Anonymous Permutation Routing

Paul Bunn[⋆], Eyal Kushilevitz[⋆⋆], and Rafail Ostrovsky[†]

Abstract. The Non-Interactive Anonymous Router (NIAR) model was introduced by Shi and Wu [SW21] as an alternative to conventional solutions to the anonymous routing problem, in which a set of senders wish to send messages to a set of receivers. In contrast to most known approaches to support anonymous routing (e.g. mix-nets, DC-nets, etc.), which rely on a network of routers communicating with users via interactive protocols, the NIAR model assumes a *single* router and is inherently *non-interactive* (after an initial setup phase). In addition to being non-interactive, the NIAR model is compelling due to the security it provides: instead of relying on the honesty of some subset of the routers, the NIAR model requires anonymity even if the router (as well as an arbitrary subset of senders/receivers) is corrupted by an honest-but-curious adversary.

In this paper, we present a protocol for the NIAR model that improves upon the results from [SW21] in two ways:

- Improved computational efficiency (quadratic to near linear): Our protocol matches the communication complexity of [SW21] for each sender/receiver, while reducing the computational overhead for the router to polylog overhead instead of linear overhead.
- Relaxation of assumptions: Security of the protocol in [SW21] relies on the Decisional Linear assumption in bilinear groups; while security for our protocol follows from the existence of any rate-1 oblivious transfer (OT) protocol (instantiations of which are known to exist under the DDH, QR and LWE assumptions [DGI+19,GHO20]).

Keywords. Anonymous Routing, Private-Information Retrieval, Permutation Routing, Non-Interactive Protocols.

[⋆] Stealth Software Technologies, Inc. Email: `paul@stealthsoftwareinc.com`
[⋆⋆] Computer Science Department, Technion, Israel. Email: `eyalk@cs.technion.ac.il`.
[†] UCLA Departments of Computer Science, Mathematics. Email: `rafail@cs.ucla.edu`

1 Introduction

As the collection and access of digital information in our daily lives becomes ever-more ubiquitous (internet, local networks, mobile networks, IoT), so too does the need for the development of technologies to protect access and transmission of this data. While protecting the integrity and access to sensitive data remain important tasks, there has been a growing need for *anonymity* in protecting data access and communications between users. Throughout this paper, anonymity will refer to the inability to associate which nodes in a network are communicating with each other; i.e. the unlinkability between one or more senders and the associated receiver(s). The conventional approach to providing such protection (onion routing, mix-nets, and others) relies on a network of routers relaying messages, where anonymity is only guaranteed if there are sufficiently many uncorrupted routers. A markedly different approach to this problem was recently introduced by Shi and Wu [SW21], who proposed using cryptographic techniques to hide connectivity patterns. Namely, they introduce the Non-Interactive Anonymous Router (NIAR) model, in which a set of N receiving nodes wish to receive information from a set of N sending nodes, with all information passing through a central router. Anonymity in their model is defined to be the inability to link any sender to the corresponding receiver, even if the router and (up to $N-2$) various (sender, receiver) pairs are susceptible to attack by an (honest-but-curious[1]) adversary.

There are a number of real-world scenarios in which the NIAR model as described above is relevant. The important characteristics of any such application is that a number of (sender, receiver) pairs wish to anonymously communicate with each other through a central server, where the messages to be transmitted are large and/or the communication channels are non-ephemeral/indefinite. These conditions are exhibited, for example, in the following scenarios:

Anonymous Peer-to-Peer Communication. Relevant in settings where a large set of users wish to communicate anonymously through a central server, e.g. for a Messaging app, where every communication link is established as a separate pair of (anonymous) virtual users.

Pub/Sub with Privacy. Because our solution is quasi-linear in message size, the additional overhead of storing all messages is minimal. We can therefore view the central router of the NIAR model as delivering each stream of messages it receives from the N senders into N storage units, rather than delivering them directly to receivers. In this way, the set of receivers can (privately) subscribe to an information service/source, and periodically receive updates. Furthermore, our protocol allows receivers to (privately) subscribe to *multiple* services at the

[1] Our limitation to HBC adversaries is only needed to ensure Correctness of our protocol - that receivers get the correct messages. We note that requiring HBC for correctness is unavoidable, as a malicious router can, for example, not forward any message (like in PIR and other related primitives). In terms of Security (privacy of the senders-receivers permutation): so long as the one-time Setup is performed properly, then security of our protocol will hold in the Malicious adversary setting.

same time, without revealing which services they are subscribed to.

Multi-Client PIR/PIW. In a similar spirit as the previous point, viewing the receivers as storage units, the messages being streamed from the senders can accumulate (or update previous messages), thus implementing a form of Private Information Writing (PIW). Depending on the application (in terms of which users will ultimately access/read the PIW server), hiding the linkage between which location each sender writes to versus which location each receiver reads from may require stronger security requirements, e.g. for our protocol, any receiver colluding with the central router will learn which sender it is reading from.

Oblivious Shuffle. A common scenario encountered in MPC protocols is when two or more parties are secret sharing a list of values, and need to obliviously permute the list, so that no party knows the permutation. Our protocol can be used to implement this oblivious shuffle, by viewing one party as acting as all N senders (for its list of N secret shared values), and sending the permuted shares via the "central router" (also being simulated by the sending party) to the other party (who is acting as all N receivers). This process is then reversed, with the other party sending its shares to the first party, via the same permutation. There are subtleties that need to be specified, such as ensuring that the permutation remains unknown to each party (which can be handled as part of the Setup procedure), and how to amortize the process to ensure efficiency (so the Setup does not dominate overall cost), but in general a solution in the NIAR model can be viewed as an instantiation of oblivious shuffle.

Permutation Routing with Anonymity. There has been substantial work in researching permutation routing (e.g. [AKS83,Lei84,Upf89,MS92]), which was inspired due to its relevance to parallel computing (for timing the connections between processors and memory) and fault tolerant routing. Since the NIAR model is essentially permutation routing with anonymity, any applications of permutation routing that stand to benefit from hiding the permutation are relevant to our work.

1.1 Technical Challenges

Notice that (assuming PKI) an immediate solution to anonymity in the NIAR model is to have each sender encrypt their message (under the desired receiver's public key or using a shared secret key with the recipient), send the encrypted message to the center router, and then simply have the router flood all N (encrypted) messages to each of the N receivers. While this naïve approach satisfies anonymity (as well as privacy, in that receivers only receive messages intended for them), it has the pitfall of excessive communication: $O(N)$ for each receiver, and $O(N^2)$ for the router. Shi and Wu [SW21] present a protocol which, under the Decisional Linear assumption (on certain bilinear groups), achieves anonymity with minimal *communication* overhead.

Having re-framed the goal of anonymity to the NIAR model and with the toolbox of cryptographic techniques at hand, a natural observation is that Private Information Retrieval (PIR) can be used as a potential solution. In a (single server) PIR protocol [KO97], a server stores a database DB of N elements, and a client issues a query to the server to retrieve the i^{th} element $DB[i]$, for i of its choice. Security in the PIR model means that the server does not learn any information about the index i being queried. Thus, if N senders encrypt their messages and send them to the router, we can let the router act as a PIR server with the N concatenated (encrypted) messages forming the contents of the PIR database. Each receiver can then issue a PIR query to fetch the appropriate message, and anonymity follows from the security of PIR. As with the protocol of [SW21], this solution enjoys both the requisite security features, as well as having minimal communication overhead (e.g. $\log N$ overhead, depending on the PIR protocol; see survey of PIR results in [OS07]).

An important metric in determining the feasibility of a protocol in the NIAR model is the end-to-end message transmission time, which depends on the computational burden on each user, and especially that of the central router. A significant drawback of both the protocol of [SW21][2] and the naïve PIR solution described above is that they require *quadratic* (in terms of the number of users) computation at the router. As this computation cost is likely prohibitive (or at least extremely inefficient) when there are a large number of users, we set out to explore the possibility of a NIAR protocol that maintained the minimal communication burden of the naïve PIR and [SW21] solutions, but reduced computation overhead (at the router) from $O(N^2)$ closer to the optimal $O(N)$.

Our first observation is that the NIAR model is similar to so-called "permutation routing" (see Section 2.1), but with an additional anonymity requirement. Namely, permutation routing seeks to connect N senders to N receivers through a network, which (from a communication standpoint) is what is required in the NIAR model. Our main idea was to leverage the efficient routing (and therefore minimal overhead) of a permutation-routing network, but then to administer PIR at each node to keep each routing decision hidden, thereby allowing for the anonymity required by the NIAR model. In particular, we envisioned a solution in which the central router simulates a virtual permutation-routing network by itself, where the actual path the messages take (from each of the N senders on one end of the network to the N receivers at the other end) is hidden (from the central router) by using PIR along each edge. Namely, at each node of the (virtual) network, a PIR query is applied to each of the node's outgoing edges, where the PIR query (privately) selects a message from one of the node's incoming edges.

While the above idea captures the spirit of our solution (and indeed, the idea of layering PIR on top of various routing networks/protocols may have other interesting applications for anonymizing communication), there are several complications that required additional consideration:

[2] Router computation is not explicitly measured in the protocol of [SW21], our analysis of their protocol yields $O(N^2)$ computation load on the router: their Multi-Client Functional Encryption (MCFE) protocol is invoked N times by the router, with each invocation processing N ciphertexts.

1. (Virtual) Network Size. Since each outgoing edge in the routing network is assigned a PIR query, and this PIR query is applied to a (virtual) database whose size is the number of incoming edges of the node in question, the computation cost of simulating routing in a virtual network is roughly $O(E \cdot I)$, where E is the number of edges and I is the number of incoming edges per node. Since E is necessarily at least $\Omega(N)$, having a NIAR protocol with only polylog computation overhead requires that E is at most $O(N \cdot \mathit{polylog}\ N)$ and I is $O(\mathit{polylog}\ N)$.
2. Standard PIR Won't Work. Even if network size is small $(O(N \cdot \mathit{polylog}(N)))$, if the depth (number of nodes a message passes through from sender to receiver) is not constant, then standard PIR schemes will not work, since each invocation of PIR typically has $O(\mathit{polylog}(N))$ bits in the PIR server's response, and hence the message size will incur an exponential blow-up with network depth. For example, even log-depth networks will have messages of size $O(2^{\log N}) = O(N)$ by the time they reach the last layer of the network, which is no better than the naïve PIR approach mentioned above.
3. Correctness Requires Edge-Disjoint Paths. Since PIR is being used to hide routing decisions made at each node/routing gate in the network, this requires that each outgoing edge forwards the message on (at most) one of the node's incoming edges. In particular, if any two paths connecting two different sender-receiver pairs in the permutation network contain a common edge, then correctness is compromised. Since a *random* path selection algorithm will be crucial to proving anonymity, the given (virtual) permutation network must have the property that, with high probability, a random sample of paths connecting the sender-receiver pairs are edge-disjoint.
4. Edge-Disjoint Property is Insufficient for Anonymity. While having edge-disjoint paths is necessary for correctness, it is not sufficient to ensure anonymity. For example, if the central router is colluding with (N-2) sender-receiver pairs (and therefore only needs to determine the linkage amongst the remaining two senders and two receivers), then knowledge that all paths are edge-disjoint can give the router an advantage in identifying the linkage between the remaining two senders and two receivers. Namely, the router knows (via collusion) N-2 paths, and thus can eliminate available options for the remaining two paths. For example, this attack is viable in the Beneš network (which is commonly used in permutation routing literature; see Section 3.1) making it unsuitable when anonymity is required, and justifying our usage of a more complex network. Indeed, since permutation-routing networks have been studied outside of the context of anonymity, to our knowledge there has not been any research into understanding how network properties and path selection protocols impact anonymity.

1.2 Overview of Our Results

Our solution to the NIAR problem, which blends techniques from permutation routing with techniques for hiding routing decisions made at each node of the (virtual) permutation network, overcomes the challenges outlined in the previous section as follows. By using familiar permutation-routing networks, which are inherently small ($O(N \cdot polylog(N))$), we ensure the network size is suitably small, thus addressing the first potential issue. Furthermore, a common (and well-studied) feature of many permutation-routing networks is the edge-disjoint property, which inspired our choice to use an (extended) Beneš permutation Network, thus addressing the third issue. We observe that there is an inherent tension between network topology (number of nodes, edges, and depth) in terms of achieving correctness and anonymity versus low router computation. Our solution includes carefully selecting appropriate network parameters to balance these trade-offs. Meanwhile, recent works [DGI$^+$19,GHO20,CGH$^+$21] present so-called *rate-1 PIR* protocols, which can address the second issue of exponential growth of message size per network layer.

Addressing the fourth issue is one of our key technical achievements. In spirit, the edge-disjoint property is related to anonymity, but as mentioned above, it is in general insufficient. Identifying a property that *is* sufficient (and simultaneously not over-cumbersome in terms of network size), and then using such a property to formally argue anonymity, requires some thought and careful analysis. Informally, this property states that not only are N randomly chosen permutation paths through the network edge-disjoint (w.h.p), but even if the permutation swaps the output nodes of any two input nodes and two new paths are created to join these, then the collection of the old edges plus the two new sets of edges are still edge disjoint (w.h.p); see Definition 2.

Assuming rate-1 PIR, we present in Figure 3 a routing protocol for the NIAR model that achieves $O(\log N)$ per-party communication and $O(N \cdot polylog(N))$ router computation. At a high level, our protocol dictates that the central router emulates routing in a permutation network, whereby each routing gate is (virtually) obliviously evaluated using a rate-1 PIR query/response for each outgoing edge. Our protocol consists of a setup phase in which the PIR queries that correspond to all outgoing edges of every routing gate are prepared, and then an online routing phase where a stream of (encrypted) messages are injected by the senders and routed to the receivers (re-using the setup).

A succinct comparison of our results to other relevant works is in §2.3.

2 Previous Work

2.1 Permutation Routing

In permutation routing [AKS83,Lei84,Upf89,MS92], messages from a set of N "input" nodes are routed through a network G to a set of N distinct "output" nodes. Such works attempt to identify networks G with various desired properties, and protocols within these networks that can efficiently route these messages, for any possible permutation σ that dictates which input node is connected to which output node. While our work is partially inspired by the routing networks considered in this line of work, the NIAR model is quite different than the permutation routing model, both because of the number of routers (one versus $\Theta(N \log N)$) and due to the required privacy of the permutation σ. In other words, we do not route the messages over a physical routing network (which is an iterative process that depends on the "depth" of the network), but rather we design our non-interactive routing protocol using a *virtual* sorting network.

2.2 PIR

There has been an extensive amount of work done on the original PIR problem [CGKS95,KO97] and its variants. Here, we discuss only a few of these works that are most relevant to us.

Multi-Client PIR As discussed in the introduction, the NIAR problem can be solved using multi-client PIR. Indeed, a solution to generic multi-client PIR in which the PIR server's work does not scale with the number of users would imply an efficient solution for NIAR. While no such result is known, we discuss a few relevant works and why they are insufficient for the NIAR model.

In [IP07], it is demonstrated how a single user can efficiently issue multiple queries to a PIR server. However, their results rely on a single decoding algorithm, whereas the NIAR model would require distinct decoding keys for each of the N receivers. [HOWW19] present a related notion of private anonymous data access; we note that the results in their model do not scale to the full corruption threshold ($N-2$) required in the NIAR security model. Finally, results in the related areas of Batch Codes [IKOS04] and Public-Key Encryption with amortized updates [COS10] address a different model, and consequently do not seem to be directly applicable to the NIAR model.

Rate-1 PIR A recent line of work [DGI+19,GHO20,CGH+21] has demonstrated the viability of rate-1 PIR, in which the server response is comparable in size to the database entry being fetched. Formally, for a database of N elements each of size B, rate-1 PIR means that the ratio of B to the server response size approaches 1 as $N \to \infty$. Stated differently, a rate-1 PIR scheme has an additive constant-stretch term δ_{PIR}, such that the server's response has size $B + \delta_{PIR}$. Rate-1 PIR is known to exist under the DDH, QR and LWE assumptions [DGI+19,GHO20].

Doubly Efficient PIR (DEPIR) In a recent result of Lin et al. [LMW22], they demonstrate a PIR protocol that, after a pre-processing phase that costs $O(N^{1+\epsilon})$ in server computation, enjoys polylog N communication and computation for each PIR query. If this DEPIR protocol were to be used to solve the NIAR problem (as per the straightforward application described in Section 1.1), the resulting protocol would have $O(N^{1+\epsilon})$ computation at the router for each new message packet/bit of the senders (since each database update would trigger a new "pre-processing" phase of the PIR server).

$$\begin{tikzcd}
& N\text{-2 Pairs} & \\
\begin{array}{c}\text{Sender Collusion}\\ \text{(and } N-2 \text{ Receivers)}\end{array} \arrow[ur, Rightarrow] & & \begin{array}{c}\text{Receiver Collusion}\\ \text{(and } N-2 \text{ Senders)}\end{array} \arrow[ul, Rightarrow] \\
& \text{Arbitrary} \arrow[ul, Rightarrow] \arrow[ur, Rightarrow] &
\end{tikzcd}$$

Fig. 1: *Various security requirements/settings relevant to the NIAR model. All four scenarios include collusion with router C, plus:*
- *Top Setting (N-2 Pairs): Corruption of up to N-2 (sender, receiver) pairs;*
- *Left Setting (Sender Collusion): Corruption of all senders (and N-2 receivers);*
- *Right Setting (Receiver Collusion): Corruption of all receivers (and N-2 senders);*
- *Bottom Setting (Arbitrary): Corruption of any $2N-2$ senders/receivers.*

The implication arrows indicate that a protocol that is secure in one setting is automatically secure in the other.

2.3 Comparison with Other Results in NIAR Model

The NIAR model was introduced in [SW21], which included several variants of the security requirement, and offered solutions for these variants. As mentioned, our results improve upon those of [SW21] in three main ways: (i) Reduced router overhead ($O(N \cdot polylogN)$ versus $O(N^2)$); (ii) Seemingly simpler protocol based on weaker/more standard cryptographic assumptions; (iii) Improved practical/observed efficiency (not empirically verified). On the other hand, the protocol of [SW21] provides protection in different scenarios of security requirements. Namely, in terms of Figure 1, our protocol focuses on the top and left settings, while [SW21] covers the top, right, and bottom settings. However, for all of the motivating examples discussed in the Introduction, security in the top and left settings (which our protocol provides) is sufficient.

A recent work of Fernando et al. [FSSV22] improves upon the work of [SW21], by reducing router computation to $O(N \cdot polylogN)$, which (asymptotically) matches our result. However, the other comparisons between our work and that of [SW21] are still valid; namely, our protocol benefits from simpler assumptions and protocol complexity (e.g. we do not require obfuscation) as well as practical efficiency, but ours does not offer protection against full receiver collusion.

A summary of the comparison of our results to other relevant results can be found in the table below, where $\widetilde{N} = O(N \cdot polylogN)$ denotes quasi-linear:

Protocol	**Anonymity Level**[2]	**Crypto Assumptions**	**Comm.**	**Router Comp.**
Permutation Routing	None	N/A	$\widetilde{N}$	$\widetilde{N}$
Naïve PIR	Sender Collusion	PIR	$\widetilde{N}$	N^2
DEPIR [LMW22][3]	Sender Collusion	Ring LWE	$\widetilde{N}$	$N^{1+\epsilon}$
Original NIAR [SW21]	Arbitrary	Obfuscation	$\widetilde{N}$	N^2
Improved NIAR [FSSV22]	Arbitrary	Obfuscation	$\widetilde{N}$	$\widetilde{N}$
Our Results	Sender Collusion	DDH *or* QR *or* LWE	$\widetilde{N}$	$\widetilde{N}$

3 Preliminaries

3.1 Beneš Network

(Each figure referenced here can be found in Section A, and the networks mentioned here are common in the permutation routing literature, see for example [AKS83,Lei84,Upf89,MS92]). In a butterfly network (Fig. 8), N input nodes are connected to N output nodes via a leveled network of $(1 + \log N)$ levels, each with N nodes. A Beneš network appends a second (inverted) butterfly network to the first (Fig. 10); and more generally an extended Beneš network appends many "blocks" of butterfly networks together. We continue expanding on this model by replicating each node and edge c times, which can be conceptualized as coloring them with c distinct colors (Fig. 12). Finally, our protocol will assume wide edges, which means that each edge can simultaneously route w messages (requiring specification of which of the w "slots" each message occupies).

3.2 Non-Interactive Anonymous Routing (NIAR)

We adopt the NIAR model of [SW21], in which N senders each has a series of m (e.g. single-bit) messages they wish to send to a distinct receiver *anonymously.* The anonymity guarantee refers to the unlinkability of each sender-receiver pair, and crucially it must be preserved even if the central router colludes with a subset of the senders/receivers. Depending on the application, there are various collusion patterns that may be of interest, see e.g. Figure 1.

[2] Anonymity terminology as defined in Figure 1. Namely, "Sender Collusion" refers to potential corruption of the central router, all senders, and up to $N - 2$ receivers; and "Arbitrary" refers to potential corruption of the central router and any set of up to $2N - 2$ senders/receivers.

[3] Analysis of [LMW22] in the context of the NIAR model is not done by Lin et al., and the stated characteristics of their protocol in the NIAR setting are ours.

In this paper, we demonstrate our protocol is secure against the top and left settings (in Figure 1). We do not consider the right and bottom settings (Receiver Collusion and Arbitrary) in this paper for two reasons: First, in the main application areas for the NIAR model (see Introduction above), the receivers already know the senders they wish to connect to, so anonymity of the senders (in the case that all receivers are colluding) is irrelevant. The second reason we do not consider the Receiver Collusion setting is because providing protection in this setting requires additional techniques than those considered in this paper. For example in [SW21] and [FSSV22], the protocol description, performance, and cryptographic hardness assumptions are all more complex in the Receiver Collusion setting.

Formally, the (reformulated) NIAR model of [SW21] is as follows:

(Trusted) Setup. Upon input security parameters $(1^{\lambda_c}, 1^{\lambda_s})$, number of senders/receivers N, and permutation $\sigma : [N] \to [N]$, the Setup algorithm outputs sender keys $\{pk_i\}_{i\in[N]}$, receiver keys[4] $\{(sk_i, \kappa_i)\}_{i\in[N]}$, and token q for router C: $\big(\{pk_i\}_{i\in[N]}, \{(sk_i, \kappa_i)\}_{i\in[N]}, \mathsf{q}\big) \leftarrow \mathbf{Setup}(1^{\lambda_c}, 1^{\lambda_s}, N, \sigma)$.

Once Setup has been run, the Senders $\{S_i\}$ can communicate arbitrary messages $\{m_i\} = \{m_{i,\alpha}\}$ with the Receivers $\{R_i\}$ through router C.

Send Message. Using key pk_i, each Sender S_i encodes message $m_i = m_{i,\alpha}$ (where α denotes the α^{th} bit of message m_i), and sends the result to router C: $c_{i,\alpha} \leftarrow \mathbf{Enc}_{pk_i}(m_{i,\alpha})$.

Route Message. Upon inputs $\{c_i\}_{i\in[N]}$ from each Sender S_i, and using key q, router C prepares messages $\{z_i\}_{i\in[N]}$, and sends these to each Receiver R_i: $(z_1, z_2, \ldots, z_N) \leftarrow \mathbf{Route}(\mathsf{q}, c_1, c_2, \ldots, c_N)$.

Decode Message. Using keys (sk_i, κ_i), each Receiver R_i decodes the message $z_i = z_{i,\alpha}$ received from router C, and outputs $\widetilde{m}_i = \widetilde{m}_{i,\alpha}$: $\widetilde{m}_{i,\alpha} \leftarrow \mathbf{Dec}_{sk_i}(\kappa_i, z_{i,\alpha})$.

Correctness. An oblivious permutation routing protocol has:

<u>Perfect Correctness</u>: If each receiver R_i outputs message $\widetilde{m}_i = m_i$ with probability 1.

<u>λ_c - Statistical Correctness</u>: If each receiver R_i outputs message $\widetilde{m}_i = m_i$ with probability at least $\big(1 - \frac{1}{2^{\lambda_c}}\big)$, for security parameter λ_c.

Security. Informally, anonymity means that if a subset of parties collude (including router C), the permutation σ (namely, its restriction to non-colluding parties) should remain unknown. Formally, let $\mathcal{A}$ denote a (computationally bounded, honest-but-curious) adversary. Consider the following challenge game:

1. On input security parameter λ, Adversary $\mathcal{A}$ chooses N, two distinct permutations σ_0, σ_1 on $[N]$, a set of sender indices $S_{\mathcal{A}} \subseteq [N]$ to corrupt, and a set of receiver indices $R_{\mathcal{A}} \subseteq [N]$ to corrupt, subject to the following constraints:
   (a) $|R_{\mathcal{A}}| \leq N - 2$;
   (b) σ_0 and σ_1 match for all receivers in $R_{\mathcal{A}}$: $\forall\, i \in R_{\mathcal{A}} : \sigma_0^{-1}(i) = \sigma_1^{-1}(i)$.

[4] The sender keys $\{pk_i\}$ are associated with the receiver keys $\{sk_i\}$ via the permutation σ; namely, secret key $sk_{\sigma(i)}$ can decrypt messages encrypted under pk_i.

2. Adversary $\mathcal{A}$ sends $\{\sigma_0, \sigma_1\}$ to Challenger $\mathcal{C}$.
3. Challenger $\mathcal{C}$ chooses $\sigma_b \in \{\sigma_0, \sigma_1\}$ for $b \leftarrow \{0, 1\}$ (e.g. by flipping a coin).
4. Challenger $\mathcal{C}$ chooses router token q, encryption keys $\{\mathsf{pk}_i\}_{i\in[N]}$, and decryption keys $\{\mathsf{sk}_i\}_{i\in[N]}$. $\mathcal{C}$ sends q, $\{\mathsf{pk}_i\}_{i\in S_{\mathcal{A}}}$, and $\{\mathsf{sk}_i\}_{i\in R_{\mathcal{A}}}$ to $\mathcal{A}$.
5. For each round α:
   (a) Based on knowledge of all prior ciphertexts $\{c_{i,\alpha'}\}_{\alpha'<\alpha}$ (see next step), Adversary $\mathcal{A}$ chooses messages $\{m^{(0)}_{i,\alpha}\}_{i\in[N]}$ and $\{m^{(1)}_{i,\alpha}\}_{i\in[N]}$, subject to the constraint that all messages bound for a corrupt receiver match: $\forall i$ s.t. $i = \sigma_0^{-1}(j)$ for some $j \in R_{\mathcal{A}}$: $m^{(0)}_{i,\alpha} = m^{(1)}_{i,\alpha}$. $\mathcal{A}$ sends $\{m^{(0)}_{i,\alpha}\}$, $\{m^{(1)}_{i,\alpha}\}$ to $\mathcal{C}$.
   (b) Challenger $\mathcal{C}$ outputs to $\mathcal{A}$ ciphertexts $\{c_{i,\alpha}\}_{i\in[N]}$, where each ciphertext is computed as (with b as chosen in Step 3): $c_{i,\alpha} = Enc_{\mathsf{pk}_i}(m^{(b)}_{i,\alpha})$.
6. Adversary $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ of which permutation $\{\sigma_0, \sigma_1\}$ Challenger $\mathcal{C}$ chose.

A NIAR protocol is λ_{s}-secure if the probability that any computationally bounded adversary $\mathcal{A}$ guesses b correctly is bounded by:

$$\Pr[b' = b] \;\leq\; \frac{1}{2} + \frac{1}{2^{\lambda_s}} \tag{1}$$

3.3 Emulating Oblivious Routing in a Virtual Routing Network

In this section, we present the main ideas that connect the NIAR model to the permutation routing problem. At a high level, the idea is to have the NIAR router emulate message transmission through a (virtual) routing network that supports permutation routing between N senders and receivers. In particular, we view the N senders as *input* nodes in the routing network, and the N receivers as the *output* nodes, and then choose paths through the routing network connecting each sender to its receiver. The NIAR router then passes messages from each sender to the designate receiver by routing messages along this path. Note that this entire network, except the input nodes (corresponding to the senders) and output nodes (corresponding to the receivers), together with message routing within it, is entirely simulated by the NIAR router.

In order to preserve anonymity in terms of linkage between each (sender, receiver) pair, the paths that each message takes through this (virtual) routing network must remain hidden to the NIAR router. The key primitive that we utilize to achieve this is called an oblivious routing gate.This notion is defined formally in Definition 28, but intuitively this describes a process in which the message written on each outgoing wire of a gate is (a re-encoding of) a message from one of the input wires to that gate, such that knowledge of *which* incoming wire's message was selected for each outgoing wire is unknown to the router that is instantiating the gate (see Figure 2).

Using a protocol Π_{ORG} that instantiates the oblivious routing gate paradigm at every node, the NIAR router can emulate routing each of the sender's messages through a virtual permutation network, finally delivering the (re-encoded) messages to the output nodes (receivers). Formally, we define this process via an ideal functionality we refer to as an Emulated Permutation Routing protocol Π_{EPR} (see Definition 29).

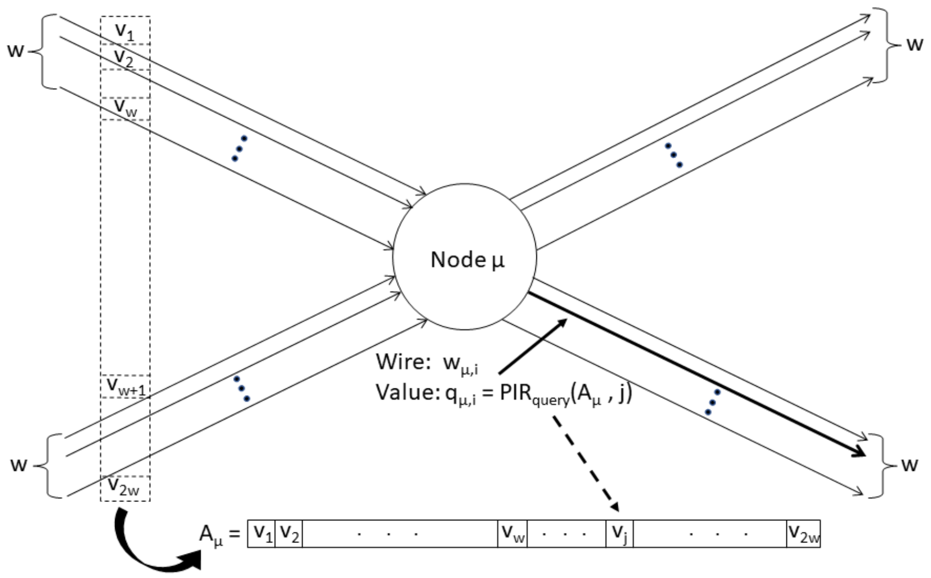
w
v_1
v_2
v_w
v_{w+1}
v_{2w}
w
Node μ
w
w
Wire: $w_{\mu,i}$
Value: $q_{\mu,i}$ = $PIR_{query}(A_\mu, j)$
A_μ = v_1 v_2 . . . v_w . . . v_j . . . v_{2w}

Fig. 2: Oblivious routing gate (Π_{ORG}) realization via PIR at node μ with $2w$ incoming and outgoing edges (used in Step (2b) of our main protocol in Fig. 3).

4 Our Protocol

4.1 Overview of Our Solution

Given N pairs of (sender, receiver) nodes and central router C, our protocol routes messages from the senders to the corresponding receivers via a virtual routing network G that C emulates where, for each node in the network, the router C obliviously executes a routing gate by simulating the functionality of a (rate-1) PIR query. Namely, (as part of trusted setup) each outgoing edge of a routing gate will have an assigned PIR query, and each incoming edge will have a value (which represents an encrypted message from one of the senders). Then the router C obliviously produces a message on each outgoing edge of the routing gate by running the associated PIR query on this wire against the (virtual) PIR database of messages (from the incoming wires). The determination of *which* incoming edge that a given PIR query (on a routing gate's outgoing edge) should specify is established offline during a setup phase, and specifically it is determined by choosing a random path $\mathcal{P}_i$, for each (sender$_i$, receiver$_i$) pair, through the (virtual) routing network G. Notice that once PIR queries are assigned (during an offline setup phase) as per all chosen paths $\{\mathcal{P}_i\}$, they may be reused indefinitely during the online routing phase to continuously route new messages for each (sender, receiver) pair. The main features of our solution are as follows:

- Correctness. Ensuring each receiver gets every message reduces to showing that the paths $\{\mathcal{P}_i\}$ connecting each (sender$_i$, receiver$_i$) pair are *edge-disjoint.*
- Privacy. Since each sender encrypts their messages under the intended receiver's public key, receivers can only decipher messages intended for them.

- Anonymity. This property is obtained so long as the paths $\{\mathcal{P}_i\}$ chosen are "sufficiently edge-disjoint" (for details see Definition 20).
- Communication. To limit the expansion of message size through each (virtual) routing gate, we employ **rate-1** PIR, which ensures the final message size is proportional to the length of the chosen path $\mathcal{P}$ through the (virtual) routing network G; and that any such path is short (i.e. of $polylogN$ length).
- End-to-End Time. Computation of central router C (which, together with communication, determines end-to-end transmission time) will depend on the size of the virtual graph $G = (V, E)$. Thus, in order to minimize computational overhead, $|E|$ should be close to N (e.g. $N \cdot polylogN$). Notice that there is inherent tension in minimizing end-to-end time versus satisfying the Correctness and Anonymity properties: the former requires small $|V|$ and $|E|$, while the latter two are readily achieved for larger $|V|$ and $|E|$. Our protocol finds appropriate (minimal) parameters to achieve correctness and anonymity, while introducing minimal end-to-end overhead.

We stress that some relaxed approaches to the NIAR problem actually fail to provide anonymity. Specifically, the approach of deploying an *arbitrary* permutation-routing network (without the extra features that we require), and the approach of just replacing each gate in the routing network (even a properly selected network) with PIR, do not seem sufficient, which we argue as follows.

While PIR is the main tool that hides (from central router C and any other parties it colludes with) the linkage between uncorrupted (sender, receiver) pairs, applying it naïvely will not provide the desired protection. Namely, if any two of the paths $\{\mathcal{P}_i\}$ through the virtual routing network have an edge in common, then a PIR query cannot be assigned to that edge, as there will be conflicting input edge indices (and conflicting messages on those edges) to select. Since, in proving anonymity, path selection must be a randomized process (in particular, edge conflicts cannot be deliberately avoided), our protocol will handle edge conflicts by producing garbage PIR queries for such edges. While this approach introduces failures in terms of delivering messages along the conflicting paths that were chosen for any such (sender, receiver) pairs, the threat to correctness is overcome by ensuring enough redundancy in the system to account for (the low probability event of) edge conflicts. However, edge conflicts (and the *lack* of edge conflicts), also threatens anonymity: for example, the router C could observe many messages from (sender, receiver) pairs it has corrupted all pass through a common node, and the router may also know that the message from an *uncorrupted* sender has some probability of passing through this same node. Thus, the presence or absence of an edge conflict on the set of outgoing edges of this node may give the router an advantage in determining if the uncorrupted sender's path goes through this node, and if so, some probabilistic advantage in knowing which outgoing edge the path used; and these advantages then threaten anonymity since the router may be able to have an advantage in guessing the ultimate destination (i.e. receiving node) of this path. Demonstrating that this approach cannot be used to give the router a non-negligible advantage in linking uncorrupted (sender, receiver) pairs will require: (i) Identifying what property a network should have to avoid this attack; (ii) Generating such a routing network that also supports the desired complexity and correctness requirements; (iii) An appropriate analysis that this property indeed proves anonymity. For example, the natural candidate property of exhibiting (with high probability on randomly chosen paths) the edge-disjoint property is insufficient, as it is susceptible to the above attack.

Figures 3 and 4 below formally describe our protocol.

4.2 Analysis of Our Protocol

Theorem 1 *Assuming the existence of rate-1 PIR, following trusted setup,*[5] *the protocol presented in Figure 3 is λ_s-secure with λ_c-statistical correctness, $O(\log N)$ per-party communication, and $O(N \text{ polylog } N)$ router computation.*

Remark. Instead of trusted setup, under appropriate cryptographic hardness assumptions the ideal functionality $\Pi_{ORG}(G, \widehat{c}, r, l, \Pi_{1\text{-}PIR})$ could instead be realized via generic secure multiparty computation (MPC) techniques. This would contribute $O(N^2 \textit{ polylog } N)$ to the asymptotic cost of the protocol (to deal the $O(N \textit{ polylog } N)$ rate-1 queries and $O(N^2 \textit{ polylog } N)$ reconstruction keys), but because $\Pi_{ORG}(G, \widehat{c}, r, l, \Pi_{1\text{-}PIR})$ is utilized only in the Setup Phase, this would be incurred as a one-time cost and would not impact cost of the Routing Phase.

Proof (Proof of Theorem 1).

Cost. Per-party computation and communication costs for the routing phase are:

Party	**Computation**	**Communication**
S_i	$Cost(\Pi_{Enc})$	c_{Enc}
R_i	$Cost(\Pi_{Dec}) + (1+b) \cdot (1+\log N) \cdot Cost(\Pi_{PIR\text{-}Rec})$	N/A
C	$M \cdot \lvert E \rvert \cdot Cost(\Pi_{PIR\text{-}Query})$	$N \cdot (2 \cdot c_{Enc} + (1+b) \cdot \delta_{PIR})$

where:

- $|E| = (2 \log N + c) \cdot (c \cdot w \cdot N \cdot (1+b))$ is the number of edges in network $B(N, b, c, w)$.
- $Cost(\Pi_{Enc})$ is the (computation) cost of encrypting a message m.
- $Cost(\Pi_{Dec})$ is the (computation) cost of decrypting a ciphertext $Enc_{pk_i}(m)$.
- c_{Enc} is the size of a ciphertext $Enc_{pk_i}(m)$.
- δ_{PIR} is the constant stretch of the underlying rate-1 PIR protocol $\Pi_{1\text{-}PIR}$.
- $Cost(\Pi_{PIR\text{-}Query})$ is the PIR server cost of $\Pi_{1\text{-}PIR}(c \cdot w, c_{Enc} + (1+b) \cdot \delta_{PIR})$.
- $Cost(\Pi_{PIR\text{-}Rec})$ is the cost of running the reconstruction algorithm (on a PIR response) for $\Pi_{1\text{-}PIR}(c \cdot w, c_{Enc} + (1+b) \cdot \delta_{PIR})$.

Correctness. The intuition for the proof is as follows: Independent of adversarial presence, we first demonstrate bounds of certain properties of routing in the Beneš network, as per the protocols described in Figures 3 and 5. Namely, we demonstrate in Corollary 16 that, with overwhelming probability, for any row

[5] Trusted setup is required for establishing public/secret key pairs for encryption and for instantiating ideal functionality $\Pi_{ORG}(G, \widehat{c}, r, l, \Pi_{1\text{-}PIR})$.

Anonymous Permutation Routing Protocol Π_{NIAR}

Input. Anonymous Permutation Routing parameters: $N = 2^n$, "central router" party C, "sender" parties $\{S_i\}_{i\in[N]}$ each with arbitrarily many messages $\{m_{i,\alpha}\}$, "receiver" parties $\{R_i\}_{i\in[N]}$, permutation $\sigma\colon [N]\to[N]$.

Output. For each party index $1 \le i \le N$, receiver $R_{\sigma(i)}$ outputs message $\widetilde{m}_i$.

Notation. Let $\lambda := \max(\lambda_c/(2\text{-}\log 3),\ 2\log N + \max(\lambda_s, 2 + \log\log N)$. Let $G = B(\widehat{N}, b, c, w)$ denote a wide-edged, extended and colored Beneš network with parameters $\widehat{N} = N$, $b = \lambda - 1$, $c = 4\cdot a_\lambda$, and $w = 1.2\cdot\lambda\cdot\log N\cdot(1+\log N)$ (for $a_\lambda := \max(2, \lambda^{1/(\log N-1)})$); see Figure 12 in Section A. For each $1 \le i \le N$, the parties $\{S_i\}$ and $\{R_i\}$ are assigned to "row" i, associating each sender S_i with I_i (the i^{th} "input" node of G, i.e. the left-most node in row i) and each receiver R_i with O_i (the i^{th} "output" node of G).

Setup Phase.

1. For each $i \in [N]$: let (pk_i, sk_i) denote a public-key/secret-key pair. Output: $S_i \leftarrow pk_i$ and $R_{\sigma(i)} \leftarrow sk_i$.
2. For each $1 \le m \le M = \lambda$:
   (a) Choose random paths through G (as per σ). For each $i \in [N]$: let $\mathcal{P}_i = \mathcal{P}_{m,i}$ denote a random path through G (namely, paths are chosen as per protocol $\Pi_{N,\sigma,G}(i)$; Figure 5).
   (b) Assign rate-1 PIR queries and keys to each edge. For each *internal* node $\mu = \mu_{\widehat{c},r,l}$ at position $(\widehat{c}, r, l)$ of G (i.e. color $\widehat{c} \in [c]$, row $r \in [N]$, and level $l \in [0, (b + (1 + b)\cdot\log N)]$; see Fig. 12 in Section A), invoke the oblivious routing gate protocol $\Pi_{ORG(G,\widehat{c},r,l,\Pi_{1\text{-}PIR})}$ (Fig. 6, §6) with inputs $\{(w_i, w'_i)\}_{i\in[N]}$, where:
      - $w_i = \perp$ if $\mu \notin \mathcal{P}_i$; otherwise $w_i \in [1, |\mathcal{I}_l|]$ is the incoming wire index to μ (as specified by $\mathcal{P}_i$).
      - $w'_i = \perp$ if $\mu \notin \mathcal{P}_i$; otherwise $w'_i \in [1, |\mathcal{O}_l|]$ is the outgoing wire index from μ (as specified by $\mathcal{P}_i$).

      For each $j \in [1, |\mathcal{O}_l|]$, let $\{q_{\mu,j}\}$ denote the rate-1 PIR queries and let $\{(\mu, j, \kappa_{\mu,j})\}$ denote the set of reconstruction keys that are output by the $\Pi_{ORG(G,\widehat{c},r,l,\Pi_{1\text{-}PIR})}$ protocol (by definition, a reconstruction key for output wire j of node μ is output by Π_{ORG} if and only if there exists a unique index $i \in [N]$ such that $w'_i = j$).
   (c) Aggregate reconstruction keys along each path. For each $i \in [N]$: let $\kappa_i = \kappa_{m,i}$ be either $\perp$ (if the reconstruction key for *any* outgoing edge $(\mu, j) \in \mathcal{P}_i$ is not output by Π_{ORG}), and otherwise let κ_i be the collection of reconstruction keys $\{(\mu, j, \kappa_{\mu,j})\}_{\mu\in\mathcal{P}_i}$ for each outgoing edge $(\mu, j) \in \mathcal{P}_i$.

   Output: $C \leftarrow \{q_{\mu,j}\}$, and for each receiver: $R_i \leftarrow \kappa_i$.

Fig. 3: Anonymous Permutation Routing protocol Π_{NIAR}.

index $i \in [N]$ there will exist (at least) one experiment $m \in [M]$ for which the path $\mathcal{P}_{m,i}$ is edge-disjoint from all other paths $\{\mathcal{P}_{m,j}\}_{j\neq i}$. Then as per protocol Π_{NIAR} specification (Step 2b of the Output Parties portion of the Routing Phase; see Figure 3), the existence of an edge-disjoint path $\mathcal{P}_i$ means that $R_{\sigma(i)}$

Anonymous Permutation Routing Protocol Π_{NIAR} (continued)

Routing Phase.
Repeat the following procedure for each successive message $\{m_{i,\alpha}\}$:

Senders $\{S_i\}$.
1. Sender S_i encrypts $m_i = m_{i,\alpha}$ under pk_i and sends $Enc_{pk_i}(m_i)$ to router C.

Central Router C.
1. For each $1 \leq m \leq M = \lambda$:
   (a) C runs the Π_{EPR_m} protocol (Fig. 7 in §6) with inputs $\{Enc_{pk_i}(m_i)\}$ (from each sender S_i's Routing Phase Step 1) and $\{q_{\mu,j}\}$ (from Setup Phase). Let $\{z_{m,i,j}\}_{i \in [0,(N\text{-}1)], j \in [1,c]}$ denote the output of Π_{EPR_m}.
   (b) C sends $\{z_{m,i,j}\}_{j \in [c]}$ to receiver R_i.

Receivers $\{R_{\sigma(i)}\}$.
1. Receiver $R_{\sigma(i)}$ initializes final output value $\widetilde{w}_i = \perp$.
2. For each $1 \leq m \leq M = \lambda$:
   (a) Let $\mathcal{P}_i = \mathcal{P}_{m,i}$, and note that $\mathcal{P}_i$ is edge-disjoint (as per Definition 8) if and only if for every node $\{\mu_l\}$ in $\mathcal{P}_i$, $R_{\sigma(i)}$ received the reconstruction key for μ_l (as per Output (ii) of Π_{ORG}).
   (b) If $\mathcal{P}_i$ *is not* edge-disjoint, or if $\widetilde{w}_i \neq \perp$ (i.e. $\widetilde{w}_i$ was set in a previous iteration $m' < m$): do nothing.
   (c) If $\mathcal{P}_i$ *is* edge-disjoint, then $R_{\sigma(i)}$ uses the reconstruction keys $\kappa_{m,i}$ to traverse $\mathcal{P}_i$ *backwards*, starting with the final value $z_{m,i,j}$ that it received from C (along the appropriate color index $j \in [c]$, as specified by $\mathcal{P}_i$) in Step 1b above. Using the reconstruction keys $\{\kappa_\mu\}$, level by level $R_{\sigma(i)}$ reconstructs to remove one layer of the PIR stretch δ_{PIR} that was added by Π_{ORG}. When $R_{\sigma(i)}$ has traversed backwards to level 0, it will have reconstructed value $Enc_{pk_i}(m_i)$. In this case, $R_{\sigma(i)}$ updates $\widetilde{w}_i$ with value $\widetilde{w}_{m,i} = Dec_{sk_i}(m_i)$.
3. $R_{\sigma(i)}$ outputs $\widetilde{w}_i$.

Fig. 4: Anonymous Permutation Routing protocol Π_{NIAR} (continued).

will update $\widetilde{w}_i \leftarrow \widetilde{w}_{m,i}$. By the correctness property of the ideal functionality of Π_{ORG}, this value will be *correct* (i.e. it will equal p_i).

Formally, with $\lambda = \max(\frac{\lambda_c}{2\text{-}\log 3},\ 2\log N + \max(\lambda_s, 2 + \log\log N)) \geq \frac{\lambda_c}{2-\log 3}$, Lemma 16 states that the probability that there exists some row index $i \in [N]$ for which $\mathcal{P}_{m,i}$ is *not* edge-disjoint for *every* experiment $m \in [M]$ is bounded by:

$$\Pr[X = 0] \ < \ \left(\frac{3}{4}\right)^{\lambda} \ \leq \ \left(\left(\frac{3}{4}\right)^{\frac{1}{2-\log 3}}\right)^{\lambda_c} \ = \ \frac{1}{2^{\lambda_c}}.$$

Security. As with the Correctness proof, we first demonstrate (probability bounds for) a version of the *edge-disjoint* property in the Beneš graph G (Section 3.1) used in Figure 3. Namely, we demonstrate in Corollary 24 that, using the parameters as per Π_{NIAR} (Figure 3), with overwhelming probability (in λ_s), for any pair of row indices $i, i' \in [N]$ and for every experiment $m \in [M]$, there will exist a block in which the chosen paths $\mathcal{P}_{m,i}$ and $\mathcal{P}_{m,i'}$ *as well as* their alternate paths $\mathcal{P}'_{m,i}$ and $\mathcal{P}'_{m,i'}$ are each edge-disjoint from all other paths in this block. Effectively, this means that for any two *uncorrupted* receiver nodes $i, i' \notin R_{\mathcal{A}}$, that for each experiment there exists some block in which the Adversary will necessarily lose all ability to distinguish between $\mathcal{P}_{m,i}$ and $\mathcal{P}_{m,i'}$ by the time these paths cross through this block. We then use a hybrid argument to show that the existence of an adversary that can distinguish between two arbitrary permutations (as per (1)) implies the existence of an adversary who can distinguish (with a smaller probability) between two permutations that differ only on two points; and then this contradicts the existence of a block in which any two paths become indistinguishable after that block.

Formally, the proof reduces the NIAR security game (with Challenger invoking the protocol Π_{NIAR} of Fig. 3) to Challenge Game 2, and then uses the indistinguishability of Challenge Game 2 (Lemma 26). To match notation of Π_{NIAR} with the communication sent to adversary $\mathcal{A}$ in the NIAR security game:

For Step 4 of the NIAR security game:

- Encryption keys $\{\mathsf{pk}_i\}$: The $\{pk_i\}$ from Step 1 of the Setup Phase (Figure 3).
- Decryption keys $\{\mathsf{sk}_i\}$: The $\{sk_i\}$ from Step 1 of the Setup Phase, together with the reconstruction keys $\{\kappa_i\} = \{(\mu, j, \kappa_{\mu,j})\}$ from Step 2b of the Setup.
- Router token q: The rate-1 PIR queries $\{q_{\mu,j}\}$ from Step 2b of the Setup.

For Step 5b of the NIAR security game:

- Ciphertexts $\{c_{i,\alpha}\}$: The encrypted messages $\{Enc_{pk_i}(m_{i,\alpha})\}$ from Sender's Step 1 of the Routing Phase (Figure 3).

First observe that indistinguishability of the distribution of ciphertexts $\{c_{i,\alpha}\} = \{Enc_{pk_i}(m_{i,\alpha})\}$ under $b = 0$ versus $b = 1$ follows from the security of the encryption scheme, together with the constraint that all messages bound for a corrupt receiver must match for $b = 0$ and $b = 1$ (see the specified constraint in Step 5a of the NIAR security game). Thus, for any ciphertext $c_{i,\alpha}$ for which Adversary $\mathcal{A}$ does *not* hold the decryption key, the security of the encryption scheme ensures indistinguishability of this as a ciphertext of $m^{(0)}_{i,\alpha}$ versus $m^{(1)}_{i,\alpha}$; and for any ciphertext $c_{i,\alpha}$ for which Adversary $\mathcal{A}$ *does* hold the decryption key, the constraint in Step 5a of the NIAR security game dictates that this ciphertext encodes a common message $m^{(0)}_{i,\alpha} = m^{(1)}_{i,\alpha}$.

Next we argue indistinguishability of the encryption keys $\{pk_i\}_{i \in S_{\mathcal{A}}}$ and the decryption keys $\{sk_i\}_{i \in R_{\mathcal{A}}}$. Notice first that due to the constraint in Step 1b of the NIAR security game, the distribution of decryption keys $\{sk_i\}_{i \in R_{\mathcal{A}}}$ looks the same for $b = 0$ and $b = 1$, since σ_0 and σ_1 necessarily agree here (i.e. they each map some index $j \in [N]$ to i. Meanwhile, for the distribution of encryption keys, we focus on indices $i \in [N]$ for which $\sigma_0(i) \neq \sigma_1(i)$. Fix any such i, and define $j = \sigma_0(i)$ and $j' = \sigma_1(i)$, so $j \neq j'$. Again due to the constraint in Step 1b of the NIAR security game, we have that neither j nor j' is in $R_{\mathcal{A}}$. This means that Adversary $\mathcal{A}$ does *not* hold the corresponding decryption key for pk_i regardless of whether $b = 0$ or $b = 1$, and thus by the security of the encryption scheme, the distribution of pk_i for $b = 0$ appears identical as the distribution when $b = 1$.

For indistinguishability of the router token $\mathsf{q} = \{q_{\mu,j}\}$: for a given $q_{\mu,j}$ for which Adversary $\mathcal{A}$ does *not* hold the corresponding reconstruction key $\kappa_{\mu,j}$, indistinguishability follows from the security of the underlying rate-1 PIR scheme.

Conversely, for a given $q_{\mu,j}$ for which Adversary $\mathcal{A}$ *does* hold the corresponding reconstruction key $\kappa_{\mu,j}$, $\mathcal{A}$ learns the input wire index that $q_{\mu,j}$ is selecting. However, notice that the paths chosen through G are independent of each other and depend only on the given (sender, receiver) indices and coin flips of path selection protocol $\Pi_{N,\sigma,G}(i)$ (Figure 5), and also that $\mathcal{A}$ knows reconstruction key $\kappa_{\mu,j}$ if and only if outgoing edge (μ, j) is on the path leading to a corrupt receiver $i \in R_{\mathcal{A}}$. Therefore, we again rely on the constraint in Step 1b of the NIAR security game to argue that σ_0 and σ_1 must agree on the (sender, receiver) indices for this path, so the input wire index that $q_{\mu,j}$ is selecting is the same.

It remains to argue indistinguishability of the reconstruction keys $\{\kappa_i\}_{i \in R_{\mathcal{A}}} = \{(\mu, j, \kappa_{\mu,j})\}$. If for a given tuple $(\mu, j, \kappa_{\mu,j})$ the last component is a *valid* reconstruction key (i.e. $\kappa_{\mu,j} \neq \perp$), then indistinguishability follows the same argument as above for the router token. On the other hand, if $\kappa_{\mu,j} \neq \perp$, then as per the Correctness property of any Π_{ORG} protocol, Adversary $\mathcal{A}$ learns that at least two distinct paths chose outgoing edge (μ, j). Since this is the exact scenario as Challenge Game 2, the proof now follows from Lemma 26.

5 Correctness and Security

In this section, we present a series of definitions and lemmas that allow us to argue our main protocol (Figures 3 and 4) satisfies the correctness and security properties of the NIAR model (Section 3.2). The main technical work lies in proving Security; this requires first defining a key property that networks can exhibit (Definition 20), then demonstrating that the Beneš Network we use satisfies this property (Corollary 24), and finally demonstrating how this property ensures security (see Challenge Games 1 and 2 in Section 5.3). As there are a number of lemmas and definitions to go through, to preserve the flow and focus on the main ideas, all proofs appear at the end of the paper.

5.1 Probabilities in a Beneš Network

The main goal of this section is to define a property of graphs that will allow us to formally argue that anonymity is achieved. As mentioned in the Introduction, this property is a stronger variant of edge-disjointness, which we call "local reversal edge-disjoint." Informally:

Definition 2 *(Informal). Given any permutation on N sets of (sender, receiver) pairs, a pairwise$_{i,j}$ reversal refers to swapping the receivers of senders i and j. When viewing a "block" of a permutation network (which also has N input nodes and N output nodes), a local pairwise$_{i,j}$ reversal refers to swapping the output nodes of two input nodes. A set of $N+2$ paths through a block, which include one path for each (sender, receiver) pair plus two extra paths connecting sender i to receiver j (and sender j to receiver i) is said to be local pairwise$_{i,j}$ reversal edge-disjoint if these $N+2$ paths are edge-disjoint. A permutation network is said to enjoy the local reversal edge-disjoint property if, for any pair of indices*

(i, j), w.h.p. there exists a block that is local pairwise$_{i,j}$ reversal edge-disjoint for $N + 2$ randomly chosen paths.

Formally, Definitions 20 and 23 define the "*local reversal edge-disjoint*" property, and it is used to prove security via Corollary 24 and in the analysis of (12)).

