$$\frac{\begin{array}{c} S_1, v = S.pop() \qquad v' = op(v) \qquad S' = S_1.push(v') \qquad ins = B[pc+1] \\ \tau_{S_1}, t = \tau_S.pop() \quad t' = P_{UNOP}(t) \quad \tau'_S = \tau_{S_1}.push(t') \end{array}}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, op \rightsquigarrow \tau_{C,} \tau'_S, \tau_{M,} \tau_{\Delta,} B, C, pc+1, S', M, \Delta, ins} \; \text{T-UNOP}$$

$$\frac{\begin{array}{c} S_1, v_1 = S.pop() \qquad S_2, v_2 = S_1.pop() \qquad v' = op(v_1, v_2) \qquad S' = S_2.push(v') \qquad ins = B[pc+1] \\ \tau_{S_1}, t_1 = \tau_{S.} pop() \quad \tau_{S_2}, t_2 = \tau_{S_1}.pop() \quad t' = P_{BINOP}(t_1, t_2) \quad \tau'_S = \tau_{S_2}.push(t') \end{array}}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, op \rightsquigarrow \tau_{C,} \tau'_S, \tau_{M,} \tau_{\Delta,} B, C, pc+1, S', M, \Delta, ins} \; \text{T-BINOP}$$

$$\frac{\begin{array}{c} S_1, v_1 = S.pop() \qquad S_2, v_2 = S_1.pop() \qquad S_3, v_3 = S_2.pop() \quad v' = op(v_1, v_2, v_3) \\ S' = S_3.push(v') \qquad ins = B[pc+1] \qquad \tau_{S_1}, t_1 = \tau_{S.} pop() \quad \tau_{S_2}, t_2 = \tau_{S_1}.pop() \\ \tau_{S_3}, t_3 = \tau_{S_2}.pop() \quad t' = P_{TERNOP}(t_1, t_2, t_3) \quad \tau'_S = \tau_{S_3}.push(t') \end{array}}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, op \rightsquigarrow \tau_{C,} \tau'_S, \tau_{M,} \tau_{\Delta,} B, C, pc+1, S', M, \Delta, ins} \; \text{T-TERNOP}$$

$$\frac{S', v = S.pop() \quad ins = B[pc+1] \quad \tau'_S, t = \tau_{S.} pop()}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, \rightsquigarrow \tau_{C,} \tau'_S, \tau_{M,} \tau_{\Delta,} B, C, pc+1, S', M, \Delta, ins} \; \text{T-POP}$$

$$\frac{S' = S.push(const) \quad ins = B[pc+1] \quad t = P_{CONST} \quad \tau'_S = \tau_{S.} push(t)}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, push(v) \rightsquigarrow \tau_{C,} \tau'_S, \tau_{M,} \tau_{\Delta,} B, C, pc+1, S', M, \Delta, ins} \; \text{T-PUSH}$$

$$\frac{\begin{array}{c} S_1, i = S.pop() \qquad v = M[i] \qquad S' = S_1.push(v) \qquad ins = B[pc+1] \\ \tau_{S_1}, t_i = \tau_{S.} pop() \quad t_v = \tau_M[i] \quad t' = P_{MEM}(t_i, t_v) \quad \tau'_S = \tau_{S_1}.push(t') \end{array}}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, mload \rightsquigarrow \tau_{C,} \tau'_S, \tau_{M,} \tau_{\Delta,} B, C, pc+1, S', M, \Delta, ins} \; \text{T-MLOAD}$$

$$\frac{\begin{array}{c} S_1, i = S.pop() \qquad S', v = S_1.pop() \qquad M' = M[i \leftarrow v] \qquad ins = B[pc+1] \\ \tau_{S_1}, t_i = \tau_{S.} pop() \quad \tau'_S, t_v = \tau_{S_1}.pop() \quad t' = P_{MEM}(t_i, t_v) \quad \tau'_M = \tau_M[i \leftarrow t'] \end{array}}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, mstore \rightsquigarrow \tau_{C,} \tau'_S, \tau'_M, \tau_{\Delta,} B, C, pc+1, S', M', \Delta, ins} \; \text{T-MSTORE}$$

