

UNIVERSIDAD DE LOS ANDES
FACULTAD DE INGENIERIA
POSTGRADO DE COMPUTACION

ORDENAMIENTO TOPOLOGICO ALGORITMO DE PRIM Y KRUSKAL

Ing. Helena Ospino Mantilla

Agenda

- ❖ Concepto de ordenamiento topológico
 - ❖ Algoritmo y ejemplo
 - ❖ Árboles de expansión mínima
 - ❖ Algoritmo de kruskal y ejemplo
 - ❖ Algoritmo de Prim y ejemplo
-

ORDENAMIENTO TOPOLOGICO

Un grafo G dirigido y sin ciclo se denomina grafo dirigido acíclico. Una ordenación topológica T de G es una ordenación lineal de los vertices de G que preserva la ordenación parcial.

ORDENAMIENTO TOPOLOGICO

Ordenación de los vértices de un grafo dirigido acíclico. (v, w) donde v esta antes que w .

Un algoritmo simple sería:

- ❖ Buscar un vértice con grado de incidencia de entrada=0.
- ❖ Se elimina este vértice con sus aristas.
- ❖ Se sigue aplicando esta estrategia al resto del grafo.
- ❖ Usa el recorrido de árbol primero en profundidad.

Sirve para la planificación de actividades.

ORDENAMIENTO TOPOLOGICO

Ordenamiento topológico (G)

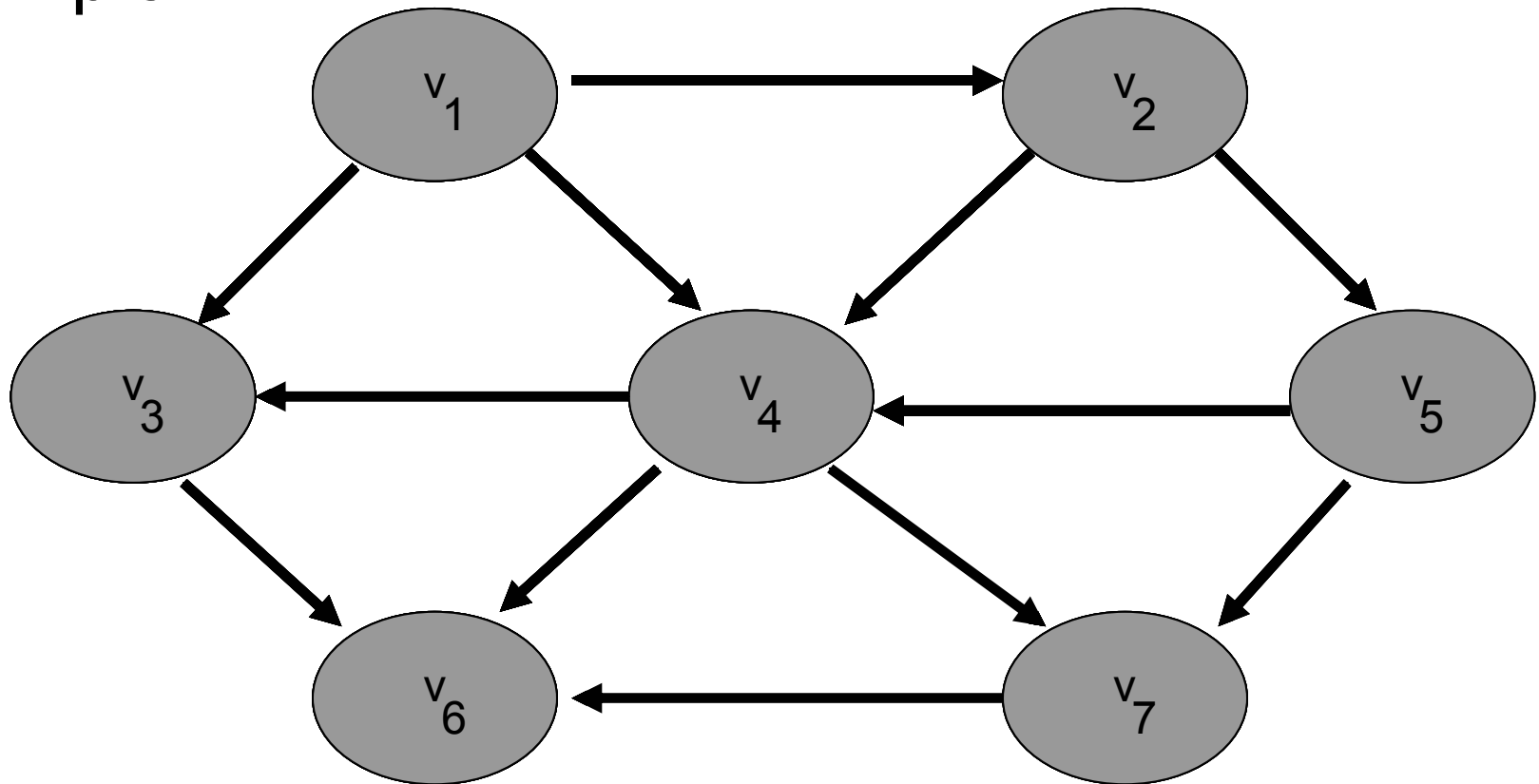
Llama búsqueda primero en profundidad para finalizar tiempos de computo $f(v)$ para cada vértice v , como cada vértice es finalizado, insértelo en la lista enlazada

Retorne la lista enlazada de vértices.

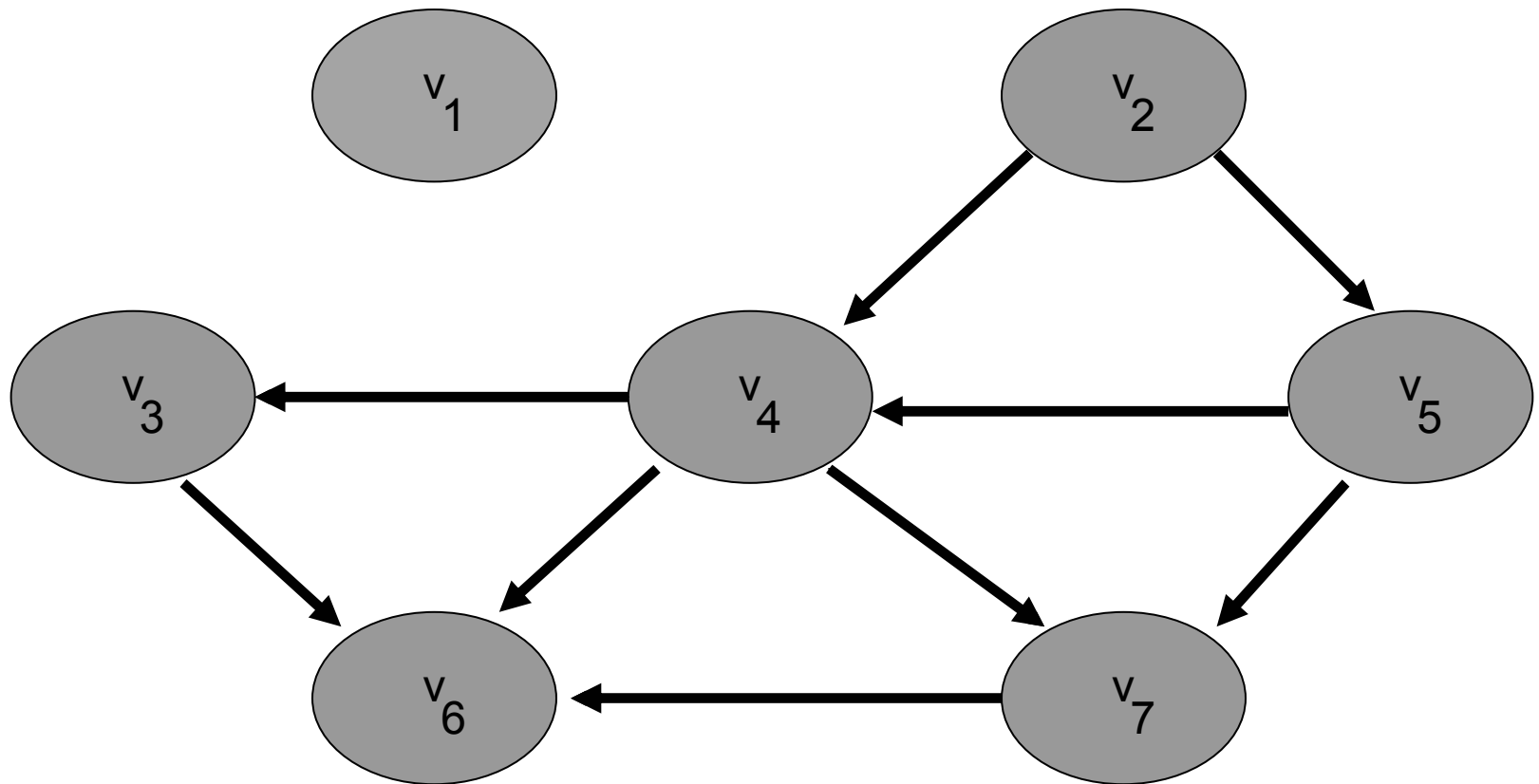
Es del orden $\Theta(V + E)$, dado que la búsqueda en profundidad toma $\Theta(V + E)$, el tiempo $O(1)$ es el tiempo de insertar cada vértice $|V|$ en la lista enlazada.

ORDENAMIENTO TOPOLOGICO

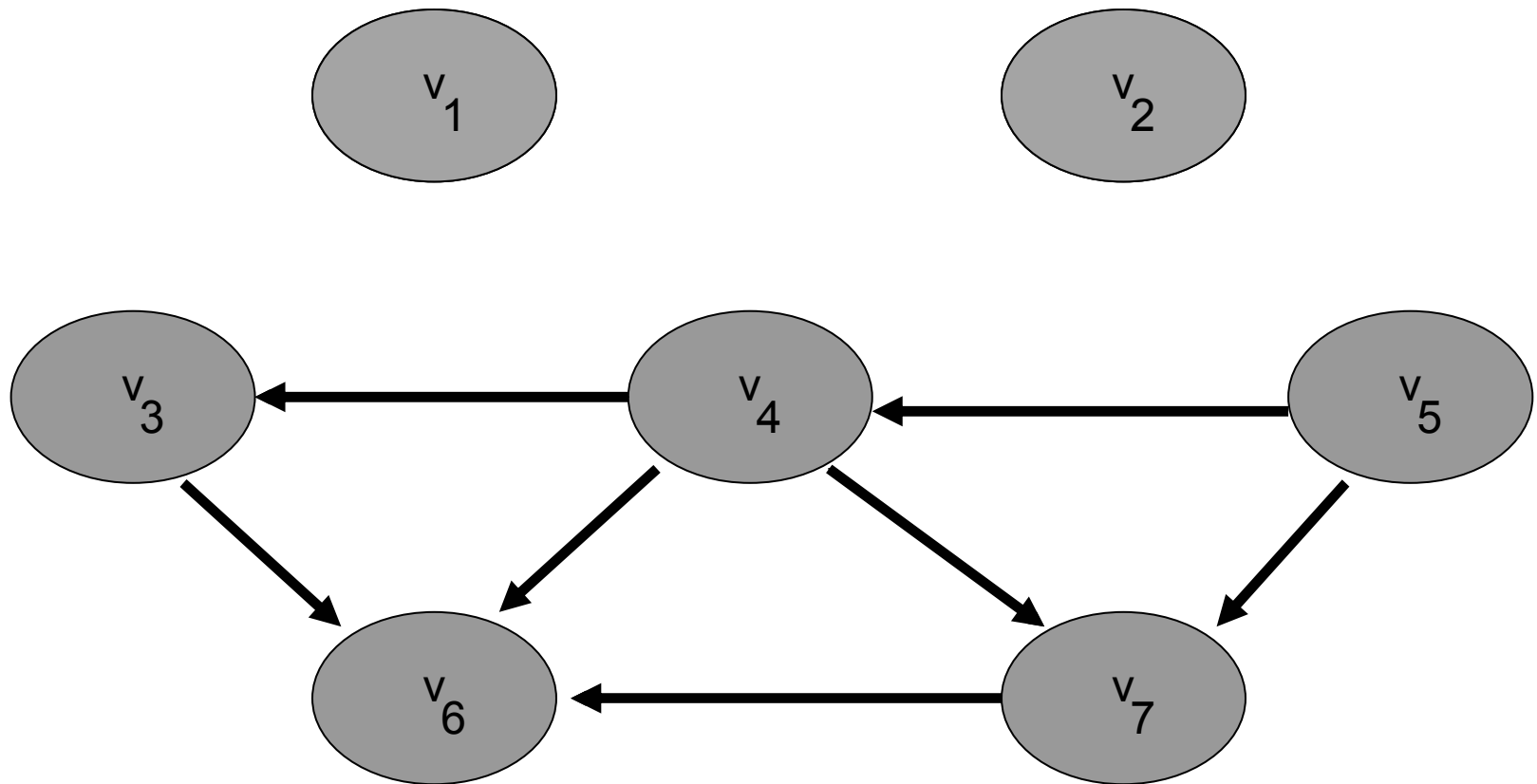
Ejemplo:



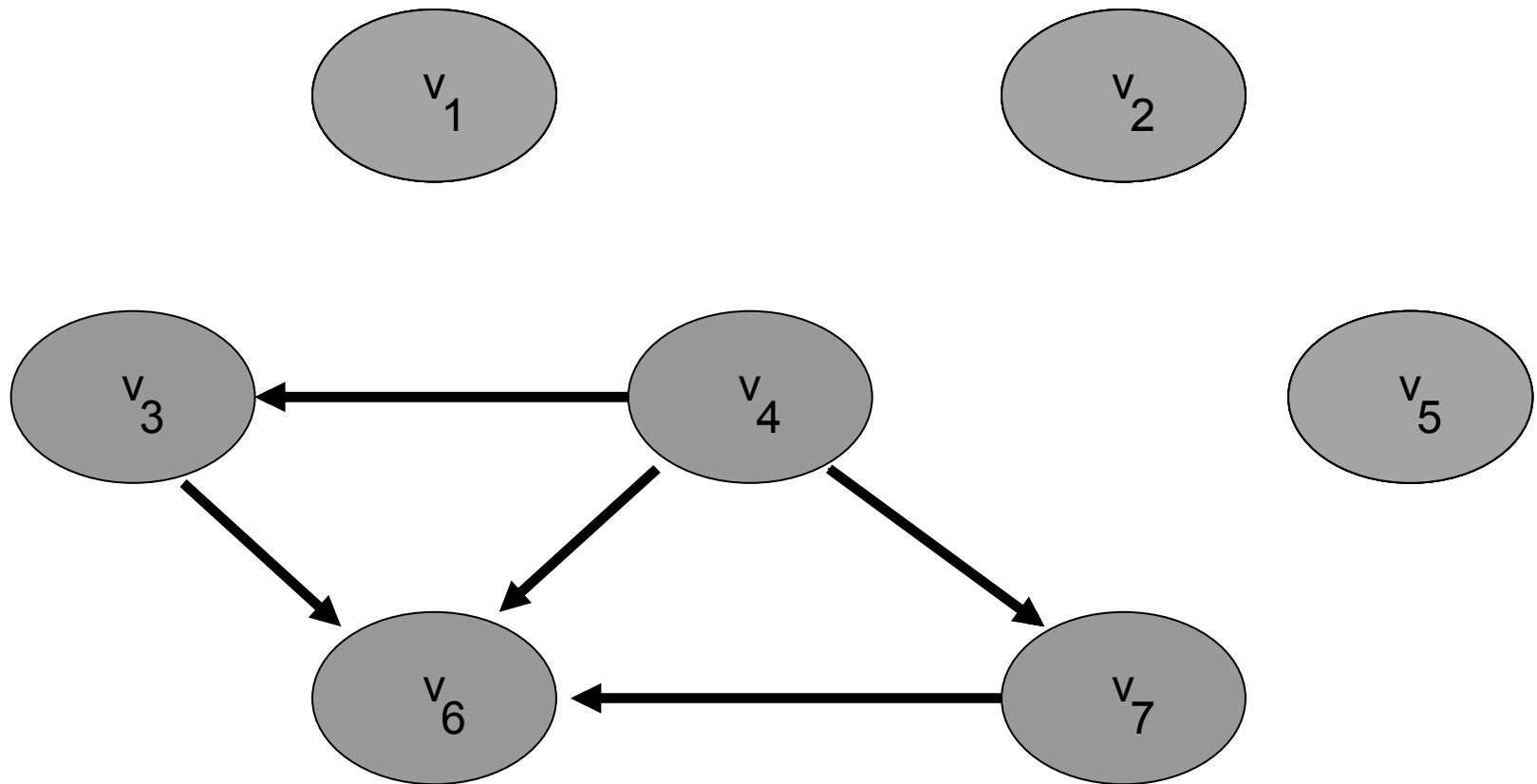
ORDENAMIENTO TOPOLOGICO



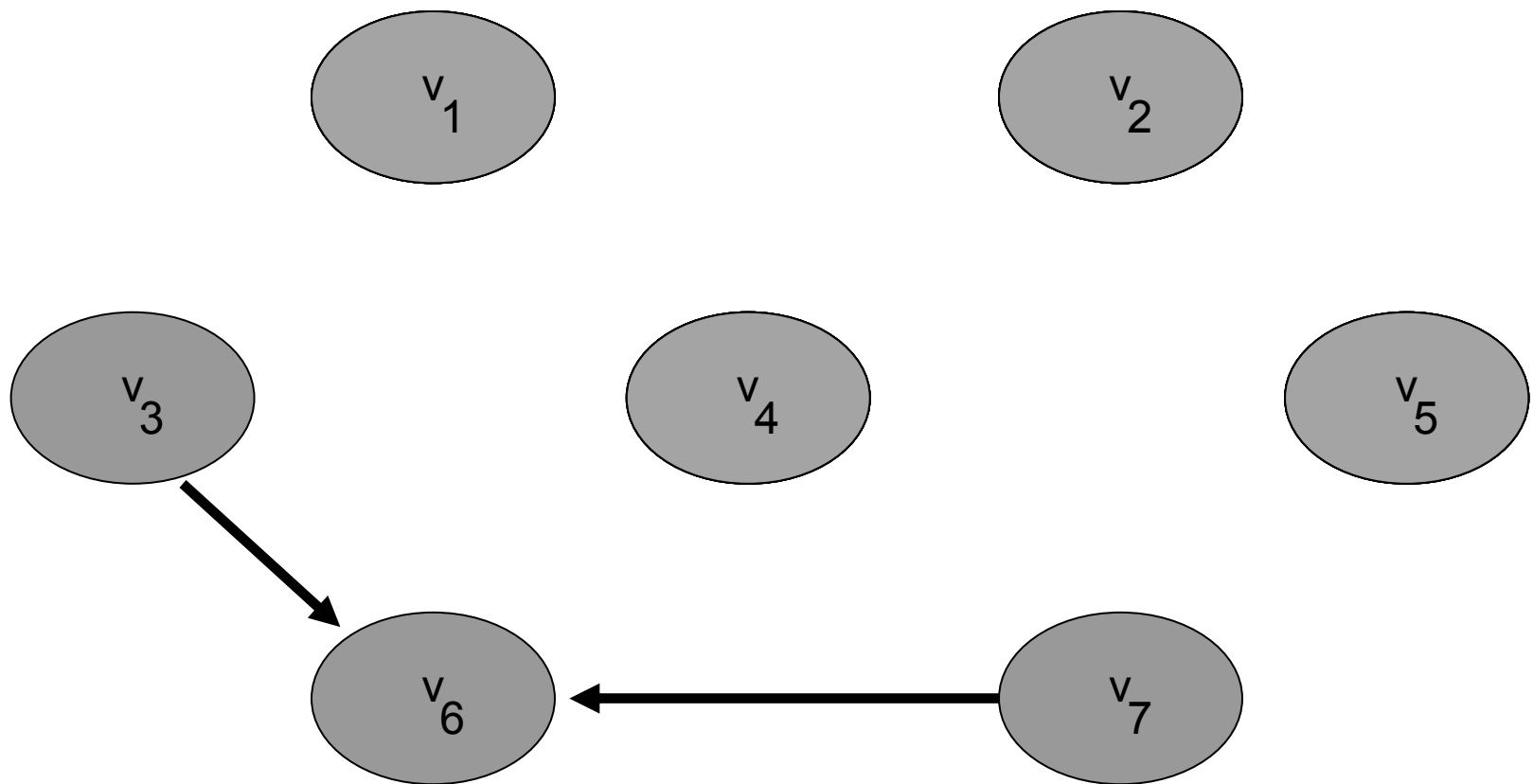
ORDENAMIENTO TOPOLOGICO



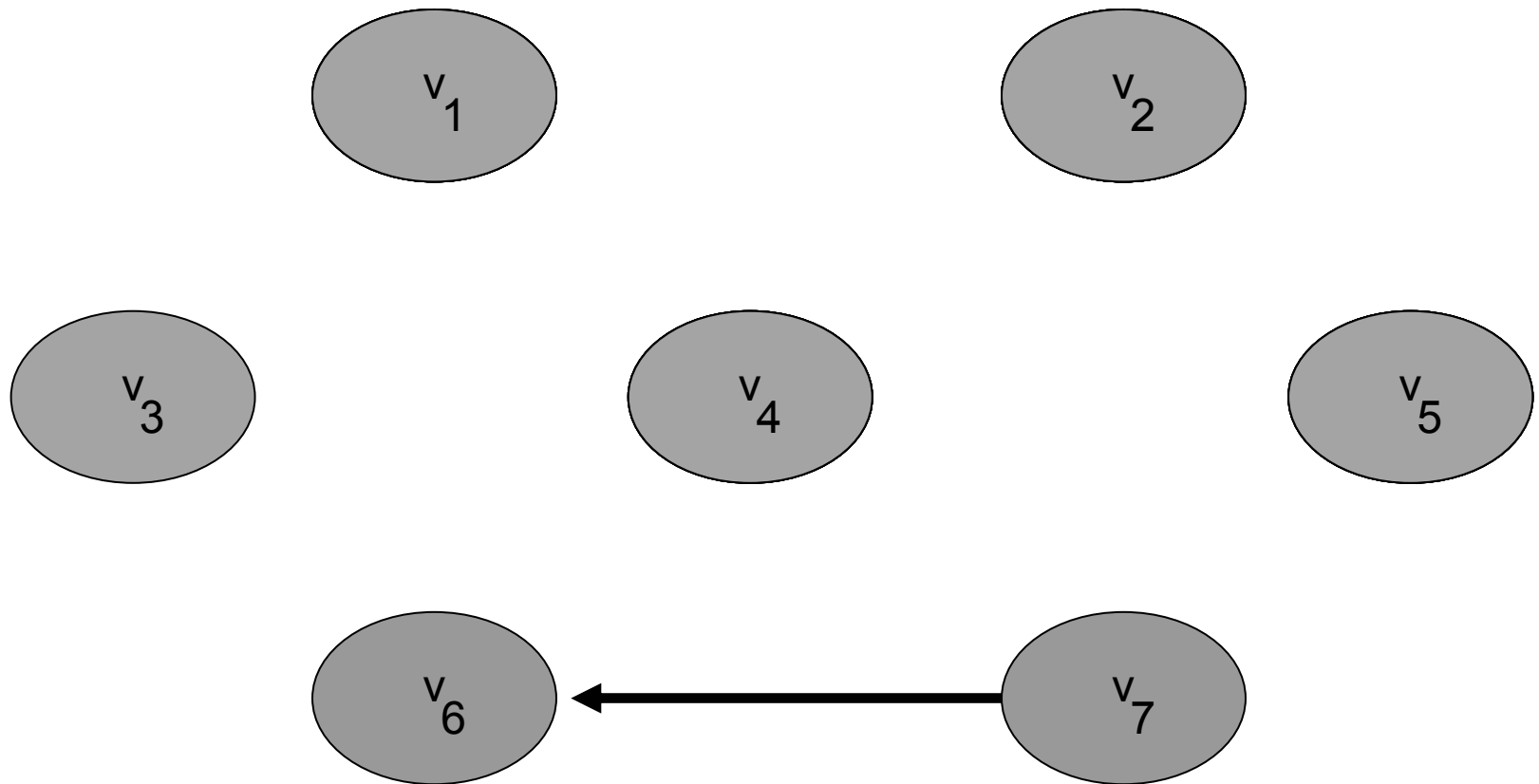
ORDENAMIENTO TOPOLOGICO



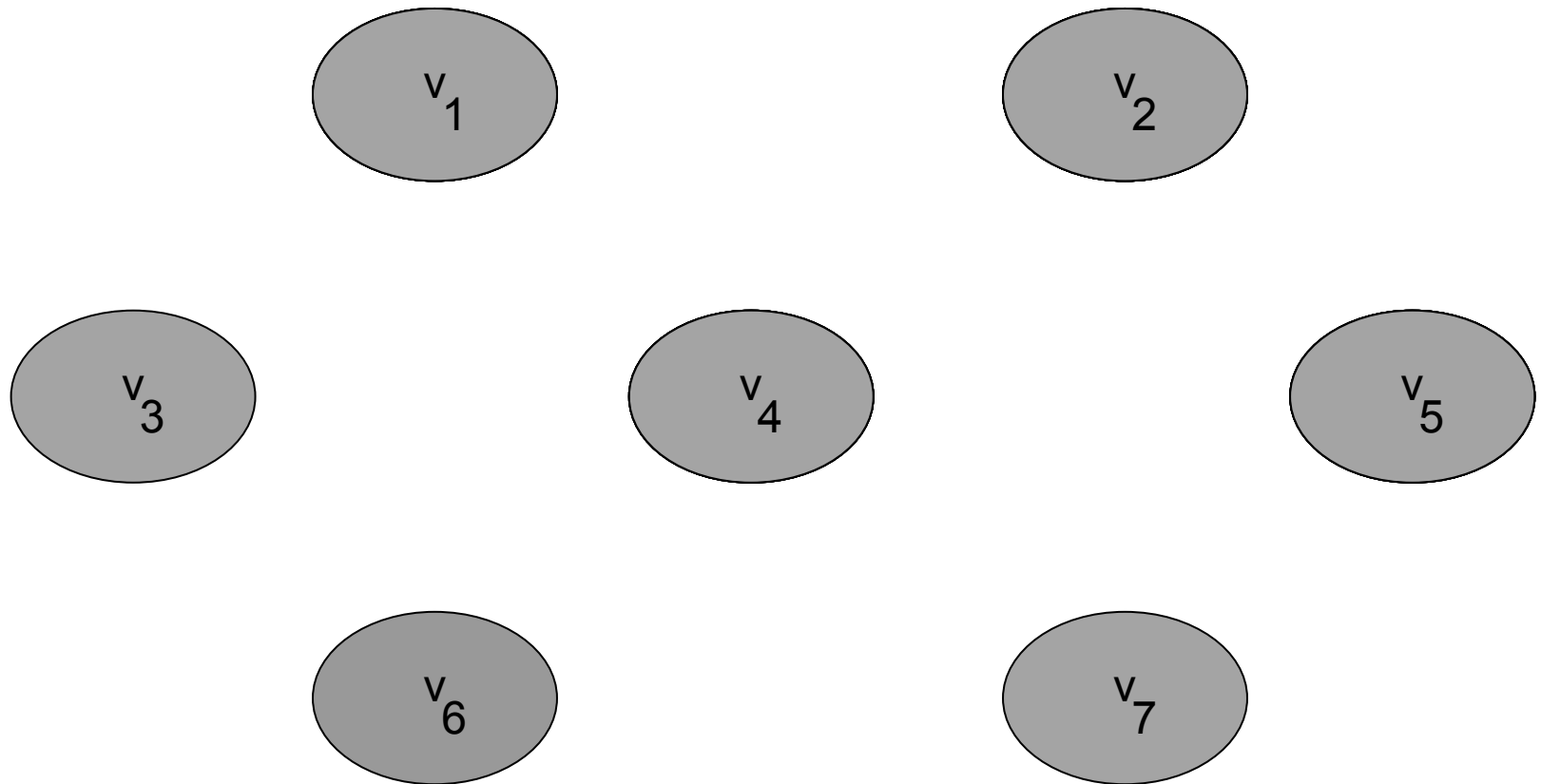
ORDENAMIENTO TOPOLOGICO



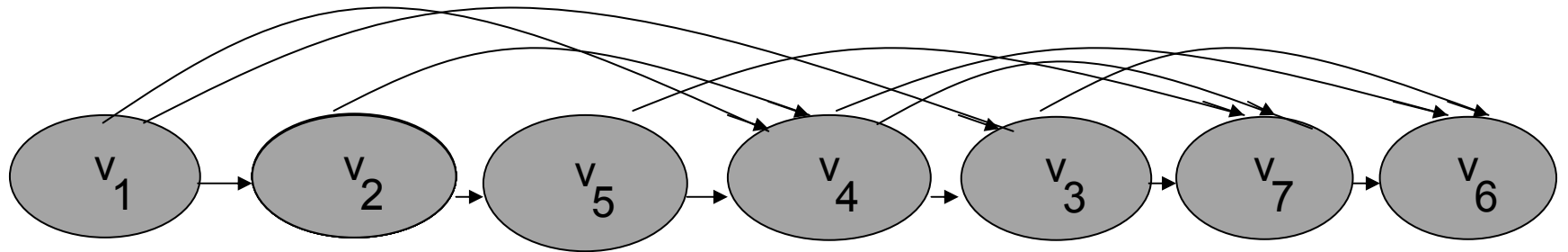
ORDENAMIENTO TOPOLOGICO



ORDENAMIENTO TOPOLOGICO



ORDENAMIENTO TOPOLOGICO



ORDENAMIENTO TOPOLOGICO

Grados de entrada de los vértices

Vértice	1	2	3	4	5	6	7
V_1	0	0	0	0	0	0	0
V_2	1	0	0	0	0	0	0
V_3	2	1	1	1	0	0	0
V_4	3	2	1	0	0	0	0
V_5	1	1	0	0	0	0	0
V_6	3	3	3	3	2	1	0
V_7	2	2	2	1	0	0	0
Encolar	V_1	V_2	V_5	V_4	V_3	V_7	V_6
Desencolar	V_1	V_2	V_5	V_4	V_3	V_7	V_6

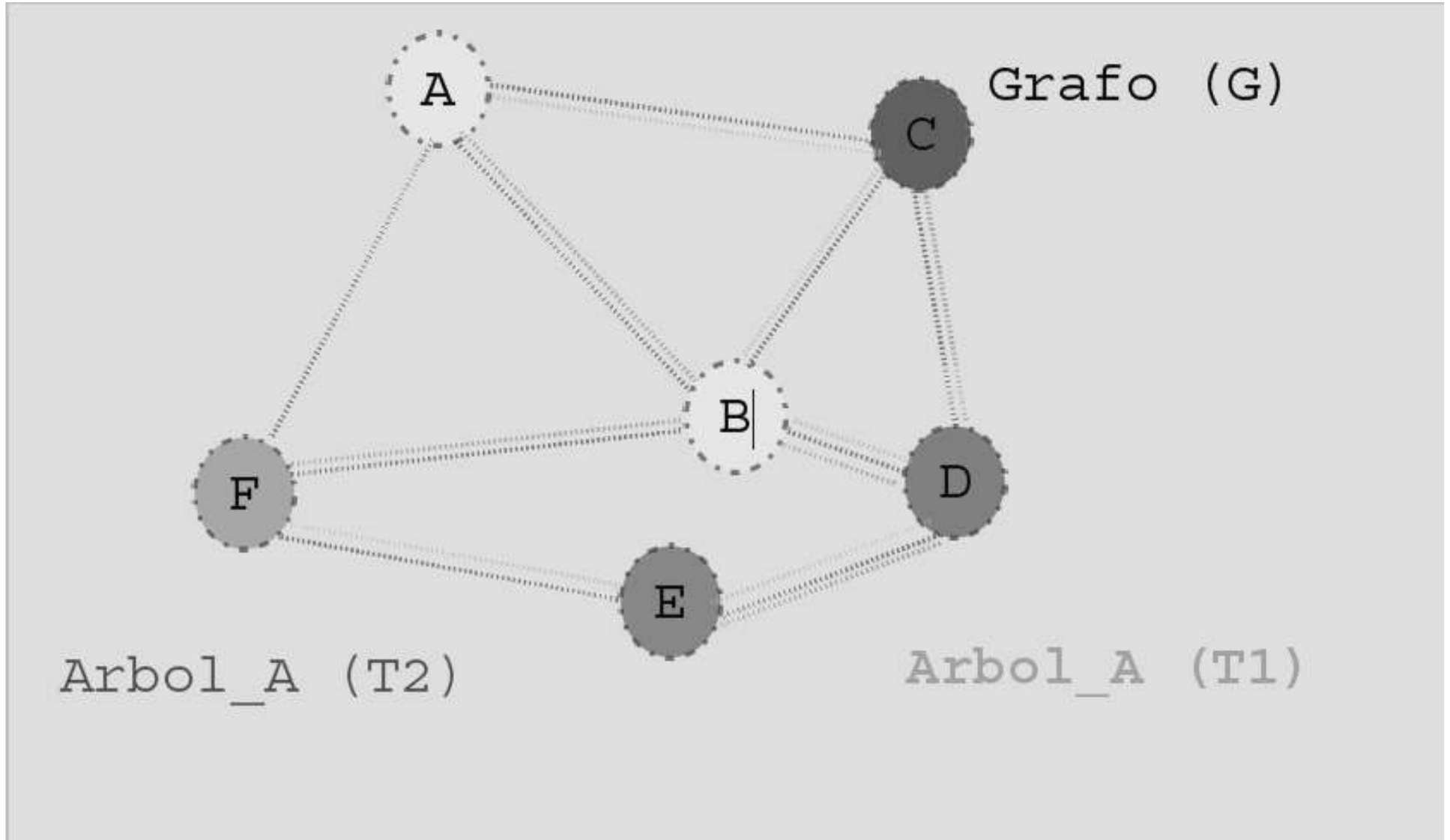
Conceptos

Grafo conexo: un grafo g es conexo si y solo si sus nodos no pueden ser divididos en dos conjuntos no vacíos n_1 y n_2 en los que sus caminos desde un punto cualquiera a otro estén en el mismo conjunto.

Árbol de expansión: Es un subconjunto del grafo, (árbol) que abarca todos los vértices del grafo que están conectados y sin ciclos.

- ✓ Todo árbol de n vértices contiene exactamente $n-1$ aristas.
 - ✓ Es un árbol pues es aciclico.
 - ✓ Es de extensión porque cubre todas las aristas.
-

ARBOL DE EXPANSION



ARBOL DE EXPANSION MINIMA

Dado un grafo G conectado con pesos, se desea con frecuencia crear un árbol de expansión T para G , tal que la suma de los pesos de las aristas del árbol en T sea tan pequeña como sea posible, por tanto representa la manera más barata de conectar todos los nodos de G .

Técnicas de árboles de expansión Mínima

- Algoritmo de Kruskall.
 - Algoritmo de Prim.
-

Algoritmo de Kruskal.

Consiste en elegir sucesivamente las aristas de mínimo peso sin formar ciclos a partir de un grafo con pesos en sus arcos.

Usa una estructura de conjuntos disjuntos para mantener los diferentes conjuntos disjuntos de elementos.