Lemma 3 *Suppose that for each input node $\{\nu_i\}_{i=1}^N$ on the butterfly network of Figure 8 (see Section A), a random path $\mathcal{P}_i$ of $\log N$ steps is performed. For any node μ_l (at level $l \in [0, \log N]$), let X_{μ_l} denote the random variable that indicates how many of the paths $\{\mathcal{P}_i\}$ pass through node μ_l. Then for any integer $k \geq 1$:*

$$\Pr[X_{\mu_l} \geq k] \ \leq \ \frac{2^l}{k!} \tag{2}$$

Lemma 4 *Suppose that for each input node[6] $\{\nu_i\}_{i=1}^N$ of a **colored** butterfly network (with replication factor c) of Figure 9 (see Section A), a random path $\mathcal{P}_i$ of $(1+\log N)$ steps is performed (the first step chooses the color $\widehat{c} \in [c]$). For any node $\mu_l = \mu_{\widehat{c},r,l}$ (at level $l \in [0, \log N]$, row $r \in [N]$, and color $\widehat{c} \in [c]$), let X_{μ_l} denote the random variable that indicates how many of the paths $\{\mathcal{P}_i\}$ pass through node μ_l. Then for any integer $k \geq 1$:*

$$\Pr[X_{\mu_l} \geq k] \ \leq \ \frac{2^l}{k! \cdot c^k} \tag{3}$$

Lemma 5 *Suppose that for each input node $\{\nu_i\}_{i=1}^N$ of a **colored** butterfly network (with replication factor c) of Figure 9 (see Section A), a random path $\mathcal{P}_i$ of $(1 + \log N)$ steps is performed (the first step chooses the color $\widehat{c} \in [c]$). For any integer $k \geq 1$, let X_k denote an indicator variable on whether there exists **any** node μ (in the entire colored butterfly network) that has more than k (of the N total) random paths $\{\mathcal{P}_i\}$ pass through it. Then:*

$$\Pr[X_k = 1] \ \leq \ \frac{2c \cdot N^2}{k! \cdot c^k} \tag{4}$$

We now extend a (colored) butterfly network by concatenating several "blocks," each block consisting of $\log N$ levels, and then finishing with one final level that is the mirror reflection of a butterfly network:

Definition 6 *An extended (colored) Beneš network with b blocks consists of b butterfly networks concatenated together, followed by a single (reflected) butterfly network. Additionally, where each pair of blocks are connected, there is a single*

[6] A colored butterfly network can be viewed as c disjoint butterfly networks overlaid on top of one another. Alternatively, we can view a colored butterfly network as a single (connected) graph by adding an extra input level (with level index -1) on the far left, consisting of N input nodes. Then there are c edges emanating from each input node, connecting it to each of the c colored nodes in level 0 of the corresponding row.

level inserted which consists of edges connecting all colors of each node (at each "row") to each other; see Figure 12 in §A. A block j*, for* $j \in [1, (1 + b)]$*, refers to the* $(1 + \log N)$ *levels (and edges) between levels* $(j - 1) \cdot (1 + \log N)$ *and* $j \cdot (1 + \log N)$*. That is, a block corresponds to a contiguous set of* $(1 + \log N)$ *levels, whose first* $\log N$ *levels are a butterfly network, and the last level is the "connecting" level that consists of all edges connecting the different colors of all nodes on the same "row."*[7] *The* input *level of a block* $j \in [1, 1 + b]$ *is level* $(j - 1) \cdot (1 + \log N)$*, and the* output *level is* $j \cdot (1 + \log N)$ *(notice the input level of block* b *is the same as the output level of block* $b - 1$*).*

The following is analogous to Lemma 5, but bounds the probability with respect to each *block* of an extended, colored Beneš network:

Lemma 7 *Let* $\sigma : [N] \rightarrow [N]$ *be an arbitrary permutation on* N *items. Suppose that for each input node* $\{\nu_i\}_{i=1}^{N}$ *of an extended, colored Beneš network with replication factor* c *and* b *blocks, a random path* $\mathcal{P}_i$ *of* $(1 + b \cdot (1 + \log N))$ *steps is performed, and then each such path is extended (from level* $(b \cdot (1 + \log N))$ *to level* $(1 + b) \cdot (1 + \log N)$*) by traversing the unique path from the current node (on level* $(b \cdot (1 + \log N))$*) to* $\sigma(i)$ *(see Figure 12 in §A). For any* $j \in [1,\ (b+1)]$ *and for any integer* $k \geq 1$*, let* $X_{j,k}$ *denote an indicator variable on whether there exists* ***any*** *node* μ_j *within block* j *(i.e. between levels* $[(j - 1) \cdot (1 + \log N),\ j \cdot (1 + \log N) - 1]$ *that has more than* k *(of the* N *total) random paths* $\{\mathcal{P}_i\}$ *pass through it. Then:*

$$Pr[X_{1,k} = 1] \;=\; Pr[X_{1+b,k} = 1] \;\leq\; \frac{2c \cdot N^2}{k! \cdot c^k}$$

$$\forall\, j \in [2,\ b] : \quad Pr[X_{j,k} = 1] \;\leq\; \frac{c \cdot N^2 \cdot (1 + \log N)}{k! \cdot c^k} \tag{5}$$

5.2 Permutation Routing Problem

We begin with the definitions that are needed to describe the Permutation Routing Problem and the desired properties that a successful solution must exhibit.

Definition 8 *Given a graph* $G = (V, E)$ *and a collection of paths* $\{\mathcal{P}_i\}$ *within the graph, we say that any given path* $\mathcal{P}_i$ *is* edge-disjoint *from the others if no edge in* $\mathcal{P}_i$ *is contained/traversed by any other path. We say the entire collection of paths* $\{\mathcal{P}_i\}$ *is* edge-disjoint *if each individual edge is edge-disjoint.*

Definition 9 *A* Permutation Routing Problem(N, σ, G) *is defined as follows: For input integer* $N \in \mathbb{N}$*, permutation* $\sigma : [N] \rightarrow [N]$*, and graph* G *that has* N *designated "input" nodes* $\{I_1, I_2, \ldots, I_N\}$ *and* N *designated "output" nodes* $\{O_1, O_2, \ldots, O_N\}$*, construct* N ***edge-disjoint*** *paths through* G *that connect each input-output pair* $(I_i, O_{\sigma(i)})$*.*

[7] In the special case of the $(1+b)^{th}$ block, the first $\log N$ levels of this block are a *reflected* butterfly network, and the last level of the block is the final "output" level of the entire network.

We extend the notion of the extended, colored Beneš network to a *wide-edged* variant, in which each edge has been replicated w times (which can equivalently be viewed as each edge having capacity w):

Definition 10 *A wide-edged, extended, colored Beneš network $B(N, b, c, w)$ is an extended and colored Beneš network (Figure 12 in §A) in which, for each level $l \in [1, (b+(1+b)\cdot\log N)]$, each edge connecting levels $(l\text{-}1, l)$ is replicated w times.*

Notice that the added *color* and *edge-width* features serve a similar purpose: they each reduce the probability of an edge conflict (i.e. increase the probability of being edge-disjoint, as per Definition 8); but they do so in slightly different ways: the *color* feature not only introduces new edges, but also additional nodes, so that once a path chooses a color for a particular block (which happens only at the start of each block, when there is a transition between levels in which each edge connects the various "colors" corresponding to the nodes on a common "row;" see Figure 12 in §A), it will not conflict (on the present block) with paths that chose another color. In contrast, the *edge-width* feature reduces the chances that two paths conflict across a given edge; but those same paths may still end up in the same node at the far end of this edge, and thus may conflict in a later edge.

Definition 11 *Given a wide-edged, extended, colored Beneš network $B(N, b, c, w)$, and given a routing algorithm $\Pi = \Pi_{N,\sigma,G=B(N,b,c,w)}$ that attempts to solve the Permutation Routing Problem (Definition 9), for each $i \in [N]$ and for each block $1 \le j \le (1 + b)$, let $\mathsf{X}_{\Pi}(\mathsf{i},\mathsf{j})$ denote the boolean random variable that indicates whether Π constructs an edge-disjoint path* ***on the $\mathbf{j^{th}}$ block*** *for the pair $(I_i, O_{\sigma(i)})$. That is, $X_\Pi(i, j) = 1$ if the path connecting I_i and $O_{\sigma(i)}$ within the j^{th} block (as specified by Π) is edge-disjoint from all other paths specified by Π.*

The algorithm in Figure 5 formalizes a naïve solution for the Permutation Routing Problem in which random paths are chosen in an extended and colored Beneš network. Namely, this algorithm specifies that each path $\mathcal{P}_i$ emanating from input I_i chooses random edges for each level through the first b blocks in an extended and colored Beneš network $B(N, b, c, w)$, and then follows the unique path from its current node on level $(b \cdot (1 + \log N))$ to the destination node $O_{\sigma(i)}$ (by choosing one of the w replicates of each edge along this path).

We now demonstrate several properties that the naïve routing protocol of Figure 5 satisfies.

Lemma 12 *Let $\Pi = \Pi_{\sigma_N}$ denote the routing algorithm of Figure 5 on a wide-edged, extended, and colored Beneš network $B(N, b, c \ge 2, w)$. Then for any $i \in [0, N]$, for any $1 \le j \le (1 + b)$, and for any $1 \le k \le N$, the probability that $X_\Pi(i, j) = 0$ (as per Definition 11) is bounded by:*

$$Pr[X_\Pi(i, j) = 0] \ \le\ (1 + \log N) \cdot \left(\frac{c \cdot N^2(1 + \log N)}{k! \cdot c^k} + \frac{k}{2w} \right) \qquad (6)$$

We now extend Definition 11 (and in particular the indicator random variable $X_\Pi(i, j) = 0$) to a statement about a path $\mathcal{P}_i$ being edge-disjoint across the *entire network G*:

Routing Algorithm $\Pi_{N,\sigma,G}$ in a Wide-Edged, Extended, and Colored Beneš Network $B(N,b,c,w)$

Input. Parameters $N = 2^n, b, c, w \in \mathbb{N}$, and a permutation $\sigma_N : [N] \to [N]$. Also, a given wide-edged, extended, and colored Beneš network $B(N,b,c,w)$ with N designated "input" nodes $\{I_1, I_2, \dots, I_N\}$ (on level -1) and N designated "output" nodes $\{O_1, O_2, \dots, O_N\}$ (on level $(1+b)\cdot(1+\log N)$).

Output. The specification of N paths $\{\mathcal{P}_i\}_{i=1}^N$ connecting each $(I_i, O_{\sigma(i)})$.

Protocol. For each row index $1 \le i \le N$, run the $\Pi_{N,\sigma,G}(i)$ sub-protocol.

Sub-protocol $\Pi_{N,\sigma,G}(i)$: Outputs path $\mathcal{P}_i$ connecting input node I_i with output node $O_{\sigma(i)}$.

1. [First Step.] Choose a uniformly random value $\widehat{c}_{-1} \in [c]$, and set the first edge in $\mathcal{P}_i$ (which connects node I_i (at level -1) to one of its c neighbors on level 0) accordingly.
2. ["Internal" Blocks.] For each block $1 \le j \le b$:
   (a) Choose a uniformly random node μ_j on the output level of block j (level $j\cdot(1+\log N)$). Namely, choose a uniformly random color $\widehat{c}_j \in [c]$ and row $r_j \in [N]$. Notice that choice of μ_j completely determines the up/down (see Fig. 12 in §A, which shows each node in a Beneš network has just two nodes – "up" or "down" – on the next layer it is connected to) specification of path $\mathcal{P}_i$ on block j. It remains only to specify, for each expanded edge, which of the w duplicates of this edge to traverse.
   (b) For each $1 \le l \le (1+\log N)$, choose a uniformly random value $w_{j,l} \in [w]$, which specifies which duplicated edge $\mathcal{P}_i$ will traverse between levels $(j-1)\cdot(1+\log N)+l-1$ and $(j-1)\cdot(1+\log N)+l$.
3. [Final Block.] For each $1 \le l \le \log N$, set the edge connecting level $(b\cdot(1+\log N)+l-1,\ b\cdot(1+\log N)+l)$ in $\mathcal{P}_i$ as follows:
   (a) Choose a uniformly random value $w_l \in [w]$.
   (b) Let $r_l \in \{0,1\}$ be determined by $\sigma(i)$; namely, $r_l = 0$ iff the $(\log N - l - 1)$ bit of the target node $\sigma(i)$'s row is 0. Set the next edge in $\mathcal{P}_i$ according to (w_l, r_l); i.e. choose the w^{th} duplicate of the "up" edge if $r_l = 0$, and otherwise choose the w^{th} duplicate of the "down" edge.
4. [Final Level.] For $l = (1+b)\cdot(1+\log N)$: set the final edge in $\mathcal{P}_i$ to be the unique edge that leads from the current node (on level l) to node $O_{\sigma(i)}$.

Fig. 5: Routing algorithm in a wide-edged, extended and colored Beneš network.

Definition 13 *Given a routing algorithm $\Pi = \Pi_{N,\sigma,G}$ that attempts to solve the Permutation Routing Problem (Definition 9), for each $i \in [N]$, let $\mathsf{X}_{\Pi}(\mathsf{i})$ denote the boolean random variable that indicates whether Π constructs an edge-disjoint path for the pair $(I_i, O_{\sigma(i)})$. That is, $X_\Pi(i) = 1$ if the path connecting I_i and $O_{\sigma(i)}$ (as specified by Π) is edge-disjoint from all other paths specified by Π.*

Lemma 14 *Let $\Pi = \Pi_{\sigma_N}$ denote the routing algorithm of Figure 5 on a wide-edged, extended, and colored Beneš network $B(N,b,c \ge 2,w)$. Then for any $i \in [N]$ and for any $1 \le k \le N$, the probability that $X_\Pi(i) = 0$ (as per Definition 13)* *is bounded by:*

$$Pr[X_{\Pi}(i) = 0] \;\leq\; (1 + b) \cdot (1 + \log N) \cdot \left(\frac{c \cdot N^2 \cdot (1 + \log N)}{k! \cdot c^k} + \frac{k}{2w} \right) \tag{7}$$

Proof. This follows immediately from Lemma 12 by applying a union bound on the $(1 + b)$ blocks of the Beneš network $B(N, b, c, w)$.

We are now ready to present the final definition and corresponding statement that will be required for the correctness property of the protocol in Figure 3.

Definition 15 *Given an (independent) collection $\{\Pi_m\}$ of M routing algorithms that attempt to solve the Permutation Routing Problem (see Definition 9) in a wide-edged, extended and colored Beneš network $B(N, b, c, w)$, let X denote the boolean random variable that indicates if, for every $i \in [N]$, there exists (at least) one experiment $m \in [M]$ in which $X_{\Pi_m}(i) = 1$ (where $X_{\Pi_m}(i)$ is the random variable in Definition 13).*

Corollary 16 *For any security parameter λ and for any input parameters $2^n = N \geq 64$, $b = \lambda - 1$, $c = 4 \cdot a_\lambda$, and $w = 1.2 \cdot \lambda \cdot \log N \cdot (1 + \log N)$ (for[8] $a_\lambda := \max(2, \lambda^{1/(\log N - 1)})$), if the Routing Algorithm of Figure 5 is repeated $M := \lambda$ times, then the probability that $X = 0$ (Definition 15) is bounded by:*

$$Pr[X = 0] \;<\; \left(\frac{3}{4}\right)^{\lambda} \tag{8}$$

Ultimately, Corollary 16 will demonstrate *correctness* of our routing protocol (3). However, for the *security* property, we will need to consider two sets of (input, output) node pairs. The following definition (which extends Definition 11, but for two sets of (input, output) pairs of nodes) will be used to capture the requisite probabilities for our security proof.

Definition 17 *Given a wide-edged, extended, colored Beneš network $B(N, b, c, w)$ and two routing algorithms $\Pi = \Pi_{N,\sigma,G=B(N,b,c,w)}$ and $\Pi' = \Pi'_{N,\sigma,G=B(N,b,c,w)}$ that attempt to solve the Permutation Routing Problem (Definition 9), for any pair of row indices $(i, i') \in [N]$ and for any block $1 \leq j \leq (1 + b)$, let $\mathsf{Y}_{\Pi,\Pi'}(\mathsf{i}, \mathsf{i}', \mathsf{j})$ denote the boolean random variable that indicates whether each of the four paths $\{\mathcal{P}_i, \mathcal{P}_{i'}, \mathcal{P}'_i, \mathcal{P}'_{i'}\}$ are edge-disjoint from all other paths on block j.*

Aside. Notice that Definition 17 is only concerned about what happens on a single block of a wide-edged, extended, and colored Beneš network $B(N, b, c, w)$. In particular, we do not actually require two routing algorithms Π, Π' to be defined on the full network $B(N, b, c, w)$ in order to evaluate whether $Y_{\Pi,\Pi'}(i, i', j)$ equals zero or one on a given block $j \in [1, 1 + b]$ (as per Definition 17); rather, we only need to know what each algorithm does on block j. Also notice that there is no requirement that the four paths be edge-disjoint *from each other.*

[8] Notice $a_\lambda = 2$ if $\lambda \leq N/2$.

Definition 18 *Given a wide-edged, extended, and colored Beneš network $G = B(N, b, c, w)$, and a routing algorithm $\Pi = \Pi_{N,\sigma,G}$ that attempts to solve the Permutation Routing Problem (Definition 9), and given any pair of row indices $i, i' \in [N]$ and any block index $j \in [1,(1+b)]$, define the* block j alternate routing algorithm $\Pi'_{i,i',j}$ *as follows:*

– *$\Pi'_{i,i',j}$ is identical to Π on the first $(j-1)$ blocks.*
– *On the j^{th} block:*
  - *For all $\widehat{i} \notin \{i, i'\}$: $\Pi'_{i,i',j}$ is identical to Π.*
  - *Let μ_i (respectively $\mu_{i'}$) denote the node on the output level (which has level index $j \cdot (1 + \log N)$) of block j that $\mathcal{P}_i$ (respectively $\mathcal{P}_{i'}$) passes through, as per Π (see Step 2a of Figure 5). Then $\Pi'_{i,i',j}$ is identical to Π except that the choice of μ_i versus $\mu_{i'}$ is swapped in Step 2a for i and i'.*[9]
– *For all blocks beyond the j^{th} block:*
  - *For all $\widehat{i} \notin \{i, i'\}$: $\Pi'_{i,i',j}$ is identical to Π.*
  - *For i, i': $\Pi'_{i,i',j}$ is identical to Π, except that it has swapped paths $\mathcal{P}_i$ and $\mathcal{P}_{i'}$.*[10]

With these definitions in hand, we provide an analogous probability bound for $Y_{\Pi,\Pi'}(i, i', j)$ as Lemma 12 provided for $X_{\Pi}(i, j)$.

Lemma 19 *Let $\Pi = \Pi_{\sigma_N}$ denote the routing algorithm of Figure 5 on a wide-edged, extended, and colored Beneš network $B(N, b, c \geq 2, w)$. Fix any pair of row indices $i, i' \in [N]$ and any block index $j \in [1,(1+b)]$, and let $\Pi' = \Pi'_{i,i',j}$ denote the "block j alternate routing protocol" (Definition 18). Then for any $1 \leq k \leq N$, the probability that $Y_{\Pi,\Pi'}(i, i', j) = 0$ (as per Definition 17) is bounded by:*

$$Pr[Y_{\Pi,\Pi'}(i, i', j) = 0] \;\leq\; 4 \cdot (1 + \log N) \cdot \left(\frac{c \cdot N^2 \cdot (1 + \log N)}{k! \cdot c^k} + \frac{k}{2w} \right) \tag{9}$$

Just as $X_{\Pi}(i,j)$ (Definition 11) and the corresponding bound for it (Lemma 12) were extended from variables/statements about *blocks* to variables/statements about the *entire network* (in the corresponding Definition 13 and Lemma 14), we likewise extend $Y_{\Pi,\Pi'}(i, i', j)$ (Definition 17) and the corresponding Lemma 19 to variables/statements about the entire network. However, these extensions differ slightly from before, as ultimately we only need the *existence* of a block that satisfies the key property, as opposed to requiring that *all blocks* satisfy some property.

