1 Generating Couplings from DiPA

Consider a well-formed DiPA $\mathcal{A} = (Q, \Sigma, \Gamma, q_{init}, \{\texttt{insample}, \texttt{insample}', \texttt{x}\}, P, \delta)$. For all adjacent inputs $X \sim X'$ and each output $\sigma \in \texttt{range}(\mathcal{A})$, we want to create a coupling $\mathcal{A}(X)\{(a,b) : a = \sigma \implies b = \sigma\}^{\#(\varepsilon_\sigma,0)}\mathcal{A}(X')$. If such a coupling exists for each σ and $X \sim X'$, then we have proved that the program corresponding to $\mathcal{A}$ is $(\max_{\sigma\in\texttt{range}(\mathcal{A})} \varepsilon_\sigma, 0)$-differentially private.

1.1 An algorithmic process for generating coupling proofs

1.1.1 Overview

In order to show that $\mathcal{A}$ is differentially private, we must compare, for every pair of neighbouring datasets $X \sim X'$ and for each possible output σ of $\mathcal{A}$, the probability that $\mathcal{A}(X) = \sigma$ and that $\mathcal{A}(X') = \sigma$. As previously mentioned, in order to show that $\mathcal{A}$ is ε-DP, it is sufficient to show for each σ that the lifting $\mathcal{A}(X)\{(a,b) : a = \sigma \implies b = \sigma\}^{\#(\varepsilon_\sigma,0)}\mathcal{A}(X')$ exists such that $\varepsilon = \max_\sigma\{\varepsilon_\sigma\}$.

We assume that every state of $\mathcal{A}$ reads in input for convenience [1].

Because of output distinction and determinism, each output σ uniquely determines a path ρ_σ in $\mathcal{A}$. After fixing some $X \sim X'$ and $\sigma \in \texttt{range}(\mathcal{A})$, our high level procedure thus proceeds as follows: split ρ_σ at each assignment transition so that we get a bounded number of segments of ρ_σ. For each segment, the first transition in the segment is an assignment transition and the remaining transitions are non-assignment transitions. We can then choose between three "coupling strategies" for the segment - in essence, we choose to assign zero cost to either $\texttt{insample} < \texttt{x}$ or $\texttt{insample} \geq \texttt{x}$ transitions, or neither (to accomodate certain output patterns). Because of the initialization condition, we know that the first transition of any path will be an assignment transition, so these segments truly partition the path.

After creating coupling proofs for each segment in isolation, we can recombine segments together in sequence. However, at each assignment transition, because we must couple the Laplace noise added to both satisfy the guard of the assignment transition (dependent on the *previous* value of x) and also couple the same noise to influence the *new* value of x, there are certain constraints on which coupling strategy can be chosen for a segment that depend on the strategy selected for the previous segment and the guard of the assignment transition (see section 1.4 for details).

We claim that, if a DiPA $\mathcal{A}$ is well-formed, then there exists some upper cost bound ε such that for $X \sim X'$ and for *all* outputs $\sigma \in \texttt{range}(\mathcal{A})$, we can construct the lifting $\mathcal{A}(X)\{(a,b) : a = \sigma \implies b = \sigma\}^{(\varepsilon,0)}A(X')$.

In particular, if a DiPA does not have a leaking cycle, then there exists a bound on the number of segments for any path through $\mathcal{A}$, and if a DiPA does not have a leaking pair,

[1] Note that a state that doesn't read in input can be simulated by supplying an input of 0 at that state and that states that don't read in input cannot impact the control flow of the automaton because of the non-input transition condition

a disclosing cycle, or a privacy violating path, then we can claim that there exists a global bound on the cost of a minimal coupling strategy for any path through $\mathcal{A}$.

Because there can be a very large or even potentially unbounded number of paths through $\mathcal{A}$, note that it is not actually practical to individually minimize each path.

1.1.2 Details

As mentioned, we suppose that all states in $\mathcal{A}$ are input states.

Preliminaries

Fix $X \sim X'$. We will analyze the relative behaviour of the two runs $\mathcal{A}(X)$ and $\mathcal{A}(X')$. In particular, for each $\sigma \in \texttt{range}(\mathcal{A})$, we want to construct the lifting

$$\mathcal{A}(X)\{(a,b) : a = \sigma \implies b = \sigma\}^{(\varepsilon,0)}\mathcal{A}(X')$$

Because each transition in $\mathcal{A}$ must output, this is equivalent to, for each $\sigma \in \texttt{range}(\mathcal{A})$, constructing a series of liftings

$$\mathcal{A}(X)_i\{(a,b) : a = \sigma_i \implies b = \sigma_i\}^{(\varepsilon_i,0)}\mathcal{A}(X')_i$$

for all $i \in [|\sigma|]$, where $\mathcal{A}(X)_i, \mathcal{A}(X')_i$, and σ_i are the ith characters of $\mathcal{A}(X), \mathcal{A}(X')$, and σ, respectively. Additionally, $\sum \varepsilon_i \leq \varepsilon$.

Fix an output $\tau \in \texttt{range}(\mathcal{A})$. By output distinction and determinism, τ corresponds to exactly one path $\varrho = q_0 \to q_1 \to \cdots \to q_n$ in $\mathcal{A}$. Recall that $\texttt{trans}(q_i)$ is the transition from $q_i \to q_{i+1}$ and $\texttt{guard}(q_i) = c_i$ is the guard of $\texttt{trans}(q_i)$ for $0 \leq i \leq n-1$.

Let $AT = \{i : \texttt{trans}(q_i) \text{ is an assignment transition}\}$. Note that by the initialization condition, $0 \in AT$. Let $m = |AT|$ and let $AT(k)$ be the ordering of AT for $1 \leq k \leq m$ such that $AT(k) < AT(k+1)$ for all k. Additionally, let $AT(m+1) = n$.

We want to split up ϱ into a series of subpaths separated by assignment transitions.

Splitting ϱ into segments

Recall that for a path $\varrho = q_0 \to q_1 \to \cdots \to q_n$, $\varrho[i:j]$ is the subpath $q_i \to \cdots \to q_j$. For $1 \leq k \leq m$, let ρ_k be the subpath $\varrho[AT(k) : AT(k+1)]$. Note that for each $\rho_k = a_0 \to \cdots \to a_p$, $\texttt{trans}(a_0)$ is an assignment transition and none of the other transitions in ρ_k are assignment transitions.

For a subpath $\rho = \varrho[i:j]$, let $\mathcal{A}(X)[\rho]$ be the substring $\tau[i:j]$.

For each k, there are two sets of couplings we can create for ρ_k, each with their own associated privacy cost.

Coupling strategies for each segment

Consider a subpath $\rho_k = a_0 \to \cdots \to a_p = \varrho[AT(k) : AT(k+1)]$.

Let $\sigma_k = \tau[AT(k) : AT(k+1)]$ be the substring of the output τ contributed by ρ_k.

We will construct the lifting $\mathcal{A}(X)[\rho_k]\{(a,b) : a = \sigma_k \implies b = \sigma_k\}^{\#(\varepsilon,0)}\mathcal{A}(X')[\rho_k]$, or equivalently, construct the liftings $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon_i,0)}\mathcal{A}(X')[\rho_k]_i$ for all $i \in \{0,\ldots,p-1\}$ such that $\sum \varepsilon_i \leq \varepsilon$.

Let $z_0,\ldots,z_p$ be the Laplace noise added to the input read at each state in ρ_k. Without loss of generality[2], suppose that the mean of each z_i is 0 and that the spread parameter of each z_i is $\frac{1}{\varepsilon_i}$[3]. Let $z_0',\ldots,z_p'$ be the Laplace noise added to the input at each state in ρ_k to produce $\texttt{insample}'$. Suppose that μ_i and $\frac{1}{\varepsilon_i'}$ are the mean and spread parameter, respectively, of each z_i'.

Similarly, let $\texttt{in}_i$ for $i \in \{0,\ldots,p\}$ represent the input value read at each state a_i and let x be the value of the stored variable of $\mathcal{A}$. For any of these variables v, let $v\langle 1\rangle$ represent its value in a run of $\mathcal{A}(X)$, and let $v\langle 2\rangle$ represent its value in a run of $\mathcal{A}(X')$.

Recall that because $X \sim X'$ and we assume that $\Delta q_i = 1$ for all q_i, $|\texttt{in}_i\langle 1\rangle - \texttt{in}_i\langle 2\rangle| \leq 1$ for all i. Additionally, $\texttt{insample}_i = \texttt{in}_i + x_i$ by definition.

Note that we are only (for now) considering a single stored value in x.

The first coupling strategy (call this S^L)[4] proceeds as follows:

Suppose that $\texttt{trans}(a_0)$ does not output $\texttt{insample}$. If it does, then we cannot use S^L.

Create the lifting $z_0\langle 1\rangle(=)^{\#(2\varepsilon_0,0)}z_0\langle 2\rangle + \texttt{in}_0\langle 2\rangle - \texttt{in}_0\langle 1\rangle - 1$. Note that since $\texttt{trans}(a_0)$ is an assignment transition and $\texttt{insample}_0 = \texttt{in}_0 + z_0$, this is equivalent to constructing the lifting $x\langle 1\rangle + 1(=)^{\#(2\varepsilon_0,0)}x\langle 2\rangle$.

Fix some $i \in \{1,\ldots,p-1\}$.

If $\texttt{trans}(a_i)$ outputs $\texttt{insample}$, then construct the lifting $z_i\langle 1\rangle(=)^{\#(\varepsilon_i,0)}z_i\langle 2\rangle + \texttt{in}_i\langle 2\rangle - \texttt{in}_0\langle 1\rangle$. This is equivalent to constructing the lifting $\texttt{insample}\langle 1\rangle(=)^{\#(\varepsilon_i,0)}\texttt{insample}\langle 2\rangle$.

Otherwise if $\texttt{guard}(a_i) = \texttt{insample} < \texttt{x}$, construct the lifting $z_i\langle 1\rangle(=)^{\#(0,0)}z_i\langle 2\rangle$. If $\texttt{trans}(a_i)$ doesn't output $\texttt{insample}$ and $\texttt{guard}(a_i) = \texttt{insample} \geq \texttt{x}$, construct the lifting $z_i\langle 1\rangle + 2(=)^{\#(2\epsilon_i,0)}z_i\langle 2\rangle$.

If $\texttt{trans}(a_i)$ outputs $\texttt{insample'}$, then construct the lifting $\texttt{insample}'\langle 1\rangle(=)^{\#(\varepsilon_i',0)}\texttt{insample}'\langle 2\rangle$.

As before, we claim that if $(\sigma_k)_i \in \Gamma$ and $\texttt{guard}(a_i) \in \{\texttt{insample} < \texttt{x}, \texttt{true}\}$, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(0,0)}\mathcal{A}(X')[\rho_k]_i$.

If $(\sigma_k)_i \in \Gamma$ and $\texttt{guard}(a_i) = \texttt{insample} \geq \texttt{x}$, then we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(2\varepsilon_i,0)}\mathcal{A}(X')[\rho_k]_i$.

If $(\sigma_k)_i = \texttt{insample}$ and $\texttt{guard}(a_i) \in \{\texttt{insample} < \texttt{x}, \texttt{true}\}$, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon_i,0)}\mathcal{A}(X')[\rho_k]_i$.

If $(\sigma_k)_i = \texttt{insample}$ and $\texttt{guard}(a_i) = \texttt{insample} \geq \texttt{x}$, the only lifting we can construct is $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\infty,0)}\mathcal{A}(X')[\rho_k]_i$. If this is the case, we say a_i is

[2]If the mean of the noise added at a state *isn't* zero originally, we can shift the input query at that state

[3]So the pdf of each z_i is $f(x) = \frac{\varepsilon_i}{2}\exp(-\varepsilon_i|x|)$

[4]Because it allows all "less than" guards to be traversed with zero cost

faulty (in the context of S^L).

If $(\sigma_k)_i = \texttt{insample'}$ and $\texttt{guard}(a_i) \in \{\texttt{insample} < \texttt{x}, \texttt{true}\}$, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon'_i,0)}\mathcal{A}(X')[\rho_k]_i$.

If $(\sigma_k)_i = \texttt{insample'}$ and $\texttt{guard}(a_i) = \texttt{insample} \geq \texttt{x}$, then we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(2\varepsilon_i+\varepsilon'_i,0)}\mathcal{A}(X')[\rho_k]_i$.

Thus by sequential composition of liftings, we have $\mathcal{A}(X)[\rho]\{(a,b) : a = \sigma_k \implies b = \sigma_k\}^{\#(\varepsilon_L,0)}\mathcal{A}(X')[\rho]$, where the total privacy cost ε_L of S^L is

$$\varepsilon_L = \begin{cases} \infty & \exists i : a_i \text{ is faulty} \\ 2\varepsilon_0 + \sum_{i>0:\texttt{guard}(a_i)=\texttt{insample}\geq\texttt{x}} 2\varepsilon_i + \sum_{i:(\sigma_k)_i=\texttt{insample'}} \varepsilon'_i & \text{otherwise} \end{cases}$$

The second coupling strategy (S^G) proceeds as follows:

Suppose that $\texttt{trans}(a_0)$ does not output $\texttt{insample}$. If it does, then we cannot use S^G.

Similar to before, create the lifting $z_0\langle 1\rangle(=)^{\#(2\varepsilon_0,0)}z_0\langle 2\rangle + \texttt{in}_0\langle 2\rangle - \texttt{in}_0\langle 1\rangle + 1$, which is equivalent to constructing the lifting $x\langle 1\rangle(=)^{\#(2\varepsilon_0,0)}x\langle 2\rangle + 1$.

Fix some $i \in \{1, \ldots, p-1\}$.

If $\texttt{trans}(a_i)$ outputs $\texttt{insample}$, then construct the lifting $z_i\langle 1\rangle(=)^{\#(\varepsilon_i,0)}z_i\langle 2\rangle + \texttt{in}_i\langle 2\rangle - \texttt{in}_0\langle 1\rangle$. This is equivalent to constructing the lifting $\texttt{insample}\langle 1\rangle(=)^{\#(\varepsilon_i,0)}\texttt{insample}\langle 2\rangle$.

Otherwise if $\texttt{guard}(a_i) = \texttt{insample} \geq \texttt{x}$, construct the lifting $z_i\langle 1\rangle(=)^{\#(0,0)}z_i\langle 2\rangle$. If $\texttt{trans}(a_i)$ doesn't output $\texttt{insample}$ and $\texttt{guard}(a_i) = \texttt{insample} < \texttt{x}$, construct the lifting $z_i\langle 1\rangle(=)^{\#(2\epsilon_i,0)}z_i\langle 2\rangle+2$.

If $\texttt{trans}(a_i)$ outputs $\texttt{insample'}$, then construct the lifting $\texttt{insample}'\langle 1\rangle(=)^{\#(\varepsilon'_i,0)}\texttt{insample}'\langle 2\rangle$.

As before, we claim that if $(\sigma_k)_i \in \Gamma$ and $\texttt{guard}(a_i) \in \{\texttt{insample} \geq \texttt{x}, \texttt{true}\}$, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(0,0)}\mathcal{A}(X')[\rho_k]_i$. (Again, see section 1.3 for details.)

If $(\sigma_k)_i \in \Gamma$ and $\texttt{guard}(a_i) = \texttt{insample} < \texttt{x}$, then we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(2\varepsilon_i,0)}\mathcal{A}(X')[\rho_k]_i$.

If $(\sigma_k)_i = \texttt{insample}$ and $\texttt{guard}(a_i) \in \{\texttt{insample} \geq \texttt{x}, \texttt{true}\}$, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon_i,0)}\mathcal{A}(X')[\rho_k]_i$.

If $(\sigma_k)_i = \texttt{insample}$ and $\texttt{guard}(a_i) = \texttt{insample} < \texttt{x}$, the only lifting we can construct is $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\infty,0)}\mathcal{A}(X')[\rho_k]_i$. If this is the case, we say a_i is *faulty* (in the context of S^G).

If $(\sigma_k)_i = \texttt{insample'}$ and $\texttt{guard}(a_i) \in \{\texttt{insample} \geq \texttt{x}, \texttt{true}\}$, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon'_i,0)}\mathcal{A}(X')[\rho_k]_i$.

If $(\sigma_k)_i = \texttt{insample'}$ and $\texttt{guard}(a_i) = \texttt{insample} < \texttt{x}$, then we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(2\varepsilon_i+\varepsilon'_i,0)}\mathcal{A}(X')[\rho_k]_i$.

Thus by sequential composition of liftings, we have $\mathcal{A}(X)[\rho]\{(a,b) : a = \sigma_k \implies b = \sigma_k\}^{\#(\varepsilon_G,0)}\mathcal{A}(X')[\rho]$, where the total privacy cost ε_G of S^G is

$$\varepsilon_G = \begin{cases} \infty & \exists i : a_i \text{ is faulty} \\ 2\varepsilon_0 + \sum_{i>0:\texttt{guard}(a_i)=\texttt{insample<x}} 2\varepsilon_i + \sum_{i:(\sigma_k)_i=\texttt{insample}'} \varepsilon'_i & \text{otherwise} \end{cases}$$

The third coupling strategy (S^N) proceeds as follows:

Create the lifting $z_0\langle 1\rangle(=)^{\#(2\varepsilon_0,0)}z_0\langle 2\rangle + \texttt{in}_0\langle 2\rangle - \texttt{in}_0\langle 1\rangle$. This is equivalent to constructing the lifting $x\langle 1\rangle(=)^{\#(2\varepsilon_0,0)}x\langle 2\rangle$.

