**Algorithm 9.7 Sum-product variable elimination for sets of rules**

**Procedure** Rule-Sum-Product-Eliminate-Var (
  $\mathcal{R}$, // Set of rules
  $Y$ // Variable to be eliminated
)
1 $\mathcal{R}^+ \leftarrow \{\rho \in \mathcal{R} : \mathit{Scope}[\rho] \ni Y\}$
2 $\mathcal{R}^- \leftarrow \mathcal{R} - \mathcal{R}^+$
3 **while** $\mathcal{R}^+ \neq \emptyset$
4 Apply one of the following actions, when applicable
5 **Rule sum:**
6 Select $\mathcal{R}_{\boldsymbol{c}} \subseteq \mathcal{R}^+$ such that
7 $\mathcal{R}_{\boldsymbol{c}} = \{\langle \boldsymbol{c}, Y = y^1; p_1\rangle, \ldots, \langle \boldsymbol{c}, Y = y^k; p_k\rangle\}$
8 no other $\rho \in \mathcal{R}^+$ is compatible with $\boldsymbol{c}$)
9 $\mathcal{R}^- \leftarrow \mathcal{R}^- \cup \sum_Y \mathcal{R}_{\boldsymbol{c}}$
10 $\mathcal{R}^+ \leftarrow \mathcal{R}^+ - \mathcal{R}_{\boldsymbol{c}}$
11 **Rule product:**
12 Select $\langle \boldsymbol{c}; p_1\rangle, \langle \boldsymbol{c}; p_2\rangle \in \mathcal{R}^+$
13 $\mathcal{R}^+ \leftarrow \mathcal{R}^+ - \{\langle \boldsymbol{c}; p_1\rangle, \langle \boldsymbol{c}; p_2\rangle\} \cup \{\langle \boldsymbol{c}; p_1 \cdot p_2\rangle\}$
14 **Rule splitting for rule product:**
15 Select $\rho_1, \rho_2 \in \mathcal{R}^+$ such that
16 $\rho_1 = \langle \boldsymbol{c}_1; p_1\rangle$
17 $\rho_2 = \langle \boldsymbol{c}_2; p_2\rangle$
18 $\boldsymbol{c}_1 \sim \boldsymbol{c}_2$
19 $\mathcal{R}^+ \leftarrow \mathcal{R}^+ - \{\rho_1, \rho_2\} \cup \text{Rule-Split}(\rho_1, \boldsymbol{c}_2) \cup \text{Rule-Split}(\rho_2, \boldsymbol{c}_1)$
20 **Rule splitting for rule sum:**
21 Select $\rho_1, \rho_2 \in \mathcal{R}^+$ such that
22 $\rho_1 = \langle \boldsymbol{c}_1, Y = y^i; p_1\rangle$
23 $\rho_2 = \langle \boldsymbol{c}_2, Y = y^j; p_2\rangle$
24 $\boldsymbol{c}_1 \sim \boldsymbol{c}_2$
25 $i \neq j$
26 $\mathcal{R}^+ \leftarrow \mathcal{R}^+ - \{\rho_1, \rho_2\} \cup \text{Rule-Split}(\rho_1, \boldsymbol{c}_2) \cup \text{Rule-Split}(\rho_2, \boldsymbol{c}_1)$
27 **return** $\mathcal{R}^-$

A different way of understanding the algorithm is to consider its application to rule sets that originate from standard table-CPDs. It is not difficult to verify that the algorithm performs exactly the same set of operations as standard variable elimination. For example, the standard operation of factor product is simply the application of rule splitting on all of the rules that constitute the two tables, followed by a sequence of rule product operations on the resulting rule pairs. (See exercise 9.16.)

To prove that the algorithm computes the correct result, we need to show that each operation performed in the context of the algorithm maintains a certain correctness invariant. Let $\mathcal{R}$ be the current set of rules maintained by the algorithm, and $\boldsymbol{W}$ be the variables that have not yet been eliminated. Each operation must maintain the following condition: