Grafos con costos - II

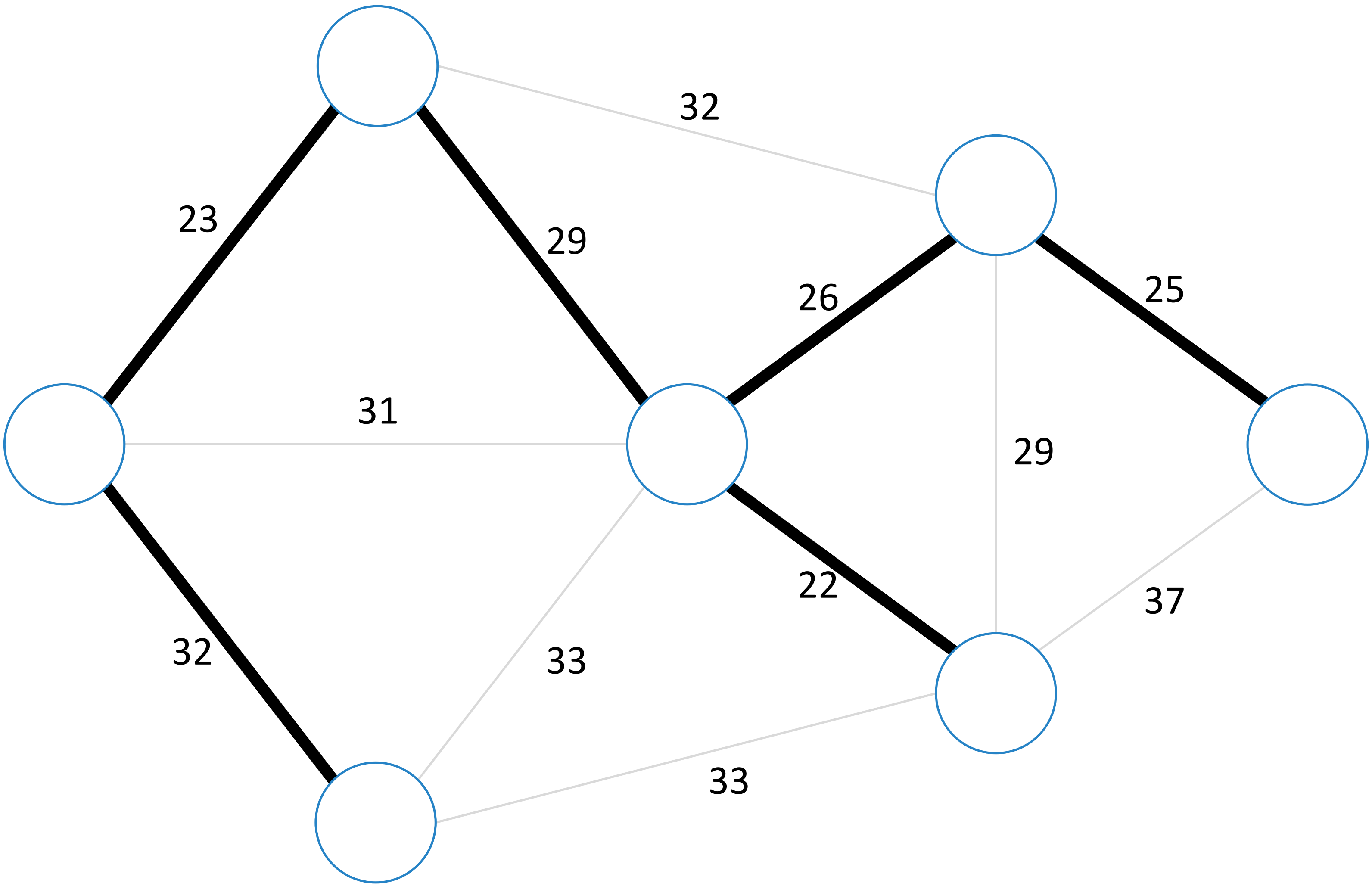
Propiedades del MST

¿Hay alguna arista que **siempre** pertenezca a un **MST**?

¿Se cumple esto recursivamente? ¿En que casos?

¿Podremos aprovecharlo en un algoritmo **codicioso**?

MST



El algoritmo de Kruskal

$\boldsymbol{kruskal}\big(\boldsymbol{G}(\boldsymbol{V},\boldsymbol{E})\big)$:

Ordenar $\boldsymbol{E}$ por costo, de menor a mayor

$\boldsymbol{T} \leftarrow \emptyset$

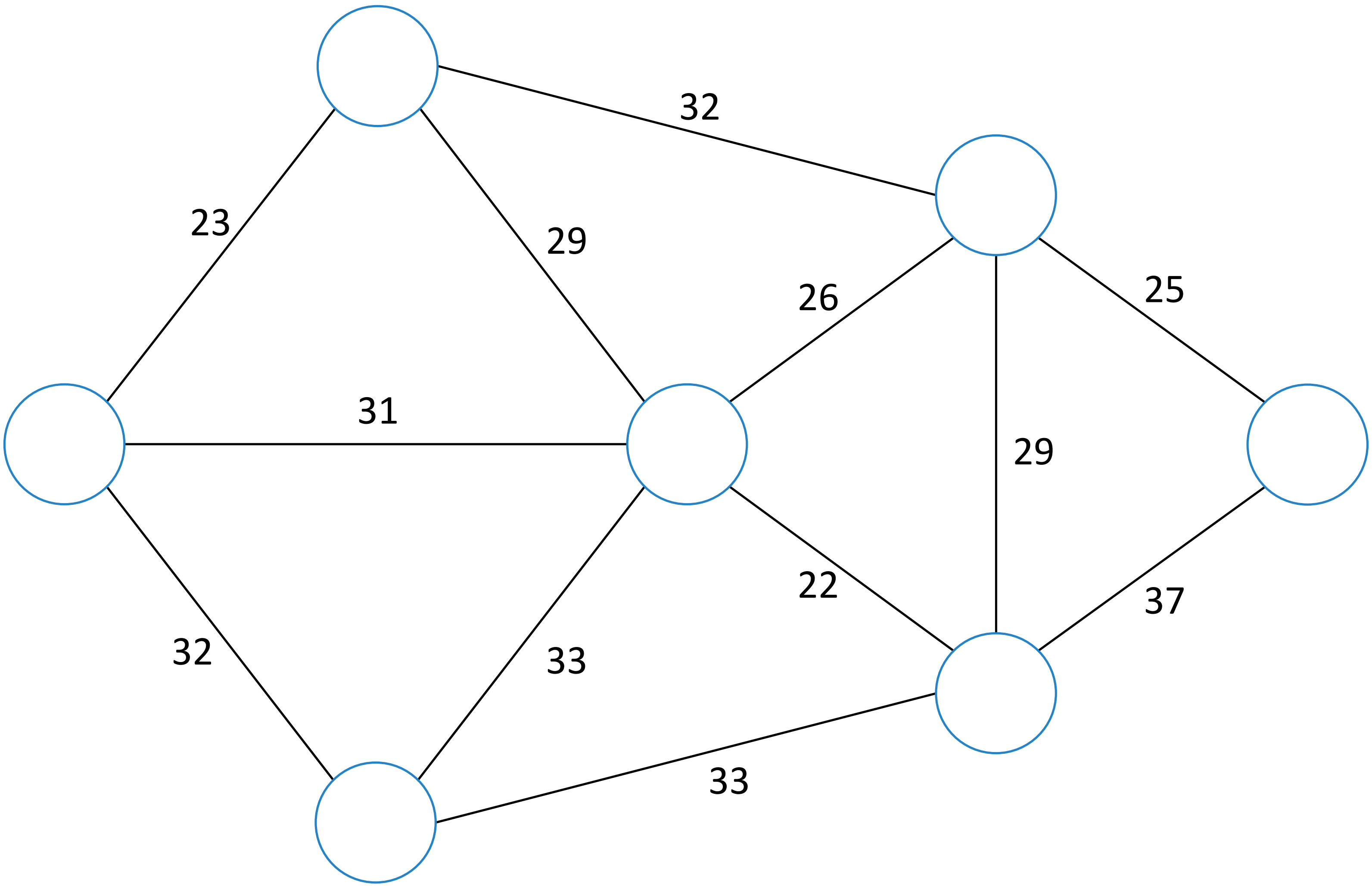
$\boldsymbol{foreach}\ \boldsymbol{e} \in \boldsymbol{E}$:

$\boldsymbol{if}$ agregar $\boldsymbol{e}$ a $\boldsymbol{T}$ no forma un ciclo:

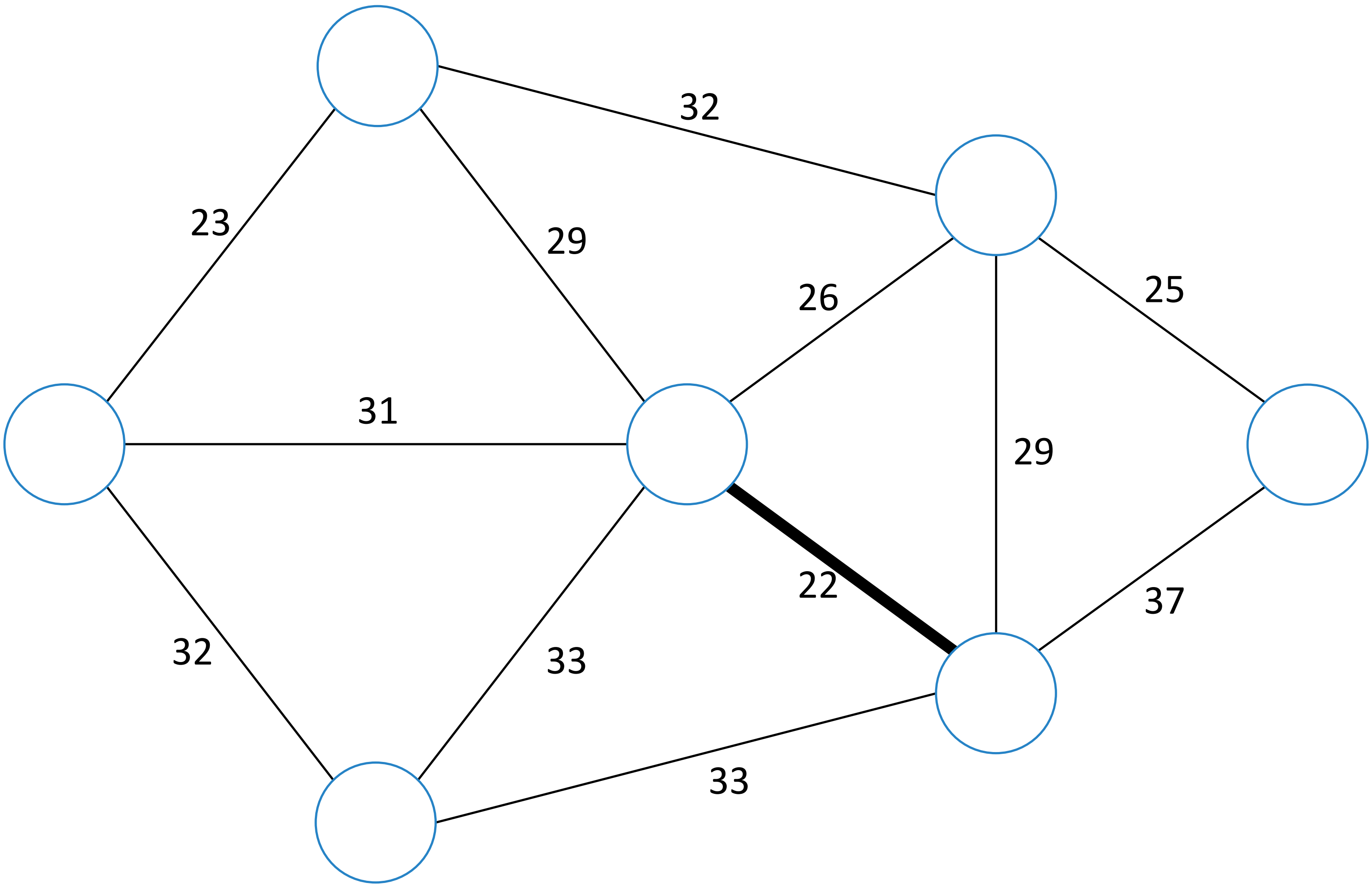
Agregar $\boldsymbol{e}$ a $\boldsymbol{T}$