Fix some $i \in \{0, \ldots, p-1\}$.

For all states a_i, construct the lifting $z_i\langle 1\rangle(=)^{\#(\varepsilon_i,0)}z_i\langle 2\rangle$.

If $(\sigma_k)_i = \texttt{insample}'$, also construct the lifting $\texttt{insample}'\langle 1\rangle(=)^{\#(\varepsilon'_i,0)}\texttt{insample}'\langle 2\rangle$.

We claim that if $(\sigma_k)_i \in \Gamma \cup \{\texttt{insample}\}$, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon_i,0)}\mathcal{A}(X')[\rho_k]_i$.

Otherwise, if $(\sigma_k)_i = \texttt{insample'}$, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon_i+\varepsilon'_i,0)}\mathcal{A}(X')[\rho_k]_i$.

Thus by sequential composition of liftings, we have $\mathcal{A}(X)[\rho]\{(a,b) : a = \sigma_k \implies b = \sigma_k\}^{\#(\varepsilon_L,0)}\mathcal{A}(X')[\rho]$, where the total privacy cost ε_N of S^N is

$$\varepsilon_N = \sum_i \varepsilon_i + \sum_{i:(\sigma_k)_i=\texttt{insample}'} \varepsilon'_i.$$

Combining segments together

Note that we have not actually explicitly guaranteed that $\mathcal{A}(X)[\rho_k]_0 = (\sigma_k)_0 \implies \mathcal{A}(X')[\rho_k]_0 = (\sigma_k)_0$. This situation is not immediately generalizable from the previous analysis because the transition that $\mathcal{A}(X)$ (or $\mathcal{A}(X')$) takes from state a_0 is dependent on the *previous* value of x.

Luckily, for most cases, coupling the new x values together will also allow us to satisfy the transition guard.

However, there are some cases where this is impossible. Combined with the fact that if the assignment transition outputs `insample`, we must choose the coupling strategy S^N, this leads to the following constraints on segments ρ_k, ρ_{k+1}:

- If the assignment transition of ρ_k outputs `insample`, then we can only use S^N for ρ_k.
- If S^G was used for ρ_k and the assignment guard of ρ_{k+1} is $\texttt{insample} < \texttt{x}$, then S^G must be used for ρ_{k+1}.
- If S^L was used for ρ_k and the assignment guard of ρ_{k+1} is $\texttt{insample} \geq \texttt{x}$, then S^L must be used for ρ_{k+1}.

- If S^N was used for ρ_k and the assignment guard of ρ_{k+1} is $\texttt{insample} \geq \texttt{x}$, then either S^N or S^L must be used for ρ_{k+1}.
- If S^N was used for ρ_k and the assignment guard of ρ_{k+1} is $\texttt{insample} < \texttt{x}$, then either S^N or S^G must be used for ρ_{k+1}.

For a detailed case analysis, see section 1.4.

Assuming that these two constraints are satisfied, combining segments together is relatively straightforward. From each segment ρ_k, we have used either S^G, S^L, or S^N to construct the lifting $\mathcal{A}(X)[\rho_k]\{(a,b) : a = \sigma_k \implies b = \sigma_k\}^{\#(\varepsilon^{(k)},0)}\mathcal{A}(X')[\rho_k]$ for some $\varepsilon^{(k)}$. Then because $\tau = \sigma_1 \cdot \sigma_1 \cdot \ldots \cdot \sigma_m$, by sequential composition, we can construct the overall lifting $\mathcal{A}(X)\{(a,b) : a = \tau \implies b = \tau\}^{\#(\varepsilon_\tau,0)}\mathcal{A}(X')$, where $\varepsilon_\tau = \sum \varepsilon_k$.

If $\max_{\tau \in \texttt{range}(\mathcal{A})}\{\varepsilon_\tau\}$ is finite, this is a proof that $\mathcal{A}$ is $\max\{\varepsilon_\tau\}$-differentially private, as desired.

1.2 Privacy

We begin with some preliminary definitions to help us construct *segment families*, which will allow us to group the potentially infinite number of paths through $\mathcal{A}$ into a finite number of families that we can analyze individually.

Definition 1.1. A **terminal state** of a DiPA $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, X, P, \delta)$ is a state $q \in Q$ such that $\delta(q, c)$ is not defined for any guard condition c[a]. Let $\texttt{term}(\mathcal{A})$ denote the set of terminal states of $\mathcal{A}$.

[a] This definition assumes that for all states q, if $\delta(q, \texttt{insample} < \texttt{x})$ is defined, then $\delta(q, \texttt{insample} \geq \texttt{x})$ is as well. Technically this is not required, but adding this requirement doesn't change the expressive power of DiPAs.

Definition 1.2. A **complete path** of a DiPA $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, X, P, \delta)$ is a path in $\mathcal{A}$ that begins at q_0 and ends at a terminal state in $\mathcal{A}$. Complete paths represent a single execution of the program corresponding to $\mathcal{A}$.

Definition 1.3. Consider a DiPA $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, X, P, \delta)$. Let $q_i, q_j \in Q$ be such that there exists a assignment transition from q_i and either q_j is a terminal state or there exists an assignment transition from q_j. Then we define $seg(q_i, q_j)$ as the set of all paths $\rho = a_1 \to \ldots \to a_m$ in $\mathcal{A}$ that begin at q_i and end at q_j such that the only assignment transition in ρ is $\texttt{trans}(q_i)$ and the path $a_1 \to \ldots \to a_{m-1}$ is acyclic. Note that q_i can equal q_j.

For a segment $s \in seg(q_i, q_j)$, let $\texttt{trans}(s)$ refer to the (only) assignment transition in s and let $\texttt{guard}(s)$ be the guard of the (only) assignment transition in s.

Note that we allow for $q_i = q_j$ in the definition, but there can be no cycles *internal* to a path in $seg(q_i, q_j)$.

Definition 1.4. For a DiPA $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, X, P, \delta)$, let $\texttt{assn}(\mathcal{A})$ be the set of states $q \in Q$ such that there exists an assignment transition from q. Overloading notation, define

$$seg(\mathcal{A}) = \bigcup_{q_i \in \texttt{assn}(\mathcal{A})} \bigcup_{q_j \in \texttt{assn}(\mathcal{A}) \cup \texttt{term}(\mathcal{A})} seg(q_i, q_j)$$

Proposition 1.5. *For all $q_i \in \texttt{assn}(\mathcal{A}), q_j \in \texttt{assn}(\mathcal{A}) \cup \texttt{term}(\mathcal{A})$, $|seq(q_i, q_j)|$ is finite.*

Corollary 1.6. *For all DiPAs $\mathcal{A}$, $|seg(\mathcal{A})|$ is finite.*

Definition 1.7. For a path $\rho = a_1 \to \ldots \to a_n$, let $acyclic(\rho)$ be ρ with all cycles (except potentially a cycle where $a_1 = a_n$) removed. That is, $acyclic(\rho)$ is the path constructed iteratively by the following process:

1. If $\exists i \neq j \in [n]$ where $a_i = a_j$ and $\{a_i, a_j\} \neq \{a_1, a_n\}$, remove the cycle found between a_i and a_j; i.e. let $\rho = a_1 \to \ldots \to a_i \to a_{j+1} \to \ldots \to a_n$.
2. Repeat until no such i, j exist in ρ[a].

[a] Do I need to justify that $acyclic(\rho)$ is a function?

Definition 1.8. Suppose $\texttt{trans}(q_i)$ and $\texttt{trans}(q_j)$, $q_i, q_j \in Q$ are two assignment transitions in a DiPA $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, X, P, \delta)$ such that $seg(q_i, q_j) \neq \emptyset$ and let $s \in seg(q_i, q_j)$. Then define the **segment family** of s, notated $segF(s)$, as the set of all paths ρ from q_i to q_j such that the only assignment transition in ρ is $\texttt{trans}(q_i)$ and $acyclic(\rho) = s$.

Definition 1.9. Let $G = (V, E)$ be the underlying (directed) graph of a DiPA $\mathcal{A}$. For all paths ρ of $\mathcal{A}$, let $G_\rho = (V_\rho, E_\rho)$ be the subgraph of G corresponding to ρ. For all $s \in seg(\mathcal{A})$, define the **subgraph corresponding to** s, notated $G_s = (V_s, E_s)$, as a subgraph of G with $V_s = \bigcup_{\rho \in segF(s)} V_\rho$ and $E_s = \bigcup_{\rho \in segF(s)} E_\rho$[a].

[a] Does "the subgraph of G corresponding to ρ" need to be separately defined?