Es de orden $O(A \log V)$

Pseudocódigo Kruskal

Kruskal (G)

$T \leftarrow \emptyset$

for each vertex $v \in V[G]$

do **hacer-conjunto**(v)

 ordenar las aristas de E en orden nondecreasing ordene por peso w

for cada arista $(u, v) \in E$, tome en orden nondecreasing por peso

do **if** FIND-SET(u) \neq FIND-SET(v)

then $A \leftarrow A \cup \{(u, v)\}$

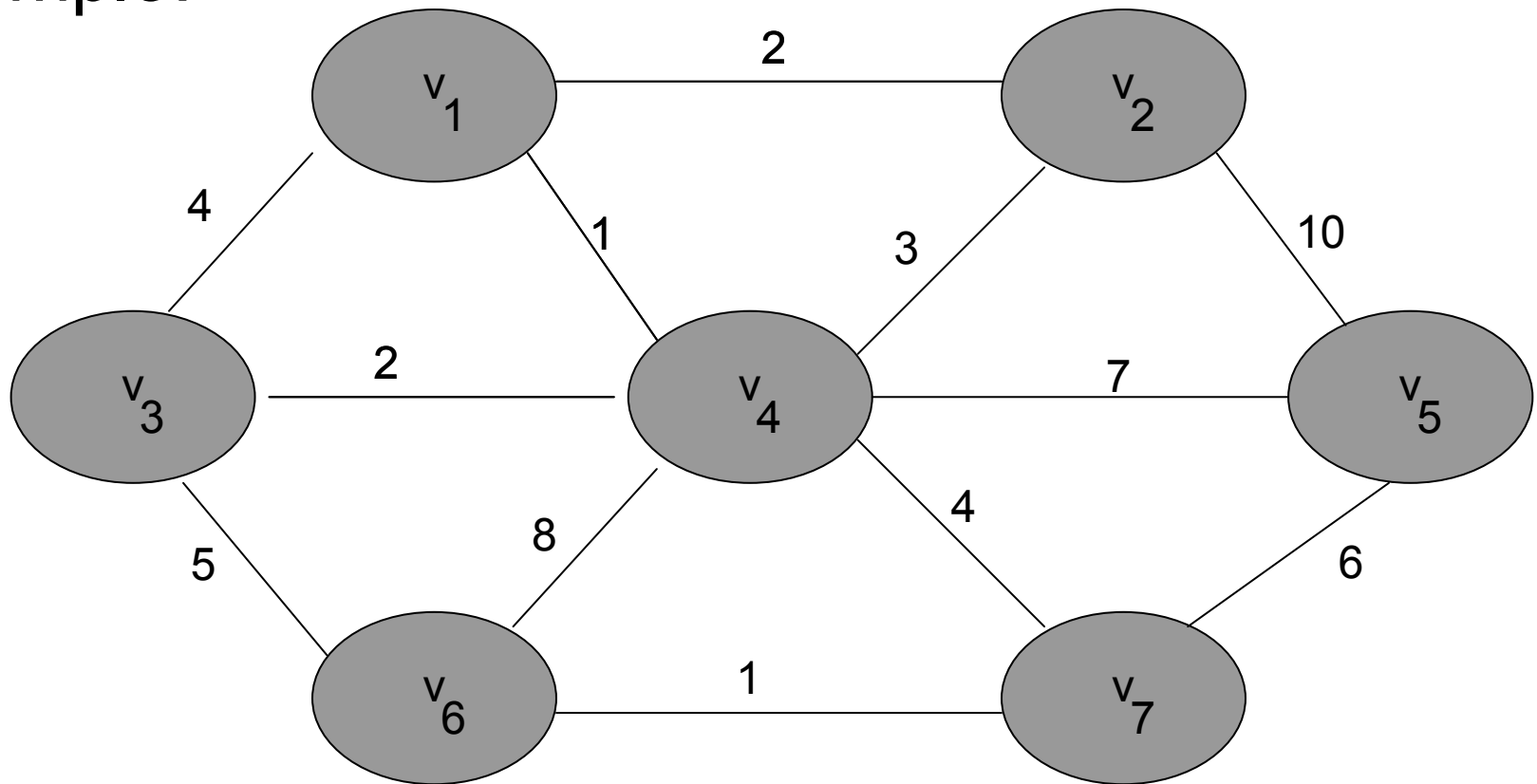
 UNION(u, v)

return T

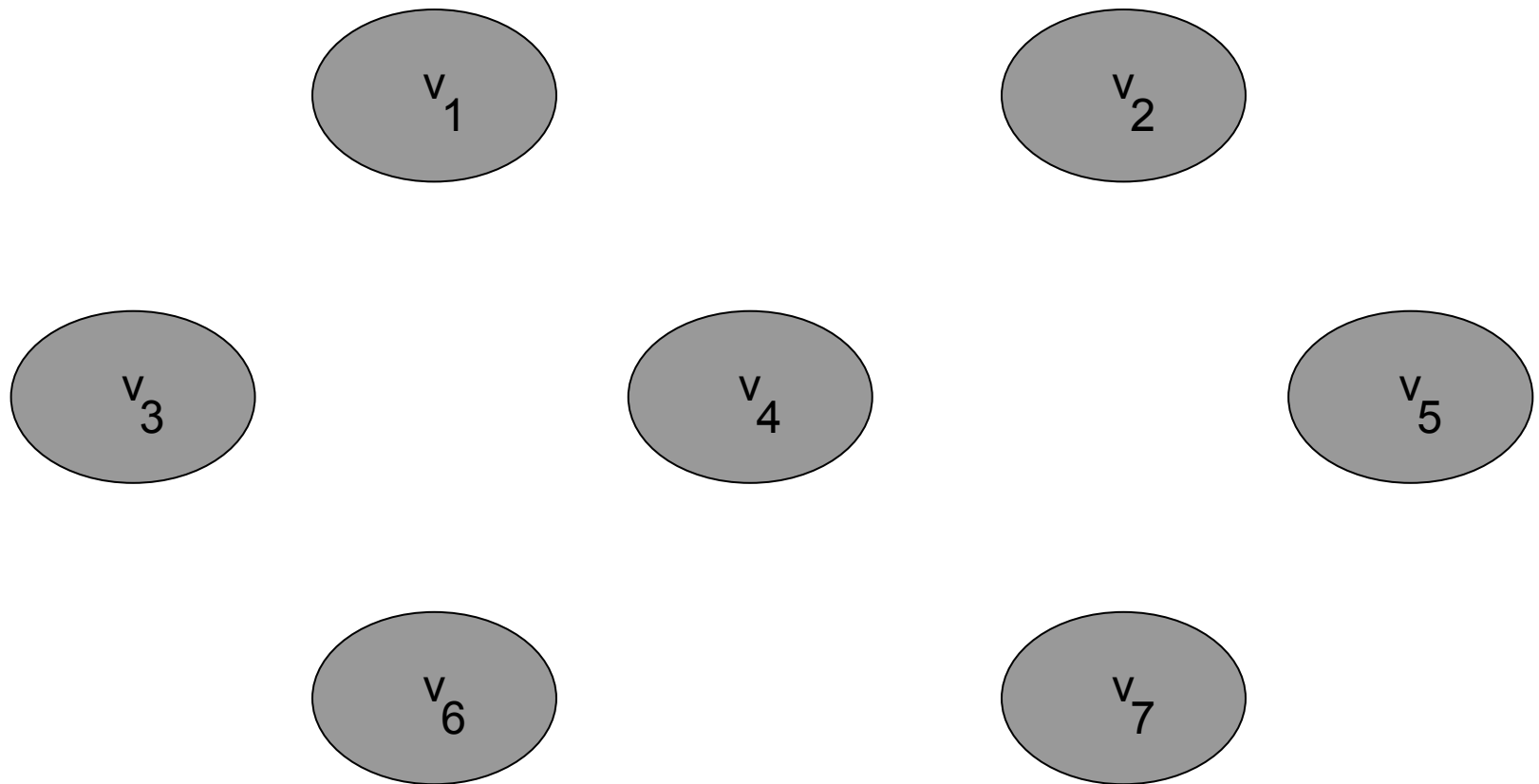
Es de orden $O(A \log V)$

Algoritmo de Kruskal.

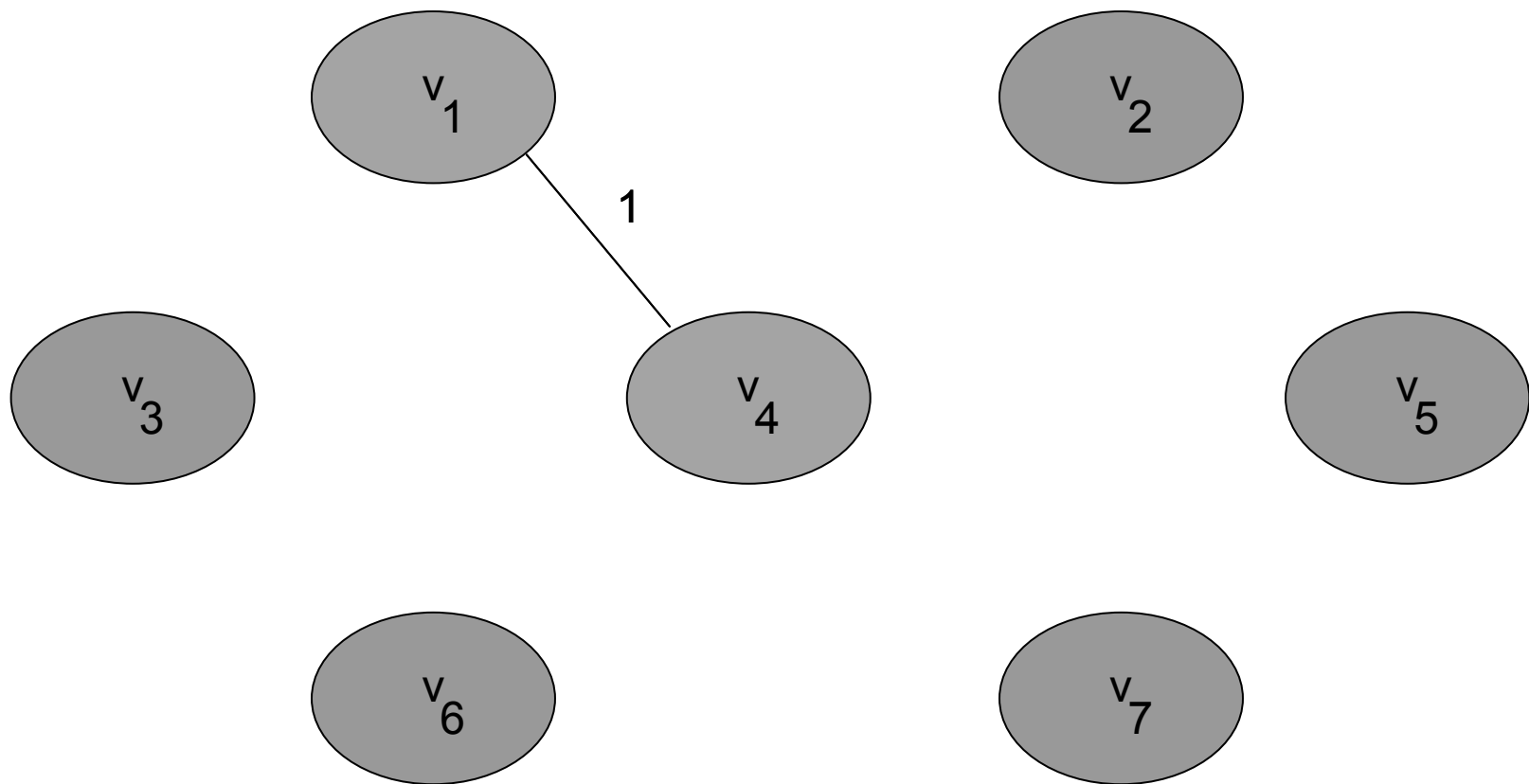
Ejemplo:



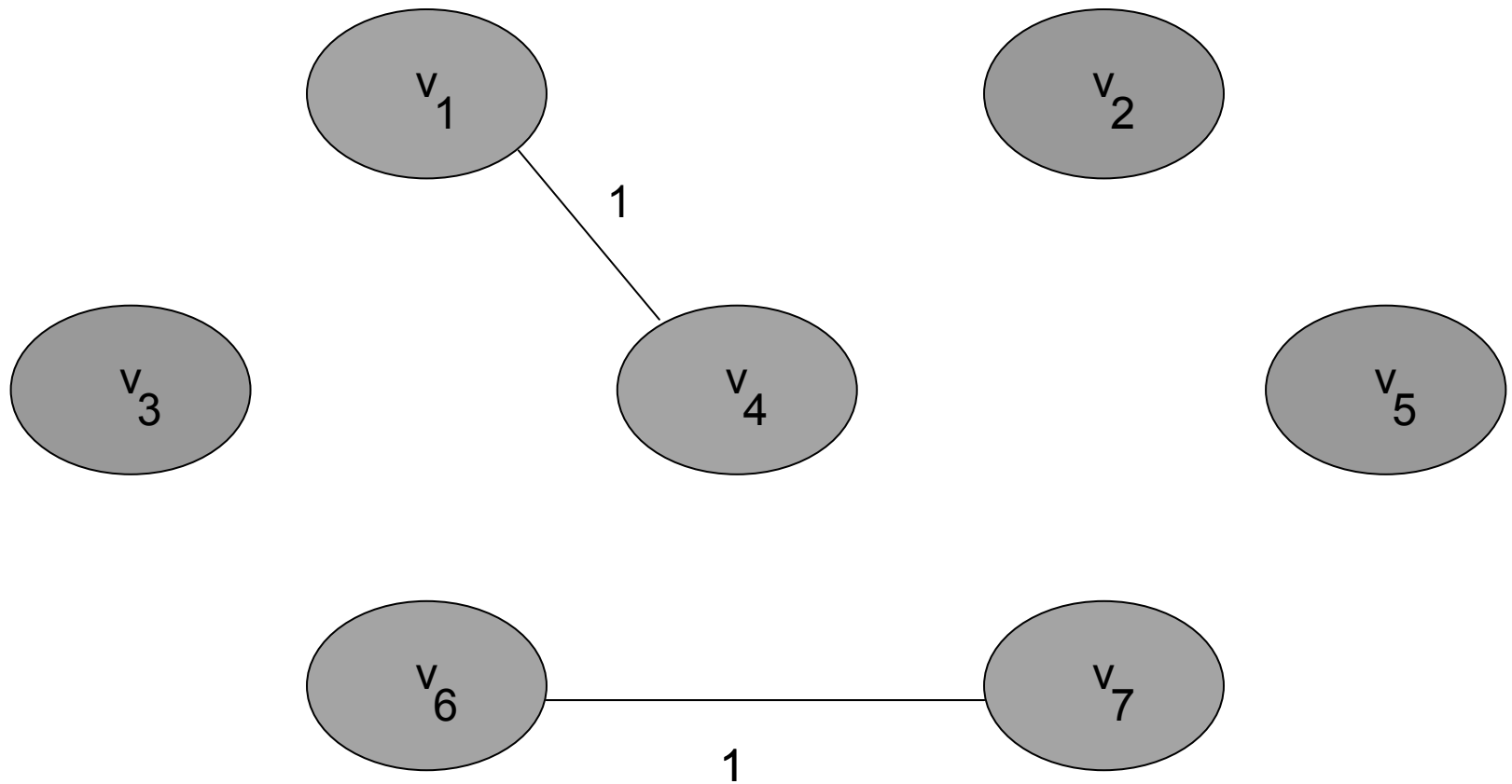
Algoritmo de Kruskal.



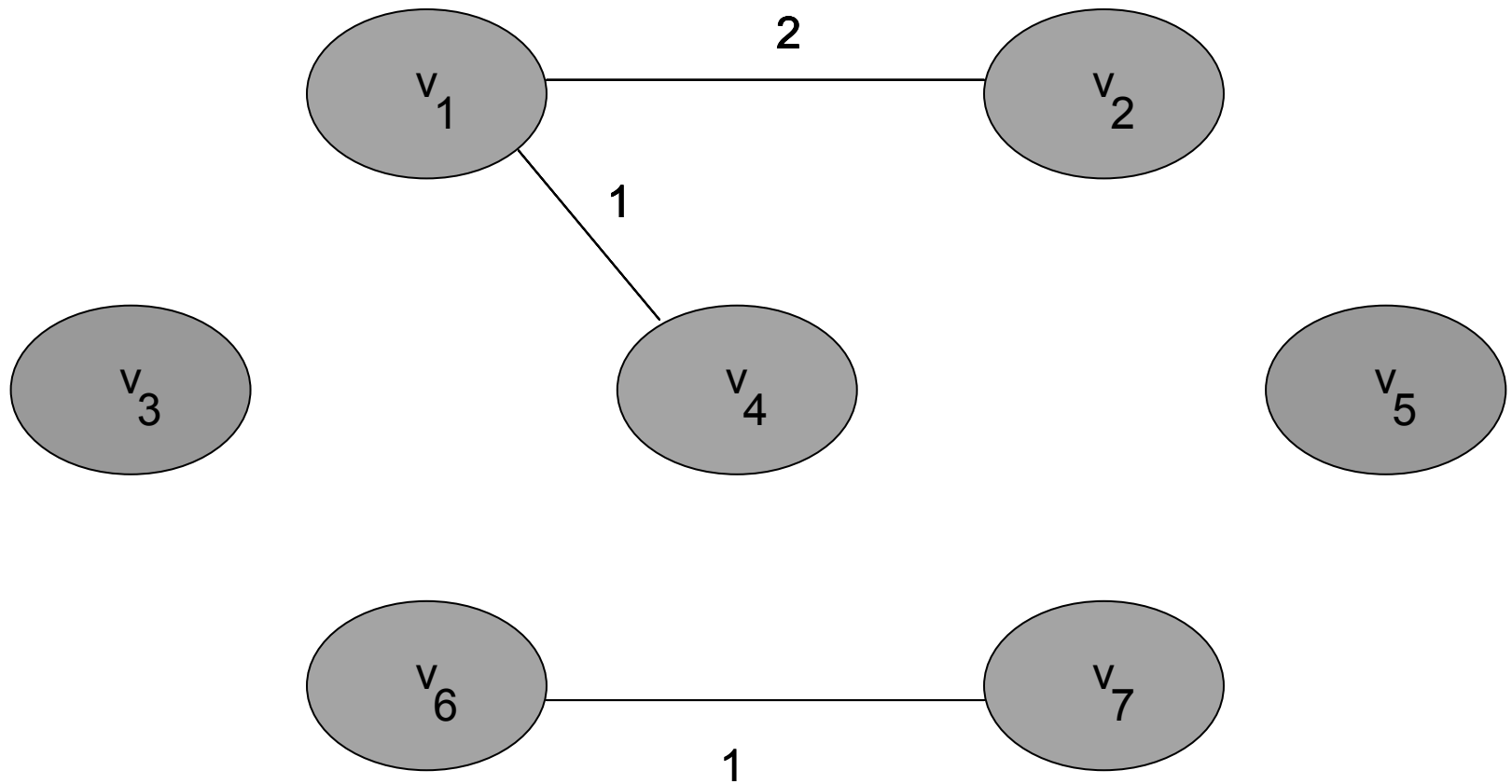
Algoritmo de Kruskal.



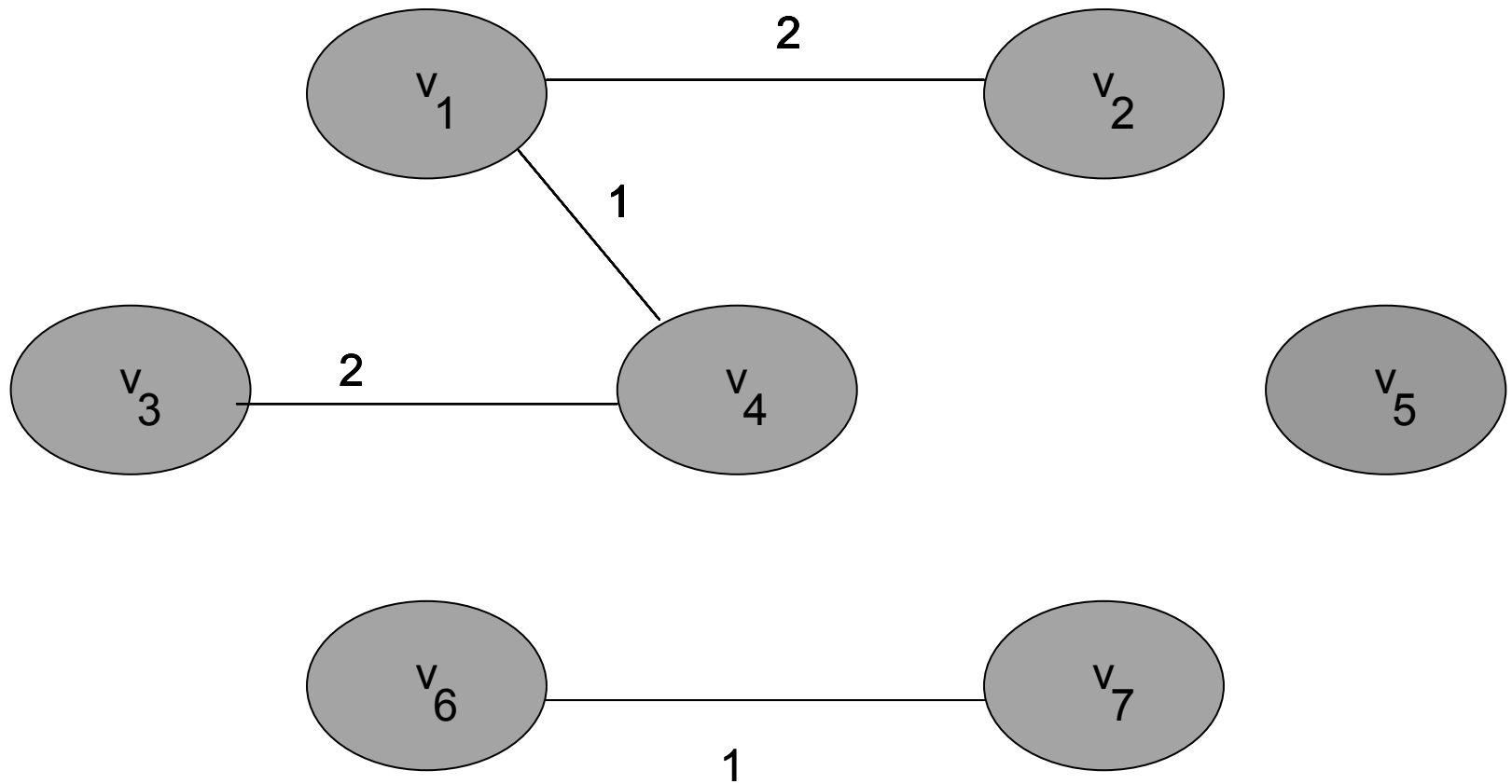
Algoritmo de Kruskal.



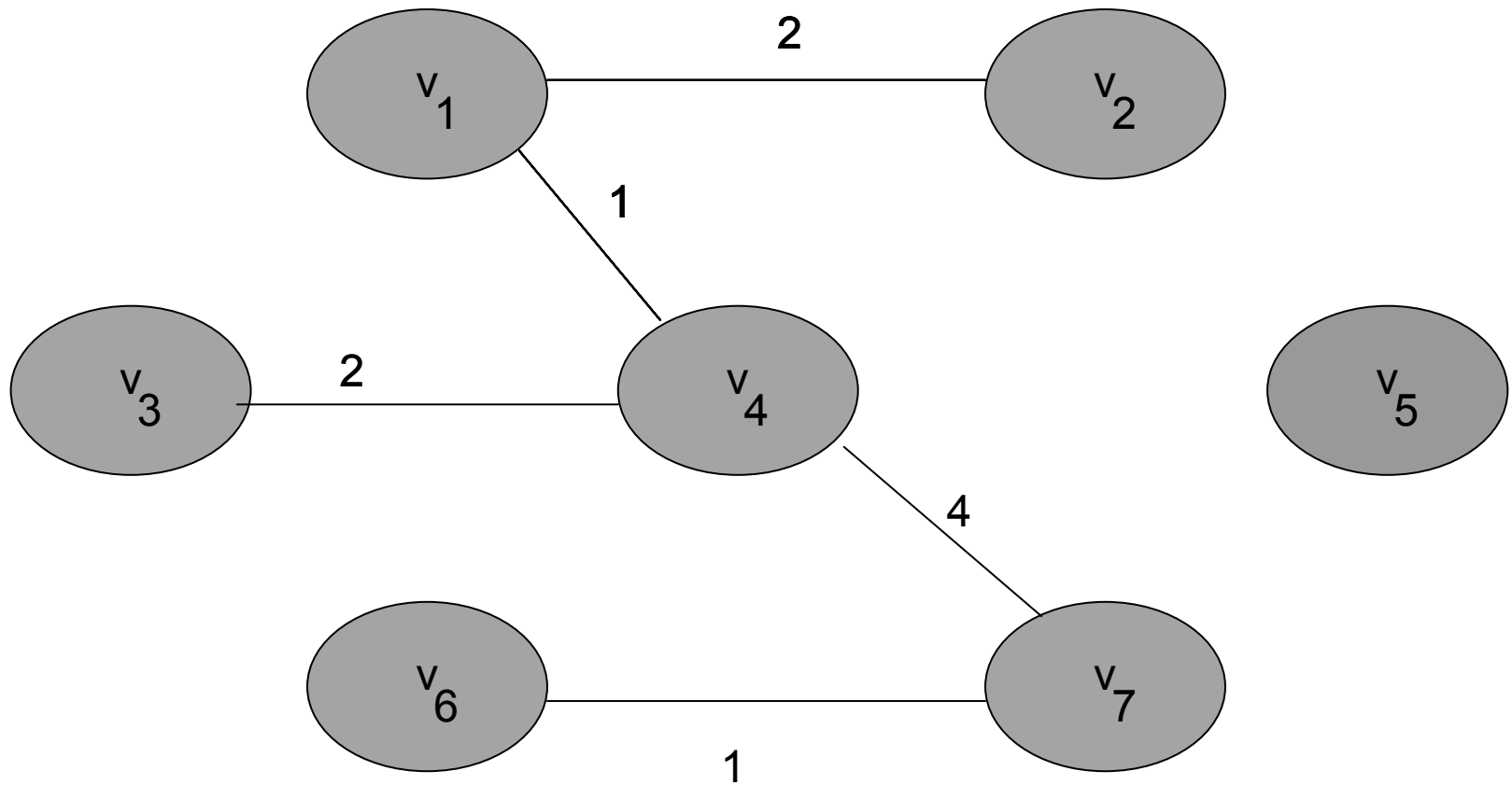
Algoritmo de Kruskal.