$\boldsymbol{return}\ \boldsymbol{T}$

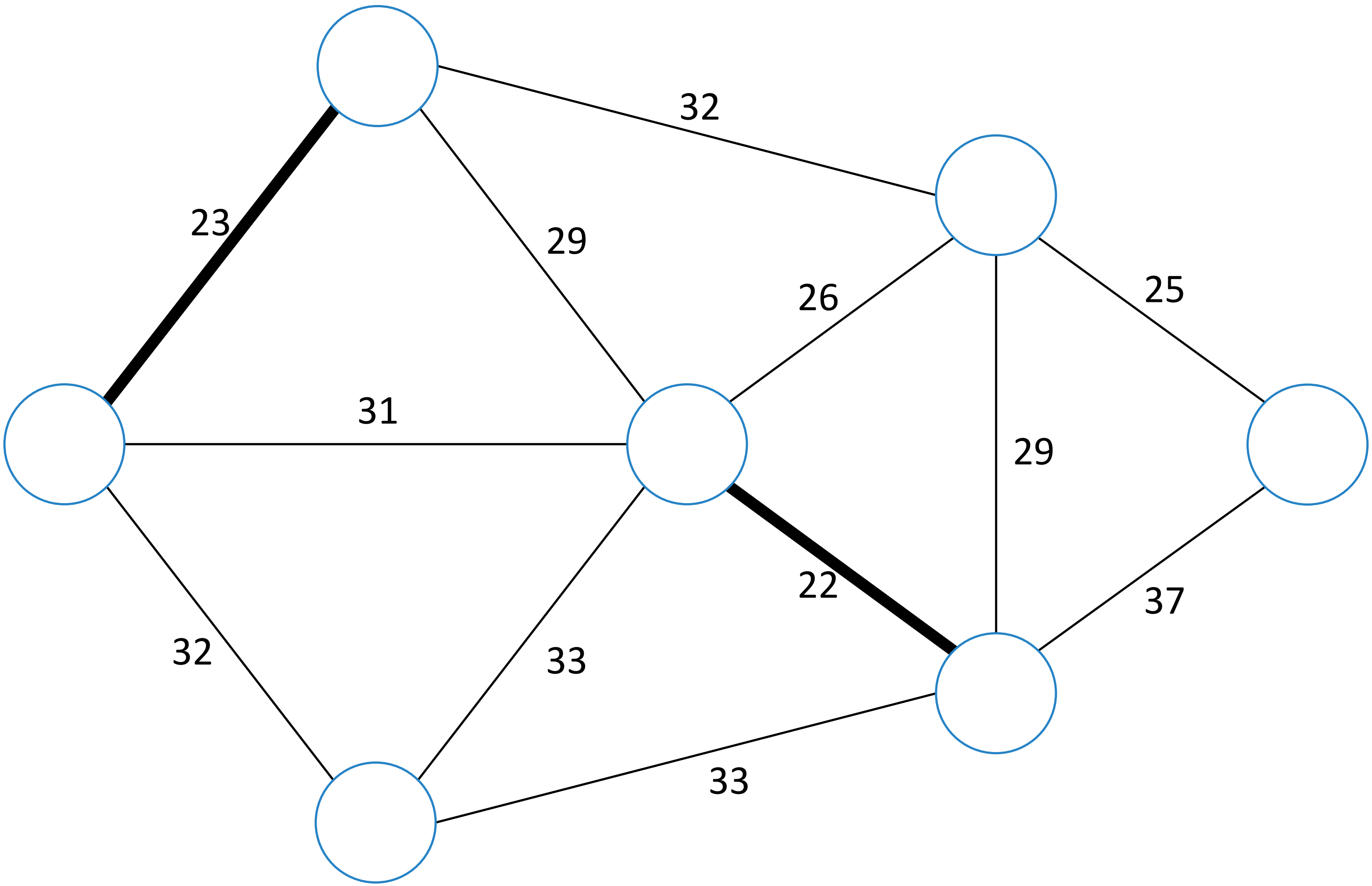
Kruskal en acción



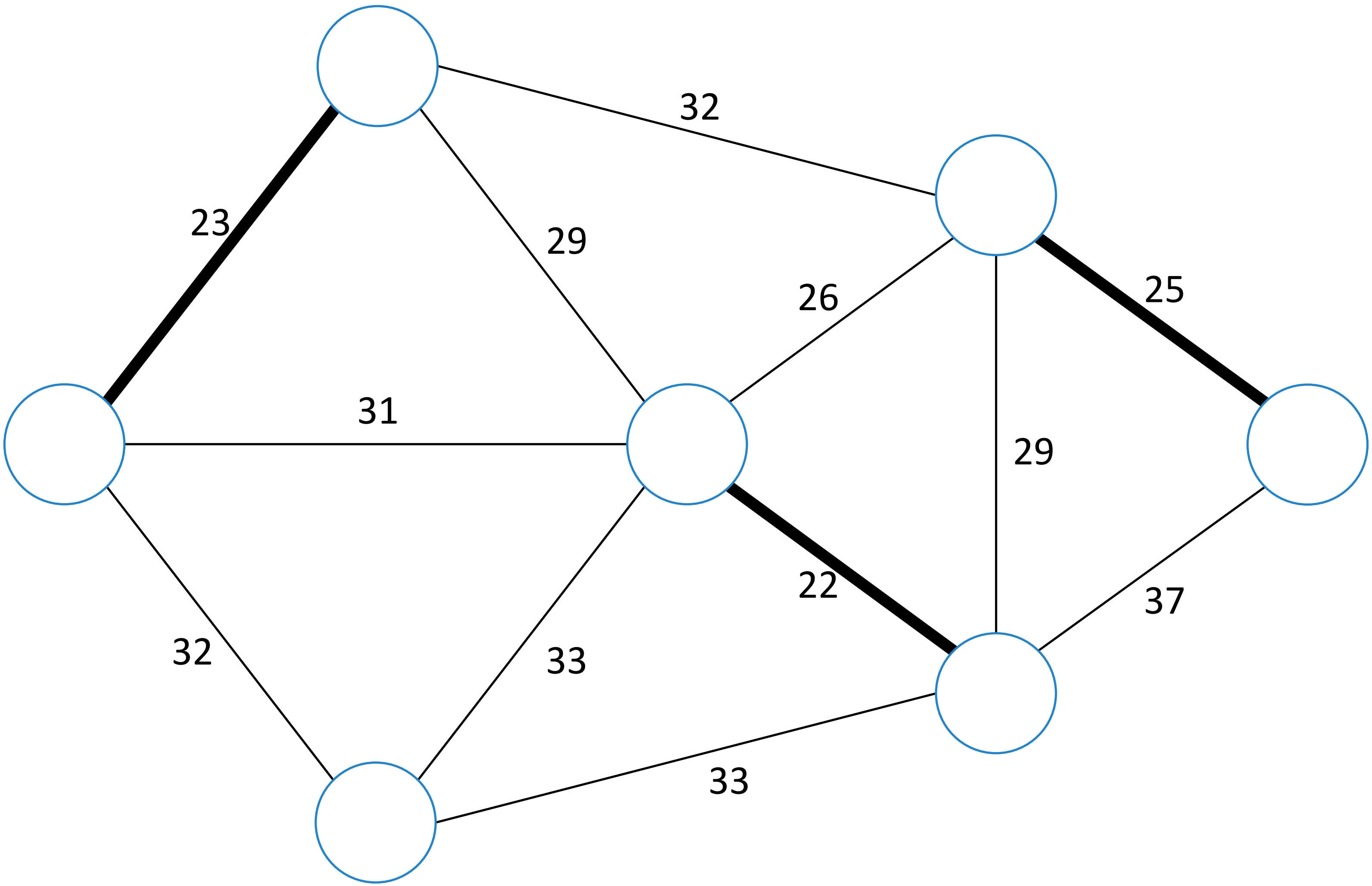
Kruskal en acción



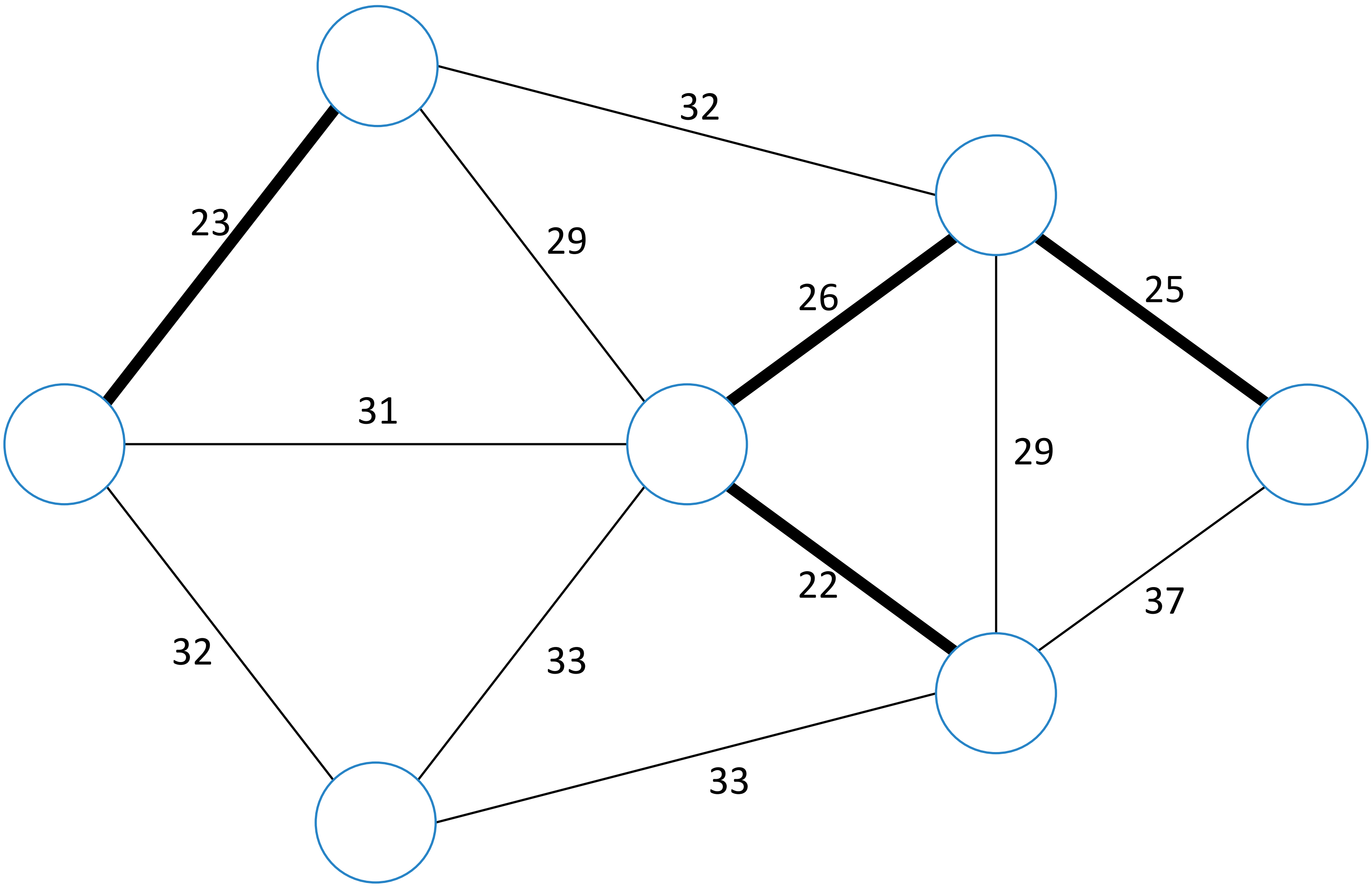
Kruskal en acción



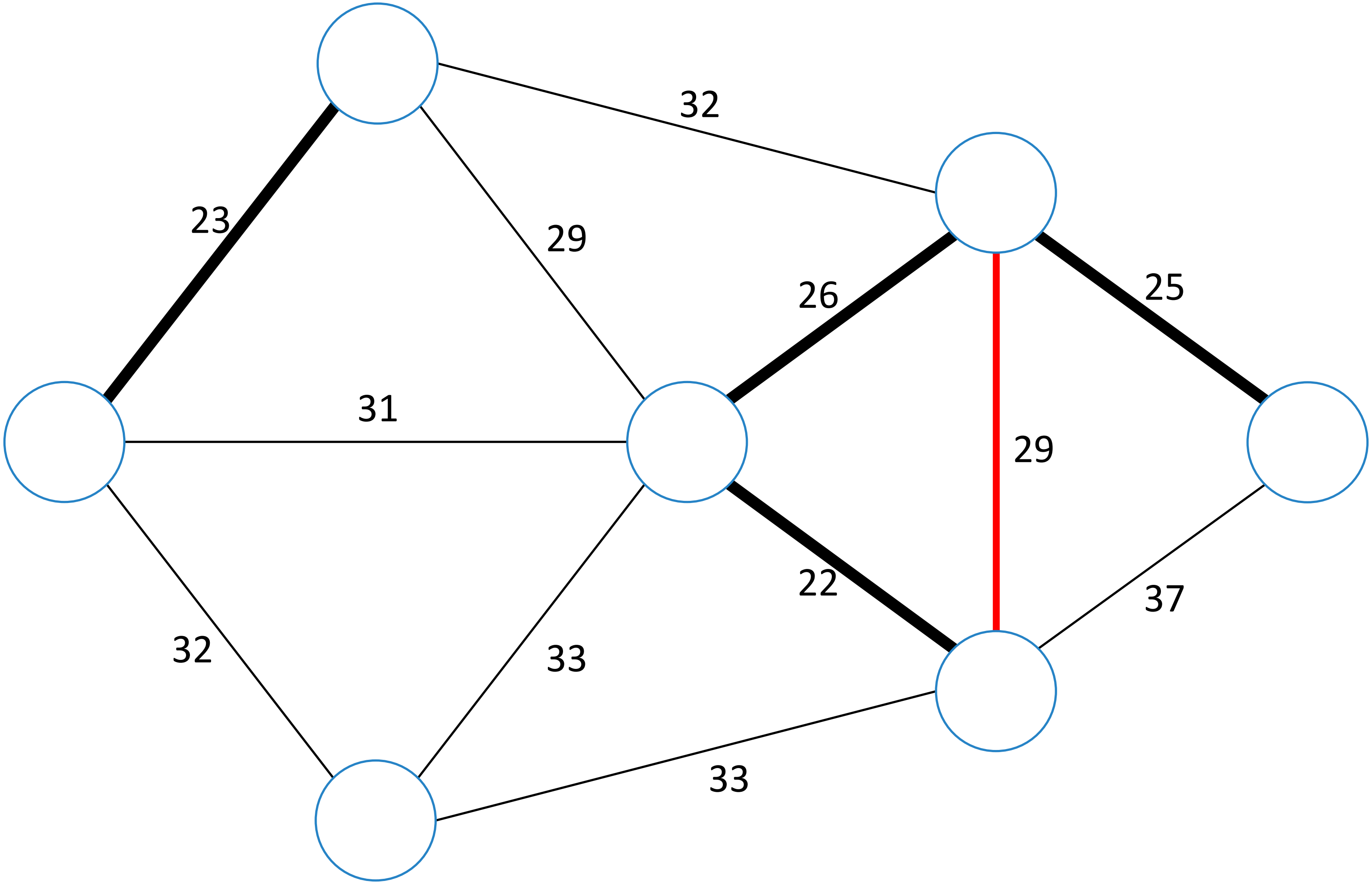
Kruskal en acción



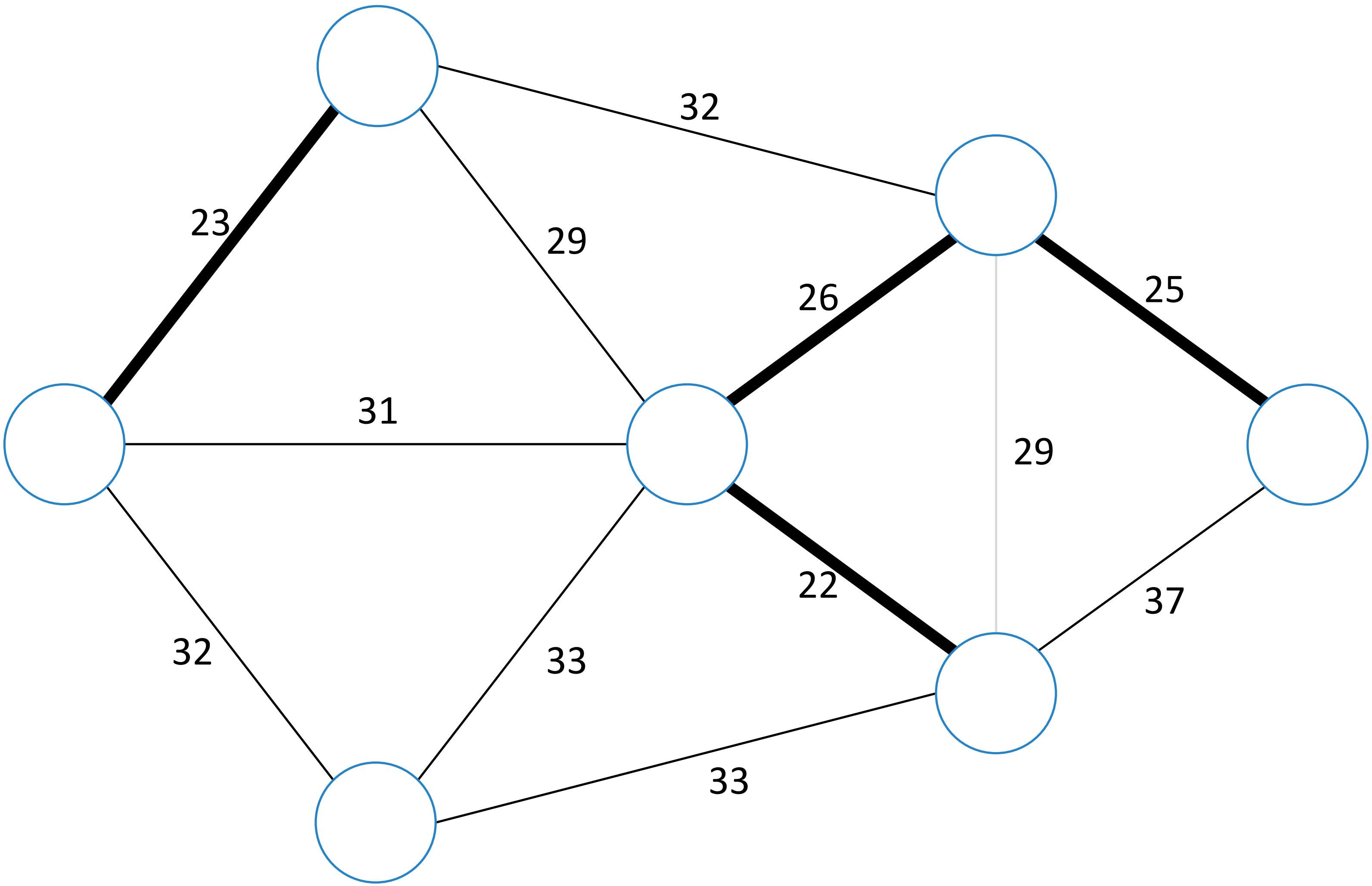