Proposition 1.10. *For all $s, s' \in seg(\mathcal{A})$ where $s \neq s'$, G_s and $G_{s'}$ are edge-disjoint. Further, if $G = (V, E)$ is the underlying graph of $\mathcal{A}$, $V = \bigcup_{s \in \mathcal{A}} V_s$ and $E = \bigcup_{s \in \mathcal{A}} E_s$, where $G_s = (V_s, E_s)$ for all s.*

Definition 1.11. For a segment $s \in seg(\mathcal{A})$, the **G-cost** of s is

$$cost_G(s) = \sup\{\varepsilon_G^{(\rho)} : \rho \in segF(s)\}$$

and the **L-cost** of s is

$$cost_L(s) = \sup\{\varepsilon_L^{(\rho)} : \rho \in segF(s)\},$$

where $\varepsilon_G^{(\rho)}$ (resp. $\varepsilon_L^{(\rho)}$) is the privacy cost of using the coupling strategy S^G (resp. S^L) for the path ρ (see section 1.1).

If either set is unbounded, take the supremum to be ∞.

Proposition 1.12. *For a segment $s \in seg(\mathcal{A})$, $cost_G(s) = \infty$ iff G_s has a cycle with* *a transition with guard* $\texttt{insample} < \texttt{x}$ *and* $cost_L(s) = \infty$ *iff* G_s *has a cycle with a transition with guard* $\texttt{insample} \geq \texttt{x}$.

Proposition 1.13. *Every complete path* ρ *in a DiPA* $\mathcal{A}$ *can be partitioned into a sequence of subpaths* ρ_i *such that* $\rho_i \in segF(s)$ *for some* $s \in seg(\mathcal{A})$.

Proof. Fix a complete path $\rho = q_0 \to \ldots \to q_n$ in $\mathcal{A}$. Let $AT = \{i : \texttt{trans}(q_i)$ is an assignment transition or q_i is a terminal state$\} = (a_1, \ldots, a_m)$ be the ordered set of the indices of all $m-1$ assignment transitions in ρ as well as the terminal state of ρ. Note that because ρ is complete, $a_1 = 0$ (i.e. $\texttt{trans}(q_0)$ is an assignment transition) and $a_m = n$, since q_n is a terminal state in $\mathcal{A}$.

Recall that $\rho[i : j]$ is the subpath $q_i \to \ldots \to q_j$ of ρ.

Then partition ρ into $m-1$ subpaths ρ_i, where for all $1 \leq i < m$, $\rho_i = \rho[a_i : a_{i+1}]$. Since $a_1 = 0$ and $a_m = n$ and all a_i are ordered, ρ_i is a partition of ρ.

Now consider some $\rho_i = \rho[a_i : a_{i+1}]$. Then $acyclic(\rho_i) \in seg(q_{a_i}, a_{i+1})$, since the only assignment transition in ρ_i is $\texttt{trans}(q_{a_i})$ by construction. Thus by definition, $\rho_i \in segF(s)$ for some $s \in seg(\mathcal{A})$, specifically when $s = acyclic(\rho_i)$. $\square$

Proposition 1.14. *If a DiPA* $\mathcal{A}$ *terminates and has no leaking cycles, then no assignment transition in* $\mathcal{A}$ *lies on a cycle in* $\mathcal{A}$.

Proof. Because we suppose $\mathcal{A}$ terminates, there can be no cycle in $\mathcal{A}$ whose transitions all have guard $\texttt{true}$. In other words, every cycle in $\mathcal{A}$ must contain a transition whose guard is either $\texttt{insample} < \texttt{x}$ or $\texttt{insample} \geq \texttt{x}$.

Then note that no transitions on a cycle can be an assignment transition. To see this, suppose that there exists a cycle C with a assignment transition $\texttt{trans}(q)$. Because $\mathcal{A}$ terminates, there must exist some $\texttt{trans}(q')$ in C such that $\texttt{guard}(q') \in \{\texttt{insample} < \texttt{x}, \texttt{insample} \geq \texttt{x}\}$. But then C is a leaking cycle, which is a contradiction. $\square$

Lemma 1.15. *If a terminating DiPA* $\mathcal{A}$ *has no leaking cycles, then there exists a global bound* $N \in \mathbb{N}$ *such that every complete path* ρ *in* $\mathcal{A}$ *can be partitioned into a sequence of at most* N *subpaths* ρ_i *such that for all* i, $\rho_i \in segF(s_i)$ *for some* $s_i \in seg(\mathcal{A})$.

Proof. Because $\mathcal{A}$ is a finite automaton, there are a finite number of assignment transitions in $\mathcal{A}$. Let T be the set of assignment transitions in $\mathcal{A}$ and let $N = |T|$, so we can index T as $t_1, \ldots, t_N$.

We claim that for every complete path ρ in $\mathcal{A}$, each assignment transition t_i can appear in ρ at most one time.

Suppose for the sake of contradiction that there exists some complete path $\rho = q_0 \to \ldots \to q_n$ such that the assignment transition t appears in ρ (at least) twice. Let $\texttt{trans}(q_i) = \texttt{trans}(q_j) = t, i \neq j$ be the transitions in ρ where t appears. This implies that $q_i = q_j$ for $i \neq j$, meaning that $\texttt{trans}(q_i)$ lies on a cycle. However, because $\mathcal{A}$ has no leaking cycles and terminates, by proposition 1.14, no assignment transition in $\mathcal{A}$ can lie on a cycle. Thus, each assignment transition t_i can appear in any complete path at most one time.

Then for any complete path ρ in $\mathcal{A}$, we can partition ρ into subpaths ρ_i such that $\rho_i \in segF(s_i)$ for some $s_i \in seg(\mathcal{A})$ as in proposition 1.13. Because each assignment transition t_i can appear in any complete path at most one time and, in the construction in proposition 1.13, each ρ_i has exactly one assignment transition, there can be at most N subpaths ρ_i. $\square$

Definition 1.16. For a DiPA $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, X, P, \delta)$ and for any two segments $s \neq s' \in seg(\mathcal{A})$, s' **follows** s if the last state of s is the first start of s'. More formally, define the relation $\hookrightarrow$ such that $s \hookrightarrow s'$ if $s \in seg(q, q')$ and $s' \in seg(q', q^*)$ for $q, q', q^* \in Q$.

We can formulate this as a constraint satisfication problem.

For each segment s_i, we want to assign a strategy S_i from $\{S^N, S^L, S^G\}$ such that the following constraints hold:

1. Constraints for valid couplings
   (a) For all s_i, if $\texttt{trans}(s_i)$ outputs `insample`, then $S_i = S^N$.
   (b) For all s_i, s_j such that $s_i \hookrightarrow s_j$,
      i. If $\texttt{guard}(s_j) = \texttt{insample} < \texttt{x}$ and $S_i = S^G$, then $S_j = S^G$.
      ii. If $\texttt{guard}(s_j) = \texttt{insample} \geq \texttt{x}$ and $S_i = S^L$, then $S_j = S^L$.
      iii. If $\texttt{guard}(s_j) = \texttt{insample} < \texttt{x}$ and $S_i = S^N$, then $S_j \neq S^L$.
      iv. If $\texttt{guard}(s_j) = \texttt{insample} \geq \texttt{x}$ and $S_i = S^N$, then $S_j \neq S^G$.
   (c) For all segments s_i, there is no transition $\texttt{trans}(a_k)$ in s_i that is *faulty*, i.e.:
      i. If s_i contains a $\texttt{insample} < \texttt{x}$ transition that outputs `insample`, then $S_i \neq S^G$.
      ii. If s_i contains a $\texttt{insample} \geq \texttt{x}$ transition that outputs `insample`, then $S_i \neq S^L$.
      iii. If s_i contains a $\texttt{insample} \geq \texttt{x}$ and $\texttt{insample} < \texttt{x}$ transition that both output `insample`, then $S_i = S^N$[5].
2. Constraints for finite cost
   (a) For all segments s_i, no cycle in s_i has a transition that outputs `insample` or `insample'`.
   (b) If s_i has an `L`-cycle, then $S_i = S^L$.
   (c) If s_i has a `G`-cycle, then $S_i = S^G$.

[5]This follows directly from (i) and (ii)

If such an assignment over all s_i exists, then we claim that we have created a coupling proof that (a) is valid and (b) demonstrates that $\mathcal{A}$ is ε-DP for some finite ε. Validity follows from sections 1.3 and 1.4. Finiteness follows from prop 1.12[6].

Theorem 1.17. *If $\mathcal{A}$ is well-formed, there exists such an assignment over s_i.*

This proof is incomplete because I gave up on doing like a 20 case analysis, but the general idea is that if a contradiction between two constraints is forced, then there must exist some "bad" graph structure. In general, take a glance at section 1.4 which has some notes on what each contradiction would correspond to. (For example, if S_i is forced to be S^G because it has a `G`-cycle, but S_{i+1} is forced to take S^L because it has a `L`-cycle AND this violates constraint (1bi) then a leaking pair is formed - the path between the two cycles must be a `AL`-path to violate constraint (1bi))

In general, the constraints that include S^N are related to privacy violating paths and the constraints between S^L and S^G relate to leaking pairs. Constraint (2a) is directly related to disclosing cycles, and the lack of leaking cycles is assumed implicitly because we consider a bounded number of segments for each path.

