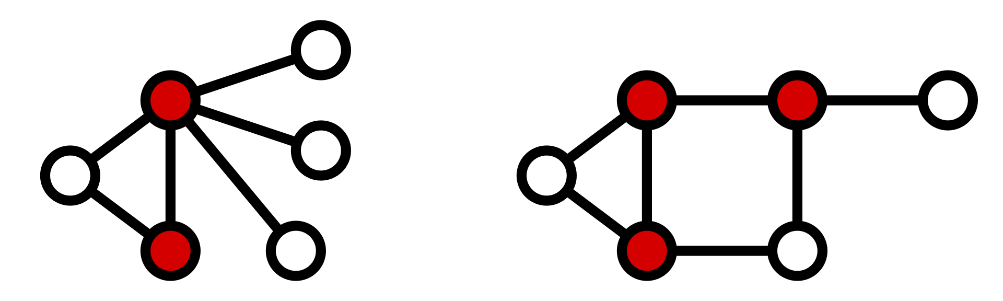
Vertex Cover

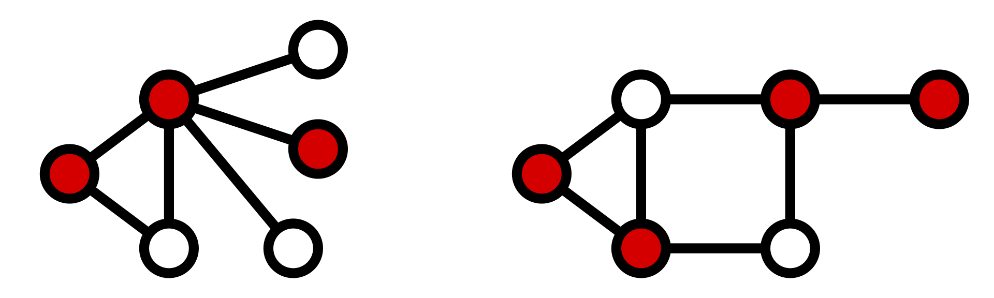
A **vertex cover** of a graph is a set of vertices that includes at least one endpoint of every edge of the graph.

Example:



A **minimum vertex cover** is a vertex cover of smallest possible size.The vertex cover number *τ*{\displaystyle \tau } is the size of a minimum vertex cover, i.e. *τ*{\displaystyle \tau =|V'|} = |*V'*|.

The following figure shows examples of minimum vertex covers in the previous graphs.



Exact evaluation:

For bipartite graphs, the equivalence between vertex cover and maximum matching described by Kőnig's theorem allows the bipartite vertex cover problem to be solved in polynomial time.

For tree graphs, an algorithm finds a minimal vertex cover in polynomial time by finding the first leaf in the tree and adding its parent to the minimal vertex cover, then deleting the leaf and parent and all associated edges and continuing repeatedly until no edges remain in the tree.

Fixed-parameter tractability:

An *exhaustive search algorithm* can solve the problem in time 2*knO*(1), where *k* is the size of the vertex cover. Vertex cover is therefore fixed-parameter tractable, and if we are only interested in small *k*, we can solve the problem in polynomial time. One algorithmic technique that works here is called *bounded search tree algorithm*, and its idea is to repeatedly choose some vertex and recursively branch, with two cases at each step: place either the current vertex or all its neighbours into the vertex cover. The algorithm for solving vertex cover that achieves the best asymptotic dependence on the parameter runs in time *O*(1.2738k + (k\*n)){\displaystyle O(1.2738^{k}+(k\cdot n))}.

Pseudocode:

APPROXIMATION-VERTEX-COVER(G)=

C = ∅

E'= G.E

**while** E' ≠ ∅: #如果邊集合不等於空集合

#讓(u, v)表示為集合E' 中任意一條邊

let (u, v) be an arbitrary edge of E'

#將{u, v} 兩點中，其中一點加入vertex cover C中

C = C ∪ {u, v}

#移除集合E' 中所有和u 或 v 相連的所有邊

remove from E' every edge incident on either u or v

**return** C