Sin pérdida de generalidad, se puede suponer que todas las aristas de G tienen un peso diferente. En efecto, alcanza con extender $c(\cdot)$ a una función de pesos que $c'(\cdot)$ tal que $c'(v) = (c(v), v)$, donde v es el identificador del vértice (i.e., un numero en $[1, n] \cap \mathbb{N}$). Bajo la hipótesis de que todos los pesos son distintos, el algoritmo que consiste en insertar todas las aristas candidatas posibles a F en cada iteración computa un árbol generador mínimo por el inciso a). Este algoritmo fue propuesto por Borůvka en 1926, mucho antes de que Prim y Krukal propusieran los suyos.

c) Describir una implementación simple del algoritmo de Borůvka que requiera $O(m \log n)$ tiempo cuando un grafo G con n vértices y m aristas es dado.

El algoritmo comienza examinando cada vértice y añadiendo el arco de menor peso desde ese vértice a otro en el grafo, sin tener en cuenta los arcos ya agregados, y continua uniendo estos grupos de la misma manera hasta que se completa un árbol que cubra todos los vértices. Si denominamos a cada vértice o conjunto de vértices conectados como «componente», el pseudocódigo del Algoritmo de Boruvka es:

- Comenzar con un grafo conexo *G* en el que todos sus arcos tengan distinto peso, y un conjunto vacío de arcos *T*.
- Mientras los vértices de *G* conectados por *T* sean disjuntos: → en c/iteración mergeo de a 2 componentes → $\log n$ iteraciones
  - Crear un conjunto vacío de arcos *S*
  - Para cada componente: $O(m)$
    - Crear un conjunto vacío de arcos *S*
    - Para cada vértice v en el componente:
      - Agregar el arco de menor peso desde el vértice v a otro vértice en un componente disjunto a *S*
    - Añadir el arco de menor peso en *S* a *E*
  - Añadir el conjunto resultante *E* a *T*.
- El conjunto de aristas resultante *T* es el árbol de expansión mínimo de *G*.

Es como un Kruskal con prioridad → ventaja de pesos todos distintos → no tiene que desempatar