Proof. Since $\mathcal{A}$ has no leaking cycles, it is meaningful to consider a finite number of segments s_i.

Since $\mathcal{A}$ has no disclosing cycles, constraint (2a) is always satisfied.

Suppose that there is no satisfying assignment.

Fix some candidate assignment of all S_i. Then there must be a constraint of S_i that is violated. We claim that we can "fix" any candidate assignment by iteratively decreasing the number of constraints that are violated.

Constraint (1a) is violated

Suppose that condition (1a) is violated for some segment s_j, so $\texttt{trans}(s_j)$ outputs `insample` but $S_j \neq S^N$. Naively, to fix this constraint, we must change S_j to S^N. This may, however, lead to further constraints being violated.

Consider some segment $s_i \hookrightarrow s_j$. If $S_i = S^G$ and $\texttt{guard}(s_j) = \texttt{insample} < \texttt{x}$, then constraint (1bi) is now violated, since $S_j \neq S^G$. Changing $S_i = S^N$ or $S_i = S^L$ means that (1bi) is no longer violated, but again brings up new possible issues. First, if s_i has a `G`-cycle, then constraint (2c) is violated. However, if s_i has a `G`-cycle, then there is an `AL`$-$path from the cycle to $\texttt{trans}(s_j)$, which has guard $\texttt{insample} < \texttt{x}$ and outputs `insample`, which creates a privacy violating path (by the third condition). Thus, changing S_i cannot violate constraint (2c).

If $\texttt{guard}(s_i) = \texttt{insample} < \texttt{x}$ and there exists $s_h \hookrightarrow s_i$ where $S_h = S^G$, we must also change the assigned strategy for S_h. As before, if this is impossible by constraint (2c), then there must exist a privacy violating path in $\mathcal{A}$. This process can be repeated inductively until

[6] This is outdated and doesn't include discussion of when `insample` is output, but the general claim still holds

changing the assigned strategy for a given segment no longer violates any further constraints, or we demonstrate the existence of a privacy violating path. The same reasoning holds symmetrically for if $\texttt{guard}(s_j) = \texttt{insample} \geq \texttt{x}$ and $S_i = S^L$; wherein changing $S_i = S^N$ would create a violation of constraint (1bii).

Now consider some segment $s_j \hookrightarrow s_k$. If $\texttt{guard}(s_k) = \texttt{insample} < \texttt{x}$ and $S_k = S^L$, then changing $S_j = S^N$ would violate constraint (1ciii). By a similar process as before, we can change s_k and all subsequent relevant segments (otherwise, a privacy violating path would be created).

Thus, if a candidate assignment has a segment assignment that violates constraint (1a), then we can find a similar candidate assignment such that that segment's assignment does not violate constraint (1a) and, in particular, the total number of segments with violated constraints decreases.

Note: these are done out of order because the earlier ones are easier to reason about in isolation

Constraint (1ci) is violated

Suppose that constraint (1a) is not violated, since otherwise we can fix (1a) first.

Suppose that condition (1ci) is violated for some s_j; i.e. s_j contains an $\texttt{insample} < \texttt{x}$ transition that outputs `insample` but $S_j = S^G$. Note that s_j cannot contain a `G`-cycle , since a privacy violating path would be created. Since (1a) is not violated, $\texttt{trans}(s_j)$ does not output `insample`. Note that s_j cannot contain both a $\texttt{insample} \geq \texttt{x}$ transition that outputs `insample` and an `L`-cycle , since a privacy violating path would be created for similar reasons.

If s_j contains a $\texttt{insample} \geq \texttt{x}$ transition that outputs `insample`, then choose $S_j = S^N$. This could possibly lead to violations of constraints (1biii) or (1biv). If constraint (1biii) is violated for some segment $s_j \hookrightarrow s_k$, then, if s_k does not have either an `L`-cycle or a `G`-cycle , we can set $S_k = S^N$. Otherwise, if s_k has a `G`-cycle , then set $S_k = S^G$. Note that s_k cannot have an `L`-cycle , since otherwise a privacy violating path would be created (similarly, s_k cannot have both a $\texttt{insample} < \texttt{x}$ transition that outputs `insample` and a `G`-cycle). Then we can "propagate" this constraint onwards. The same reasoning holds for constraint (1biv).

Otherwise, if s_j ...

Constraint (1cii) is violated

This can be fixed symmetrically to (1ci)

Constraint (2b) is violated

Constraint (2c) is violated

This can be fixed symmetrically to (2c)

Constraint (1bi) is violated

Suppose that condition (1a) is violated for some segment s_j, so there exists some $s_i \hookrightarrow s_j$ such that $\texttt{guard}(s_j) = \texttt{insample} < \texttt{x}$, $S_i = S^G$, and $S_j \neq S^G$.

- either we can change S_i and "propagate" all changes back (and then we are done), or there is an `AG`-path from a `G`-cycle to s_j.

- if there is such a path, then we must change $S_j = S^G$. This can possibly lead to (1a) being violated. However, this means that a privacy violating path exists from the `G`-cycle to the assignment transition of s_j. This can also lead to constraint (1ci) being violated; this would mean, however, that a similar privacy violating path exists from the `G`-cycle to the `insample` $<$ `x` transition in s_j that outputs `insample`. Additionally, this can lead to constraint (2b) being violated; this would mean that there is a leaking pair from the `G`-cycle to the `L`-cycle in s_j.

Finally, this can also lead to (1bi) being violated for $s_j \hookrightarrow s_k$. To fix this, have $S_k = S^G$. Then by the same reasoning as before, either no more constraints can be violated, or we can "propagate" constraint (1bi) until there are no more violations.

Constraint (1bii) is violated

This can be fixed symmetrically to constraint (1bi)

Constraint (1biii) is violated

We assume that none of the previously analyzed constraints are violated (before we start changing assignments).

Suppose that constraint (1biii) is violated for some segment s_j, so there exists some $s_i \hookrightarrow s_j$ such that $\texttt{guard}(s_j) = \texttt{insample} < \texttt{x}$, $S_i = S^N$, and $S_j = S^L$. Note that s_j cannot contain both an `L`-cycle and a `G`-cycle . If s_j has an `L`-cycle , then we change $S_i \neq S^N$. This may violate constraint (1ciii). However, if it does, then there is a privacy violating path from the `insample` $\geq$ `x` transition in s_i that outputs `insample` to the `L`-cycle in s_j.

Note that s_i cannot have a `G`-cycle , since otherwise a leaking pair would be created from the `G`-cycle to the `L`-cycle in s_j. Thus, changing S_i to S^L will not violate constraint (2c). It could, however, violate constraints (1a), (1bi), (1biii), or (1cii). If it violates constraint (1a), then a privacy violating path is created; if constraint (1bi) is violated, then, as before, either we can change the previous segment assignment or a leaking pair is created; if constraint (1biii) is violated, then, similarly, a privacy violating path is created; and finally, if (1bii) is violated, then a privacy violating path is created. In all of these cases, we are actually "propagating" changes until no more constraints are violated, which must happen or otherwise $\mathcal{A}$ is not well-formed.

Otherwise, if s_j does not have an `L`-cycle , we can still consider changing s_i to either S^L or S^G.

Finally, we can change S_j to either S^G or S^N. If s_i contains a `insample` $<$ `x` transition that outputs `insample`, then choose S_i to be S^N. Note that s_i cannot contain a `G`-cycle , since otherwise a privacy violating path would be created between that `G`-cycle and the `insample` $<$ `x` output transition. Then possibly constraints (1biii) and (1biv) are violated.

First, consider the case where we change S_i to S^G. Then possibly constraints (1bi) or (1ci) are violated. If (1ci) is violated, then

Otherwise, if we change S_i to S^N, possibly constraints (1biii), (1biv), or (2c) are violated.

Constraint (1biv) is violated

This can be fixed symmetrically to (1biii)

A similar argument can be made for all other constraints [details tbd]. □

1.3 Lifting construction details

Because of output distinction and determinism, $\mathcal{A}(X)$ takes the transition $\texttt{trans}(a_i)$ if and only if $\mathcal{A}(X)_i = (\sigma_k)_i$ and similarly, $\mathcal{A}(X')$ takes the transition $\texttt{trans}(a_i)$ if and only if $\mathcal{A}(X')_i = (\sigma_k)_i$.

