1 **Input:** V, the variable whose gradient should be added to G and *grad_table* should be calculated

2 **Input:** $\mathcal{G}$, the computational graph to modify

3 **Input:** $\mathcal{G}'$, the restriction of $\mathcal{G}$ to nodes that participate in the gradient

4 **Input:** *grad_table*, a data structure mapping nodes to their gradients

5 **if** $V \in grad\_table$ **then**

6   **Return** $grad\_table[V]$

7 $i \leftarrow 1$

8 $children \leftarrow get\_consumers(V, \mathcal{G}')$

9 **foreach** $C \in children$ **do**

10      $op \leftarrow get\_operation(C)$

11      $D \leftarrow build\_grad(C, \mathcal{G}, \mathcal{G}', grad\_table)$

12      $parents \leftarrow get\_inputs(C, \mathcal{G}')$

13      $G^{(i)} \leftarrow op.bprop(parents, V, D)$

14      $i \leftarrow i + 1$

15 $G \leftarrow \sum_i G^i$

16 $grad\_table[V] \leftarrow G$

17 insert G and the operations creating it into $\mathcal{G}$

18 **Return** G