Algorithm 1 Boruvka
Entrada: $G = (V, A); C$
Salida: $G^T = (V^T, A^T)$

$A^T \leftarrow \emptyset$
$M \leftarrow V$
while $|A^T| < (|V| - 1)$ **do**
    $S \leftarrow \emptyset$
    for cada componente $k \in M$ **do**
        $S' \leftarrow \emptyset$
        for cada nodo $i \in V$ tal que $M_i = k$ **do**
            seleccionar un arco $(i, j) \in A$ con $j \in V$ tal que $M_j \neq k$ cuyo c_{ij} sea mínimo
            if existe el arco (i, j) **then**
                $S' \leftarrow S' \cup \{(i, j)\}$
            end if
        end for
        seleccionar arco $(i, j) \in S'$ cuyo c_{ij} sea mínimo
        $S \leftarrow S \cup \{(i, j)\}$
    end for
    for cad arco $(i, j) \in S$ **do**
        $A^T \leftarrow A^T \cup \{(i, j)\}$
        $M_l \leftarrow M_i$ para todo $l \in \{1, ..., |M|\}$ tal que $M_l = M_j$
    end for
end while
$V^T \leftarrow V$

Algorithm 2 Dijkstra
Entrada: $G = (V, A); D, s \in V$
Salida: $G^T = (V^T, A^T); W$

$A^T \leftarrow \emptyset$
$V^T \leftarrow \{s\}$
$W_{1,|V|} \leftarrow 0$
while $|V^T| < |V|$ **do**
    seleccionar arcos $(i, j) \in A$ con $i \in V^T$ y $j \in VV^T$
    elegir un arco (i, j) de los anteriores tal que $d_{ij} + W_i$ sea mínimo
    $A^T \leftarrow A^T \cup \{(i, j)\}$
    $V^T \leftarrow V^T \cup \{j\}$
    $W_j \leftarrow d_{ij} + W_j$
end while

Algorithm 3 Bellman-Ford
Entrada: $G = (V, A); D; s \in V$ **Salida:** $G^T = (V^T, A^T); W$

$A^T \leftarrow \emptyset$
$W_{1:|V|} \leftarrow \infty; W_s \leftarrow 0$
$P \leftarrow \emptyset$
for cada nodo $i \in V\{s\}$ **do**
    for cada arco $(i, j) \in A$ **do**
        if $W_j > W_i + d_{ij}$ **then**
            $W_j \leftarrow W_i + d_{ij}$
            $P_j \leftarrow i$
        end if
    end for
end for
for cada arco $(i, j) \in A$ **do**
    if $W_j > W_i + d_{ij}$ **then**
        No hay solución
    end if
end for
for cada nodo $j \in V\{s\}$ **do**
    $i \leftarrow P_j$
    seleccionar arco (i, j) de A
    $A^T \leftarrow A^T \cup \{(i, j)\}$
end for
$V^T \leftarrow V$

Algorithm 4 Kruskal
Entrada: $G = (V, A); C$
Salida: $G^T = (V^T, A^T)$

$A^T \leftarrow \emptyset$
$M \leftarrow V$
while $|A^T| < (|V| - 1)$ **do**
    seleccionar un arco $(i, j) \in A$ cuyo c_{ij} sea mínimo
    if $M_i \neq M_j$ **then**
        $A^T \leftarrow A^T \cup \{(i, j)\}$
        $M_k \leftarrow M_i$ para todo $k \in \{1, ..., |M|\}$ tal que $M_k = M_j$
    end if
    $A \leftarrow A \backslash \{(i, j)\}$
end while
$V^T \leftarrow V$

Algorithm 5 Prim
Entrada: $G = (V, A); C; s \in V$
Salida: $G^T = (V^T, A^T)$

$A^T \leftarrow \emptyset$
$V^T \leftarrow \{s\}$
while $|V^T| < |V|$ **do**
    seleccionar un arco $(i, j) \in A$ con $i \in V^T$ y $j \in VV^T$ cuyo c_{ij} sea mínimo
    $A^T \leftarrow A^T \cup \{(i, j)\}$
    $V^T \leftarrow V^T \cup \{(j)\}$
end while

Algorithm 6 Edmonds
Entrada: $G = (V, A); C; s \in V$
Salida: $G^{T'} = (V^{T'}, A^{T'})$

1: $A^{T'} \leftarrow \emptyset$
2: $k \leftarrow 1; V_k \leftarrow V; A_k \leftarrow A; C_k \leftarrow C$
3: **while** $A^{T'} = \emptyset$ **do**
4: **for** cada nodo $j \in V\{s\}$ **do**
5: ente los arcos de A_k incidentes en j seleccionar el coste c_{kij} mínimo
6: restar coste c_{kij} a todos los arcos de A_k que inciden en j
7: **end for**
8: seleccionar arcos de A_k con coste 0
9: **if** forman una arborescencia de expansión **then**
10: guardar arborescencia en A^T
11: **else if** seleccionar arcos de A_k con coste 0 que formen un ciclo **then**
12: fusionar nodos y arcos del cilo en un supernodo
13: $k \leftarrow k + 1; V_k \leftarrow V_{k-1}; A_k \leftarrow A_{k-1}; C_k \leftarrow C_{k-1}$
14: **end if**
15: **end while**
16: **while** $k > 1$ **do**
17: deshacer supernodo de la etapa k y recuperar grafo, costes y ciclo de la etapa $k - 1$
18: seleccionar arcos del grafo en la etapa $k - 1$ que se correspondan con arcos en $A^{T'}$
19: seleccionar arcos del ciclo en la etapa $k - 1$ excepto el que incide en un nodo ya alcanzado
20: guardar arcos seleccionados en una nueva arborescencia en $A^{T'}$
21: $k \leftarrow k - 1$
22: **end while**
23: $V^{T'} \leftarrow V$

Algorithm 7 Gusfield
Entrada: $G = (V, A); Z$
Salida: $G^{T'} = (V^{T'}, A^{T'}); Z^T$

1: $V^T \leftarrow \{1\}$
2: $A^T \leftarrow \emptyset$
3: $Z^T_{|V|X|V|} \leftarrow \infty$
4: **for** cada nodo $i \in V1$ **do**
5: $V^T_k \leftarrow V^T; A^T_k \leftarrow A^T$
6: **while** $|V^T_k| > 1$ **do**
7: borrar un arco $(i, j) \in A^T_k$ cuyo $a_{i,j}$ sea mínimo
8: determinar componentes T_1 y T_2 de V^T_k y A^T_k tal que $i \in T_1$ y $j \in T_2$
9: obtener corte mínimo entre i y j en el grafo $G = (V, A)$ original
10: determinar componentes S_1 y S_2 de V^T y A^T tal que $i \in S_1$ y $j \in S_2$
11: **if** nodo $i \in S_1$ **then**
12: $V^T_k \leftarrow$ nodos de la componente T_1
13: $A^T_k \leftarrow$ arcos de la componente T_1
14: **else**
15: $V^T_k \leftarrow$ nodos de la componente T_2
16: $A^T_k \leftarrow$ arcos de la componente T_2
17: **end if**
18: **end while**
19: añadir arco (i, k), con $k \in V^T_k$, al árbol A^T
20: obtener corte mínimo entre i y k en el grafo $G = (V, A)$ original
21: $Z^T_{ik} \leftarrow$ capacidad del corte mínimo $i - k$
22: $V^T \leftarrow V^T \cup \{i\}$
23: **end for**