This section is also not complete, but you should get the idea pretty quickly

1.3.1 S^L

Case: $\texttt{trans}(a_i)$ outputs $\sigma \in \Gamma$

Now if $\texttt{guard}(a_i) = \texttt{true}$, then the outputs of $\mathcal{A}(X)$ and $\mathcal{A}(X')$ must be equal, so $\mathcal{A}(X)[\rho_k]_i = (\sigma_k)_i \implies \mathcal{A}(X')[\rho_k]_i = (\sigma_k)_i$ trivially, so we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(0,0)}\mathcal{A}(X')[\rho_k]_i$

If $i > 0$ and $\texttt{guard}(a_i) = \texttt{insample} < \texttt{x}$, then

$$\begin{aligned}
\mathcal{A}(X)[\rho_k]_i = (\sigma_k)_i &\implies \mathcal{A}(X) \text{ takes } \texttt{trans}(a_i) \\
&\implies \texttt{in}_i\langle 1\rangle + z_i\langle 1\rangle < x\langle 1\rangle \\
&\implies \texttt{in}_i\langle 1\rangle + z_i\langle 2\rangle < x\langle 2\rangle - 1 \\
&\implies \texttt{in}_i\langle 1\rangle + 1 + z_i\langle 2\rangle < x\langle 2\rangle \\
&\implies \texttt{in}_i\langle 2\rangle + z_i\langle 2\rangle < x\langle 2\rangle \\
&\implies \mathcal{A}(X') \text{ takes } \texttt{trans}(a_i) \\
&\implies \mathcal{A}(X')[\rho_k]_i = (\sigma_k)_i
\end{aligned}$$

so we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(0,0)}\mathcal{A}(X')[\rho_k]_i$.

Similarly, if $i > 0$ and $\texttt{guard}(a_i) = \texttt{insample} \geq \texttt{x}$, then

$$\begin{aligned}
\mathcal{A}(X)[\rho_k]_i = (\sigma_k)_i &\implies \texttt{in}_i\langle 1\rangle + z_i\langle 1\rangle \geq x\langle 1\rangle \\
&\implies \texttt{in}_i\langle 1\rangle + z_i\langle 2\rangle - 2 \geq x\langle 2\rangle - 1 \\
&\implies \texttt{in}_i\langle 1\rangle - 1 + z_i\langle 2\rangle \geq x\langle 2\rangle \\
&\implies \texttt{in}_i\langle 2\rangle + z_i\langle 2\rangle \geq x\langle 2\rangle \\
&\implies \mathcal{A}(X')[\rho_k]_i = (\sigma_k)_i
\end{aligned}$$

so we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(2\varepsilon_i,0)}\mathcal{A}(X')[\rho_k]_i$.

Case: $\texttt{trans}(a_i)$ outputs insample

If $\texttt{guard}(a_i) = \texttt{true}$, the outputs of $\mathcal{A}(X)$ and $\mathcal{A}(X')$ are equal by the lifting of $\texttt{insample}\langle 1\rangle$ and $\texttt{insample}\langle 2\rangle$. Thus, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon_i,0)}\mathcal{A}(X')[\rho_k]_i$.

If $i > 0$ and $\texttt{guard}(a_i) = \texttt{insample} < \texttt{x}$, then

$$\begin{aligned}
\mathcal{A}(X)[\rho_k]_i = (\sigma_k)_i &\implies \texttt{insample}\langle 1\rangle < x\langle 1\rangle \\
&\implies \texttt{insample}\langle 2\rangle < x\langle 2\rangle - 1 \\
&\implies \texttt{insample}\langle 2\rangle < x\langle 2\rangle \\
&\implies \mathcal{A}(X')[\rho_k]_i = (\sigma_k)_i
\end{aligned}$$

so we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon_i,0)}\mathcal{A}(X')[\rho_k]_i$.

Similarly, if $i > 0$ and $\texttt{guard}(a_i) = \texttt{insample} \geq \texttt{x}$, then

$$\begin{aligned}
\mathcal{A}(X)[\rho_k]_i = (\sigma_k)_i &\implies \texttt{in}_i\langle 1\rangle + z_i\langle 1\rangle \geq x\langle 1\rangle \\
&\implies \texttt{in}_i\langle 1\rangle + z_i\langle 2\rangle \geq x\langle 2\rangle - 1 \\
&\implies \texttt{in}_i\langle 1\rangle + 1 + z_i\langle 2\rangle \geq x\langle 2\rangle
\end{aligned}$$

We cannot derive the desired implication in this case, so we can only construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\infty,0)}\mathcal{A}(X')[\rho_k]_i$. We say a_i is *faulty* (in the context of S^L).

Case: $\texttt{trans}(a_i)$ outputs $\texttt{insample}'$

If $\texttt{guard}(a_i) = \texttt{true}$, the outputs of $\mathcal{A}(X)$ and $\mathcal{A}(X')$ are equal by the lifting of $\texttt{insample}'\langle 1\rangle$ and $\texttt{insample}'\langle 2\rangle$. Thus, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon_i',0)}\mathcal{A}(X')[\rho_k]_i$.

If $i > 0$ and $\texttt{guard}(a_i) = \texttt{insample} < \texttt{x}$, then

$$\begin{aligned}
\mathcal{A}(X)[\rho_k]_i = (\sigma_k)_i &\implies \texttt{insample}\langle 1\rangle < x\langle 1\rangle \\
&\implies \texttt{insample}\langle 2\rangle < x\langle 2\rangle - 1 \\
&\implies \texttt{insample}\langle 2\rangle < x\langle 2\rangle \\
&\implies \mathcal{A}(X')[\rho_k]_i = (\sigma_k)_i
\end{aligned}$$

Further, since we have the lifting $\texttt{insample}'\langle 1\rangle(=)^{\#(\varepsilon_i',0)}\texttt{insample}'\langle 2\rangle$, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon_i',0)}\mathcal{A}(X')[\rho_k]_i$.

Similarly, if $i > 0$ and $\texttt{guard}(a_i) = \texttt{insample} \geq \texttt{x}$, then

$$\begin{aligned}
\mathcal{A}(X)[\rho_k]_i = (\sigma_k)_i &\implies \texttt{in}_i\langle 1\rangle + z_i\langle 1\rangle \geq x\langle 1\rangle \\
&\implies \texttt{in}_i\langle 1\rangle + z_i\langle 2\rangle - 2 \geq x\langle 2\rangle - 1 \\
&\implies \texttt{in}_i\langle 1\rangle - 1 + z_i\langle 2\rangle \geq x\langle 2\rangle \\
&\implies \texttt{in}_i\langle 2\rangle + z_i\langle 2\rangle \geq x\langle 2\rangle \\
&\implies \mathcal{A}(X')[\rho_k]_i = (\sigma_k)_i
\end{aligned}$$

so we can similarly construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(2\varepsilon_i+\varepsilon_i',0)}\mathcal{A}(X')[\rho_k]_i$.

1.3.2 S^G

Case: $\texttt{trans}(a_i)$ outputs $\sigma \in \Gamma$

If $\texttt{guard}(a_i) = \texttt{true}$, then the outputs of $\mathcal{A}(X)$ and $\mathcal{A}(X')$ must be equal, so $\mathcal{A}(X)[\rho_k]_i = (\sigma_k)_i \implies \mathcal{A}(X')[\rho_k]_i = (\sigma_k)_i$ trivially, so we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(0,0)}\mathcal{A}(X')[\rho_k]_i$

If $\texttt{guard}(a_i) = \texttt{insample} \geq \texttt{x}$, then

$$\begin{aligned}
\mathcal{A}(X)[\rho_k] = (\sigma_k)_i &\implies \texttt{in}_i\langle 1\rangle + z_i\langle 1\rangle \geq x\langle 1\rangle \\
&\implies \texttt{in}_i\langle 1\rangle + z_i\langle 2\rangle \geq x\langle 2\rangle + 1 \\
&\implies \texttt{in}_i\langle 1\rangle - 1 + z_i\langle 2\rangle \geq x\langle 2\rangle \\
&\implies \texttt{in}_i\langle 2\rangle + z_i\langle 2\rangle \geq x\langle 2\rangle \\
&\implies \mathcal{A}(X')[\rho_k] = (\sigma_k)_i
\end{aligned}$$

so we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(0,0)}\mathcal{A}(X')[\rho_k]_i$.

Similarly, if $i > 0$ and $\texttt{guard}(a_i) = \texttt{insample} < \texttt{x}$, then

$$\begin{aligned}
\mathcal{A}(X)[\rho_k] = (\sigma_k)_i &\implies \texttt{in}_i\langle 1\rangle + z_i\langle 1\rangle < x\langle 1\rangle \\
&\implies \texttt{in}_i\langle 1\rangle + z_i\langle 2\rangle + 2 < x\langle 2\rangle + 1 \\
&\implies \texttt{in}_i\langle 1\rangle + 1 + z_i\langle 2\rangle < x\langle 2\rangle \\
&\implies \texttt{in}_i\langle 2\rangle + z_i\langle 2\rangle < x\langle 2\rangle \\
&\implies \mathcal{A}(X')[\rho_k] = (\sigma_k)_i
\end{aligned}$$

so we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(2\varepsilon_i,0)}\mathcal{A}(X')[\rho_k]_i$.

