**Algorithm 1** Algorithm to determine optimal parallelization strategy for an operator DAG

**procedure** $\textsc{FindOptStrategy}(G = (V, E),\ \mathit{nprocs})$
    **for all** $v \in V$ **do**
        $v.\mathit{neighbors} \leftarrow \{u : (u, v) \in E \vee (v, u) \in E\}$
        $v.\mathit{processed} \leftarrow \mathsf{False}$

        ▷ $v.\mathit{dim}$: Itertion domain dimension of the operation in $v$
        $v.\mathit{configs} \leftarrow \{(c_1, c_2, \ldots, c_{v.dim}) : \forall_{i \in [1, v.dim]} c_i \in \mathbb{Z}^* \wedge \prod_{i \in [1, v.dim]} c_i \le \mathit{nprocs}\}$
    **end for**

    $\mathit{tbl} \leftarrow \emptyset$
    $S \leftarrow \{v : v \in V \wedge v.\mathit{processed} = \mathsf{False}\}$ ▷ Set of unprocessed vertices
    **while** $S \neq \emptyset$ **do**
        ▷ Choose a vertex with least no. of unprocessed neighbors from $S$
        **for all** $v \in S$ **do**
            $v.\mathit{unprocessed\_neighbors} \leftarrow \{u : u, v \in S \wedge u \in v.\mathit{neighbors}\}$
        **end for**
        $v_{min} \leftarrow \arg\min_{v \in S} |v.\mathit{unprocessed\_neighbors}|$

        ▷ Each element in $C$ is a set of pairs $\{(u_1, cfg_1), \ldots, (u_n, cfg_n)\}$, where $u_i$ is an unprocessed neighbor of $v_{min}$ and $cfg_i$ is a valid configuration of $u_i$
        $C \leftarrow \prod_{u \in v_{min}.\mathit{unprocessed\_neighbors}} \{(u, cfg) : cfg \in u.\mathit{configs}\}$

        ▷ For each combination of configuration of unprocessed neighbors of $v_{min}$, find a config of $v_{min}$ that minimizes the cost
        **for all** $c \in C$ **do**
            $c_{min} \leftarrow \arg\min_{c_v \in v_{min}.configs}(\textsc{Cost}(v_{min}, c_v, c))$ ▷ Refer Alg. 2
            $\mathit{tbl} \leftarrow \mathit{tbl} \cup \{(v_{min}, c, c_{min})\}$
        **end for**

        $v_{min}.\mathit{processed} \leftarrow \mathsf{True}$
        $S \leftarrow S \setminus \{v_{min}\}$
    **end while**

**return** $\textsc{OptConfig}(G, \mathit{tbl})$ ▷ Refer Alg. 3
**end procedure**

**Algorithm 2** Algorithm that returns the cost when configuration $c$ is assigned to a vertex $v$, and its unprocessed neighbors have configurations $U_c$

**procedure** COST($v, c, U_c$)
**end procedure**

**Algorithm 3** Algorithm that extracts an optimal configuration from the dynamic programming table generated in Alg. 1

**procedure** OPTCONFIG($G, tbl$)
**end procedure**