Definition 20 *Given a wide-edged, extended, and colored Beneš network $G = B(N, b, c, w)$, and given two routing algorithms $\Pi = \Pi_{N,\sigma,G}$ and $\Pi' = \Pi'_{N,\sigma,G}$ that attempt to solve the Permutation Routing Problem (Definition 9), for any*

[9] Notice that if $\mu_i = \mu_{i'}$, then $\Pi'_{i,i',j}$ is identical to Π (for *all* paths $\{\mathcal{P}_i\}$) on all blocks through j (including block j).

[10] Swapping paths is only necessary for the sake of making sure the paths link up/connect between blocks (since output node μ_i and $\mu_{i'}$ were swapped in block j). However, as was noted in the Aside note following Definition 17, the details of what $\Pi'_{i,i',j}$ does beyond block j will be irrelevant for the context of Lemmas 19 and 22.

pair of row indices $(i, i') \in [N]$*, let* $\mathsf{Y}_{\Pi,\Pi'}(\mathsf{i},\mathsf{i}')$ *denote the boolean random variable that indicates whether there exists some block* $j \in [1,(1+b)]$ *in which the four paths* $\{\mathcal{P}_i, \mathcal{P}_{i'}, \mathcal{P}'_i, \mathcal{P}'_{i'}\}$ *are each edge-disjoint from all other paths on block* j*.*

Definition 21 *Given a wide-edged, extended, and colored Beneš network* $G = B(N,b,c,w)$*, and a routing algorithm* $\Pi = \Pi_{N,\sigma,G}$ *that attempts to solve the Permutation Routing Problem (Definition 9), and given any pair of row indices* $i, i' \in [N]$*, define the* alternate routing algorithm $\Pi'_{\mathsf{i},\mathsf{i}'}$ *as follows:*

1. $\forall\, j \in [1,(1+b)]$*, let* $\Pi'_j = \Pi'_{i,i',j}$ *denote the block* j *alternate routing algorithm (Definition 18).*
2. *Construct* $\Pi'_{i,i'}$ *from the family of alternate routing algorithms* $\{\Pi_j\}$ *as follows:*
   a. *If there exists an index* $j \in [1,(1+b)]$ *such that* $Y_{\Pi,\Pi'_j}(i,i',j) = 1$ *(as per Definition 11), then let* $\Pi'_{i,i'} = \Pi'_j$ *(for the minimal* j *satisfying* $Y_{\Pi,\Pi'_j}(i,i',j) = 1$*).*
   b. *Otherwise, define* $\Pi'_{i,i'} = \Pi$*.*

Lemma 22 *Let* $\Pi = \Pi_{\sigma_N}$ *denote the routing algorithm of Figure 5 on a wide-edged, extended, and colored Beneš network* $B(N,b,c \geq 2,w)$*, let* $i,i' \in [N]$ *be any two row indices, and let* $\Pi' = \Pi'_{i,i'}$ *be the alternate routing algorithm (as per Definition 21). Then for any* $1 \leq k \leq N$*, the probability that* $Y_{\Pi,\Pi'}(i,i') = 0$ *(as per Definition 20) is bounded by:*

$$Pr[Y_{\Pi,\Pi'}(i,i') = 0] \;\leq\; \left(4 \cdot (1+\log N) \cdot \left(\frac{c \cdot N^2 (1+\log N)}{k! \cdot c^k} + \frac{k}{2w}\right)\right)^{(1+b)} \tag{10}$$

We are now ready to present the final definition and corresponding statement that will be required for the security proof of the protocol in Figure 3.

Definition 23 *Given an (independent) collection* $\{\Pi_m\}$ *of* M *routing algorithms that attempt to solve the Permutation Routing Problem (Definition 9) in a wide-edged, extended and colored Beneš network* $B(N,b,c,w)$*, let* Y *denote the boolean random variable that indicates if, for every* Π_m *and every pair of row indices* $i,i' \in [N]$*, that* $Y_{\Pi_m,\Pi'_m}(i,i') = 1$ *(where* $\Pi'_m = \Pi'_{m,i,i'}$ *is the alternate routing algorithm (Definition 21) and* $Y_{\Pi_m,\Pi'_m}(i,i')$ *is the corresponding random variable (Definition 20)).*

Corollary 24 *For*[11] *any security parameter* $\lambda \geq 8$ *and any input parameters* $2^n = N \geq 64$*,* $b = \lambda - 1$*,* $c = 4 \cdot a_\lambda$*, and* $w = 1.2 \cdot \lambda \cdot \log N \cdot (1+\log N)$ *(for* $a_\lambda := \max(2, \lambda^{1/(\log N - 1)})$*), if the Routing Algorithm of Figure 5 is repeated* $M := \lambda$ *times, then the probability that* $Y = 0$ *(Definition 23) is bounded by:*

$$Pr[Y = 0] \;<\; \frac{\lambda \cdot N^2}{4^\lambda} \tag{11}$$

[11] Notice that these parameter values all match those in the hypothesis of Corollary 16.

5.3 Security

Succinctly, security (anonymity) will follow for the routing protocol of Figure 3 from:

$$\begin{aligned} \text{Corollary 24} &\Rightarrow (!\exists \mathcal{A} \text{ with non-negligible advantage in Challenge Game 1}) \\ &\Rightarrow (!\exists \mathcal{A} \text{ with non-negligible advantage in Challenge Game 2}) \\ &\Rightarrow (\text{Routing Protocol of Figure 3 is secure (per Definition 9)}) \quad (12) \end{aligned}$$

In this section, we define Challenge Games 1 and 2, and then demonstrate the first two implications in (12) (the third implication was already presented in the proof of Theorem 1).

Challenge Game 1

Input Parameters:

- Number of input/output nodes $2^n = N \geq 64$.
- Security parameter $\lambda \geq 8$.
- A wide-edged, extended and colored Beneš network $G = B(N, b, c, w)$, with parameters as per Corollaries 16 and 24: $b = \lambda - 1$, $c = 4 \cdot a_\lambda$, and $w = 1.2 \cdot \lambda \cdot \log N \cdot (1 + \log N)$ (for $a_\lambda := \max(2, \lambda^{1/(\log N - 1)})$).
- There are N "global input nodes" on level -1 of the Beneš network $G = B(N, b, c, w)$, which are denoted: $\mathcal{I} = \{I_1, I_2, \ldots, I_N\}$, and N global output nodes $\mathcal{O} = \{O_1, O_2, \ldots, O_N\}$.
- Set the experiment replication amount $M = \lambda$.

Challenge Game:

1. Challenger $\mathcal{C}$ chooses a permutation σ on N elements $\sigma : [N] \to [N]$.
2. For each experiment $m \in [M]$: Challenger $\mathcal{C}$ performs the routing algorithm $\Pi_m = \Pi_{m,N,\sigma_b,G}$ (for $G = B(N, b, c, w)$) of Figure 5. For each $i \in [N]$, let $\mathcal{P}_{m,i}$ denote the path chosen (by Π_m) that connects nodes $(I_i, \sigma_b(I_i))$.
3. Let Y be the boolean random variable from Definition 23. If $Y = 0$, Challenger $\mathcal{C}$ aborts (Adversary $\mathcal{A}$ wins).
4. Challenger $\mathcal{C}$ chooses any two distinct indices $i, i' \in [N]$, and gives[12] $\sigma|_{[N]\setminus\{i,i'\}}$ to Adversary $\mathcal{A}$, which is the mapping of σ on all indices *except* i and i'. Notice that since σ is a permutation, Adversary $\mathcal{A}$ now has complete knowledge of σ, *except* for what σ does to i and i'. In particular, there are two range indices $\sigma(i), \sigma(i') \in [N]$ that are *not* mapped to (based on what $\mathcal{C}$ gives to $\mathcal{A}$). Let τ denote the permutation that is identical to σ, except that it swaps where i and i' are mapped to (so $\tau(i) = \sigma(i')$ and $\tau(i') = \sigma(i)$). Notice that after this step, Adversary $\mathcal{A}$ knows that the permutation chosen by Challenger $\mathcal{C}$ is either σ or τ.
5. (If this step is reached) Since $Y = 1$, for each run $1 \leq m \leq M$ of the experiment, we have that alternate routing algorithm $\Pi'_{m,i,i'}$ must have been constructed as per Step 2a of Definition 21 (as opposed to Step 2b). Therefore,

[12] This information is also available indirectly from what $\mathcal{C}$ gives to $\mathcal{A}$ in Step 5a below.

let $j_m \in [1, (1+b)]$ denote the block index for which $\Pi'_{m,i,i'}$ is defined as in Step 2a; i.e. j_m (is the minimal index that) satisfies $Y_{\Pi_m, \Pi'_m, j_m}(i, i', j_m) = 1$. Then, for each experiment $m \in [M]$:

(a) [Block Index]: Challenger $\mathcal{C}$ gives Adversary $\mathcal{A}$ the block index j_m (recall this is the first block for which $Y_{\Pi_m, \Pi'_m, j_m}(i, i', j_m) = 1$).

(b) [All Non-Interesting Paths]: Challenger $\mathcal{C}$ gives Adversary $\mathcal{A}$ all paths $\{\mathcal{P}_{m,\hat{i}}\}_{\hat{i} \notin \{i,i'\}}$.

(c) [Interesting Paths *Before* Block j_m]: Challenger $\mathcal{C}$ gives Adversary $\mathcal{A}$, *through the first* (j_m-1) *blocks only*, paths $\mathcal{P}_{m,i}$ and $\mathcal{P}_{m,i'}$.

(d) [Interesting Paths + Alternate Paths for Block j_m]: Denote the two sub-paths of $\mathcal{P}_{m,i}$ and $\mathcal{P}_{m,i'}$ that are restricted to block j_m (i.e. just the edges of these paths within block j_m) and their two alternate sub-paths (as specified by alternate routing protocol $\Pi'_{i,i'}$ (Definition 21)) as: $\{\mathcal{P}_{m,i,j_m}, \mathcal{P}_{m,i',j_m}, \mathcal{P}'_{m,i,j_m}, \mathcal{P}'_{m,i',j_m}\}$.Then Challenger $\mathcal{C}$ gives Adversary $\mathcal{A}$ the *unordered* set $\{\mathcal{P}_{m,i,j_m}, \mathcal{P}_{m,i',j_m}, \mathcal{P}'_{m,i,j_m}, \mathcal{P}'_{m,i',j_m}\}$.

(e) [(Unordered) Interesting Paths *Beyond* Block j_m]: For each level with index $j_m \cdot (1 + \log N) \leq l \leq (1+b) \cdot (1 + \log N)$ in $G = B(N, b, c, w)$ that lies *after* block j_m, Challenger $\mathcal{C}$ gives Adversary $\mathcal{A}$ the *unordered* set of edges $\{\mathcal{P}_{m,i,l}, \mathcal{P}_{m,i',l}\}_l$, where $\mathcal{P}_{m,i,l}$ (resp. $\mathcal{P}_{m,i',l}$) denotes the l^{th} edge on the path $\mathcal{P}_{m,i}$ (resp. on the path $\mathcal{P}_{m,i'}$). In other words, $\mathcal{A}$ learns the edges (beyond block j_m) traversed by paths $\mathcal{P}_{m,i}$ and $\mathcal{P}_{m,i'}$, but $\mathcal{A}$ is *not* explicitly told which edges belong to which path ($\mathcal{P}_{m,i}$ versus $\mathcal{P}_{m,i'}$).

6. Adversary $\mathcal{A}$ outputs a guess whether Challenger's permutation was σ or τ.

The Adversary $\mathcal{A}$ wins Challenge Game 1 either if Challenger $\mathcal{C}$ aborts in Step 3, or if $\mathcal{A}$'s output guess in Step 6 is correct.

The main result for Challenge Game 1 (which is the first implication in (12)) is:

Lemma 25 *The probability that an (unbounded) Adversary $\mathcal{A}$ wins Challenge Game 1 is bounded by:*

$$Pr[\mathcal{A} \textit{ wins Challenge Game 1}] \leq \frac{1}{2} + \frac{\lambda \cdot N^2}{4^\lambda} \qquad (13)$$

Challenge Game 2

Input Parameters:

- Number of input/output nodes $2^n = N \geq 64$.
- Security parameter λ_s. Let $\lambda := 2 \log N + \max(\lambda_s, 2 + \log \log N)$.
- A wide-edged, extended and colored Beneš network $G = B(N, b, c, w)$, with parameters as per Corollaries 16 and 24: $b = \lambda - 1$, $c = 4 \cdot a_\lambda$, and $w = 1.2 \cdot \lambda \cdot \log N \cdot (1 + \log N)$ (for $a_\lambda := \max(2, \lambda^{1/(\log N - 1)})$).
- There are N "global input nodes" on level -1 of the Beneš network $G = B(N, b, c, w)$, which are denoted: $\mathcal{I} = \{I_1, I_2, \ldots, I_N\}$, and N global output nodes $\mathcal{O} = \{O_1, O_2, \ldots, O_N\}$.
- Set the experiment replication amount $M = \lambda$.

Challenge Game:

1. On input security parameter λ, Adversary $\mathcal{A}$ chooses N, two distinct permutations σ_0, σ_1 on $[N]$, a set of sender indices $S_{\mathcal{A}} \subseteq [N]$ to corrupt, and a set of receiver indices $R_{\mathcal{A}} \subseteq [N]$ to corrupt; subject to constraints:
   (a) $|R_{\mathcal{A}}| \leq N - 2$;
   (b) σ_0 and σ_1 match for all receivers in $R_{\mathcal{A}}$: $\forall\, i \in R_{\mathcal{A}} : \sigma_0^{-1}(i) = \sigma_1^{-1}(i)$.
2. Adversary $\mathcal{A}$ sends $\{\sigma_0, \sigma_1\}$ to a Challenger $\mathcal{C}$.
3. Challenger $\mathcal{C}$ chooses $b \in \{0,1\}$ and selects $\sigma_b \in \{\sigma_0, \sigma_1\}$.
4. For each experiment $m \in [M]$:
   (a) Challenger $\mathcal{C}$ performs the routing algorithm $\Pi_m = \Pi_{m,N,\sigma_b,G}$ (for $G = B(N,b,c,w)$) of Figure 5. For each $i \in [N]$, let $\mathcal{P}_{m,i}$ denote the path chosen (by Π_m) that connects nodes $(I_i, O_{\sigma_b(i)})$.
   (b) Adversary $\mathcal{A}$ is given the following information:
      – For each $i \in R_{\mathcal{A}}$: all edges $e \in \mathcal{P}_{m,i}$ that are edge-disjoint from all other paths $\mathcal{P}_{m,j}$ (for $j \neq i$).
      – The list of edges $\{e\} \in G$ that have at least two distinct paths $\mathcal{P}_{m,i}, \mathcal{P}_{m,i'}$ pass through them, with $i' \neq i$ and $i \in R_{\mathcal{A}}$. Notice that $\mathcal{A}$ is given only the identity of the set of edges $\{e\}$; in particular, $\mathcal{A}$ *is not* given the information of *which* (nor even *how many*) indices in $[N] \setminus R_{\mathcal{A}}$ traverse each such edge.
5. Let Y be the boolean random variable from Definition 23. If $Y = 0$, Challenger $\mathcal{C}$ aborts (Adversary $\mathcal{A}$ wins).
6. Adversary $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$ of which permutation $\{\sigma_0, \sigma_1\}$ Challenger $\mathcal{C}$ chose.

We say that the Adversary wins the above challenge if its output is correct.

The main result for Challenge Game 2 (which is the second implication in (12)) is:

Lemma 26 *The probability that an (unbounded) Adversary $\mathcal{A}$ wins Challenge Game 2 is bounded by: Pr[$\mathcal{A}$ wins Challenge Game 2]* $\leq \frac{1}{2} + \frac{1}{2^{\lambda_S}}$.

6 Subprotocol Definitions and Constructions

In this section, we present the formal definitions of the two subprotocols referenced in our main NIAR protocol: An oblivious routing gate protocol Π_{ORG}, and an emulated permutation routing protocol Π_{EPR}. We begin by formally defining the input and output requirements of an oblivious routing gate:

Definition 27 *A* routing gate evaluation *protocol $\Pi_{RG_{Enc}}$ for routing gate $\mu \in G$ takes as input values $\{v_j\}$ on each incoming edge of μ and encoded indices $\{q_j\}$ on each outgoing edge of μ (where each q_j encodes the index of an incoming edge), and outputs values $\{z_j\}$ for each outgoing edge of μ, where each $z_j = Enc(v_{q_j})$ is an encoding of the value on the appropriate input wire (as specified by q_j). A* routing gate reconstruction *protocol $\Pi_{RG_{Dec}}$ takes as input reconstruction keys $\{\kappa_{\mu,j}\}$ and values $\{z_{\mu,j}\}$ (one key-value pair $(\kappa_{\mu,j}, z_{\mu,j})$ for each outgoing edge of μ), and outputs values $\widehat{v}_{\mu,j}$ for each outgoing edge of μ.*

Definition 28 *Let $\mu \in G$ denote a gate in a permutation-routing network, and let $\mathcal{I}_\mu$ and $\mathcal{O}_\mu$ denote its incoming and outgoing edges (respectively). An* oblivious routing gate (ORG) *protocol for $\mu \in G$ takes as input, for each outgoing edge $\widehat{e}_\mu \in \mathcal{O}_\mu$, an associated index $j_{\widehat{e}_\mu} \in \perp \cup [1,\ |\mathcal{I}_\mu|]$, and outputs:*

i. Encodings $\{q_{\widehat{e}_\mu}\}$ of the input indices $\{j_{\widehat{e}_\mu}\}$;
ii. Specification of a routing gate evaluation protocol $\Pi_{RG_{Enc}}$;
iii. Reconstruction keys $\{\kappa_{\widehat{e}_\mu}\}$;
iv. Specification of a routing gate reconstruction protocol $\Pi_{RG_{Dec}}$ protocol;

with the outputs subject to the constraints:

Message Independence. Outputs $\{q_{\widehat{e}_\mu}\}$ are defined independently from (later-specified) messages $\mathcal{M}_\mu$.

Reusability. Outputs $(\{q_{\widehat{e}_\mu}\}, \{\kappa_{\widehat{e}_\mu}\}, \Pi_{RG_{Enc}}, \Pi_{RG_{Dec}})$ can be reused (indefinitely) for new messages $\{\mathcal{M}_{\mu,1}, \mathcal{M}_{\mu,2}, \ldots\}$, without compromising correctness or security properties for each subsequent run.