Kruskal en acción



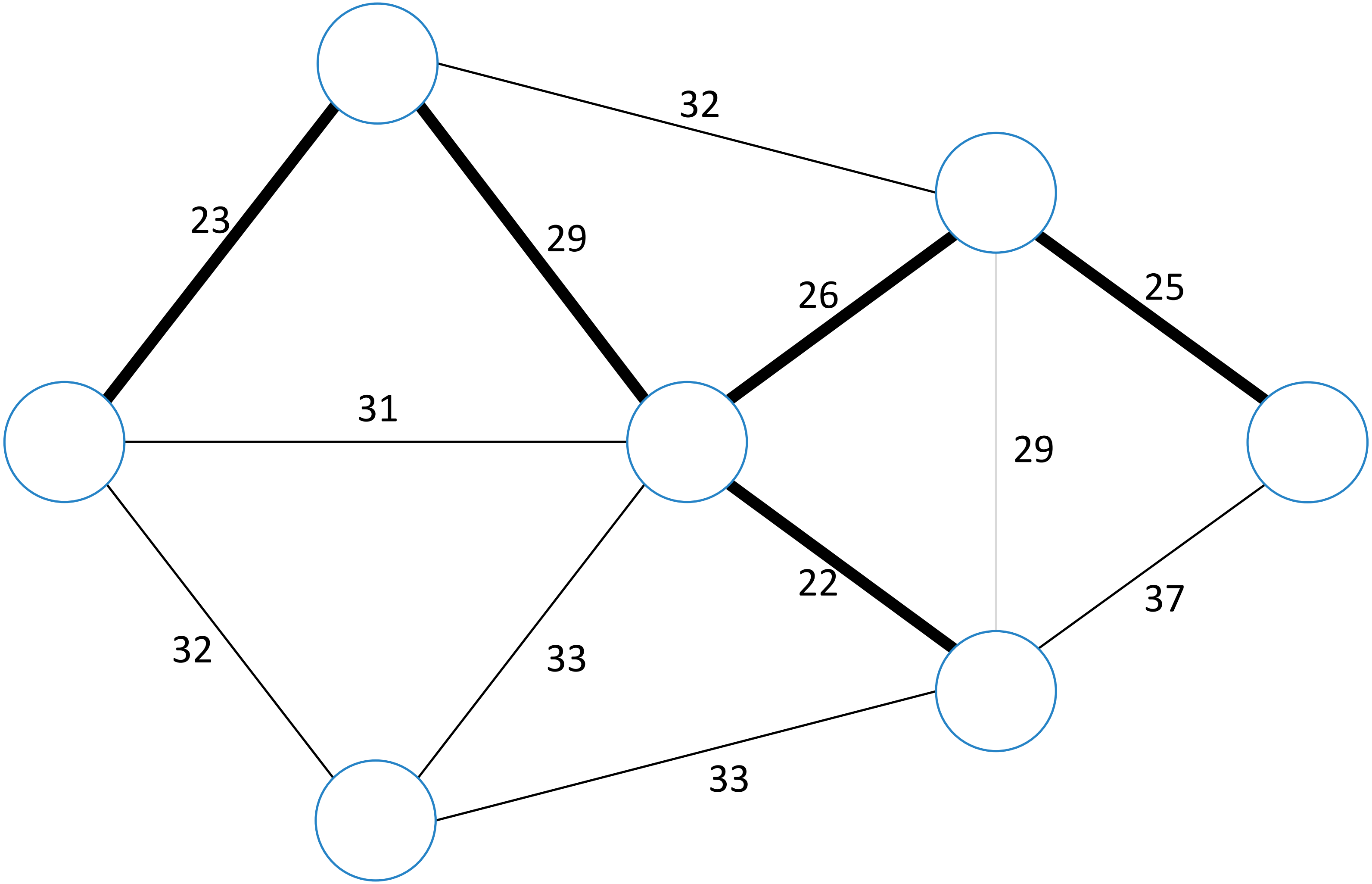
Kruskal en acción



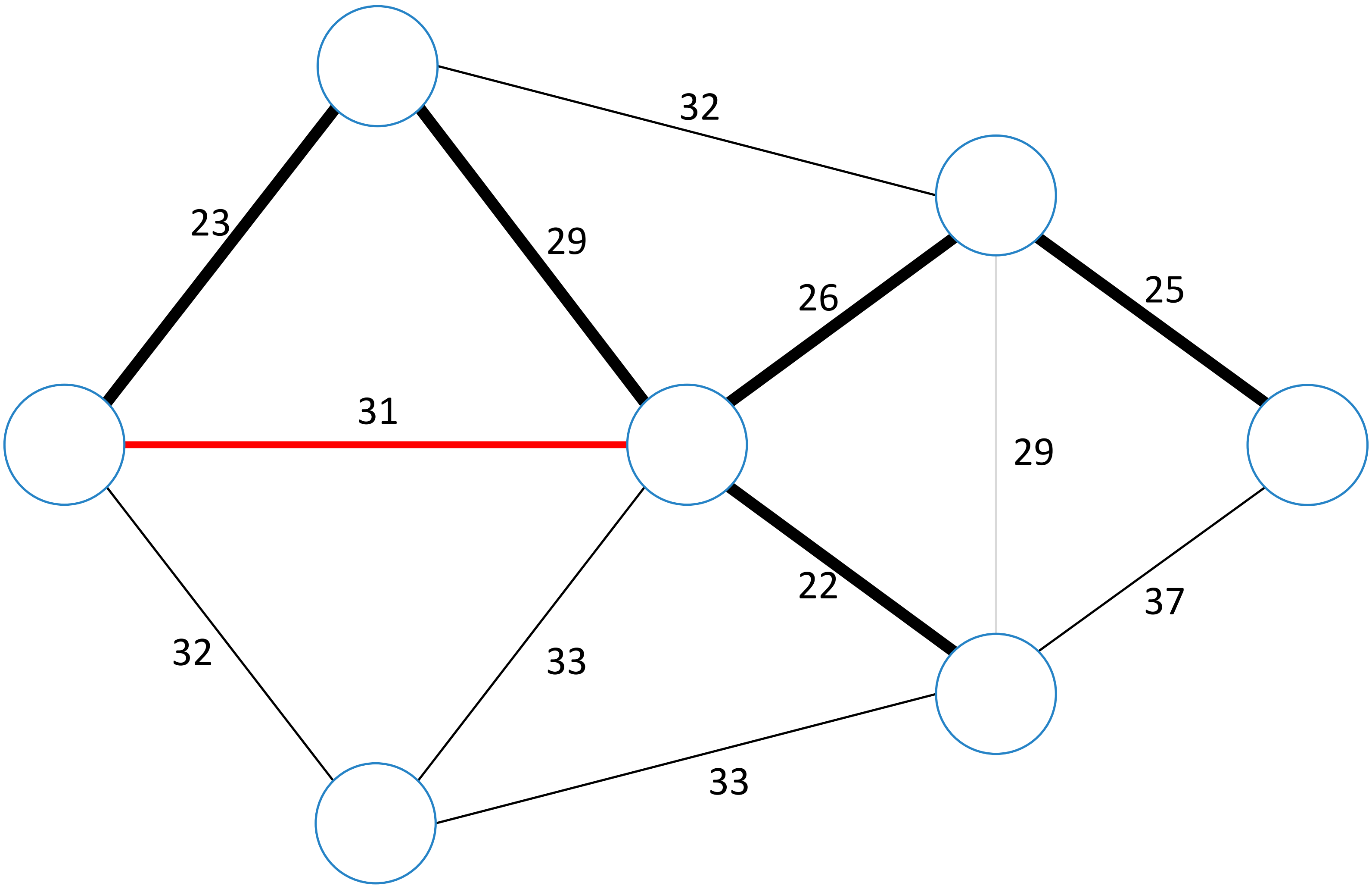
Kruskal en acción



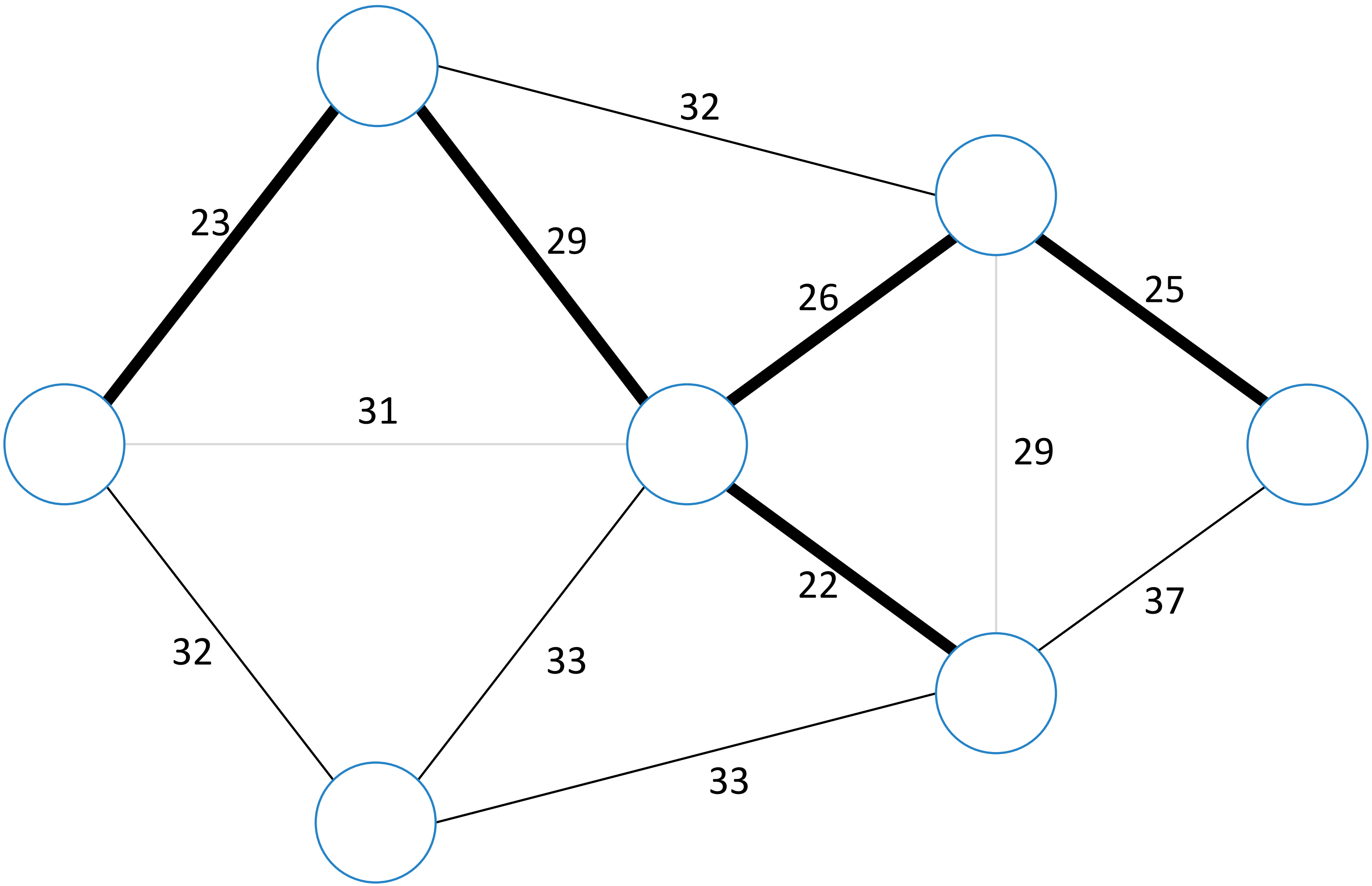
Kruskal en acción



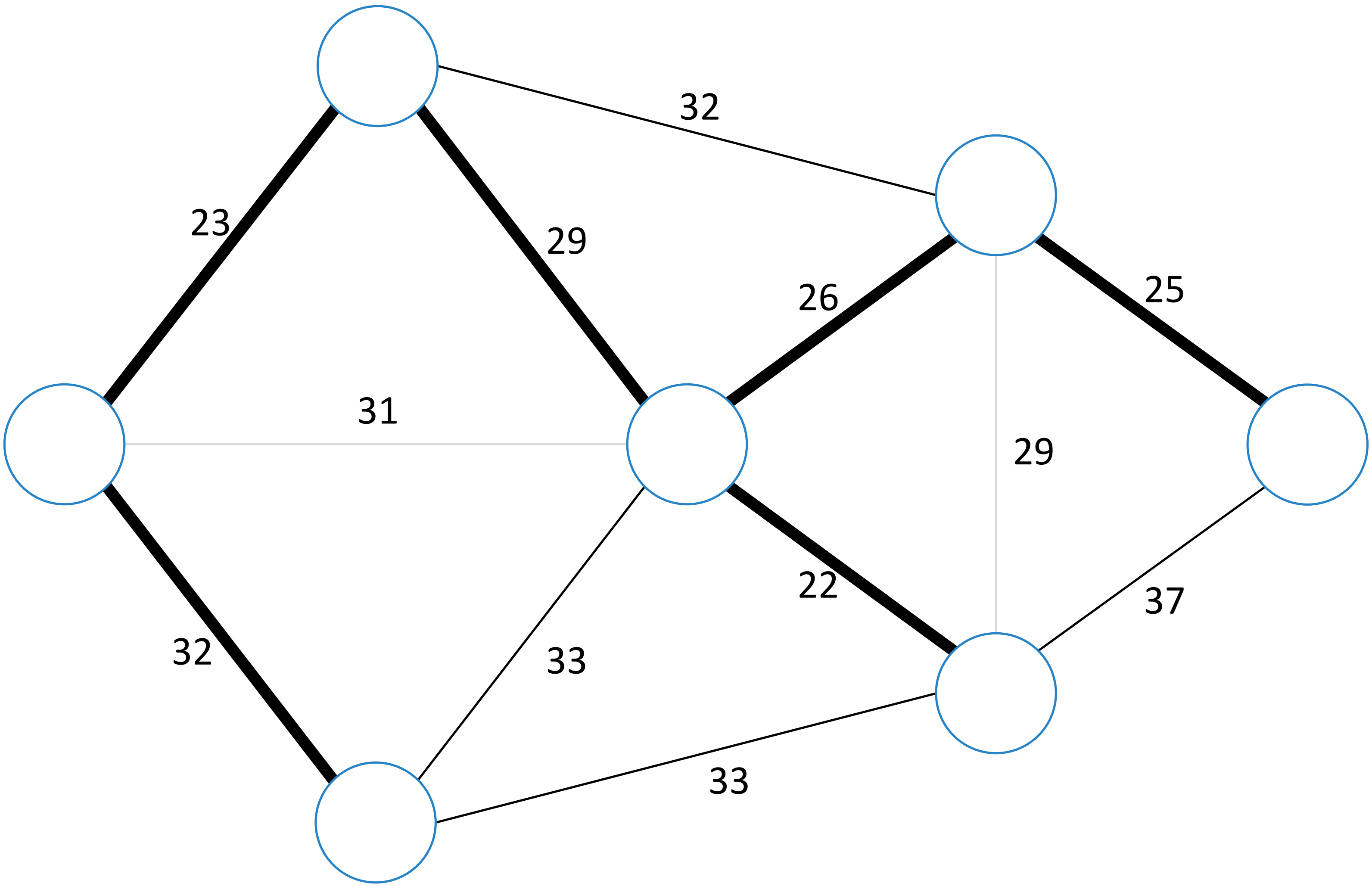
Kruskal en acción



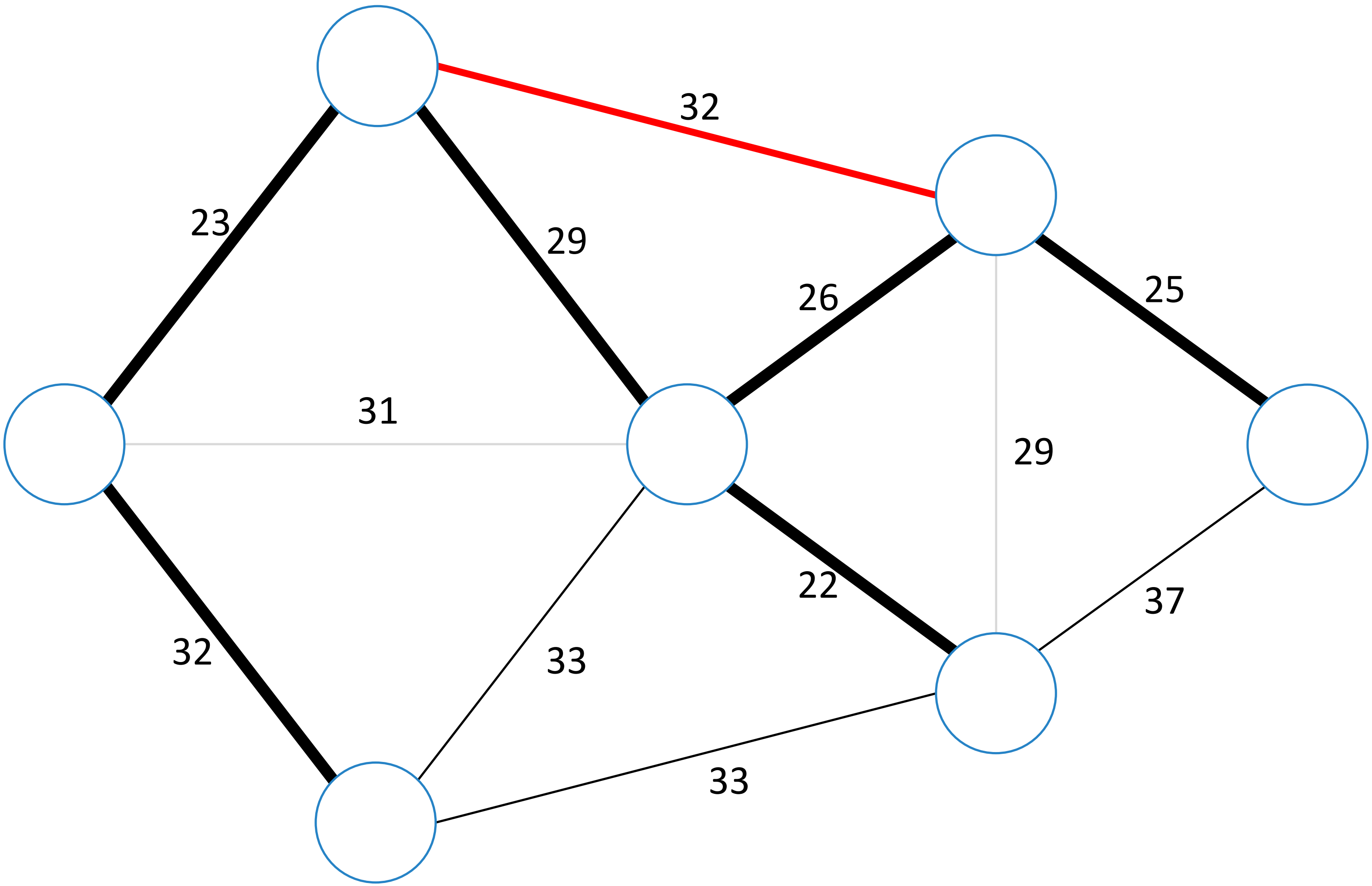