Case: $\texttt{trans}(a_i)$ outputs $\texttt{insample}$

If $\texttt{guard}(a_i) = \texttt{true}$, the outputs of $\mathcal{A}(X)$ and $\mathcal{A}(X')$ are equal by the lifting of $\texttt{insample}\langle 1\rangle$ and $\texttt{insample}\langle 2\rangle$. Thus, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon_i,0)}\mathcal{A}(X')[\rho_k]_i$.

If $i > 0$ and $\texttt{guard}(a_i) = \texttt{insample} \geq \texttt{x}$, then

$$\begin{aligned}
\mathcal{A}(X)[\rho_k]_i = (\sigma_k)_i &\implies \texttt{insample}\langle 1\rangle \geq x\langle 1\rangle \\
&\implies \texttt{insample}\langle 2\rangle \geq x\langle 2\rangle + 1 \\
&\implies \texttt{insample}\langle 2\rangle \geq x\langle 2\rangle + 1 \\
&\implies \mathcal{A}(X')[\rho_k]_i = (\sigma_k)_i
\end{aligned}$$

so we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon_i,0)}\mathcal{A}(X')[\rho_k]_i$.

Similarly, if $i > 0$ and $\texttt{guard}(a_i) = \texttt{insample} < \texttt{x}$, then

$$\begin{aligned}
\mathcal{A}(X)[\rho_k]_i = (\sigma_k)_i &\implies \texttt{in}_i\langle 1\rangle + z_i\langle 1\rangle < x\langle 1\rangle \\
&\implies \texttt{in}_i\langle 1\rangle + z_i\langle 2\rangle < x\langle 2\rangle + 1 \\
&\implies \texttt{in}_i\langle 1\rangle - 1 + z_i\langle 2\rangle < x\langle 2\rangle
\end{aligned}$$

We cannot derive the desired implication in this case, so we can only construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\infty,0)}\mathcal{A}(X')[\rho_k]_i$. We say a_i is *faulty* (in the context of S^L).

Case: $\texttt{trans}(a_i)$ **outputs** $\texttt{insample}'$

If $\texttt{guard}(a_i) = \texttt{true}$, the outputs of $\mathcal{A}(X)$ and $\mathcal{A}(X')$ are equal by the lifting of $\texttt{insample}'\langle 1\rangle$ and $\texttt{insample}'\langle 2\rangle$. Thus, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon'_i,0)}\mathcal{A}(X')[\rho_k]_i$.

If $i > 0$ and $\texttt{guard}(a_i) = \texttt{insample} \geq \texttt{x}$, then

$$\begin{aligned}
\mathcal{A}(X)[\rho_k]_i = (\sigma_k)_i &\implies \texttt{insample}\langle 1\rangle \geq x\langle 1\rangle \\
&\implies \texttt{insample}\langle 2\rangle \geq x\langle 2\rangle + 1 \\
&\implies \texttt{insample}\langle 2\rangle \geq x\langle 2\rangle \\
&\implies \mathcal{A}(X')[\rho_k]_i = (\sigma_k)_i
\end{aligned}$$

Further, since we have the lifting $\texttt{insample}'\langle 1\rangle(=)^{\#(\varepsilon'_i,0)}\texttt{insample}'\langle 2\rangle$, we can construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(\varepsilon'_i,0)}\mathcal{A}(X')[\rho_k]_i$.

Similarly, if $i > 0$ and $\texttt{guard}(a_i) = \texttt{insample} < \texttt{x}$, then

$$\begin{aligned}
\mathcal{A}(X)[\rho_k]_i = (\sigma_k)_i &\implies \texttt{in}_i\langle 1\rangle + z_i\langle 1\rangle < x\langle 1\rangle \\
&\implies \texttt{in}_i\langle 1\rangle + z_i\langle 2\rangle + 2 < x\langle 2\rangle + 1 \\
&\implies \texttt{in}_i\langle 1\rangle + 1 + z_i\langle 2\rangle < x\langle 2\rangle \\
&\implies \texttt{in}_i\langle 2\rangle + z_i\langle 2\rangle < x\langle 2\rangle \\
&\implies \mathcal{A}(X')[\rho_k]_i = (\sigma_k)_i
\end{aligned}$$

so we can similarly construct the lifting $\mathcal{A}(X)[\rho_k]_i\{(a,b) : a = (\sigma_k)_i \implies b = (\sigma_k)_i\}^{\#(2\varepsilon_i+\varepsilon'_i,0)}\mathcal{A}(X')[\rho_k]_i$.

1.3.3 S^N

Case: $\texttt{trans}(a_i)$ **outputs** $\sigma \in \Gamma$

Case: $\texttt{trans}(a_i)$ **outputs** $\texttt{insample}$

Case: $\texttt{trans}(a_i)$ **outputs** $\texttt{insample}'$

1.4 Dealing with when the first/assignment guard is not $\texttt{true}$

We are given a coupling from the previous assignment transition between $x\langle 1\rangle$ and $x\langle 2\rangle$ and we would like to have the freedom to couple $x'\langle 1\rangle$ and $x'\langle 2\rangle$ (the value of x after the current assignment transition) using either coupling scheme. However, the guard of the assignment transition itself must be satisfied. We show that in 6 out of the 8 cases, it is possible to couple $x'\langle 1\rangle$ and $x'\langle 2\rangle$ how we'd like while still satisfying the assignment transition guard. In cases 3 and 6, such a coupling is not possible.

3 possible choices to analyze (18 cases total):

- Either the assignment guard is $\texttt{insample} < \texttt{x}$ or $\texttt{insample} \geq \texttt{x}$
- The previous coupling either had $x\langle 1\rangle + 1 = x\langle 2\rangle$ (S^L), $x\langle 1\rangle = x\langle 2\rangle + 1$ (S^G), or $x\langle 1\rangle = x\langle 2\rangle$ (S^N).
- The new coupling we want to construct either has $x'\langle 1\rangle + 1 = x'\langle 2\rangle$, $x'\langle 1\rangle = x'\langle 2\rangle + 1$, or $x'\langle 1\rangle = x'\langle 2\rangle$.

Recall that $\texttt{insample} = \texttt{in} + z$ for some Laplace noise z and that $x' = \texttt{insample}$.

Case 1: $\texttt{insample} < \texttt{x}$**,** $x\langle 1\rangle + 1 = x\langle 2\rangle$**,** $x'\langle 1\rangle + 1 = x'\langle 2\rangle$ **(**$S^L \to S^L$**)**

We want that $x\langle 1\rangle > \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle > \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle - 1$. Then

$$\begin{aligned} x\langle 1\rangle > \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle - 1 > \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle - 1 \\ &\implies x\langle 2\rangle > \texttt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$

Case 2: $\texttt{insample} < \texttt{x}$**,** $x\langle 1\rangle + 1 = x\langle 2\rangle$**,** $x'\langle 1\rangle = x'\langle 2\rangle + 1$**(**$S^L \to S^G$**)**

We want that $x\langle 1\rangle > \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle > \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle + 1$. Then

$$\begin{aligned} x\langle 1\rangle > \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle - 1 > \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle + 1 \\ &\implies x\langle 2\rangle > \texttt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$

Case 3: $\texttt{insample} < \texttt{x}$**,** $x\langle 1\rangle + 1 = x\langle 2\rangle$**,** $x'\langle 1\rangle = x'\langle 2\rangle$ **(**$S^L \to S^N$**)**

We want that $x\langle 1\rangle > \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle > \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle$. Then

$$\begin{aligned} x\langle 1\rangle > \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle > \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle + 1 \\ &\implies x\langle 2\rangle > \texttt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$

Case 4: $\texttt{insample} < \texttt{x}$**,** $x\langle 1\rangle = x\langle 2\rangle + 1$**,** $x'\langle 1\rangle + 1 = x'\langle 2\rangle$ **(**$S^G \to S^L$**)**

We want that $x\langle 1\rangle > \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle > \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle - 1$. Then

$$\begin{aligned} x\langle 1\rangle > \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle + 1 > \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle - 1 \\ &\implies x\langle 2\rangle > \texttt{in}\langle 2\rangle + z\langle 2\rangle - 2 \end{aligned}$$

which is bad – the scenario where both are “forced” is a leaking pair.