Additionally, the oblivious routing gate paradigm requires ***correctness*** *and* ***security*** *properties:*

Correctness. For each $1 \leq j \leq |\mathcal{O}_l|$, output values $(q_j, \kappa_{\mu,j})$ satisfy:

***Case 1:** $j \notin \{w'_i\}_{S_i}$. In this case (no output wire index w'_i indicated wire index j), there is no correctness requirement for q_j (except that it is a valid rate-1 PIR query), and $\kappa_{\mu,j} = \perp$.*
***Case 2:** $j = w'_i$ **for exactly one** $i \in [N]$. In this case (exactly one output wire index w'_i is equal to j), q_j is a PIR query corresponding to position w_i, and $\kappa_{\mu,j}$ is the corresponding reconstruction key.*
***Case 3:** $j = w'_i = w'_{i'}$. This case (two or more output wires $\{w'_i, w'_{i'}\}$ both equal j) is the same as Case 1.*

Security. Consider the following security game featuring a (polynomially bounded) adversary $\mathcal{A}$:

1. *$\mathcal{A}$ sees all output values $\{q_{\mu,j}\}_{j\in[1,|\mathcal{O}_l|]}$.*
2. *$\mathcal{A}$ specifies any subset $R_\mathcal{A} \subset [N]$, subject only to the constraint that $|R_\mathcal{A}| \leq N-2$. Let $\mathcal{T} := [N] \setminus R_\mathcal{A}$ denote the set of indices in $[N]$ that are not selected as part of $R_\mathcal{A}$, and let $\mathcal{O}_\mathcal{A} := \{w'_i \mid w'_i \neq \perp\}_{i\in R_\mathcal{A}}$ denote the set of (non-$\perp$) output wire indices for all $i \in R_\mathcal{A}$. Further partition $\mathcal{T}$ as:*
   (a) $\mathcal{T}_1 \subseteq \mathcal{T} = \{i \mid w'_i = \perp\}$: The subset of $\mathcal{T}$ consisting of indices $i \in [N]$ whose output wire index $w'_i = \perp$
   (b) $\mathcal{T}_2 \subseteq \mathcal{T} \setminus \mathcal{T}_1 = \{i \mid w'_i \in \mathcal{O}_\mathcal{A}\}$: The subset of $\mathcal{T}$ consisting of indices $i \in [N]$ whose output wire index w'_i (is not equal to $\perp$ and) equals the output wire index $w'_{i'}$ for some index $i' \in R_\mathcal{A}$
   (c) $\mathcal{T}_3 = \mathcal{T} \setminus (\mathcal{T}_1 \cup \mathcal{T}_2)$: The subset of $\mathcal{T}$ consisting of indices $i \in [N]$ whose output wire index $w'_i \notin (\mathcal{O}_\mathcal{A} \cup \perp)$
3. *$\forall i \in (R_\mathcal{A} \cup \mathcal{T}_2)$, $\mathcal{A}$ sees $(w_i, w'_i) \in (\perp \cup \{1, \ldots, |\mathcal{I}_l|\}) \times (\perp \cup \{1, \ldots |\mathcal{O}_l|\})$.*
4. *$\mathcal{A}$ chooses $j \notin (R_\mathcal{A} \cup \mathcal{T}_2)$, and outputs guess (w_j, w'_j) for j's input and output wire indices.*

5. $\mathcal{A}$ wins if (at least) one of its guesses $\{w_j, w'_j\}$ is both correct and not[13] *the special value $\perp$.*

A Π_{ORG} protocol is secure *if any (polynomially bounded) adversary $\mathcal{A}$'s probability of winning the above security game is bounded by:*

$$\begin{aligned} Pr[\mathcal{A} \textit{ wins via correct } w_j] &\leq \frac{1}{|\mathcal{I}_l|} \qquad \textit{and} \\ Pr[\mathcal{A} \textit{ wins via correct } w'_j] &\leq \frac{1}{|\mathcal{O}_l| - |\mathcal{O}_\mathcal{A}|} \end{aligned}$$

Succinctly, a Π_{ORG} protocol is secure if the outputs $\{q_{\widehat{e}_\mu}\}$ reveal nothing about the inputs $\{j_{\widehat{e}_\mu}\}$ they encode; and it is correct if the message written on each outgoing edge gets decoded to the correct message from the appropriate incoming edge.

Figure 6 depicts a possible instantiation of an oblivious routing gate with subprotocols $(\Pi_{RG_{Enc}}, \Pi_{RG_{Dec}})$ set as (PIR-server, PIR-client) protocols (with output values $\{q_{\widehat{e}_\mu}\}$ the PIR queries and $\{\kappa_{\widehat{e}_\mu}\}$ the client's reconstruction keys). Correctness and Security of the resulting Π_{ORG} protocol follow from the correctness and security of the underlying PIR protocol.

[13] Observe that Adversary $\mathcal{A}$ *cannot* win if $\mathcal{T}_3 = \varnothing$.

Oblivious Routing Gate in Beneš Network G via rate-1 PIR

Setup (see Figure 2).

– Parameters for a wide-edged, extended and colored Beneš network $G = B(N, b, c, w)$: $N = 2^n, b, c, w$.
– Color index $\widehat{c} \in [1, c]$, row index $r \in [N]$, level index $l \in [0, (b+(1+b)\cdot\log N))]$.
– Let $\mu = \mu_{\widehat{c},r,l}$ denote the node at position $(\widehat{c}, r, l)$ of G (i.e. color $\widehat{c}$, row r, and level l; see Figure 12 in §A).
– Let $\mathcal{I} = \mathcal{I}_l$ denote the set of input wires to node μ, and notice for network G:

$$|\mathcal{I}_l| = \begin{cases} 1 & \text{if } l = 0 \\ c \cdot w & \text{if } l = j \cdot (1 + \log N) \text{ (for arbitrary } j \in [1, b]) \\ 2 \cdot w & \text{otherwise} \end{cases} \tag{14}$$

– Let $\mathcal{O} = \mathcal{O}_l$ denote the set of output wires from node μ, and notice (by definition of Beneš network G):

$$|\mathcal{O}_l| = \begin{cases} 1 & \text{if } l = (1 + b) \cdot (1 + \log N) - 1 \\ c \cdot w & \text{if } l = j \cdot (1 + \log N) - 1 \text{ (for arbitrary } j \in [1, b]) \\ 2 \cdot w & \text{otherwise} \end{cases} \tag{15}$$

– When the Π_{ORG} protocol is invoked, each of the $|\mathcal{I}_l|$ input wires will have an associated value on them. Let $|v| = |v_l|$ denote the size of each value.[a]
– Let $\Pi_{1\text{-}PIR} = \Pi_{1\text{-}PIR}(|v_l|, |\mathcal{I}_l|)$ denote a rate-1 PIR protocol for $|\mathcal{I}_l|$ elements, each of size $|v_l|$.
  - Let β_l denote the size of the reconstruct key (used by PIR client to parse the server's response).
  - Let $\gamma = \gamma_l$ denote the number of bits in a query to $\Pi_{1\text{-}PIR}(|v_l|, |\mathcal{I}_l|)$.
  - Let δ_{PIR} denote the (additive) stretch constant.
– Parties $\{S_1, \ldots, S_N\}$ connected via σ to parties $\{R_1, \ldots, R_N\}$ and C.

Input. For each $i \in [N]$, values (w_i, w'_i) with "input wire index" $w_i \in (\perp \cup \{1, 2, \ldots, |\mathcal{I}_l|\})$ and "output wire index" $w'_i \in (\perp \cup \{1, 2, \ldots |\mathcal{O}_l|\})$, subject to constraint: $w_i = \perp \Leftrightarrow w'_i = \perp$.

Output. For each output wire index $1 \leq j \leq |\mathcal{O}_l|$:

i. Rate-1 PIR query $q_j = q_{\mu,j} \in \{0, 1\}^\gamma$.
ii. $(\mu, j, \kappa_{\mu,j})$ with $\kappa_{\mu,j}$ either $\perp$ or a reconstruction key for q_j (as per Correctness requirement below).

[a] When Π_{ORG} is invoked, $|v_l| = (c_{Enc} + l \cdot \delta_{PIR})$, where c_{Enc} is the ciphertext size of some scheme Enc.

Fig. 6: Instantiation of Π_{ORG} functionality via rate-1 PIR in Beneš Network G.

With the oblivious routing gate paradigm in hand, we define the emulated permutation routing functionality, which captures how the NIAR router C transmits messages through a (virtual) permutation-routing network:

Definition 29 *Let G denote a permutation-routing network: N pairs of designated (sender, receiver) nodes and a permutation $\sigma : [N] \to [N]$ connecting them. Let $\{\mathcal{P}_i\}$ denote a set of N paths that link each (sender, receiver) pair with a path through G. For every node $\mu \in G$, and for each outgoing edge $\widehat{e}_\mu \in \mathcal{O}_\mu$, define the index $j_{\widehat{e}_\mu} \in \perp \cup [1,\ |\mathcal{I}_\mu|]$ as follows: If exactly one path $\mathcal{P}_i$ traverses outgoing edge $\widehat{e}_\mu$, then $j_{\widehat{e}_\mu}$ is the index of $\mathcal{P}_i$'s incoming edge to μ. Otherwise, $j_{\widehat{e}_\mu}$ is defined to be $\perp$. Given as input: (i) A set of values $\{v_j\}$ for each outgoing edge e_j that emanates from one of the N source nodes; and (ii) A protocol Π_{ORG} instantiating the oblivious routing gate paradigm; an emulated permutation routing (EPR) protocol $\Pi_{EPR}(\Pi_{ORG}, \{v_j\})$, is defined as:*

(One-time) Setup Phase. *For each node $\mu \in G$, run $\Pi_{ORG}(\{j_{\widehat{e}_\mu}\})$, and let $\{q_{\widehat{e}_\mu}\}$ denote the output encodings (from output (i)) and $\Pi_{RG_{Enc}}$ the routing gate evaluation protocol (from output (ii)).*

(Online) Emulation Phase. *Iterate through the levels l of routing network G:*

- *For each outgoing edge $\widehat{e}_\mu$ of each node $\mu \in G$ at level l, invoke protocol $\Pi_{RG_{Enc}}(\widehat{e}_\mu, q_{\widehat{e}_\mu}, \mathcal{M}_\mu)$ on inputs:*
  - *$q_{\widehat{e}_\mu}$: From Setup Phase (above).*
  - *$\mathcal{M}_\mu = \{m_j\}_{j \in [1,\ |\mathcal{I}_\mu|]}$: Values on the input wires of μ (either from input (i) values $\{v_j\}$ if level $l = 1$ is the first level, or from the output values of $\Pi_{RG_{Enc}}$ in iteration l-1).*
- *For each incoming edge $e_{\mu,j}$ of each sink node μ, output value $z_{\mu,j}$, where $z_{\mu,j}$ denotes the value assigned to this edge (viewed as an output edge of some node on the previous level) by $\Pi_{RG_{Enc}}$ (from the last iteration).*

Figure 7 formally describes an emulated permutation routing protocol through the routing network G used in our NIAR protocol (Figures 3 and 4), based on the rate-1 PIR Π_{ORG} protocol of Figure 6. Namely, it specifies precisely what output values to write on every (virtual) edge of G.

Per-Experiment Emulated Permutation Routing Protocol Π_{EPR_m}

Input. Same parameters as parent protocol Π_{NIAR}. Also, router C has:
1. For each node $\mu = \mu_{\widehat{c},r,l}$ and each of μ's $1 \le j \le |\mathcal{O}_l|$ outgoing wires: A rate-1 PIR query $q_{\mu,j}$.
2. For each sender S_i: Ciphertext $Enc_{pk_i}(m_i)$.

Output. For each row index $1 \le r \le N$ and for each of the c input wires to the node μ_r that is on row r of the last level in G, output values $\{z_{r,j}\}_{r\in[N],j\in[c]} \in \{0,1\}^{(c_{Enc}+(1+b)\cdot(1+\log N)\cdot\delta_{PIR})}$.

Protocol. This protocol has central router C simulate sending messages through graph G, using an oblivious routing gate protocol Π_{ORG} (instantiated by rate-1 PIR, as per Figure 6) to implement oblivious routing at each gate. Namely, the simulation begins with router C assigning the values $\{Enc_{pk_i}(m_i)\}$ to each input wire of each node at row i of level 0. The values that are written on the outgoing wires of any node μ_l (for $l \ge 0$) are the (rate-1) PIR response that would result from issuing PIR query q_{μ_l}. We now formalize the protocol rules:
1. [Level -1]. For each node $\mu = \mu_{\widehat{c},r,0}$ on level 0 (for any $\widehat{c} \in [c]$ and any $r \in [N]$), C writes value $Enc_{pk_r}(m_r)$ (from Input Step 2) on each of $\mu_{\widehat{c},r,0}$'s input wires (there are $|\mathcal{I}_0| = c$) such input wires. Let A_μ denote the virtual array that holds these $|\mathcal{I}_0| = c$ values, each of size $|v_0| = c_{Enc}$ (see Figure 2).
2. [Levels $0 \le l \le (b + (1+b)\cdot\log N)$]. For each node $\mu = \mu_{\widehat{c},r,l}$ on level l:
   (a) Let A_μ denote the virtual array holding the $|\mathcal{I}_l|$ values (each value of size $|v_l| = c_{Enc}+l\cdot\delta_{PIR}$) on each input wire leading to μ. Note that this array was constructed in Step (2d) of the previous iteration, or in Step 1 if $l = 0$.
   (b) For each output wire $1 \le j \le |\mathcal{O}_l|$ of μ, C simulates running the PIR query $q_{\mu,j}$ (from Input Step 1) against A_μ. Let $z_{\mu,j}$ denote the PIR response, and notice that (by definition of rate-1) it has size $|z_{\mu,j}| = c_{Enc} + (l+1)\cdot\delta_{PIR}$.
   (c) (If $l = (b+(1+b)\cdot\log N)$ is the last iteration, skip this step and (2d) below.) For each of the $\{z_{\mu,j}\}$, cluster each contiguous set of w values together:
   $$\{(\underbrace{z_{\mu,1},\ldots,z_{\mu,w}}_{\widehat{A}_{\mu,1},}),\ (\underbrace{z_{\mu,1+w},\ldots,z_{\mu,2w}}_{\widehat{A}_{\mu,2},}),\ \ldots,\ (\underbrace{z_{\mu,1+bw},\ldots,z_{\mu,(b+1)w}}_{\widehat{A}_{\mu,b+1}})\}$$
   where $b = 1$ if $l = j\cdot(1+\log N)-1$ (for any $j \in [b]$; see (15)); else $b = c$-1.
   (d) Construct the PIR input array $A_{\mu'}$ for each μ' on the *next* level by concatenating the appropriate number[a] of arrays $\{\widehat{A}_{\mu,j}\}$ (from Step 2c), choosing the appropriate nodes μ (as dictated by which nodes μ have output wires that lead to μ', as per graph G).
3. [Level $(1+b)\cdot(1+\log N)$]. From the $l = (b + (1+b)\cdot\log N)$ iteration of the previous Step 2.b, each node μ' on level $(b+(1+b)\cdot\log N)$ had exactly 1 value $z_{\mu',1}$ that was generated (see e.g. (15)), and this value has size $|z_{\mu',1}| = c_{Enc} + (1+b)\cdot(1+\log N)\cdot\delta_{PIR}$. For any node μ on the final level $l = (1+b)\cdot(1+\log N)$, there are c nodes $\{\mu'_j\}_{j\in[c]}$ from the previous level that each have exactly one output wire leading to μ, and each such output wire has value $z_{\mu'_j,1}$ of size $|z_{\mu'_j,1}| = c_{Enc} + (1+b)\cdot(1+\log N)\cdot\delta_{PIR}$. Output these $\{z_{\mu'_j,1}\}$ as the final output for each node μ on the final level of G.

[a] Two arrays $(\widehat{A}_{\widehat{\mu},j}, \widehat{A}_{\widetilde{\mu},j})$ are concatenated if $(1+\log N) \nmid l$; otherwise c such arrays are concatenated; see e.g. (14) in Figure 6.

Fig. 7: Per-Experiment Protocol Π_{EPR_m} for Anonymous Permutation Routing.

7 Acknowledgements

This material is based upon work supported by the United States Air Force and DARPA – Distribution Statement "A" (Approved for Public Release, Distribution Unlimited) – under Contract No. FA8750-19-C-0031, DARPA under Cooperative Agreement HR0011-20-2-0025, the Algorand Centers of Excellence program managed by Algorand Foundation, NSF grants CNS-224635, CCF-2220450, CNS-2001096, US-Israel BSF grant 2018393, ISF grant 2774/20, Amazon Faculty Award, Cisco Research Award and Sunday Group. Any views, opinions, findings, conclusions or recommendations contained herein are those of the author(s) and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, the Department of Defense, the United States Air Force, the Algorand Foundation, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes not withstanding any copyright annotation therein.

References

AKS83. Miklós Ajtai, János Komlós, and Endre Szemerédi. An o(n log n) sorting network. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25-27 April, 1983*, pages 1–9. ACM, 1983.

CGH$^+$21. Melissa Chase, Sanjam Garg, Mohammad Hajiabadi, Jialin Li, and Peihan Miao. Amortizing rate-1 OT and applications to PIR and PSI. In *Theory of Cryptography - 19th Intl. Conference*, pages 126–156. Springer, 2021.

CGKS95. Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. Private information retrieval. In *36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, USA, 23-25 October 1995*, pages 41–50. IEEE Computer Society, 1995.

COS10. Nishanth Chandran, Rafail Ostrovsky, and William E. Skeith. Public-key encryption with efficient amortized updates. In *Security and Cryptography for Networks, 7th Intl. Conference*, pages 17–35. Springer, 2010.

DGI$^+$19. Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. In *Advances in Cryptology - CRYPTO 2019 - 39th Annual Intl. Cryptology Conference*, pages 3–32. Springer, 2019.

FSSV22. Rex Fernando, Elaine Shi, Pratik Soni, and Nikhil Vanjani. Non-interactive anonymous router with quasi-linear router computation. *IACR Cryptol. ePrint Arch., Paper 1395*, 2022.

GHO20. Sanjam Garg, Mohammad Hajiabadi, and Rafail Ostrovsky. Efficient range-trapdoor functions and applications: Rate-1 OT and more. In *Theory of Cryptography - 18th Intl. Conf.*, pages 88–116. Springer, 2020.

HOWW19. Ariel Hamlin, Rafail Ostrovsky, Mor Weiss, and Daniel Wichs. Private anonymous data access. In *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual Intl. Conference on the Theory and Applications of Cryptographic Techniques*, pages 244–273. Springer, 2019.

IKOS04. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Batch codes and their applications. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 262–271. ACM, 2004.

IP07. Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In *Theory of Cryptography, 4th Theory of Cryptography Conference*, pages 575–594. Springer, 2007.

KO97. Eyal Kushilevitz and Rafail Ostrovsky. Replication is NOT needed: SINGLE database, computationally-private information retrieval. In *38th Annual Symposium on Foundations of Computer Science, FOCS '97, October 19-22, 1997*, pages 364–373. IEEE Computer Society, 1997.

Lei84. Frank Thomson Leighton. Tight bounds on the complexity of parallel sorting. In *Proceedings of the 16th Annual ACM Symposium on Theory of Computing*, pages 71–80. ACM, 1984.

LMW22. Wei-Kai Lin, Ethan Mook, and Daniel Wichs. Doubly efficient private information retrieval and fully homomorphic RAM computation from ring LWE. *IACR Cryptol. ePrint Arch., Paper 1703*, 2022.