Kruskal en acción



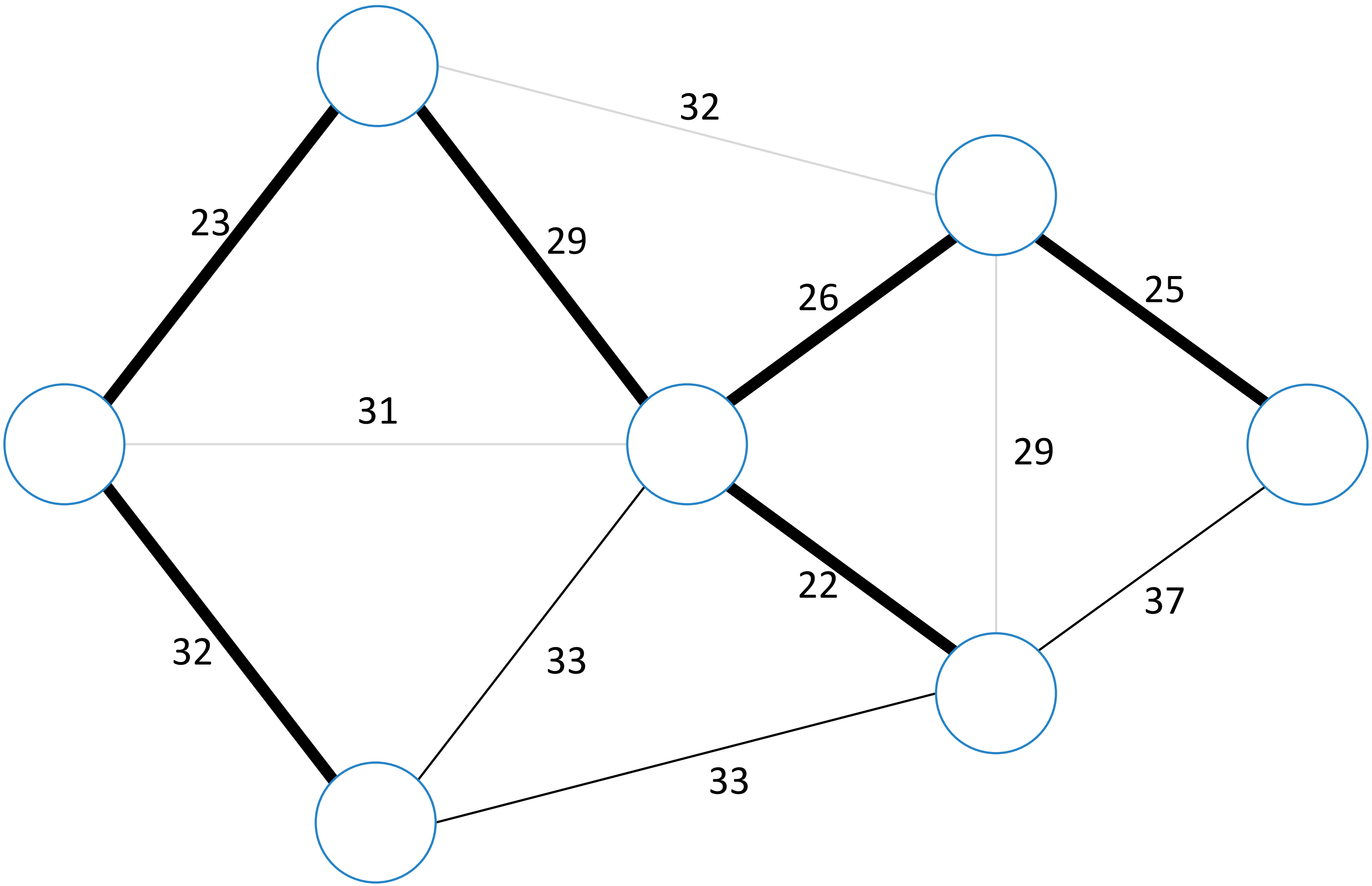
Kruskal en acción



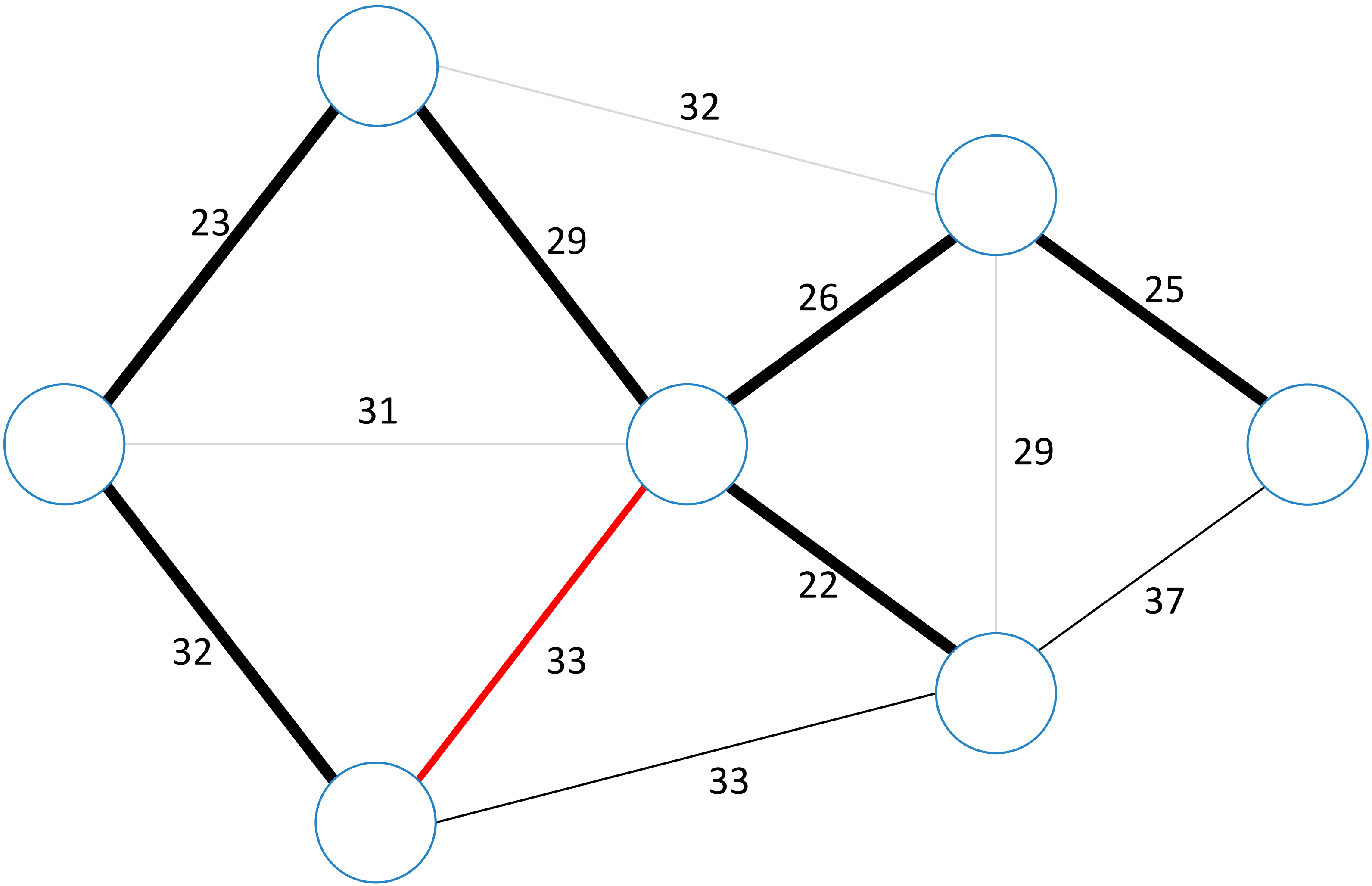
Kruskal en acción



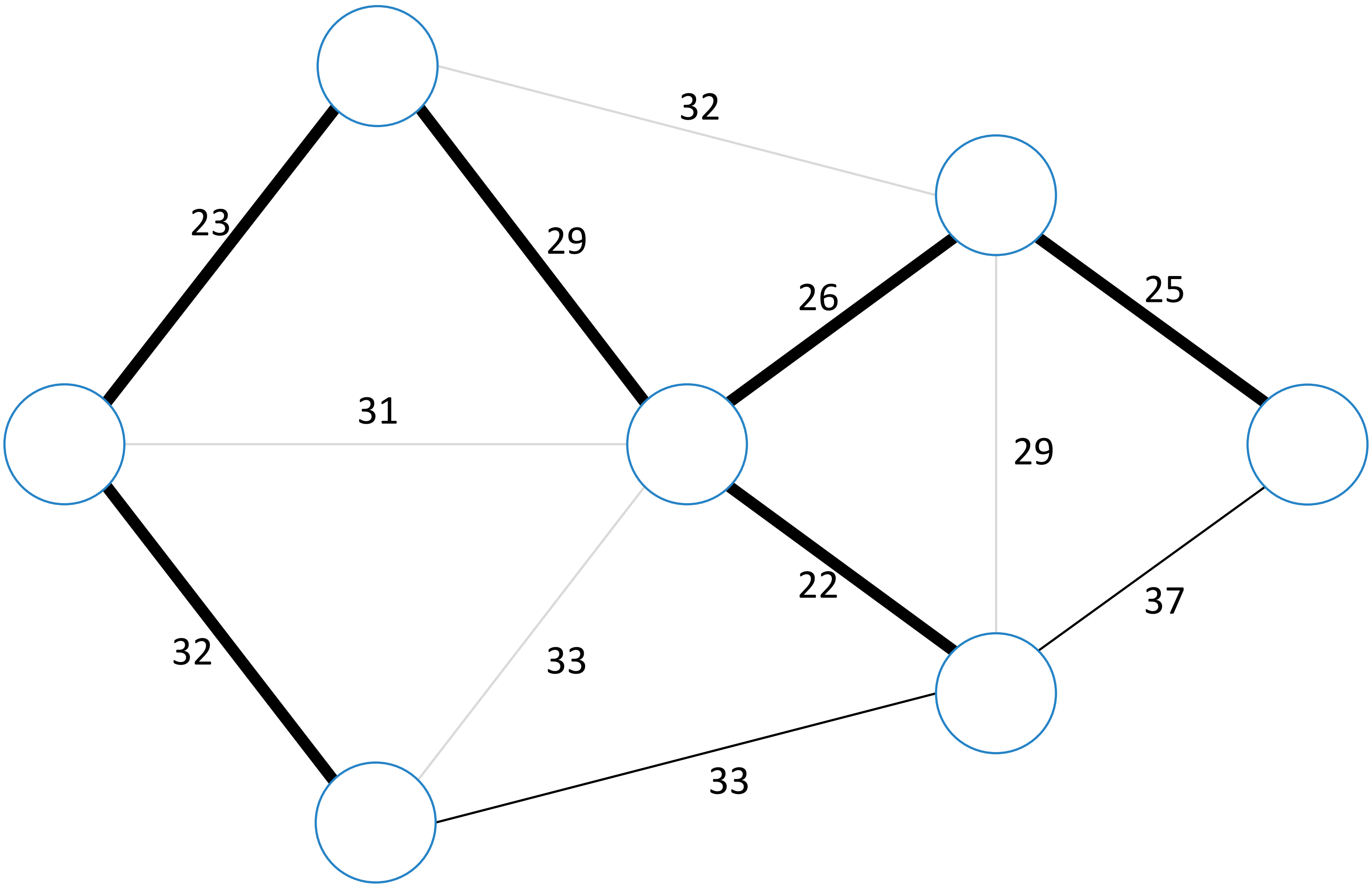
Kruskal en acción



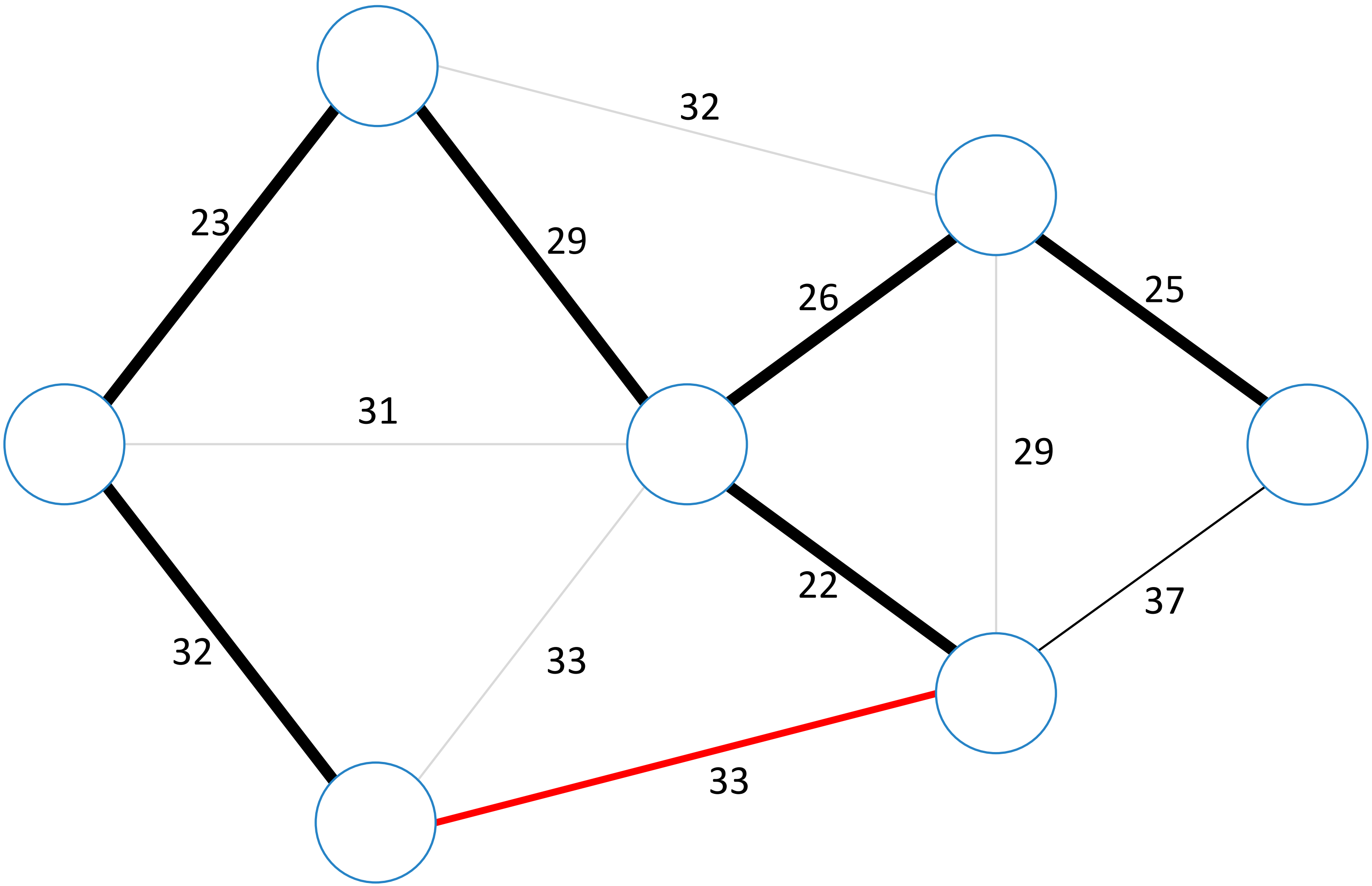
Kruskal en acción



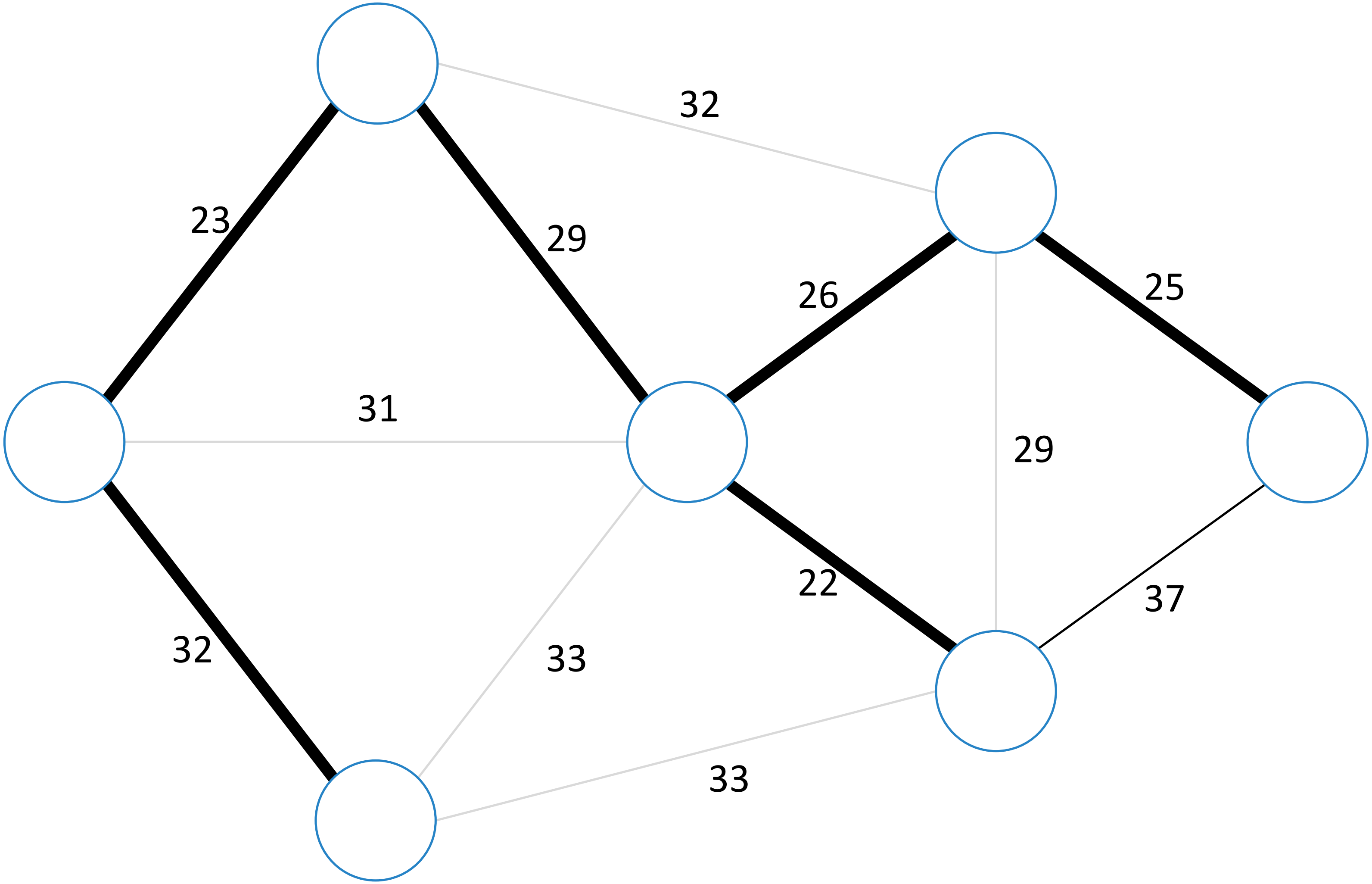