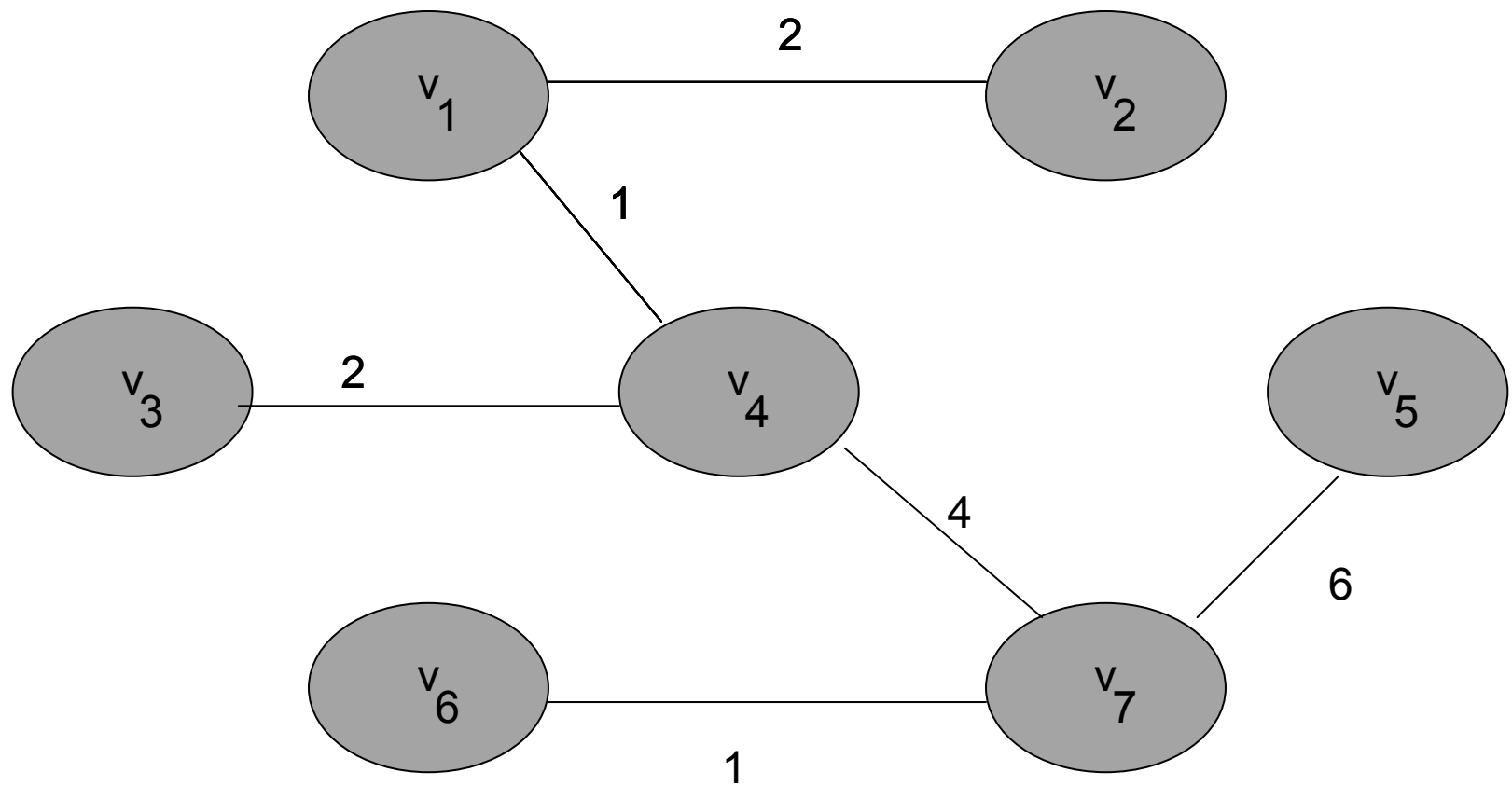
Algoritmo de Kruskal.



Algoritmo de Kruskal.



Algoritmo de Kruskal.



Algoritmo de Prim

Encuentra el árbol de expansión mínima de un grafo no dirigido, realizando una serie de pasos para incorporar al árbol una nueva arista del grafo de menor peso.

- ❖ El árbol crece en etapas sucesivas.
 - ❖ En cada etapa se elige un nodo como raíz y se agrega al árbol una arista (vértice asociado).
 - ❖ Es de orden $O(E+V \log V)$ usando montículo de Fibonacci, siempre que $V > E$ u $O(E \log V)$ usando montículo binarios.
-

Pseudocódigo (PRIM)

Para cada vértice v , $key[v]$ es el peso mínimo de cualquier arista conectada v a un vértice en el árbol; por convención, $key[v] = \infty$ si no hay tal arista. El campo $\pi[v]$ nombra a los vértices adyacentes v en el árbol.

Pseudocódigo (PRIM)

PRIM(G, w, r)