MS92. Bruce M. Maggs and Ramesh K. Sitaraman. Simple algorithms for routing on butterfly networks with bounded queues (ext. abstract). In *24th Annual ACM Symposium on Theory of Computing*, pages 150–161. ACM, 1992.

OS07. Rafail Ostrovsky and William E. Skeith. A survey of single-database private information retrieval: Techniques and applications. In *Public Key Cryptography - PKC 2007, 10th Intl. Conference on Practice and Theory in Public-Key Cryptography*, pages 393–411. Springer, 2007.

SW21. Elaine Shi and Ke Wu. Non-interactive anonymous router. In *Advances in Cryptology - EUROCRYPT 40th Annual Intl. Conf. on the Theory and Applications of Cryptographic Techniques*, pages 489–520. Springer, 2021.

Upf89. Eli Upfal. An o(log N) deterministic packet routing scheme (preliminary version). In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 241–250. ACM, 1989.

Supplementary Material

A Supplementary Figures

A.1 Butterfly and Beneš Networks

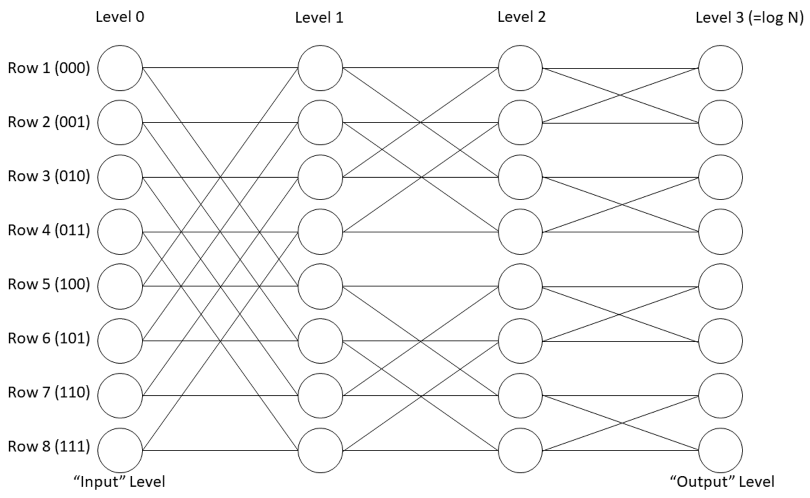

Fig. 8: Butterfly network with $N = 8$ input nodes.

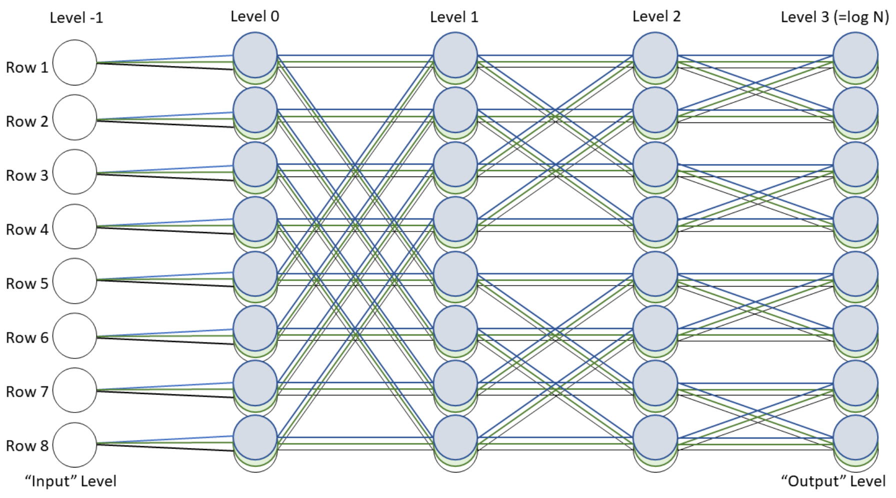

Fig. 9: Colored Butterfly network with N=8 input nodes and replication factor c=3.

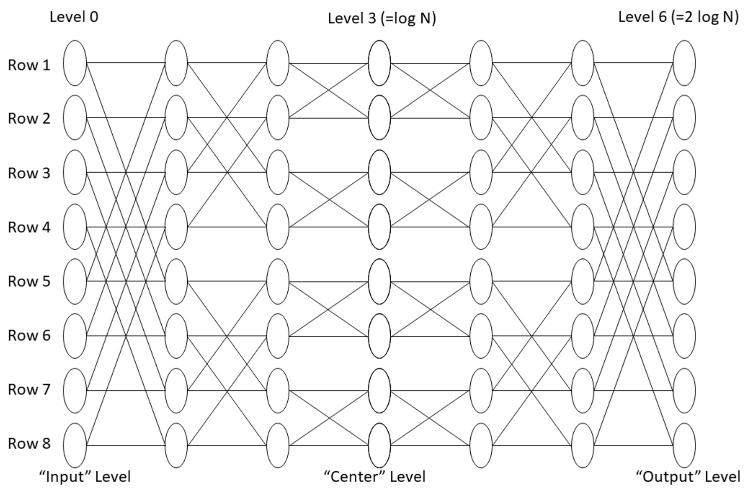

Fig. 10: Beneš network with $N = 8$ input nodes.

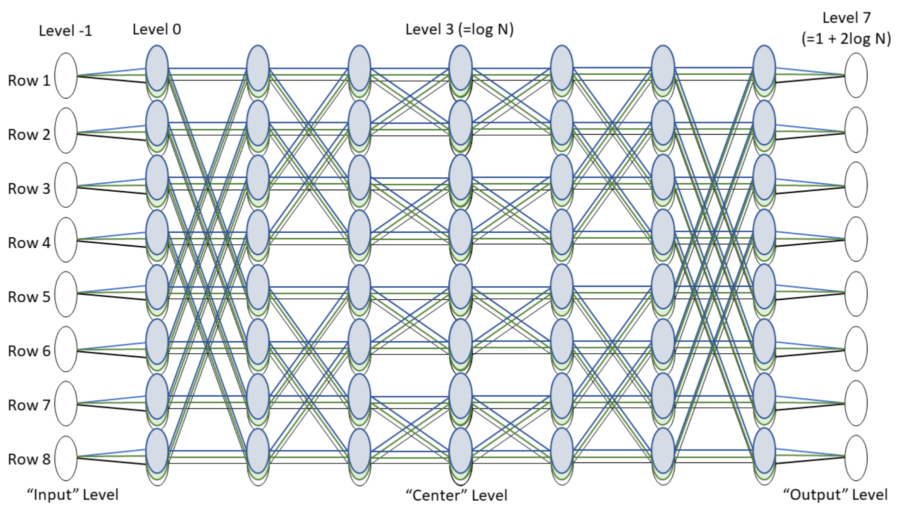

Fig. 11: Colored Beneš network with $N{=}8$ input nodes and replication factor $c{=}3$.

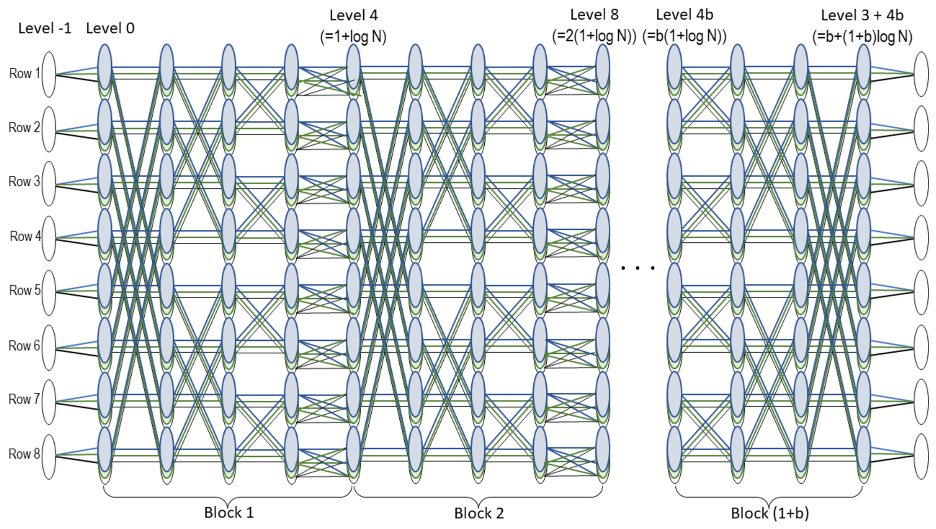

Fig. 12: Extended, Colored Beneš network with b blocks, replication factor $c = 3$, and $N = 8$.

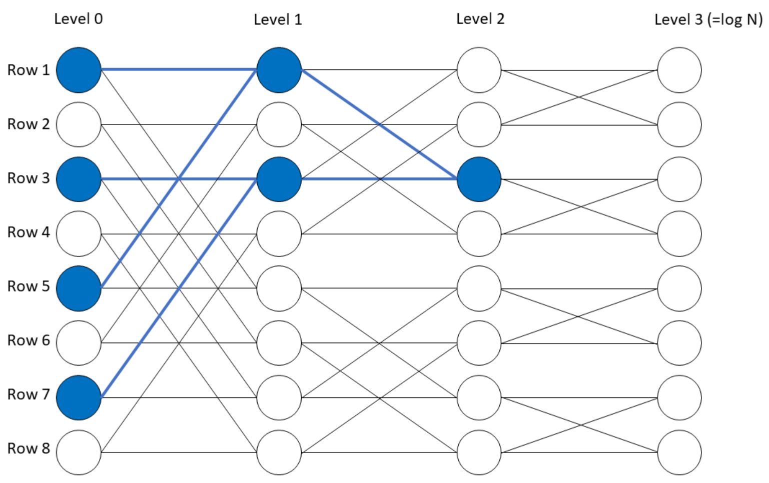

Fig. 13: Ancestors of a node on some level of a Butterfly network with $N = 8$.

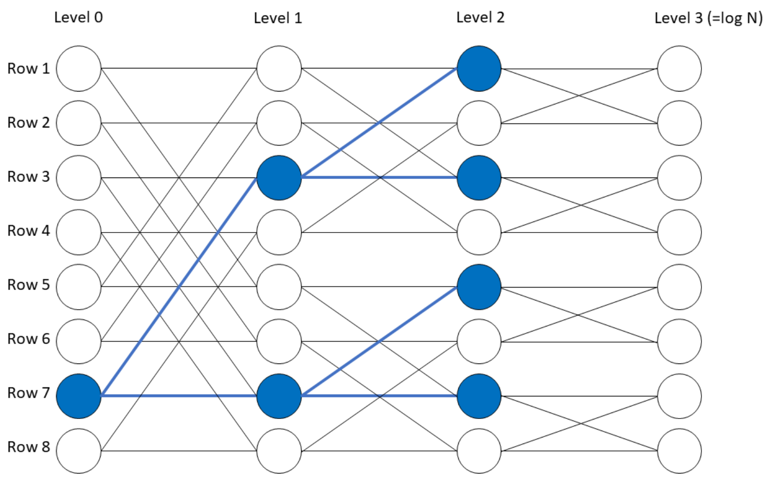

Fig. 14: Descendants of a node on some level of a Butterfly network with $N = 8$.

B Proofs

Proof (Proof of Lemma 3). Fix an arbitrary node μ_l at level $l \in [0, \log N]$. Notice that a random path $\mathcal{P}_i$ originating at input node ν_i (for $i \in [N]$) can pass through μ_l if and only if ν_i is one of the 2^l *ancestors* of μ_l (see Figure 13); and in this case, the probability that a random path $\mathcal{P}_i$ originating at ν_i passes through μ_l is exactly $1/2^l$ (see Figure 14). Since each random path is chosen independently from one another, the random variable X_{μ_l} is exactly described by the probability mass function of the binomial distribution, for $M_l := 2^l$ experiments (corresponding to the M_l ancestors of node μ_l) and probability of success $p = 1/M_l$. Furthermore, since the expected value of X_{μ_l} is $M_l \cdot p = 1$, the probability is maximized when $X_{\mu_l} = 1$. Consequently, for any $k \geq 1$:

$$\begin{aligned}
\Pr[X_\mu \geq k] &= \sum_{i=k}^{M_l} \binom{M_l}{i} \cdot p^i \cdot (1-p)^{M_l - i} \\
&\leq (M_l - k + 1) \cdot \binom{M_l}{k} \cdot p^k \cdot (1-p)^{M_l - k} \\
&\leq (M_l - k + 1) \cdot \frac{M_l \cdot (M_l - 1) \cdot \dots \cdot (M_l - k + 1)}{k!} \cdot p^k \\
&\leq \frac{M_l}{k!} \cdot M_l^k \cdot p^k = \frac{M_l}{k!} = \frac{2^l}{k!}
\end{aligned}$$

Proof (Proof of Lemma 4). The argument is analogous to the proof of Lemma 3, with each node $\mu_l = \mu_{\widehat{c},r,l}$ (at level $l \in [0, \log N]$, row $r \in [N]$, and color $\widehat{c} \in [c]$) still having $M_l = 2^l$ ancestor nodes (now at level -1), and with each ancestor having probability $p = 1/(c \cdot M_l)$ of passing through node μ_l (e.g. one of c edges are chosen from level -1 to level 0 that determines the "color"). Therefore, the analysis used in the proof of Lemma 3 can be followed exactly, plugging in $p = 1/(c \cdot M_l)$ in the last step.

Proof (Proof of Lemma 5). Since $k \geq 1$, there are clearly no nodes on levels -1 or 0 that have more than one path pass through it. The corollary therefore follows immediately from (3), by applying a union bound over all of the $c \cdot N \log N$ nodes ($c \cdot N$ nodes on each of the levels $l \in [1, \log N]$):

$$\Pr[X_k = 1] \leq (c \cdot N) \cdot \sum_{l=1}^{\log N} \frac{2^l}{k! \cdot c^k} = \left(\frac{c \cdot N}{k! \cdot c^k} \right) \cdot \sum_{l=1}^{\log N} 2^l \leq \frac{2c \cdot N^2}{k! \cdot c^k}$$

Proof (Proof of Lemma 7). For $j = 1$, (5) follows immediately from Lemma 5. Similarly for $j = 1 + b$, since σ is a permutation, then the setup (for continuing each path $\{\mathcal{P}_i\}_{i=1}^N$ from level $(b \cdot \log N)$ through the final output level $1 + (1+b) \cdot \log N$) is identical (up to reflection/mirror image) to the hypotheses of Lemma 5. Thus, it remains to demonstrate (5) for each $j \in [2,\ b]$.

The argument for all such blocks j is simply a counting argument: since each path has an equal probability of starting at any of the $(c \cdot N)$ nodes on the "input" level of block j (namely, on level $(j-1)\cdot(1+\log N)$), and from there a uniformly random path is chosen, we have that for any level $l \in [(j-1)\cdot(1+\log N),\ j\cdot(1+\log N)-1]$ contained in the j^{th} block, each path has an equal probability of passing through each of the $(c\cdot N)$ nodes on that level. For any node μ_j in the j^{th} block, let X_{μ_j} denote the random variable that counts the number of paths passing through μ_j. Then X_{μ_j} is exactly described by the probability mass function of the binomial distribution, for N experiments (corresponding to the N paths $\{\mathcal{P}_i\}$) and probability of success $p = 1/(c\cdot N)$. Furthermore, since the expected number of paths passing through μ_j is $N\cdot p = 1/c$, the probability is maximized when $X_{\mu_j} = 1/c$. Consequently, for any $k \geq 1$:

$$\begin{aligned}
\Pr[X_{\mu_j} \geq k] &\leq (N-k)\cdot \Pr[X_{\mu_j} = k] \\
&\leq (N-k)\cdot \binom{N}{k} \cdot p^k \cdot (1-p)^{N-k} \\
&\leq (N-k)\cdot \frac{N\cdot(N-1)\cdot\dots\cdot(N-k+1)}{k!}\cdot p^k \\
&\leq \frac{N\cdot(N-1)\cdot\dots\cdot(N-k)}{k!}\cdot\left(\frac{1}{cN}\right)^k \leq \frac{N}{k!\cdot c^k} \qquad (16)
\end{aligned}$$

Thus, applying a union bound on the $cN(1+\log N)$ nodes in block j:

$$\Pr[X_{j,k}] \leq (cN(1+\log N))\cdot \Pr[X_{\mu_j} \geq k] \leq \frac{c\cdot N^2\cdot(1+\log N)}{k!\cdot c^k}$$

Proof (Proof of Lemma 12). Fix any $i \in [N]$, $j \in [(1+b)]$, and $k \in [N]$. Let $\mathcal{P}_i$ denote the path selected by Π_{σ_N} connecting $(I_i, O_{\sigma(i)})$ for a given run of the algorithm in Figure 5. Let e_l be an edge in $\mathcal{P}_i$ that is in block j; that is, e_l connects levels $(l, l+1)$ for some level $l \in [(j-1)\cdot(1+\log N),\ (j\cdot(1+\log N)-1)]$. We first argue that the probability that e_l is **not** edge-disjoint (that is, that some other path $\mathcal{P}_{i'}$ specified by Π_{σ_N} also contains e_l, for $i' \neq i$) is bounded by:

$$\Pr[e_l \in \mathcal{P}_{i'} \text{ for } i' \neq i] \leq \frac{c\cdot N^2(1+\log N)}{k!\cdot c^k} + \frac{k}{2w} \qquad (17)$$

To prove (17), let μ_l denote the node on level l that lies on $\mathcal{P}_i$ (so the left-hand node of edge e_l), and let Z_{k,μ_l} denote the boolean random variable indicating whether μ_l has more than k total paths that pass through it (as specified by the given run of Π_{σ_N}). Consider:

$$\begin{aligned}
\Pr[e_l \in \mathcal{P}_{i'} \text{ for } i' \neq i] &\leq \Pr[e_l \in \mathcal{P}_{i'} \text{ for } i' \neq i \mid Z_{k,\mu_l} = 1]\cdot \Pr[Z_{k,\mu_l} = 1] + \\
&\quad \Pr[e_l \in \mathcal{P}_{i'} \text{ for } i' \neq i \mid Z_{k,\mu_l} = 0]\cdot \Pr[Z_{k,\mu_l} = 0] \\
&\leq \Pr[Z_{k,\mu_l} = 1] + \Pr[e_l \in \mathcal{P}_{i'} \text{ for } i' \neq i \mid Z_{k,\mu_l} = 0] \\
&\leq \frac{c\cdot N^2(1+\log N)}{k!\cdot c^k} + \frac{k}{2w} \qquad (18)
\end{aligned}$$

where the first term of (18) comes from (5), and the second comes from applying a union bound: since there are at most k other paths that pass through μ_l (since

$Z_{k,\mu_l} = 0$ for this term), the probability that any of these (at most k) paths also traverses edge e_l is bounded by $1/2w$ (since there are the two[14] choices of "up/down," and (expansion factor) w choices for which duplicate of each edge.

Using (17), we conclude the proof by applying a union bound on the $(1+\log N)$ levels in block j.