Kruskal en acción



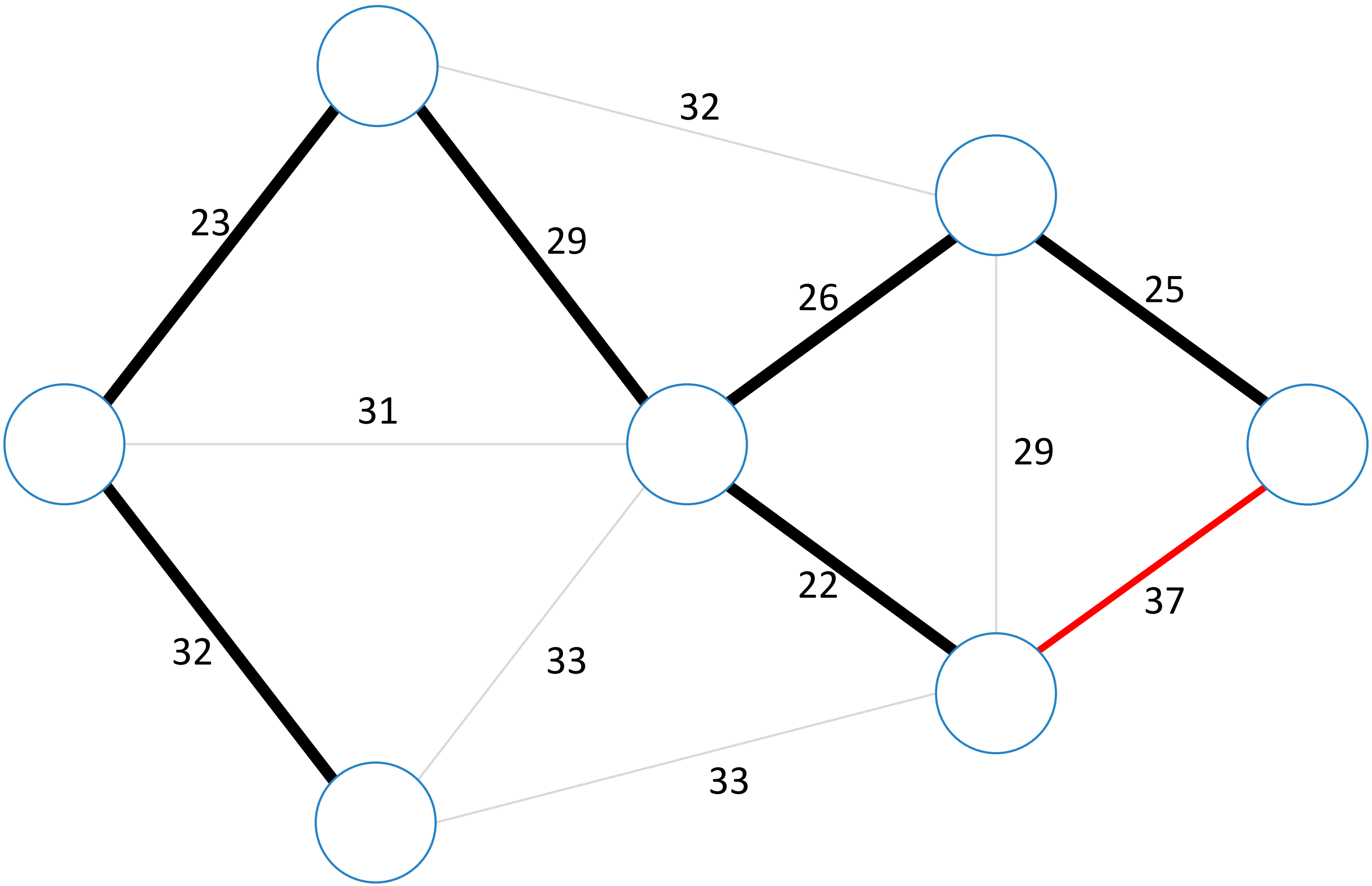
Kruskal en acción



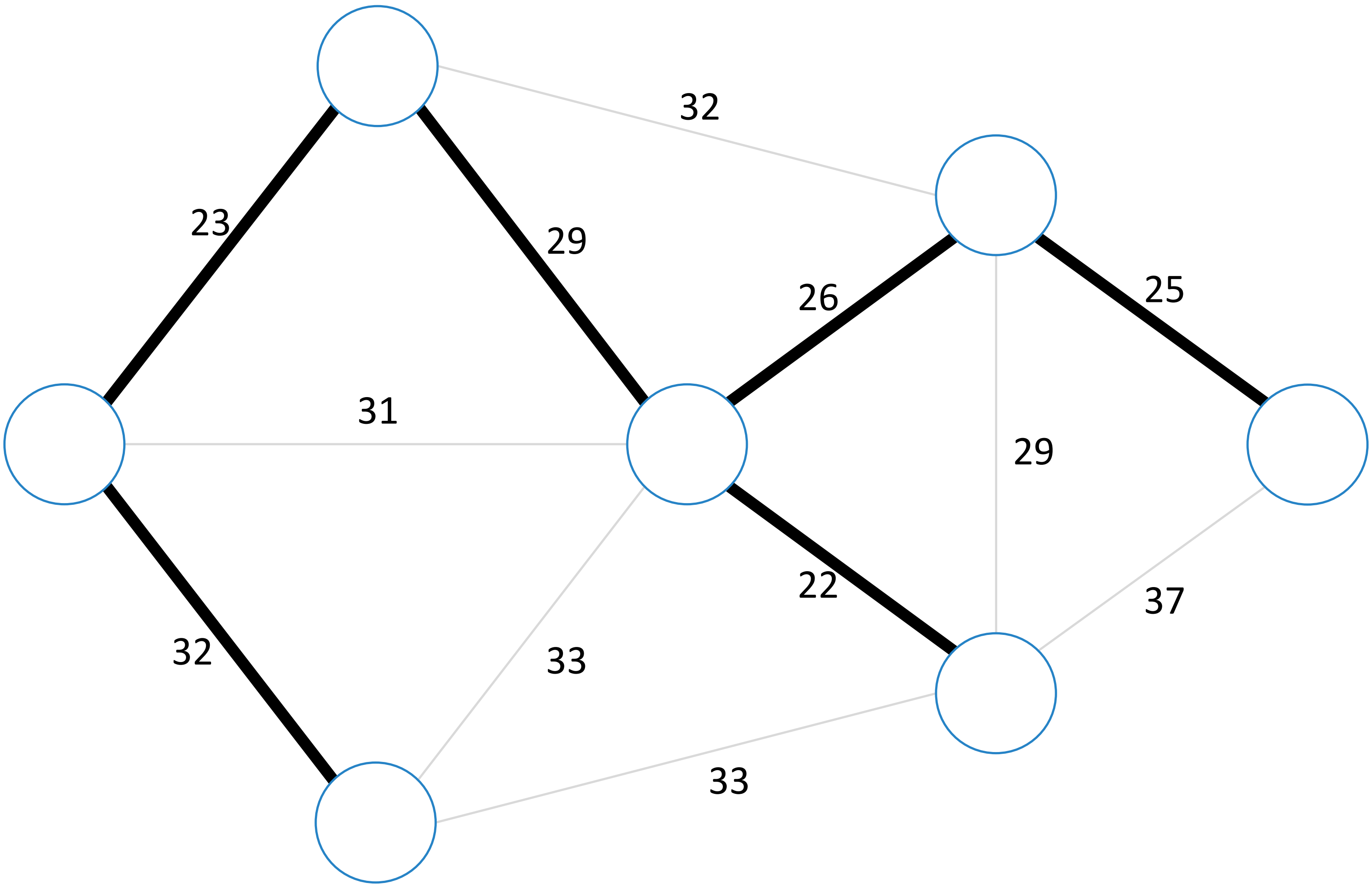
Kruskal en acción



Kruskal en acción



Kruskal en acción



Correctitud

Demuestra que el algoritmo de Kruskal es correcto

Tip: demuestra por separado que el resultado es