for each $u \in V[G]$

do $key[u] \leftarrow \infty$

$\pi[u] \leftarrow \text{NIL}$

$key[r] \leftarrow 0$

$Q \leftarrow V[G]$

while $Q \neq \emptyset$

do $u \leftarrow \text{EXTRACT-MIN}(Q)$

for each $v \in \text{Adj}[u]$

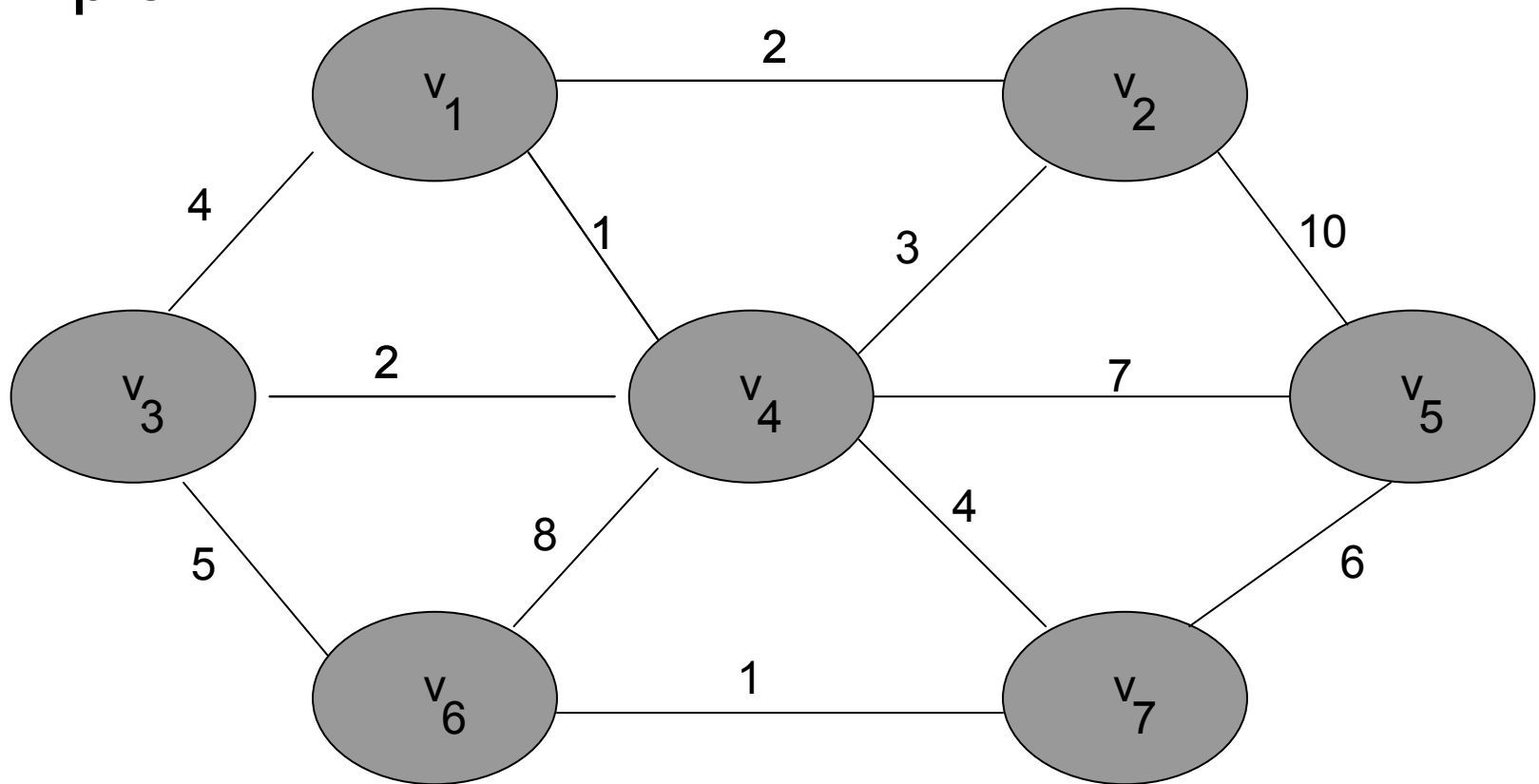
do if $v \in Q$ and $w(u, v) < key[v]$

then $\pi[v] \leftarrow u$

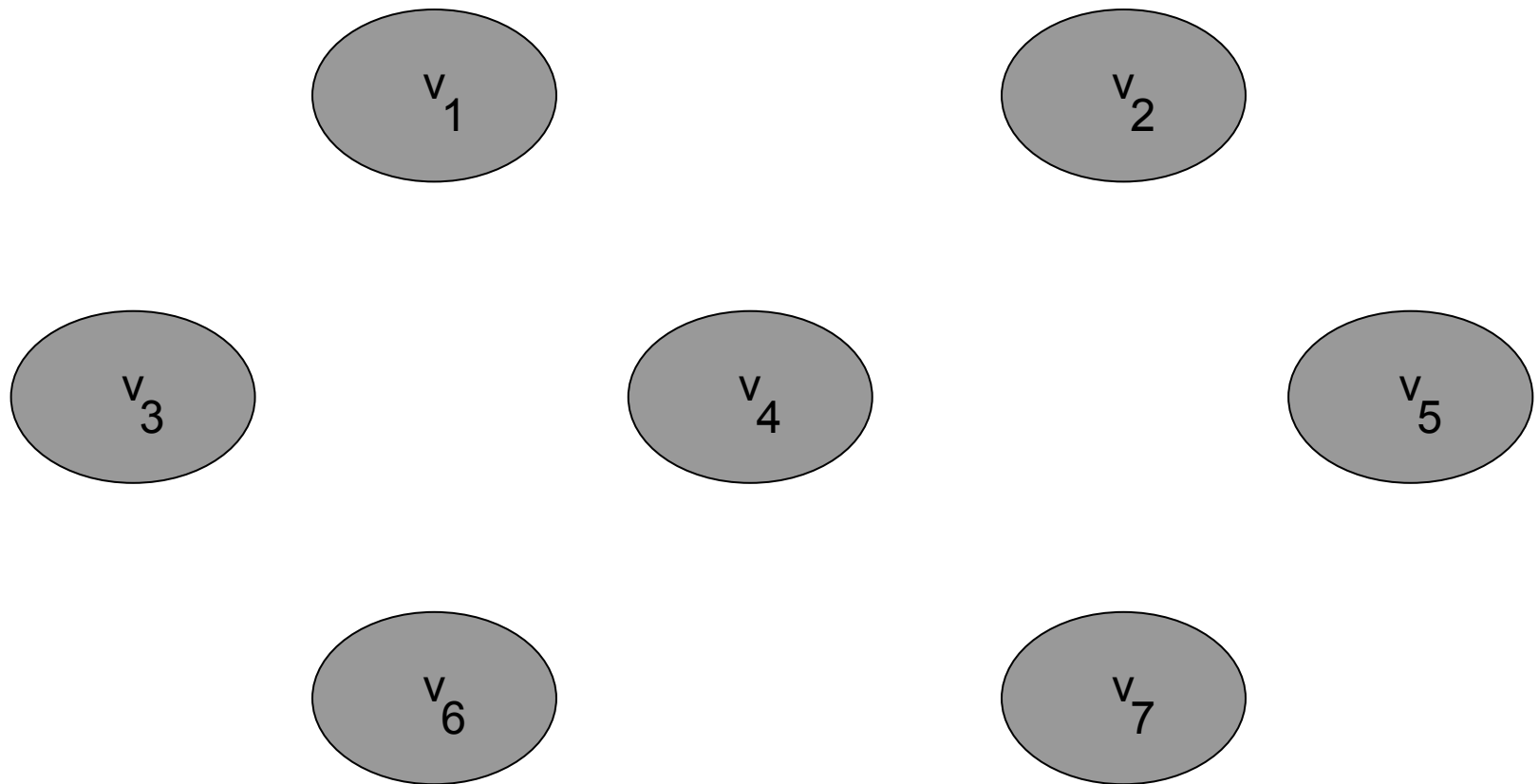
$key[v] \leftarrow w(u, v)$

Algoritmo de Prim

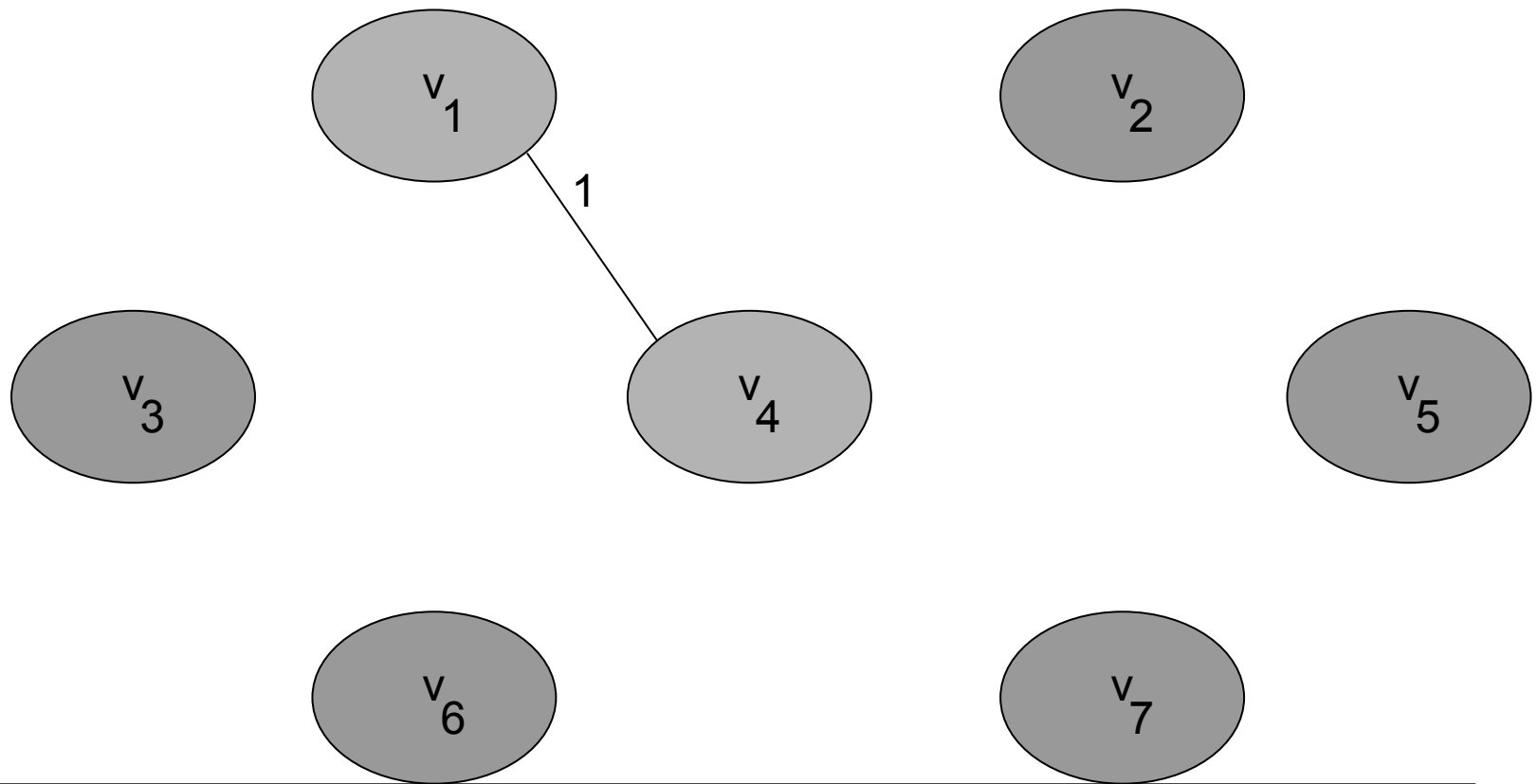
Ejemplo:



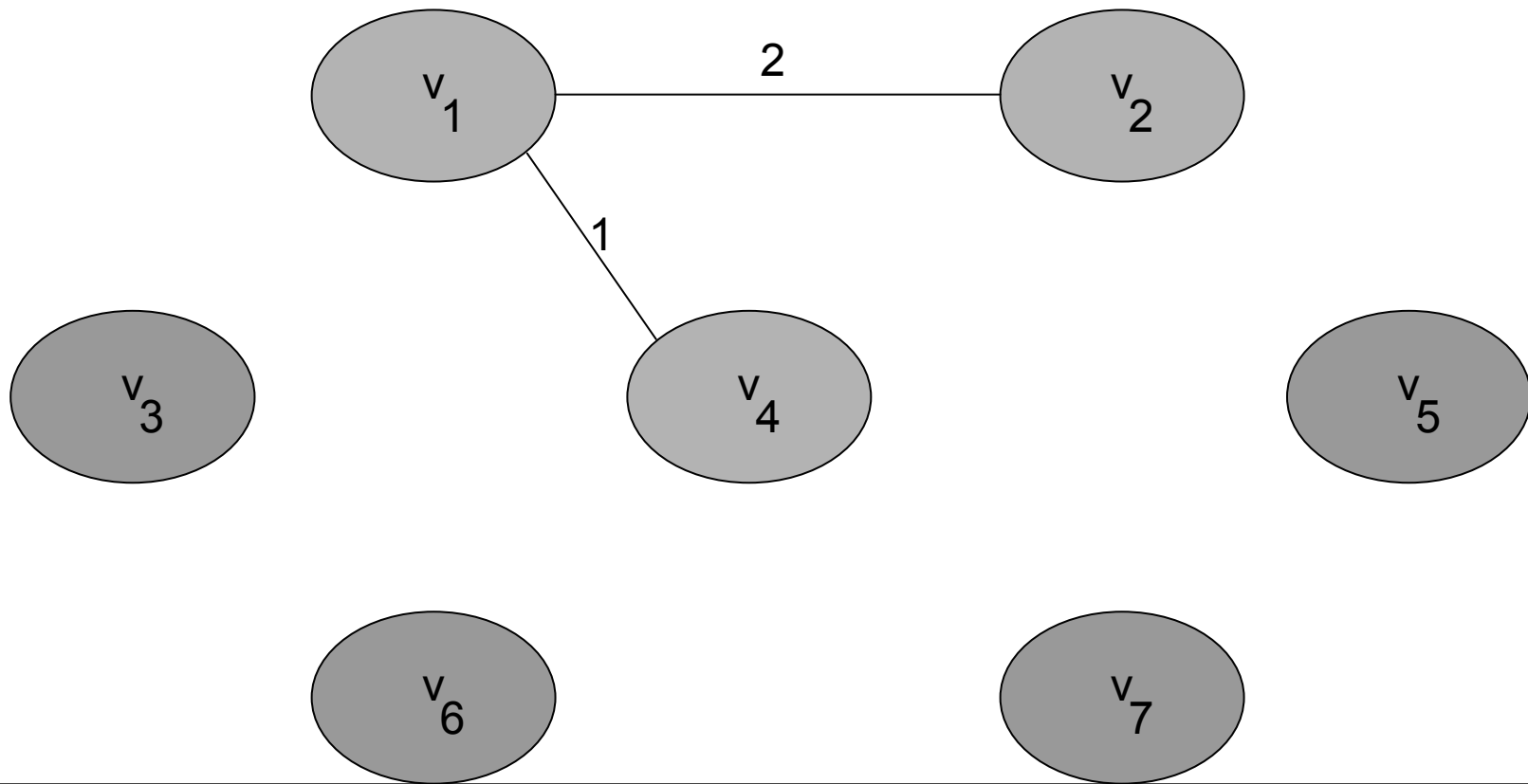
Algoritmo de Prim



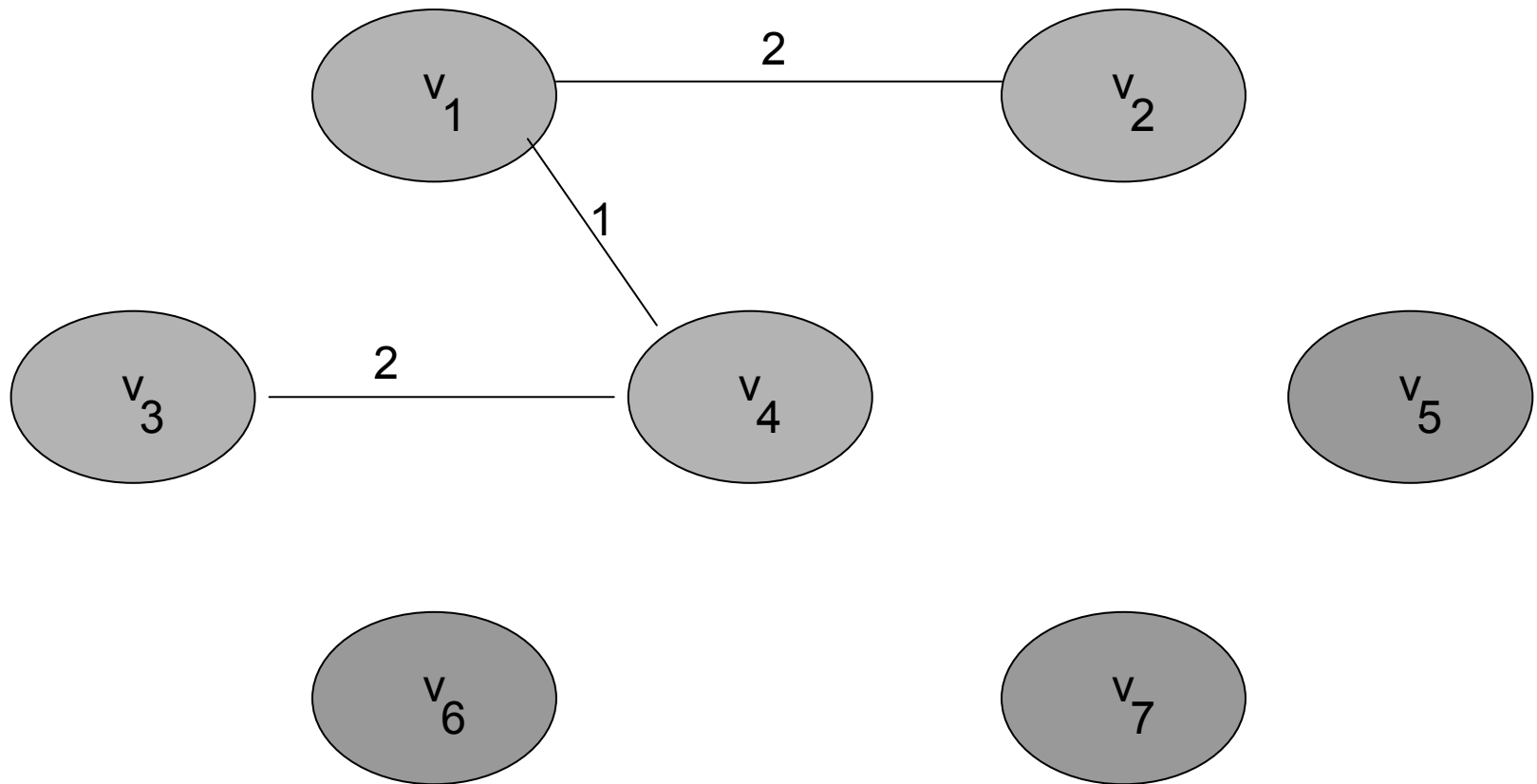
Algoritmo de Prim



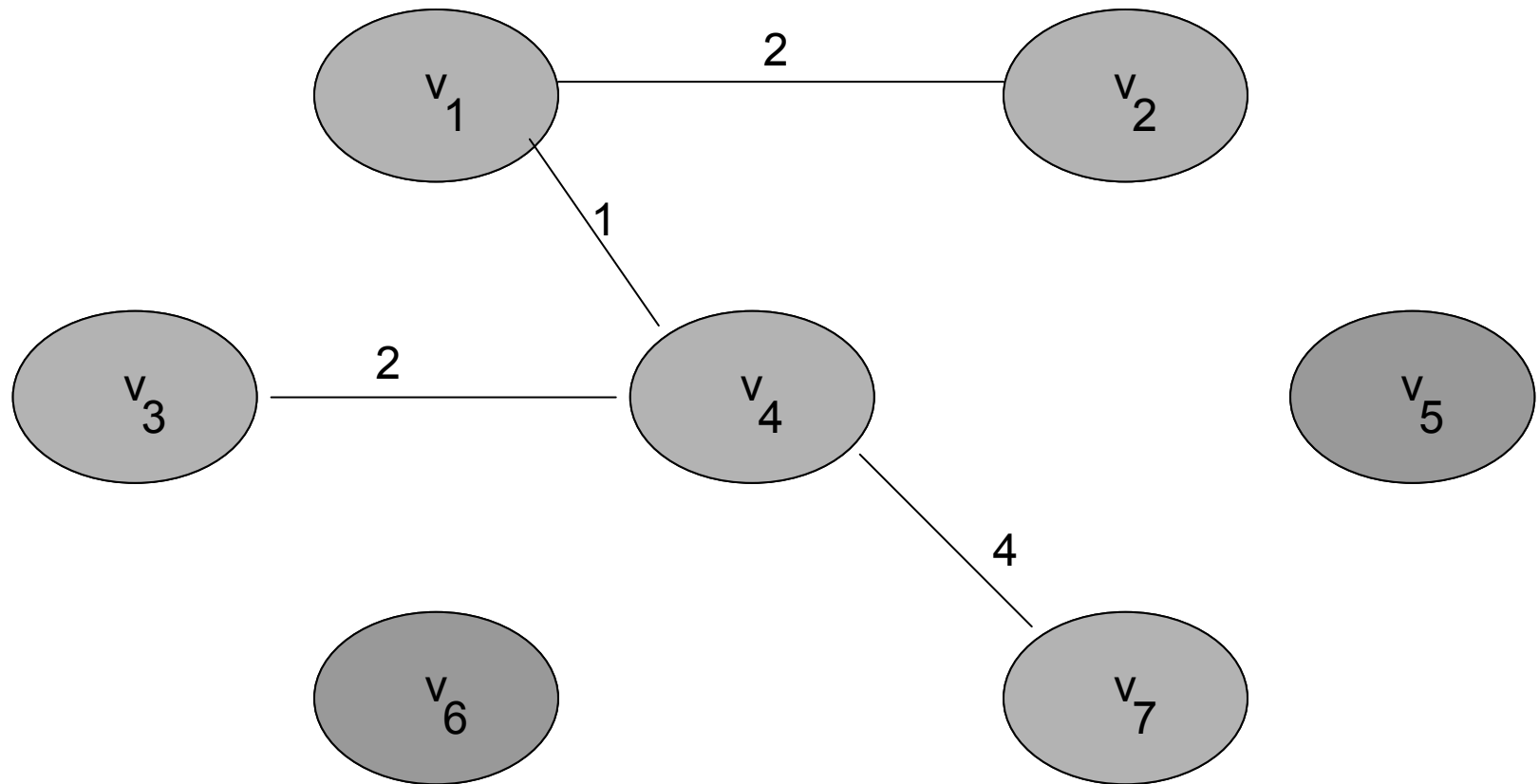
Algoritmo de Prim



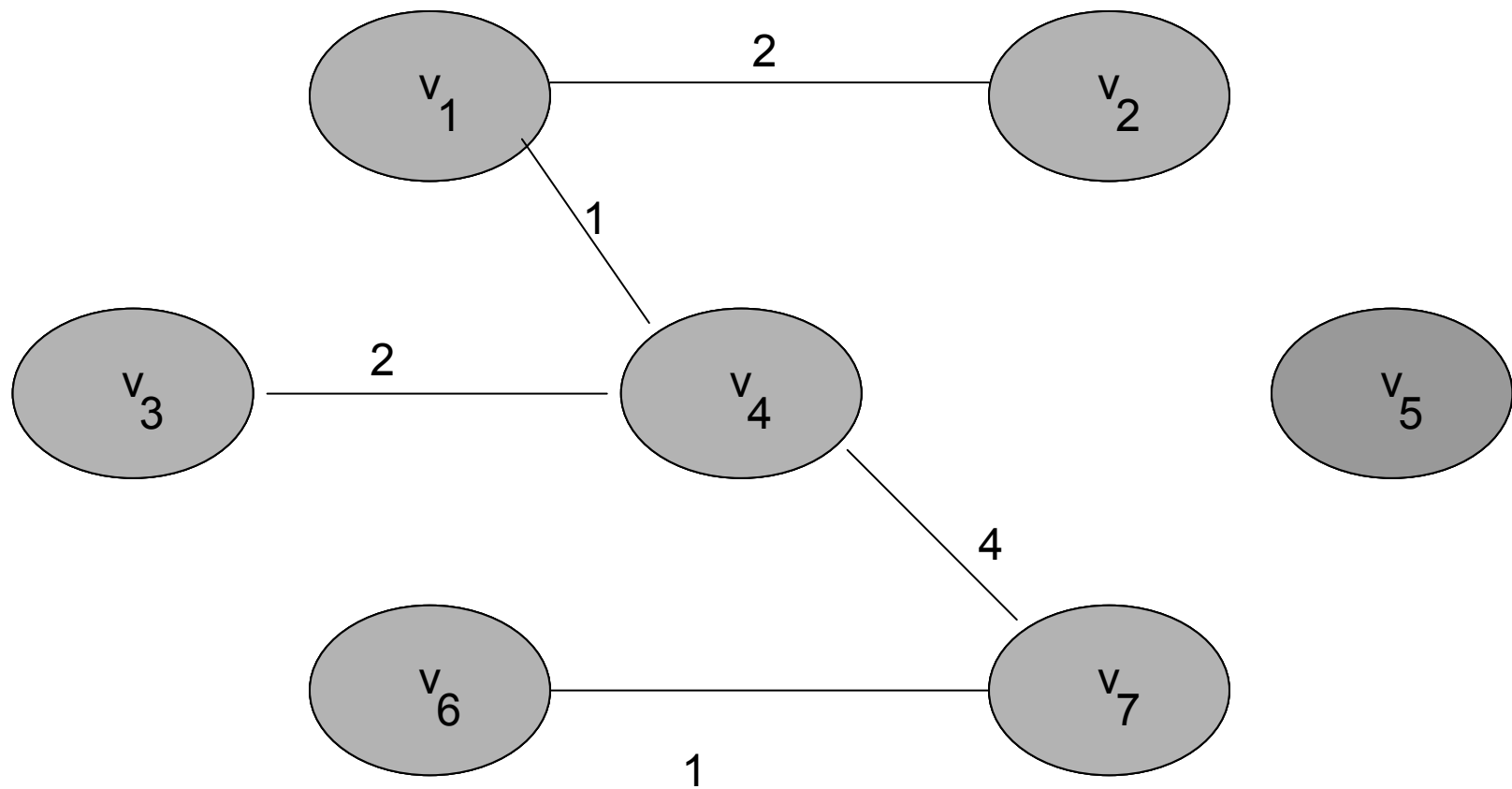
Algoritmo de Prim



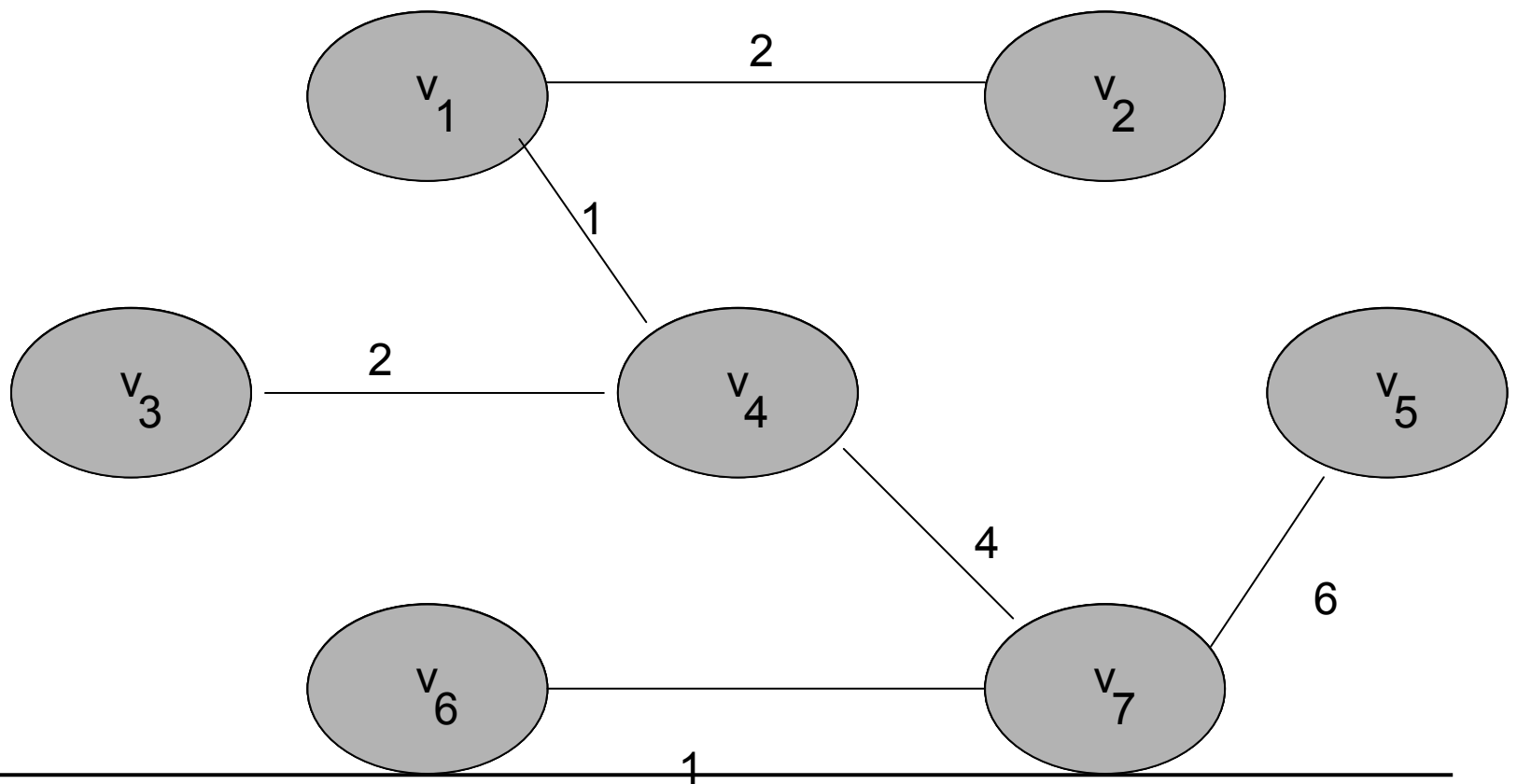
Algoritmo de Prim



Algoritmo de Prim



Algoritmo de Prim



GRACIAS!!!
