APPENDIX A
EFFICIENT IMPLEMENTATION OF THE STOCHASTIC π-CALCULUS CONSTRUCTION

From Definition 2, one can notice that redundant communication absorptions may be initiated between two parallel processes, consuming non-necessary resources, especially when high stochasticity absorptions are used. This issue can be fixed by attaching each process a unique identifier: when a process initiates a communication absorption, it additionally sends its id; the inputting process ignores the initiation if the id has already been received. We point out that this improvement does not add complexity to the construction.

Given a process definition, a channel id is added as parameter. For each choice i being an input or output, a list of ids ki is added to its parameters tuple ρ. Each ki will contain the process ids that have already initiated a communication absorption with the process. To each race condition, we add the capability of inputting on a status of the channel absorption, it additionally sends its id; the inputting process ignores the initiation if the id has already been received. When a process is replaced by another, it additionally sends its id (already present in the list, at position n). This is formally defined by the following expression, replacing (5).

\[ [A(\bar{z})]_e = \sum_{i \in I} C_i \]
\[ A(\bar{z}, id, \bar{\rho}) = \sum_{i \in I} [C_i]_e \]
\[ + id(pid, a', s).([s = 0] A(\bar{z}, id, \bar{\rho} \setminus \bar{\rho}_j)) \]
\[ \| [s \neq 0] A(\bar{z}, id, \bar{\rho}\{k_{j_n} = \text{pid}\}) ) \]

with \( \bar{\rho} \cap n(A(\bar{z})) = \emptyset \), and \( a'_{j_n} = a', \bar{\rho} \setminus \bar{\rho}_j \) stands for removing the j\textsuperscript{th} element of each list.

When a process is replaced by another, it additionally outputs its id with every private channels a' and the status 0 on every process id present in its k lists: every communication absorption with the process has to be stopped. This statement is achieved by replacing the occurrence of \([P]_e\) by the expression (21) in (6), (7), and (8). In the same manner, the calls to \( A_i(\bar{z}, id, \bar{\rho}) \) are replaced by \( A_i(\bar{z}, id, \bar{\rho}) \).

\[ ([P]_e | \prod_{i \in I} \prod_{n=1}^{\#k_i} (id, a'_{i_n}, 0)) \]

where \( \prod_{j=1}^{n} P_j = P_1 \ | \ . \ | \ P_n \).

To initiate a communication absorption, the outputting process additionally sends its id \( \bar{a}'(a', a''', id) \) becomes \( \bar{a}'(a', a''', id) \) in (7). The inputting process checks then if a communication with the received id pid has already been accepted. If pid ∈ k, a communication has already been accepted, so the process informs the sender that the new communication absorption has to be stopped (this is done by outputting both the channel a' and status 0 on pid). If pid ∉ k, a new communication absorption is instantiated, and the process lets the sender know its id.