- Un árbol
- De cobertura
- Mínimo

Un detalle no menor

$kruskal(G(V,E))$:

Ordenar E por costo, de menor a mayor

$T \leftarrow \emptyset$

$foreach\ e \in E$:

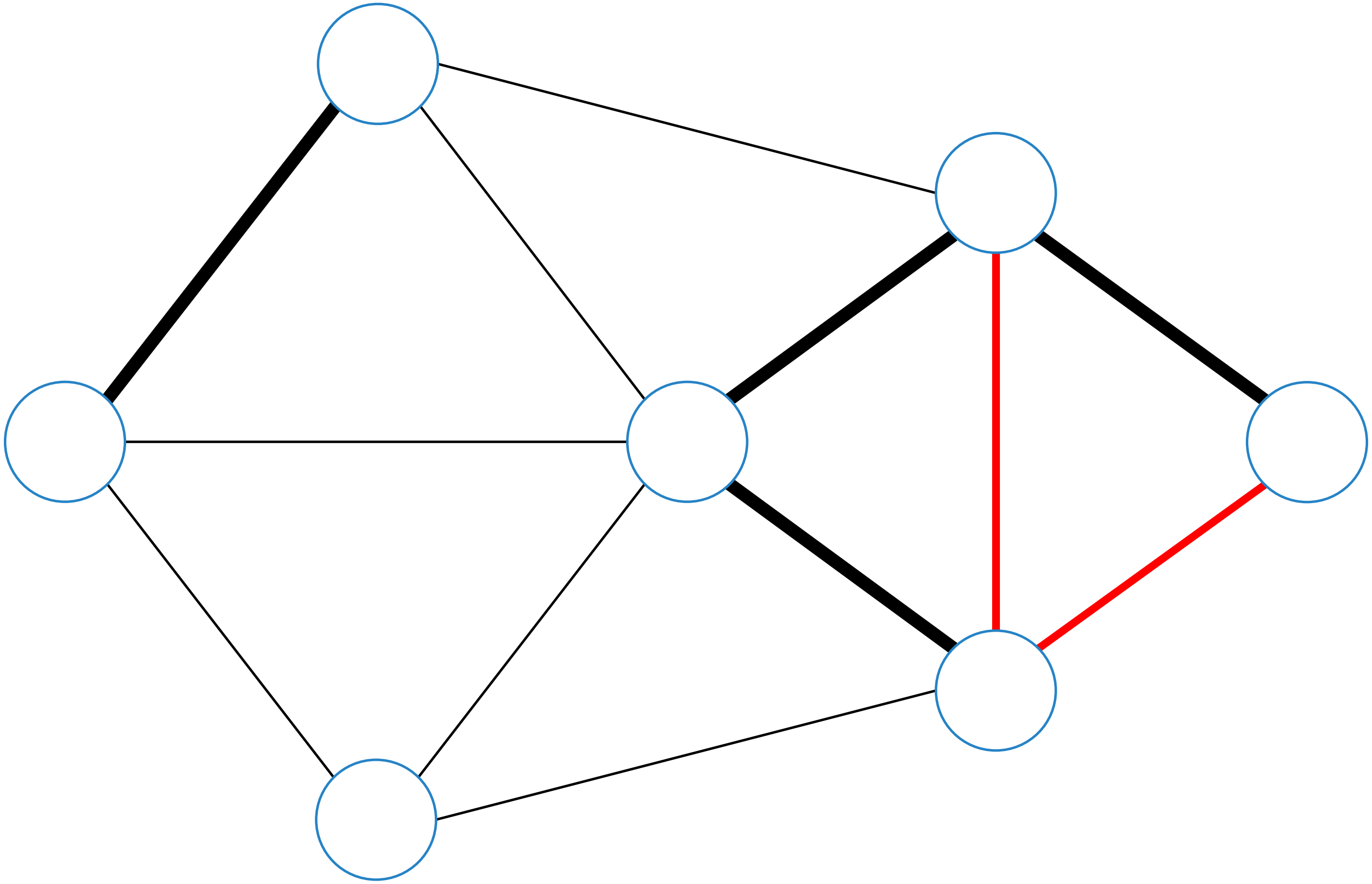
if agregar e a T no forma un ciclo:

Agregar e a T

$return\ T$

¿Cómo revisamos esto de manera eficiente?

