Algorithm 1: Edmonds-MaxWeightedMatching

```
Data: G = (V,E) un grafo, función de pesos w : E → R_{≥0}
Result: Emparejamiento M ⊆ E de peso máximo
Inicialización:
foreach v ∈ V do
    y(v) ← máx{w(e) : e incidente en v};
M ← ∅;
while true do
    // Construir el subgrafo de igualdad
    E* ← {(u, v) ∈ E : y(u) + y(v) = w(u, v)};
    Definir G* = (V, E*);
    P ← FindAugmentingPath(G*, M);
    if P ≠ NIL then
        M ← M ⊕ P;
    else
        // Calcular la cantidad mínima de ajuste dual
        δ ← mín{ y(u) + y(v) − w(u, v) : u ∈ T_even, v ∉ T };
        if δ = ∞ then
            return M;
        foreach v en el árbol T (formado durante la búsqueda) do
            if v está etiquetado EVEN then
                y(v) ← y(v) − δ;
            else
                y(v) ← y(v) + δ;
```

Algorithm 2: FindAugmentingPath(G^*, M)

```
Data: G* = (V, E*): subgrafo de igualdad; M: emparejamiento actual
Result: Un camino aumentante P o NIL si no existe
foreach v ∈ V do
    label(v) ← UNLABELED,    parent(v) ← NIL;
foreach vértice libre v (no saturado por M) do
    label(v) ← EVEN;
    Encolar v en la cola Q;
while Q no está vacía do
    u ← Q.dequeue();
    foreach v adyacente a u en G* do
        if v está UNLABELED then
            label(v) ← ODD;
            parent(v) ← u; if v está emparejado (existe w tal que
             (v, w) ∈ M) then
                label(w) ← EVEN;
                parent(w) ← v; Encolar w en Q;
            else
                return ReconstructPath(v, parent);
        else if v ya está etiquetado EVEN y pertenece a un árbol
         distinto al de u then
            return CombinePaths(ReconstructPath(u, parent),
             ReconstructPath(v, parent));
        else if v ya está etiquetado EVEN y pertenece al mismo árbol
         que u then
            // Se detecta un blossom
            B ← DetectBlossom(u, v, parent);
            Contraer B en G* y actualizar M, las etiquetas y la
             estructura del árbol;
            Continuar la búsqueda en el grafo contraído;
return NIL;
```

Notas adicionales:

- La operación $M \oplus P$ denota que se alternan las aristas del camino P (se añaden las que no están en M y se quitan las que ya están en M).
- Las subrutinas ReconstructPath, DetectBlossom, Contract y ExpandBlossom se encargan de reconstruir el camino aumentante, identificar y contraer los blossoms y posteriormente expandirlos para obtener el emparejamiento en el grafo original.
- El conjunto T_{even} corresponde a los vértices etiquetados como EVEN en el árbol alternante construido durante la búsqueda.

- El ajuste dual (cálculo de δ) "relaja" algunas aristas para que nuevas se incluyan en el subgrafo de igualdad.