Case 5: $\texttt{insample} < \texttt{x}$**,** $x\langle 1\rangle = x\langle 2\rangle + 1$**,** $x'\langle 1\rangle = x'\langle 2\rangle + 1$ **(**$S^G \to S^G$**)**

We want that $x\langle 1\rangle > \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle > \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \mathtt{in}\langle 2\rangle - \mathtt{in}\langle 1\rangle + z\langle 2\rangle + 1$. Then

$$\begin{aligned} x\langle 1\rangle > \mathtt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle + 1 > \mathtt{in}\langle 1\rangle + \mathtt{in}\langle 2\rangle - \mathtt{in}\langle 1\rangle + z\langle 2\rangle + 1 \\ &\implies x\langle 2\rangle > \mathtt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$

Case 6: $\mathtt{insample} < \mathtt{x}$, $x\langle 1\rangle = x\langle 2\rangle + 1$, $x'\langle 1\rangle = x'\langle 2\rangle$ $(S^G \to S^N)$

We want that $x\langle 1\rangle > \mathtt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle > \mathtt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \mathtt{in}\langle 2\rangle - \mathtt{in}\langle 1\rangle + z\langle 2\rangle$. Then

$$\begin{aligned} x\langle 1\rangle > \mathtt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle + 1 > \mathtt{in}\langle 1\rangle + \mathtt{in}\langle 2\rangle - \mathtt{in}\langle 1\rangle + z\langle 2\rangle \\ &\implies x\langle 2\rangle + 1 > \mathtt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$

which is bad - this corresponds to a privacy violating path.

Case 7: $\mathtt{insample} < \mathtt{x}$, $x\langle 1\rangle = x\langle 2\rangle$, $x'\langle 1\rangle + 1 = x'\langle 2\rangle$ $(S^N \to S^L)$

We want that $x\langle 1\rangle > \mathtt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle > \mathtt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \mathtt{in}\langle 2\rangle - \mathtt{in}\langle 1\rangle + z\langle 2\rangle - 1$. Then

$$\begin{aligned} x\langle 1\rangle > \mathtt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle > \mathtt{in}\langle 1\rangle + \mathtt{in}\langle 2\rangle - \mathtt{in}\langle 1\rangle + z\langle 2\rangle - 1 \\ &\implies x\langle 2\rangle > \mathtt{in}\langle 2\rangle + z\langle 2\rangle - 1 \end{aligned}$$

which is bad - this again corresponds to a privacy violating path.

Case 8: $\mathtt{insample} < \mathtt{x}$, $x\langle 1\rangle = x\langle 2\rangle$, $x'\langle 1\rangle = x'\langle 2\rangle + 1$ $(S^N \to S^G)$

We want that $x\langle 1\rangle > \mathtt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle > \mathtt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \mathtt{in}\langle 2\rangle - \mathtt{in}\langle 1\rangle + z\langle 2\rangle + 1$. Then

$$\begin{aligned} x\langle 1\rangle > \mathtt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle > \mathtt{in}\langle 1\rangle + \mathtt{in}\langle 2\rangle - \mathtt{in}\langle 1\rangle + z\langle 2\rangle + 1 \\ &\implies x\langle 2\rangle > \mathtt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$

Case 9: $\mathtt{insample} < \mathtt{x}$, $x\langle 1\rangle = x\langle 2\rangle$, $x'\langle 1\rangle = x'\langle 2\rangle$ $(S^N \to S^N)$

We want that $x\langle 1\rangle > \mathtt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle > \mathtt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \mathtt{in}\langle 2\rangle - \mathtt{in}\langle 1\rangle + z\langle 2\rangle$. Then

$$\begin{aligned} x\langle 1\rangle > \mathtt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle > \mathtt{in}\langle 1\rangle + \mathtt{in}\langle 2\rangle - \mathtt{in}\langle 1\rangle + z\langle 2\rangle \\ &\implies x\langle 2\rangle > \mathtt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$

Case 10: $\mathtt{insample} \geq \mathtt{x}$, $x\langle 1\rangle + 1 = x\langle 2\rangle$, $x'\langle 1\rangle + 1 = x'\langle 2\rangle$ $(S^L \to S^L)$

We want that $x\langle 1\rangle \leq \mathtt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle \leq \mathtt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle - 1$. Then

$$\begin{aligned} x\langle 1\rangle \leq \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle - 1 \leq \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle - 1 \\ &\implies x\langle 2\rangle \leq \texttt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$

Case 11: $\texttt{insample} \geq \texttt{x}$, $x\langle 1\rangle + 1 = x\langle 2\rangle$, $x'\langle 1\rangle = x'\langle 2\rangle + 1$ $(S^L \to S^G)$

We want that $x\langle 1\rangle \leq \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle \leq \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle + 1$. Then

$$\begin{aligned} x\langle 1\rangle \leq \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle - 1 \leq \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle + 1 \\ &\implies x\langle 2\rangle \leq \texttt{in}\langle 2\rangle + z\langle 2\rangle + 2 \end{aligned}$$

which is bad - like case 4, this corresponds to a leaking pair.

Case 12: $\texttt{insample} \geq \texttt{x}$, $x\langle 1\rangle + 1 = x\langle 2\rangle$, $x'\langle 1\rangle = x'\langle 2\rangle$ $(S^L \to S^N)$

We want that $x\langle 1\rangle \leq \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle \leq \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle$. Then

$$\begin{aligned} x\langle 1\rangle \leq \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle - 1 \leq \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle \\ &\implies x\langle 2\rangle \leq \texttt{in}\langle 2\rangle + z\langle 2\rangle + 1 \end{aligned}$$

which is bad - this again corresponds to a privacy violating path.

Case 13: $\texttt{insample} \geq \texttt{x}$, $x\langle 1\rangle = x\langle 2\rangle + 1$, $x'\langle 1\rangle + 1 = x'\langle 2\rangle$ $(S^G \to S^L)$

We want that $x\langle 1\rangle \leq \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle \leq \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle - 1$. Then

$$\begin{aligned} x\langle 1\rangle \leq \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle + 1 \leq \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle - 1 \\ &\implies x\langle 2\rangle \leq \texttt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$

Case 14: $\texttt{insample} \geq \texttt{x}$, $x\langle 1\rangle = x\langle 2\rangle + 1$, $x'\langle 1\rangle = x'\langle 2\rangle + 1$ $(S^G \to S^G)$

We want that $x\langle 1\rangle \leq \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle \leq \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle + 1$. Then

$$\begin{aligned} x\langle 1\rangle \leq \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle + 1 \leq \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle + 1 \\ &\implies x\langle 2\rangle \leq \texttt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$

Case 15: $\texttt{insample} \geq \texttt{x}$, $x\langle 1\rangle = x\langle 2\rangle + 1$, $x'\langle 1\rangle = x'\langle 2\rangle$ $(S^G \to S^N)$

We want that $x\langle 1\rangle \leq \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle \leq \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle$. Then

$$\begin{aligned} x\langle 1\rangle \le \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle + 1 \le \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle \\ &\implies x\langle 2\rangle \le \texttt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$

Case 16: $\texttt{insample} \ge \texttt{x}$**,** $x\langle 1\rangle = x\langle 2\rangle$**,** $x'\langle 1\rangle + 1 = x'\langle 2\rangle$ $(S^N \to S^L)$

We want that $x\langle 1\rangle \le \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle \le \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle - 1$. Then

$$\begin{aligned} x\langle 1\rangle \le \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle \le \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle - 1 \\ &\implies x\langle 2\rangle \le \texttt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$

Case 17: $\texttt{insample} \ge \texttt{x}$**,** $x\langle 1\rangle = x\langle 2\rangle$**,** $x'\langle 1\rangle = x'\langle 2\rangle + 1$ $(S^N \to S^G)$

We want that $x\langle 1\rangle \le \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle \le \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle + 1$. Then

$$\begin{aligned} x\langle 1\rangle \le \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle \le \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle + 1 \\ &\implies x\langle 2\rangle \le \texttt{in}\langle 2\rangle + z\langle 2\rangle + 1 \end{aligned}$$

which is bad - this corresponds to a privacy violating path.

Case 18: $\texttt{insample} \ge \texttt{x}$**,** $x\langle 1\rangle = x\langle 2\rangle$**,** $x'\langle 1\rangle = x'\langle 2\rangle$ $(S^N \to S^N)$

We want that $x\langle 1\rangle \le \texttt{in}\langle 1\rangle + z\langle 1\rangle \implies x\langle 2\rangle \le \texttt{in}\langle 2\rangle + z\langle 2\rangle$

We must couple $z\langle 1\rangle = \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle$. Then

$$\begin{aligned} x\langle 1\rangle \le \texttt{in}\langle 1\rangle + z\langle 1\rangle &\implies x\langle 2\rangle \le \texttt{in}\langle 1\rangle + \texttt{in}\langle 2\rangle - \texttt{in}\langle 1\rangle + z\langle 2\rangle \\ &\implies x\langle 2\rangle \le \texttt{in}\langle 2\rangle + z\langle 2\rangle \end{aligned}$$