. Richa, editor, *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, volume 91 of *LIPICs*, pages 18:1–18:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:[10.4230/LIPICS.DISC.2017.18](https://doi.org/10.4230/LIPICS.DISC.2017.18).

- [GHK18] Mohsen Ghaffari, David G. Harris, and Fabian Kuhn. On derandomizing local distributed algorithms. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 662–673. IEEE Computer Society, 2018. doi:[10.1109/FOCS.2018.00069](https://doi.org/10.1109/FOCS.2018.00069).
- [GKM17] Mohsen Ghaffari, Fabian Kuhn, and Yannic Maus. On the complexity of local distributed graph problems. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 784–797. ACM, 2017. doi:[10.1145/3055399.3055471](https://doi.org/10.1145/3055399.3055471).
- [GPS88] Andrew V. Goldberg, Serge A. Plotkin, and Gregory E. Shannon. Parallel symmetry-breaking in sparse graphs. *SIAM Journal on Discrete Mathematics*, 1(4):434–446, 1988. doi:[10.1137/0401044](https://doi.org/10.1137/0401044).
- [GRB22] Christoph Grunau, Václav Rozhon, and Sebastian Brandt. The landscape of distributed complexities on trees and beyond. In Alessia Milani and Philipp Woelfel, editors, *PODC '22: ACM Symposium on Principles of Distributed Computing, Salerno, Italy, July 25 - 29, 2022*, pages 37–47. ACM, 2022. doi:[10.1145/3519270.3538452](https://doi.org/10.1145/3519270.3538452).
- [GS17] Mohsen Ghaffari and Hsin-Hao Su. Distributed degree splitting, edge coloring, and orientations. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2505–2523. SIAM, 2017. doi:[10.1137/1.9781611974782.166](https://doi.org/10.1137/1.9781611974782.166).
- [Lin92] Nathan Linial. Locality in distributed graph algorithms. *SIAM Journal on Computing*, 21(1):193–201, 1992. doi:[10.1137/0221015](https://doi.org/10.1137/0221015).
- [Mar16] Andrew Marks. A determinacy approach to Borel combinatorics. *Journal of the American Mathematical Society*, 29(2):579–600, 2016. doi:[10.1090/jams/836](https://doi.org/10.1090/jams/836).
- [Nao91] Moni Naor. A lower bound on probabilistic algorithms for distributive ring coloring. *SIAM Journal on Discrete Mathematics*, 4(3):409–412, 1991. doi:[10.1137/0404036](https://doi.org/10.1137/0404036).
- [NS95] Moni Naor and Larry J. Stockmeyer. What can be computed locally? *SIAM Journal on Computing*, 24(6):1259–1277, 1995. doi:[10.1137/S0097539793254571](https://doi.org/10.1137/S0097539793254571).
- [Pel00] David Peleg. *Distributed Computing: A Locality-Sensitive Approach*. Society for Industrial and Applied Mathematics, 2000. doi:[10.1137/1.9780898719772](https://doi.org/10.1137/1.9780898719772).
- [RG20] Václav Rozhon and Mohsen Ghaffari. Polylogarithmic-time deterministic network decomposition and distributed derandomization. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 350–363. ACM, 2020. doi:[10.1145/3357713.3384298](https://doi.org/10.1145/3357713.3384298).
- [Suo20] Jukka Suomela. Landscape of locality (invited talk). In Susanne Albers, editor, *17th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT 2020, June 22-24, 2020, Tórshavn, Faroe Islands*, volume 162 of *LIPICs*, pages 2:1–2:1. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:[10.4230/LIPICS.SWAT.2020.2](https://doi.org/10.4230/LIPICS.SWAT.2020.2).