Proof (Proof of Lemma 14). This follows immediately from Lemma 12 by applying a union bound on the $(1 + b)$ blocks of the Beneš network $B(N, b, c, w)$.

Proof (Proof of Corollary 16). First, we examine the RHS of (7) with the parameter values as specified and with $k = \log N$:

$$\begin{aligned}(1+b)\cdot(1+\log N)\cdot\left(\frac{cN^2\cdot(1+\log N)}{k!\cdot c^k}+\frac{k}{2w}\right) &= \lambda\cdot\left(\frac{4a_\lambda N^2\cdot(1+\log N)^2}{(\log N)!\cdot a_\lambda^{\log N}\cdot 4^{\log N}}+\frac{\log N}{96\lambda\log N}\right)\\ &= \lambda\cdot\left(\frac{4\cdot(1+\log N)^2}{(\log N)!\cdot a_\lambda^{\log N-1}}+\frac{1}{2.4\lambda}\right)\\ &< \lambda\cdot\left(\frac{1}{3\lambda}+\frac{5}{12\lambda}\right) \;=\; \frac{3}{4} \qquad (19)\end{aligned}$$

where we have used in the inequality that $\frac{4\cdot(1+\log N)^2}{(\log N)!} < 1/3$ (which holds for any $N \geq 64$) and that $a_\lambda^{\log N-1} \geq \lambda$ (which is immediate by definition of a_λ). Consider:

$$\begin{aligned}\Pr[X=0] \;&\leq\; \Pr[\forall\, m \in [M] :\; X_{\Pi_m}(i) = 0]\\ &\leq\; \left((1+b)\cdot(1+\log N)\cdot\left(\frac{c\cdot N^2\cdot(1+\log N)}{k!\cdot c^k}+\frac{k}{2w}\right)\right)^M\end{aligned}$$

where the inequality is from Lemma 14. We then use (19) and substituting $M = \lambda$ to conclude the proof.

Proof (Proof of Lemma 19). We can use a similar argument as was done for Lemma 12 to bound the probability that any one of the paths $\{\mathcal{P}_i, \mathcal{P}_{i'}, \mathcal{P'}_i, \mathcal{P'}_{i'}\}$ is *not* edge-disjoint. Specifically, observe that:

- $\forall \widehat{i} \notin \{i, i'\}$: $\mathcal{P}_{\widehat{i}} = \mathcal{P}'_{\widehat{i}}$ (by construction of Π'), and hence any of the four paths in $\{\mathcal{P}_i, \mathcal{P}_{i'}, \mathcal{P}'_i, \mathcal{P}'_{i'}\}$ is edge-disjoint from $\mathcal{P}'_{\widehat{i}}$ if and only if it is edge-disjoint from $\mathcal{P}_{\widehat{i}}$.
- For any of the four paths in $\{\mathcal{P}_i, \mathcal{P}_{i'}, \mathcal{P'}_i, \mathcal{P'}_{i'}\}$, the setup is identical to the setup in the hypotheses of Lemma 12, *except* that there are now $N + 2$ total paths through block j (instead of just N paths). Namely, $N-2$ of the N paths specified by $\mathcal{P}'$ exactly overlap with the corresponding $N - 2$ paths specified by $\mathcal{P}$, plus the (up to) four distinct paths: $\{\mathcal{P}_i, \mathcal{P}_{i'}, \mathcal{P'}_i, \mathcal{P'}_{i'}\}$.

[14] In the case that e_l traverses the final level of the block, then instead of two there are c choices for edge (corresponding to each edge leading to each color node), and by assumption $c \geq 2$.

However, notice that Definition 17 does not require the four paths $\{\mathcal{P}_i, \mathcal{P}_{i'}, \mathcal{P}'_i, \mathcal{P}'_{i'}\}$ to be edge-disjoint from each other. Therefore, when counting the number of paths that pass through a given node (see e.g. the analysis leading to (16)), we can ignore these four paths. More specifically, the bounds from Lemma 7 (and more specifically, in (5)) can be used as-is (they still provide an upper-bound, although they could be slightly tightened to leverage that there are only $N-2$ relevant paths instead of the N paths that were used in the analysis leading to (16)) when bounding $\Pr[Z_{k,\mu_l} = 1]$, as per the analysis in the proof of Lemma 12 (see (18)). Consequently, the analysis used in the proof of Lemma 12 remains valid, and hence, for any of the four paths, the probability that the path is edge-disjoint (from the $N-2$) other paths in $\{\mathcal{P}_{\widehat{i}}\}_{\widehat{i} \neq i,i'}$ on block j is bounded by $(1+\log N) \cdot \left(\frac{c \cdot N^2 \cdot (1+\log N)}{k! \cdot c^k} + \frac{k}{2w}\right)$. We conclude the proof by applying a union bound on the individual probabilities, for the four paths $\{\mathcal{P}_i, \mathcal{P}_{i'}, \mathcal{P}'{}_i, \mathcal{P}'{}_{i'}\}$.

Proof (Proof of Lemma 22). For any index $\widehat{j} \in [1,(1+b)]$, let $\Pi_{\widehat{j}}$ denote the "block j alternate routing algorithm" as per Definition 18, with corresponding random variable $Y_{\Pi,\Pi'_{\widehat{j}}}(i,i',\widehat{j})$ (as per Def. 17). Then by Lemma 19, the probability that $Y_{\Pi,\Pi'_{\widehat{j}}}(i,i',\widehat{j}) = 0$ is bounded by $4 \cdot (1+\log N) \cdot \left(\frac{c \cdot N^2 \cdot (1+\log N)}{k! \cdot c^k} + \frac{k}{2w}\right)$. Therefore, the probability that $Y_{\Pi,\Pi'_j}(i,i',j) = 0$ for *all* block indices $j \in [1,(1+b)]$ is bounded by $\left(4 \cdot (1+\log N) \cdot \left(\frac{c \cdot N^2 \cdot (1+\log N)}{k! \cdot c^k} + \frac{k}{2w}\right)\right)^{(1+b)}$. The lemma therefore follows from the observation that $Y_{\Pi,\Pi'}(i,i') = 1$ if and only if Π' is defined as per Step (2a) in Definition 21; i.e. if there exists $j \in [1,(1+b)]$ such that $Y_{\Pi,\Pi'_j}(i,i',j) = 1$.

Proof (Proof of Corollary 24). First, notice that with the parameters as in the hypothesis and with $k = \log N$, the inner-term (everything except the exponent) on the RHS of (10) becomes:

$$\begin{aligned}
4\cdot(1+\log N)\cdot\left(\frac{cN^2\cdot(1+\log N)}{k!\cdot c^k}+\frac{k}{2w}\right) &= 4\cdot\left(\frac{4a_\lambda N^2\cdot(1+\log N)^2}{(\log N)!\cdot a_\lambda^{\log N}\cdot 4^{\log N}}+\frac{\log N}{2.4\lambda\log N}\right)\\
&= 4\cdot\left(\frac{4\cdot(1+\log N)^2}{(\log N)!\cdot a_\lambda^{\log N-1}}+\frac{5}{12\lambda}\right)\\
&< 4\cdot\left(\frac{1}{3\cdot N/2}+\frac{5}{96}\right)\\
&\le 4\cdot\left(\frac{1}{96}+\frac{5}{96}\right)\\
&= \frac{1}{4} \qquad (20)
\end{aligned}$$

where we have used in the first inequality that $\frac{4\cdot(1+\log N)^2}{(\log N)!} < 1/3$ (which holds for any $N \geq 64$), that $a_\lambda^{\log N-1} \geq N/2$ (which is immediate from the definition of

a_λ), and that $\lambda \geq 8$ (by hypothesis); and in the second inequality that $N \geq 64$. Consider:

$$\begin{aligned}
\Pr[Y=0] &= \Pr[\exists (m,i,i') \in [M] \times [N] \times [N] : Y_{\Pi_m,\Pi'_m}(i,i') = 0] \\
&\leq M \cdot N^2 \cdot \left(4 \cdot (1+\log N) \cdot \left(\frac{c \cdot N^2(1+\log N)}{k! \cdot c^k} + \frac{k}{2w}\right)\right)^{(1+b)} \\
&\leq \lambda \cdot N^2 \cdot \left(\frac{1}{4}\right)^{\lambda}
\end{aligned}$$

where the first inequality is from a union bound combined with Lemma 22; and the second inequality is from (20) and substituting $M = \lambda$ and $b = \lambda - 1$.

Proof (Proof of Lemma 25). By Corollary 24, the probability that Adversary $\mathcal{A}$ wins in Step 3 is bounded by $\frac{1}{2 \cdot 2^\lambda}$. Thus, we need only show that the probability that Adversary $\mathcal{A}$ wins in Step 6 (i.e. that $\mathcal{A}$ correctly chooses between σ and τ) equals 1/2. This in turn follows immediately from the definition of $Y_{\Pi,\Pi'}(i,i',j)$ (Definition 17): Since Adversary $\mathcal{A}$ was given an *unordered* set $\{\mathcal{P}_{m,i,j_m}, \mathcal{P}_{m,i',j_m}, \mathcal{P}'_{m,i,j_m}, \mathcal{P}'_{m,i',j_m}\}$ in Step 5d, there is no way for the Adversary $\mathcal{A}$ to distinguish the actual path $\mathcal{P}_{m,i,j_m}$ (as per Π_m) from its alternate path $\mathcal{P}'_{m,i,j_m}$ (as per Π'_m); and ditto for $\mathcal{P}_{m,i',j_m}$ and its alternate path $\mathcal{P}'_{m,i',j_m}$. Namely, if we let $\mu_{m,i}, \mu_{m,i'}$ denote the two (not necessarily distinct) "block j_m *input* level" nodes (i.e. the nodes at level $(j_m - 1) \cdot (1 + \log N)$) that $\mathcal{P}_{m,i}$ and $\mathcal{P}_{m,i'}$ pass through, and similarly let $\nu_{m,i}, \nu_{m,i'}$ denote the two (not necessarily distinct) "block j_m *output* level" nodes (i.e. the nodes at level $(j_m \cdot (1 + \log N))$) that $\mathcal{P}_{m,i}$ and $\mathcal{P}_{m,i'}$ pass through, then notice:

– Within block j_m: $\mathcal{P}_{m,i,j_m}$ starts at $\mu_{m,i}$ and ends at $\nu_{m,i}$.
– Within block j_m: $\mathcal{P}_{m,i',j_m}$ starts at $\mu_{m,i'}$ and ends at $\nu_{m,i'}$.
– Within block j_m: $\mathcal{P}'_{m,i,j_m}$ starts at $\mu_{m,i}$ and ends at $\nu_{m,i'}$.
– Within block j_m: $\mathcal{P}'_{m,i',j_m}$ starts at $\mu_{m,i'}$ and ends at $\nu_{m,i}$.

Also, define the two sets of paths: "primary" paths $\mathcal{P}_{m,0} := \{\mathcal{P}_{m,i,j_m},\ \mathcal{P}_{m,i',j_m}\}$ and "alternate" paths $\mathcal{P}_{m,1} := \{\mathcal{P}'_{m,i,j_m},\ \mathcal{P}'_{m,i',j_m}\}$. Then notice that the Adversary $\mathcal{A}$ can guess the correct permutation σ versus τ in Step 6 if and only if it can distinguish between $\mathcal{P}_{m,0}$ versus $\mathcal{P}_{m,1}$, in terms of which correspond to the "primary" paths and which is the "alternate" paths. Thus, it is sufficient to show that this probability is 1/2.

Consider the following facts:

F1.a The probability that Challenger $\mathcal{C}$ chose $\mathcal{P}_{m,i,j_m}$ (and hence $\mathcal{P}'_{m,i,j_m}$ was the "alternate") equals the probability that Challenger $\mathcal{C}$ instead chose $\mathcal{P}'_{m,i,j_m}$ (and hence $\mathcal{P}_{m,i,j_m}$ would have been the "alternate").

F1.b The probability that Challenger $\mathcal{C}$ chose $\mathcal{P}_{m,i',j_m}$ (and hence $\mathcal{P}'_{m,i',j_m}$ was the "alternate") equals the probability that Challenger $\mathcal{C}$ instead chose $\mathcal{P}'_{m,i',j_m}$ (and hence $\mathcal{P}_{m,i',j_m}$ would have been the "alternate").

F2.a Amongst the two paths in $\mathcal{P}_{m,0}$: One starts at $\mu_{m,i}$ and the other starts at $\mu_{m,i'}$; and one ends at $\nu_{m,i}$ and the other ends at $\nu_{m,i'}$.

F2.b Amongst the two paths in $\mathcal{P}_{m,1}$: One starts at $\mu_{m,i}$ and the other starts at $\mu_{m,i'}$; and one ends at $\nu_{m,i}$ and the other ends at $\nu_{m,i'}$.

F3.a Both paths in $\mathcal{P}_{m,0}$ are edge-disjoint (on block j_m) from all other paths $\{\mathcal{P}_{m,\widehat{i}}\}_{\widehat{i} \notin \{i,i'\}}$.

F3.b Both paths in $\mathcal{P}_{m,1}$ are edge-disjoint (on block j_m) from all other paths $\{\mathcal{P}_{m,\widehat{i}}\}_{\widehat{i} \notin \{i,i'\}}$.

Given the above facts, there is no way for (even an unbounded) Adversary $\mathcal{A}$ to distinguish whether the two paths in $\mathcal{P}_{m,0}$ are the "primary" paths (selected by Π_m) with $\mathcal{P}_{m,1}$ as the "alternate" paths (selected by Π'_m), or vice-versa, since:

- A priori, the two paths in $\mathcal{P}_{m,0}$ are equally likely to be chosen by Π_m as the "primary" paths as the two paths in $\mathcal{P}_{m,1}$ (Fact F1).
- The information of block index j_m (dealt in Step 5a) is no help in distinguishing,
- The information of non-interesting paths (dealt in Step 5b) is no help in distinguishing, since $\mathcal{P}_{m,0}$ and $\mathcal{P}_{m,1}$ are similarly edge-disjoint (on block j_m) from all other paths $\{\mathcal{P}_{m,\widehat{i}}\}_{\widehat{i} \notin \{i,i'\}}$ (Fact F3).
- The information of paths $\mathcal{P}_{m,i,j_m}$ and $\mathcal{P}_{m,i',j_m}$ *before* block j_m (dealt in Step 5c) is no help in distinguishing, since both $\mathcal{P}_{m,0}$ and $\mathcal{P}_{m,1}$ have one path that starts at $\mu_{m,i}$ and one that starts at $\mu_{m,i'}$ (Fact F2).
- The information dealt in Step 5d is no help in distinguishing, since the provided list of paths $\{\mathcal{P}_{m,i,j_m}, \mathcal{P}_{m,i',j_m}, \mathcal{P}'_{m,i,j_m}, \mathcal{P}'_{m,i',j_m}\}$ is *unordered.*
- The (*unordered*) information of which edges lie in $\mathcal{P}_{m,i,j_m}$ and $\mathcal{P}_{m,i',j_m}$ *beyond* block j_m (dealt in Step 5e) is no help in distinguishing, since both $\mathcal{P}_{m,0}$ and $\mathcal{P}_{m,1}$ have one path that ends at $\nu_{m,i}$ and one that ends at $\nu_{m,i'}$ (Fact F2).

Therefore, none of the information provided by the Challenger $\mathcal{C}$ (nor any other a priori bias) provides any information to Adversary $\mathcal{A}$ that allows it to distinguish between $\mathcal{P}_{m,0}$ and $\mathcal{P}_{m,1}$ (in terms of which are the "primary" paths and which are the "alternate" paths). Consequently, the probability that Adversary $\mathcal{A}$ can distinguish between $\mathcal{P}_{m,0}$ and $\mathcal{P}_{m,1}$ (in terms of which are the "primary" paths and which are the "alternate" paths) is 1/2.

Proof (Proof of Lemma 26). (Proof by contradiction.)
Suppose there exists a (computationally unbounded) adversary $\mathcal{A}$ that can win the security challenge game with probability greater than $\frac{1}{2} + \frac{1}{2^{\lambda_s}}$. We will show this leads to a contradiction.

Reduction. Without loss of generality, we may assume that σ_0 and σ_1 differ in exactly two indices $i \neq i'$, so that $\sigma_0(i) = \sigma_1(i')$, and $\sigma_0(i') = \sigma_1(i)$, and for all other indices $j \neq i, i'$, $\sigma_0(j) = \sigma_1(j)$ (this reduction gives the adversary an extra factor of N advantage).

Proof (Proof of Reduction.). Consider the following chain of permutations:

$$(\sigma_0 =)\tau_0,\ \tau_1,\ \tau_2,\ \ldots,\ \tau_N(= \sigma_1), \tag{21}$$

where each τ_{i+1} is defined iteratively from τ_i, starting with $\tau_0 = \sigma_0$. Then given τ_i, define τ_{i+1} to be the permutation that matches τ_i at all indices *except* (possibly) in positions i and $j \geq i$, where $j := \sigma_0^{-1}(\sigma_1(i))$. Namely, $\tau_{i+1}(i) := \sigma_1(i)$, and $\tau_{i+1}(j) := \sigma_0(i)\ (= \tau_i(i))$. Then it is easy to demonstrate the for any $1 \leq i \leq N$, permutations τ_{i-1} and τ_i differ in (at most) two places.

We now apply a standard hybrid argument, to show that the existence of an Adversary who wins the security challenge game with σ_0, σ_1 implies the existence of an Adversary $\mathcal{A}'$ who can distinguish between τ_i and τ_{i+1} (for some $i \in [0, N]$), with advantage:

$$\begin{aligned} \Pr[\mathcal{A} \text{ outputs } b \text{ correctly}] &> \frac{1}{2} + \frac{1}{2^{\lambda_s}} \quad \Rightarrow \\ \Pr[\mathcal{A}' \text{ distinguishes between } \tau_i \text{ and } \tau_{i+1}] &> \frac{1}{2} + \frac{1}{N \cdot 2^{\lambda_s}} = \frac{1}{2} + \frac{1}{2^{\lambda}} \end{aligned} \tag{22}$$

The proof by contradiction is now complete, as the existence of adversary $\mathcal{A}'$ with advantage as per (22) violates (13), since for $\lambda = 2\log N + \max(\lambda_s, 2 + \log\log N)$, we have:

$$\frac{\lambda \cdot N^2}{4^{\lambda}} = \frac{1}{2^{\lambda}} \cdot \frac{N^2}{2^{2\log N}} \cdot \frac{\lambda}{2^{\alpha}} \leq \frac{1}{2^{\lambda}}$$

where $\alpha := \max(\lambda_s, 2 + \log\log N)$, and we have used that $2^{\alpha} \geq \lambda$ since:

If $\lambda_s \geq 2\log N$: Then $\lambda \leq 2\lambda_s \Rightarrow \log\lambda \leq \lambda_s \leq \alpha$.
If $\lambda_s \leq 2\log N$: Then $\lambda \leq 4\log N \Rightarrow \log\lambda \leq 2 + \log\log N \leq \alpha$.