Observación



Agregar (u, v) forma un ciclo **ssi** u y v están en el mismo **sub-árbol**

Conjuntos Disjuntos

Un nodo puede pertenecer a un solo **sub-árbol** del grafo

Los **conjuntos** de nodos de cada sub-árbol son **disjuntos**

¿Cómo podemos modelar esto para aprovecharlo?

Kruskal con conjuntos

$\boldsymbol{kruskal}\left(\boldsymbol{G}(\boldsymbol{V}, \boldsymbol{E})\right)$:

Ordenar $\boldsymbol{E}$ por costo, de menor a mayor

$\boldsymbol{T} \leftarrow \emptyset$

$\boldsymbol{foreach}$ $(\boldsymbol{u}, \boldsymbol{v}) \in \boldsymbol{E}$:

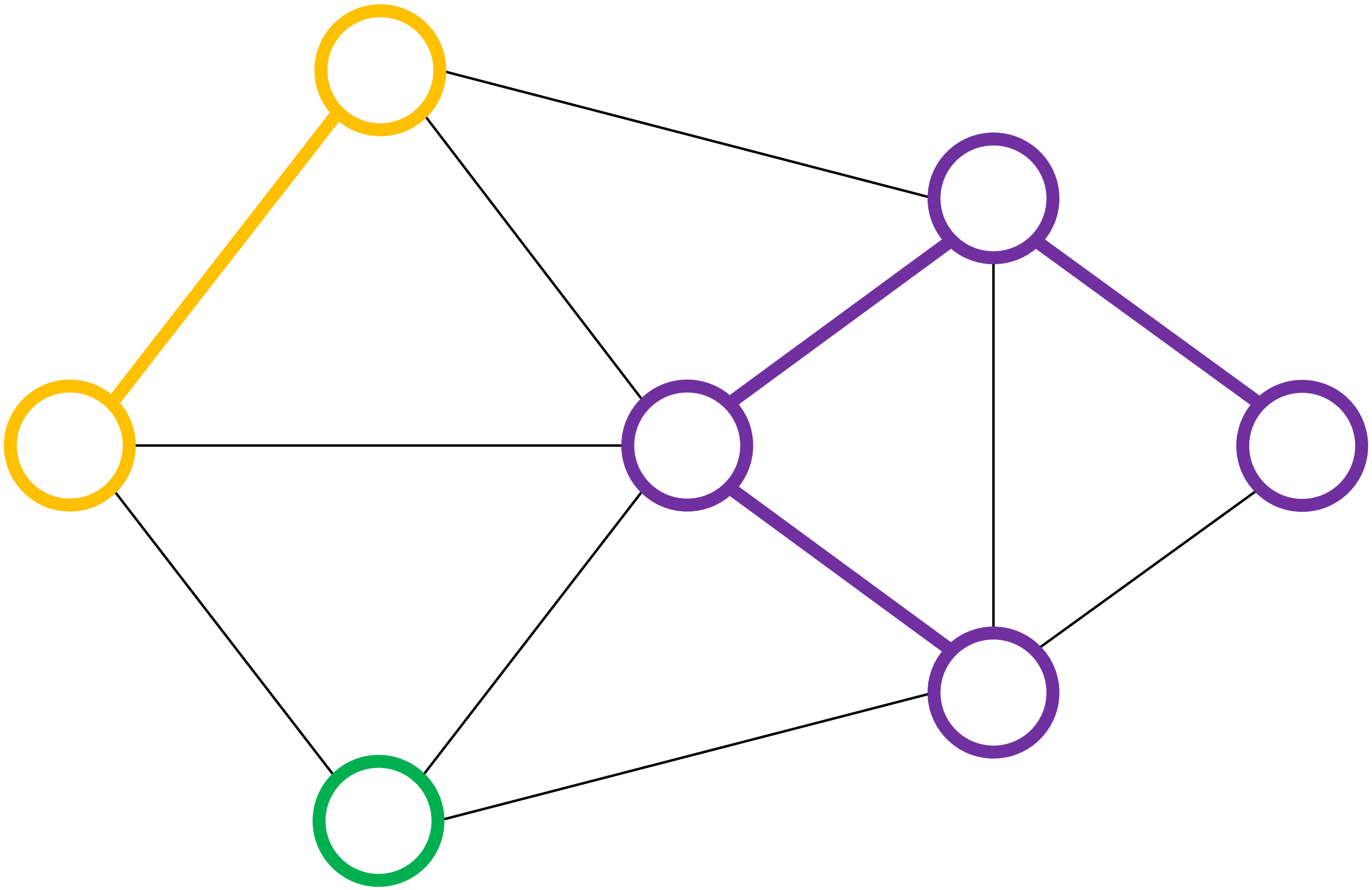
$\boldsymbol{if}$ si $\boldsymbol{u}$ y $\boldsymbol{v}$ no están en el mismo conjunto:

Agregar $(\boldsymbol{u}, \boldsymbol{v})$ a $\boldsymbol{T}$

Combinar los conjutos de $\boldsymbol{u}$ y $\boldsymbol{v}$

$\boldsymbol{return}$ $\boldsymbol{T}$

Conjuntos de sub-árboles



Agregar una arista significa unir dos conjuntos

Conjuntos Disjuntos

Nos interesan dos cosas:

- **Identificar** en que conjunto está un elemento
- **Unir** dos conjuntos

¿Cómo podemos hacer esto de manera eficiente?

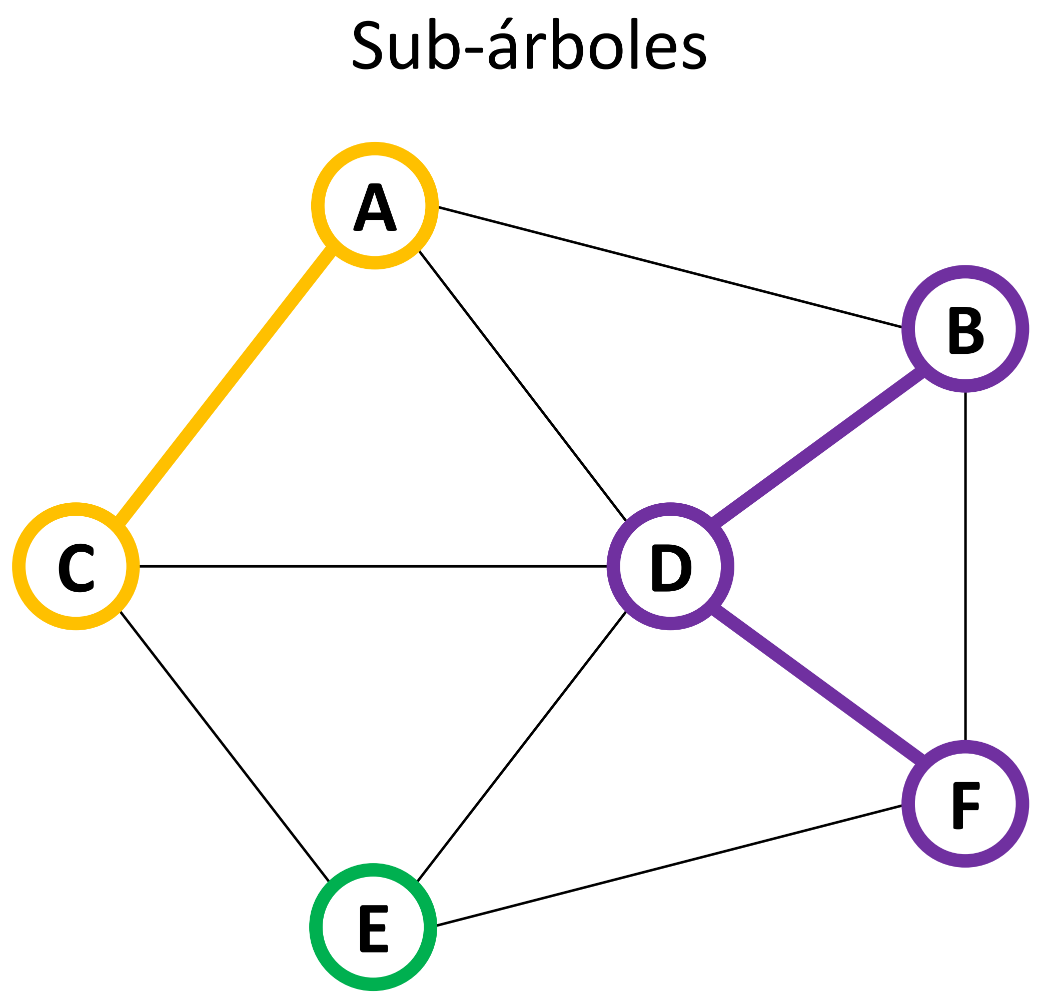
Representación

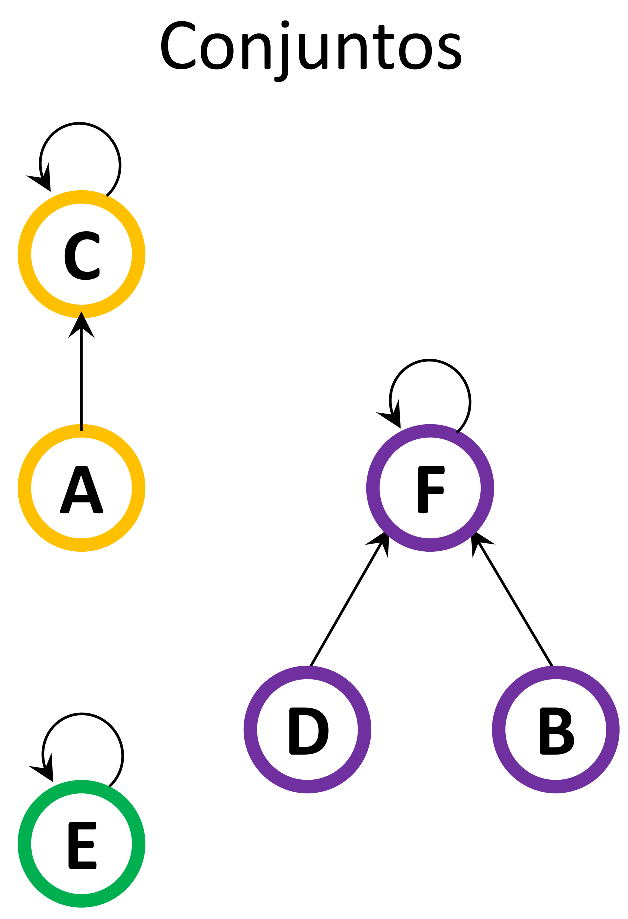
Para cada conjunto, escogemos un **representante**

Cada nodo tiene una **referencia** a su representante

Dos nodos están en el **mismo** conjunto si comparten representante

Conjuntos disjuntos





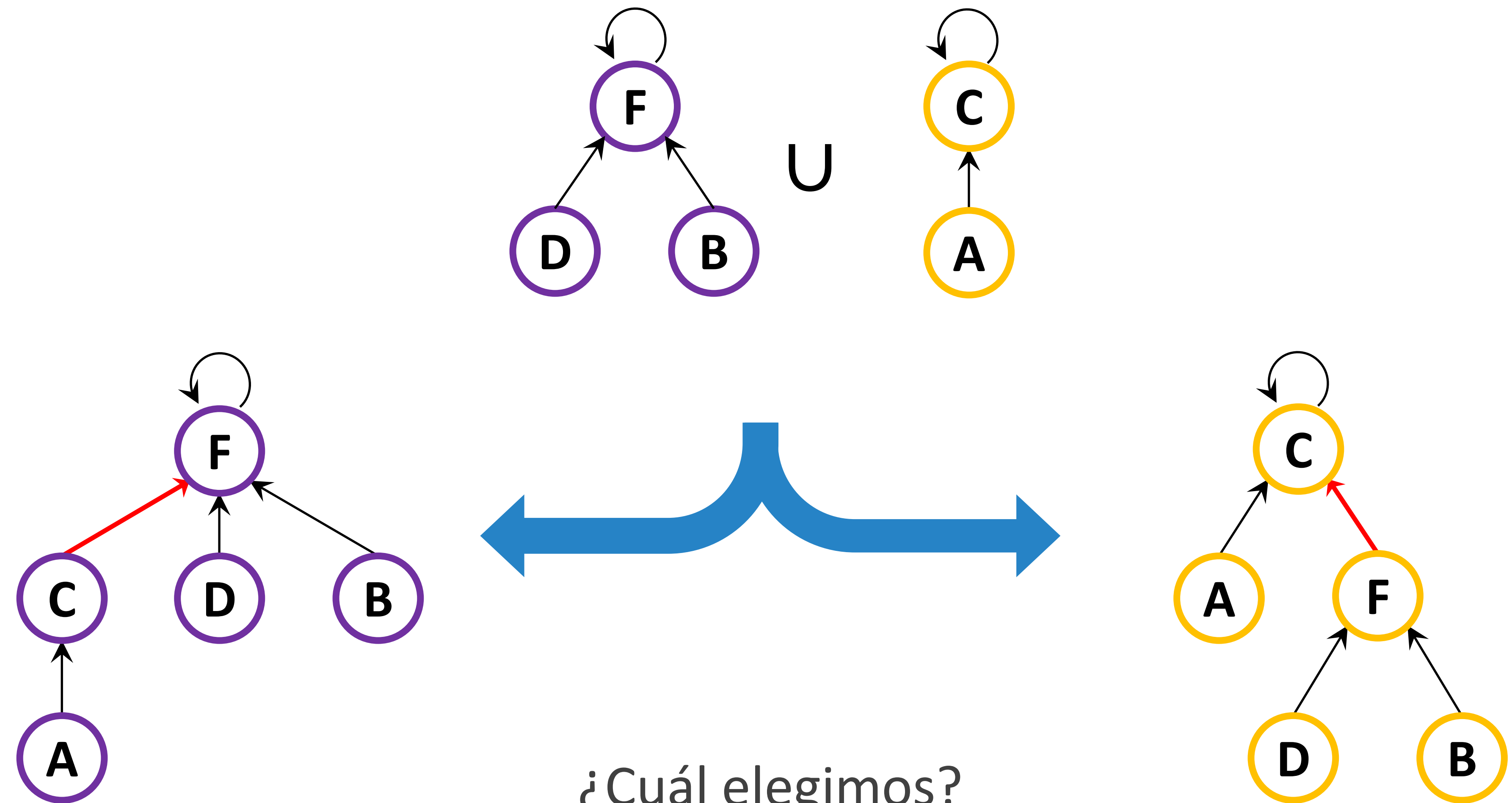
Conjuntos disjuntos

Definimos 3 funciones para esta estructura:

- $\boldsymbol{make\ set}(\boldsymbol{x})$: inicializa $\boldsymbol{x}$ como su propio representante
- $\boldsymbol{find\ set}(\boldsymbol{x})$: retorna el representante del nodo $\boldsymbol{x}$
- $\boldsymbol{union}(\boldsymbol{x}, \boldsymbol{y})$: combina los conjuntos de $\boldsymbol{x}$ e $\boldsymbol{y}$

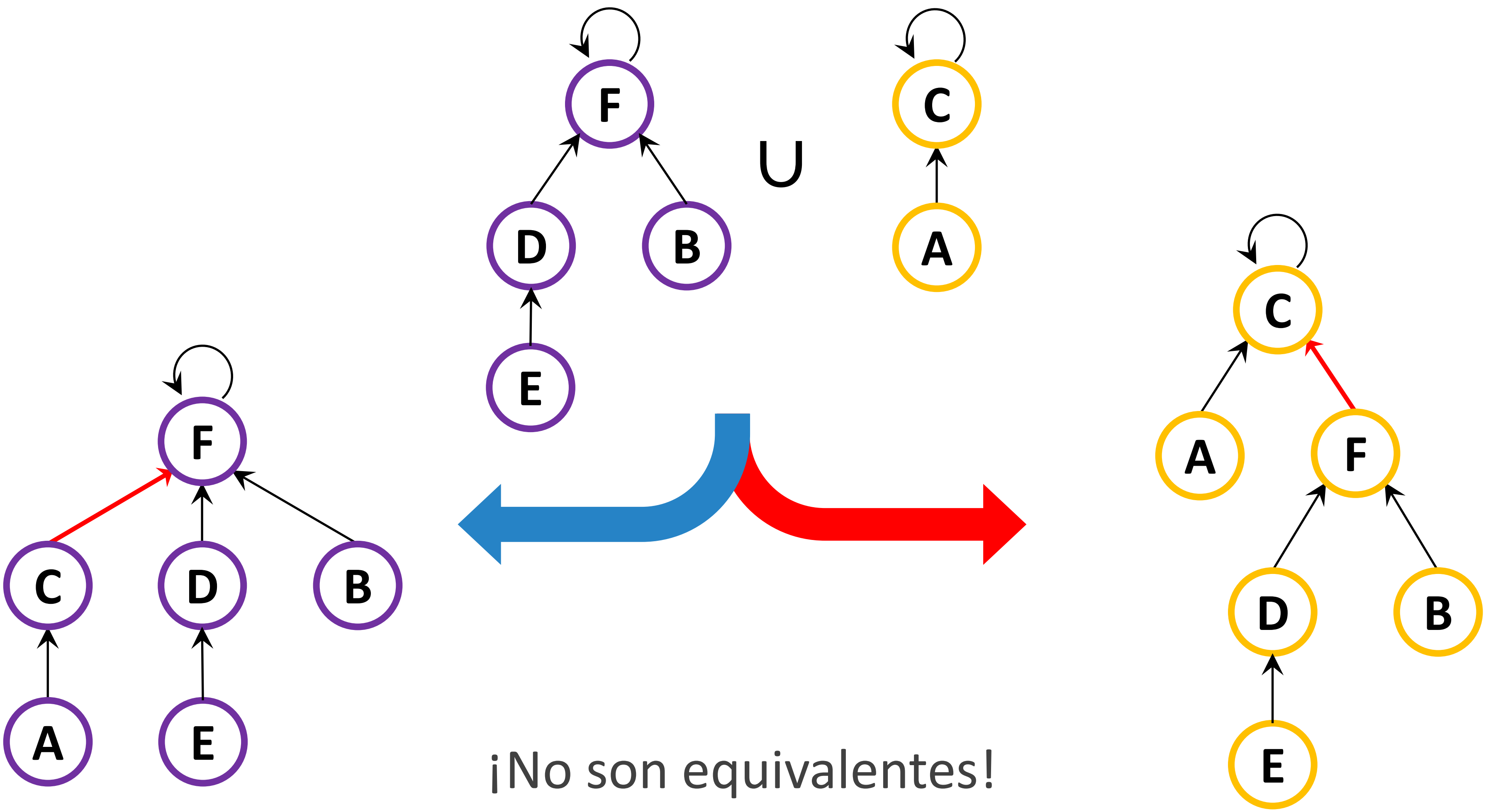
Todas son bastante directas, pero pueden mejorarse. ¿Cómo?

Unión



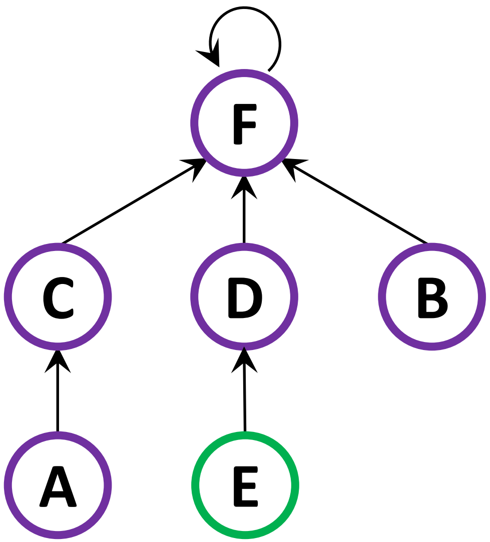
¿Cuál elegimos?

Unión



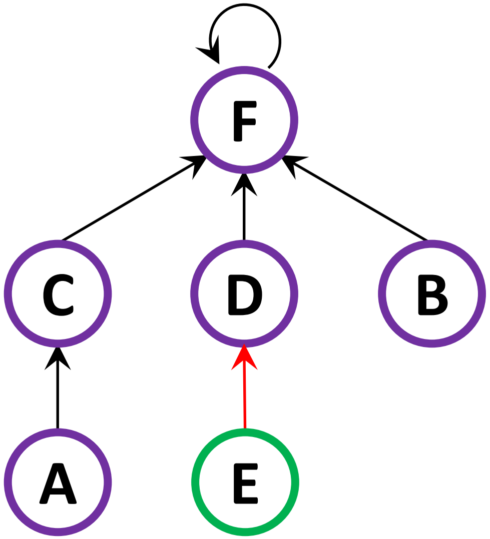
¡No son equivalentes!

Find-set



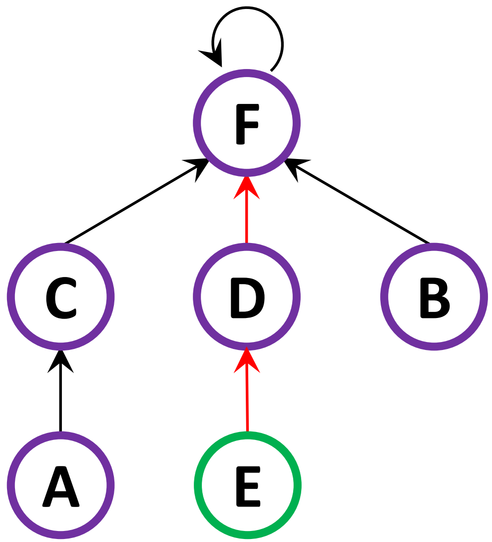
$$find\ set(\boldsymbol{E}) =$$

Find-set



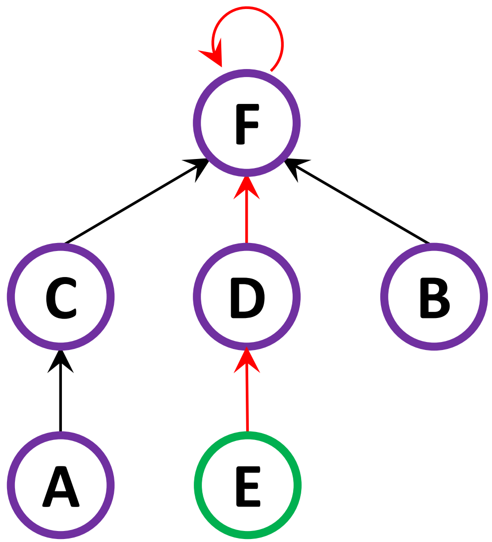
$$\boldsymbol{find\ set(E)} =$$

Find-set



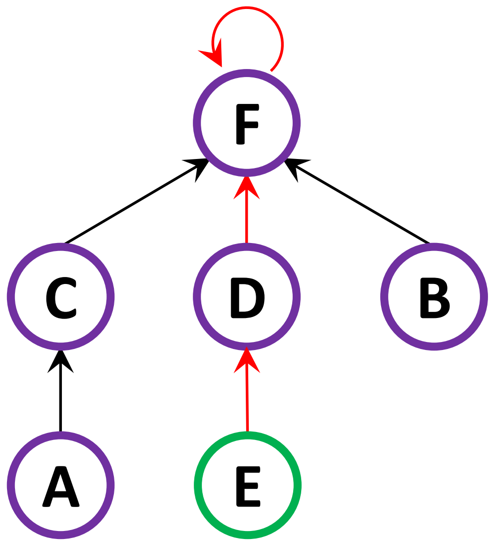
$$\boldsymbol{find\ set(E) =}$$

Find-set



$$\boldsymbol{find\ set(E)} =$$ F

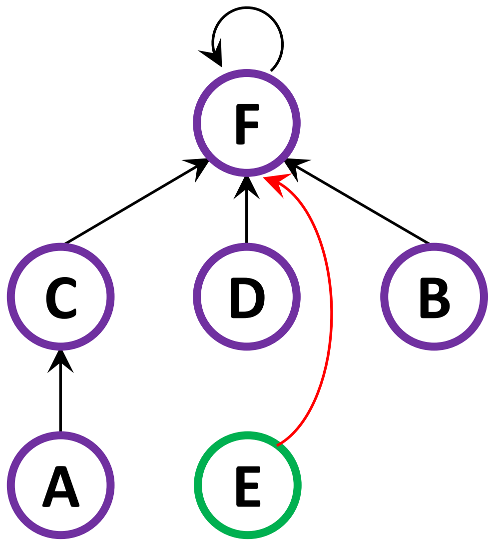
Find-set



$$find\ set(\boldsymbol{E}) = \text{F}$$

¿Cómo podemos aprovechar que ya tenemos esta información?

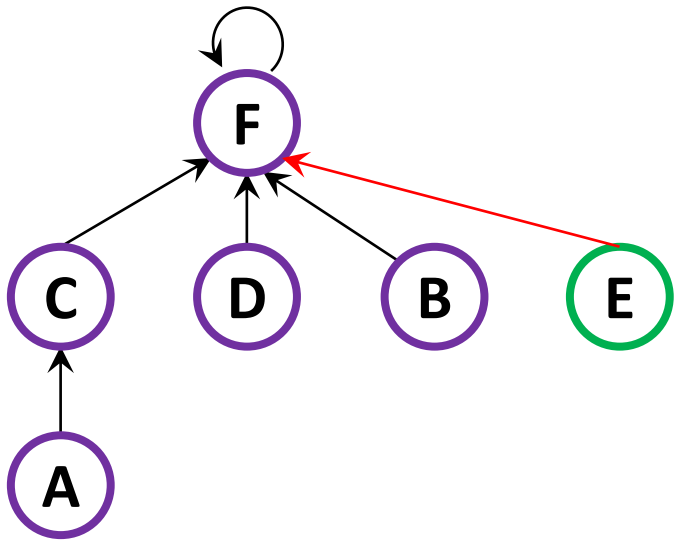
Compresión de caminos



$$\boldsymbol{find\ set(E)} = \text{F}$$

¡Acortando el camino al representante!

Compresión de caminos



$$find\ set(E) = $$ F

Complejidad

Si pretendemos operar sobre n conjuntos disjuntos

¿Cuál es la complejidad de estas operaciones?

¿Y usando las mejoras?

Kruskal con conjuntos disjuntos

$\boldsymbol{kruskal}\big(\boldsymbol{G}(\boldsymbol{V},\boldsymbol{E})\big)$:

Ordenar $\boldsymbol{E}$ por costo, de menor a mayor

$\boldsymbol{foreach}\ \boldsymbol{v} \in \boldsymbol{V}$: $\boldsymbol{make\ set}(\boldsymbol{v})$

$\boldsymbol{T} \leftarrow \emptyset$

$\boldsymbol{foreach}\ (\boldsymbol{u},\boldsymbol{v}) \in \boldsymbol{E}$:

$\boldsymbol{if}\ \boldsymbol{find\ set}(\boldsymbol{u}) \neq \boldsymbol{find\ set}(\boldsymbol{v})$:

$\boldsymbol{T} \leftarrow \boldsymbol{T} \cup \{(\boldsymbol{u},\boldsymbol{v})\}$

$\boldsymbol{union}(\boldsymbol{u},\boldsymbol{v})$

$\boldsymbol{return}\ \boldsymbol{T}$

Kruskal

¿Considerando esto, cual es la complejidad de Kruskal?