$$\frac{\begin{array}{c} S_1, i = S.pop() \qquad v = \Delta[i] \qquad S' = S_1.push(v) \qquad ins = B[pc+1] \\ \tau_{S_1}, t_i = \tau_{S.} pop() \quad t_v = \tau_\Delta[i] \quad t' = P_{STORE}(t_i, t_v) \quad \tau'_S = \tau_{S_1}.push(t') \end{array}}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, sload \rightsquigarrow \tau_{C,} \tau'_S, \tau_{M,} \tau_{\Delta,} B, C, pc+1, S', M, \Delta, ins} \; \text{T-SLOAD}$$

$$\frac{\begin{array}{c} S_1, i = S.pop() \qquad S', v = S_1.pop() \qquad \Delta' = \Delta[i \leftarrow v] \qquad ins = B[pc+1] \\ \tau_{S_1}, t_i = \tau_{S.} pop() \quad \tau'_S, t_v = \tau_{S_1}.pop() \quad t' = P_{STORE}(t_i, t_v) \quad \tau'_\Delta = \tau_\Delta[i \leftarrow t'] \end{array}}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, sstore \rightsquigarrow \tau_{C,} \tau'_S, \tau_{M,} \tau'_\Delta, B, C, pc+1, S', M, \Delta', ins} \; \text{T-SSTORE}$$

$$\frac{\begin{array}{c} v_1 = S.get(0) \quad v_2 = S.get(i) \quad S_1 = S[i \leftarrow v_1] \quad S' = S_1[0 \leftarrow v_2] \quad ins = B[pc+1] \\ t_1 = \tau_S.get(0) \quad t_2 = \tau_S.get(i) \quad \tau_{S_1} = \tau_S[i \leftarrow t_1] \quad \tau'_S = \tau_{S_1}[0 \leftarrow t_2] \end{array}}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, swap(i) \rightsquigarrow \tau_{C,} \tau'_S, \tau_{M,} \tau_{\Delta,} B, C, pc+1, S', M, \Delta, ins} \; \text{T-SWAPI}$$

$$\frac{v = S.get(i-1) \quad S' = S.push(v) \quad ins = B[pc+1] \quad t = \tau_{S.} get(i-1) \quad \tau'_S = \tau_{S.} push(t)}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, dup(i) \rightsquigarrow \tau_{C,} \tau'_S, \tau_{M,} \tau_{\Delta,} B, C, pc+1, S', M, \Delta, ins} \; \text{T-DUPI}$$

$$\frac{\begin{array}{c} S_1, i = S.pop() \qquad v = C[i] \qquad S' = S_1.push(v) \qquad ins = B[pc+1] \\ \tau_{S_1}, t_i = \tau_{S.} pop() \quad t_v = \tau_C[i] \quad t' = P_{CALLDATA}(t_i, t_v) \quad \tau'_S = \tau_{S_1}.push(t') \end{array}}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, calldataload \rightsquigarrow \tau_{C,} \tau'_S, \tau_{M,} \tau_{\Delta,} B, pc+1, S', M, \Delta, ins} \; \text{T-CALLDATALOAD}$$

$$\frac{\begin{array}{c} S_1, d = S.pop() \qquad S_2, s = S_1.pop() \qquad S', l = S_2.pop() \qquad v = C[s...s+l] \\ M' = M[d...d+l \leftarrow v] \qquad ins = B[pc+1] \qquad \tau_{S_1}, t_d = \tau_{S.} pop() \qquad \tau_{S_2}, t_s = \tau_{S_1}.pop() \\ \tau'_S, t_l = \tau_{S_2}.pop() \qquad t' = P_{CALLDATA}(t_s, t_v) \quad \tau'_M = \tau_{M_[} d...d+l \leftarrow t'] \end{array}}{\tau_C, \tau_S, \tau_M, \tau_\Delta, B, C, pc, S, M, \Delta, calldataload \rightsquigarrow \tau_{C,} \tau'_S, \tau'_M, \tau_{\Delta,} B, pc+1, S', M', \Delta, ins} \; \text{T-CALLDATACOPY}$$

Figure 1: Operational semantics of EVM assembly with taint policy P.