



グラフの探索

深さ優先探索と幅優先探索

2017年度

担当: 只木進一(工学系研究科)

グラフ探索

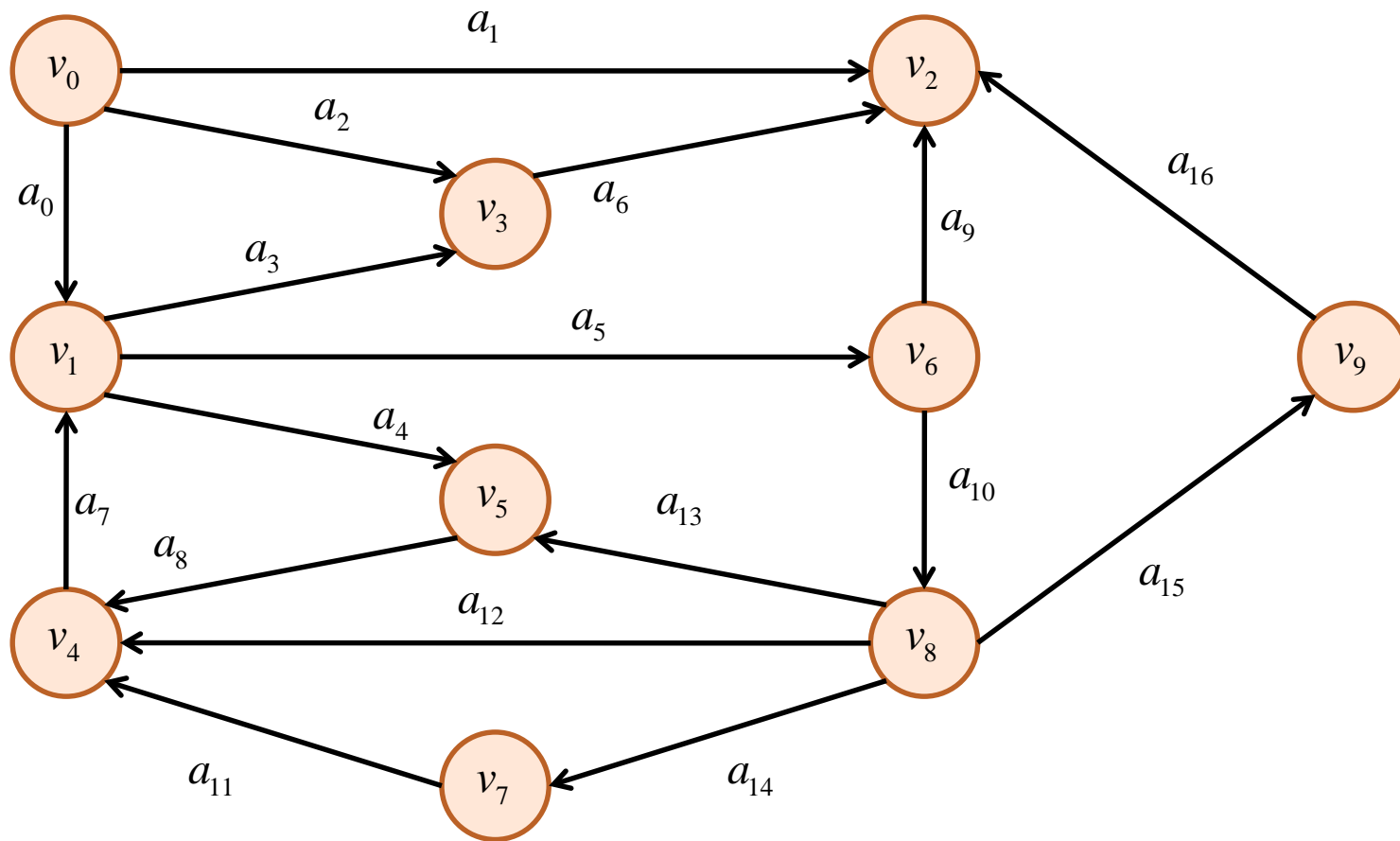
- 単純な探索
 - ある頂点に到達できるか
 - 到達できる頂点を列挙する
 - 二つのアルゴリズム
 - 深さ優先
 - 幅優先
- 単純でない探索:後述
 - 最小な木
 - 最短経路

深さ優先探索DFS (DEPTH-FIRST SEARCH)

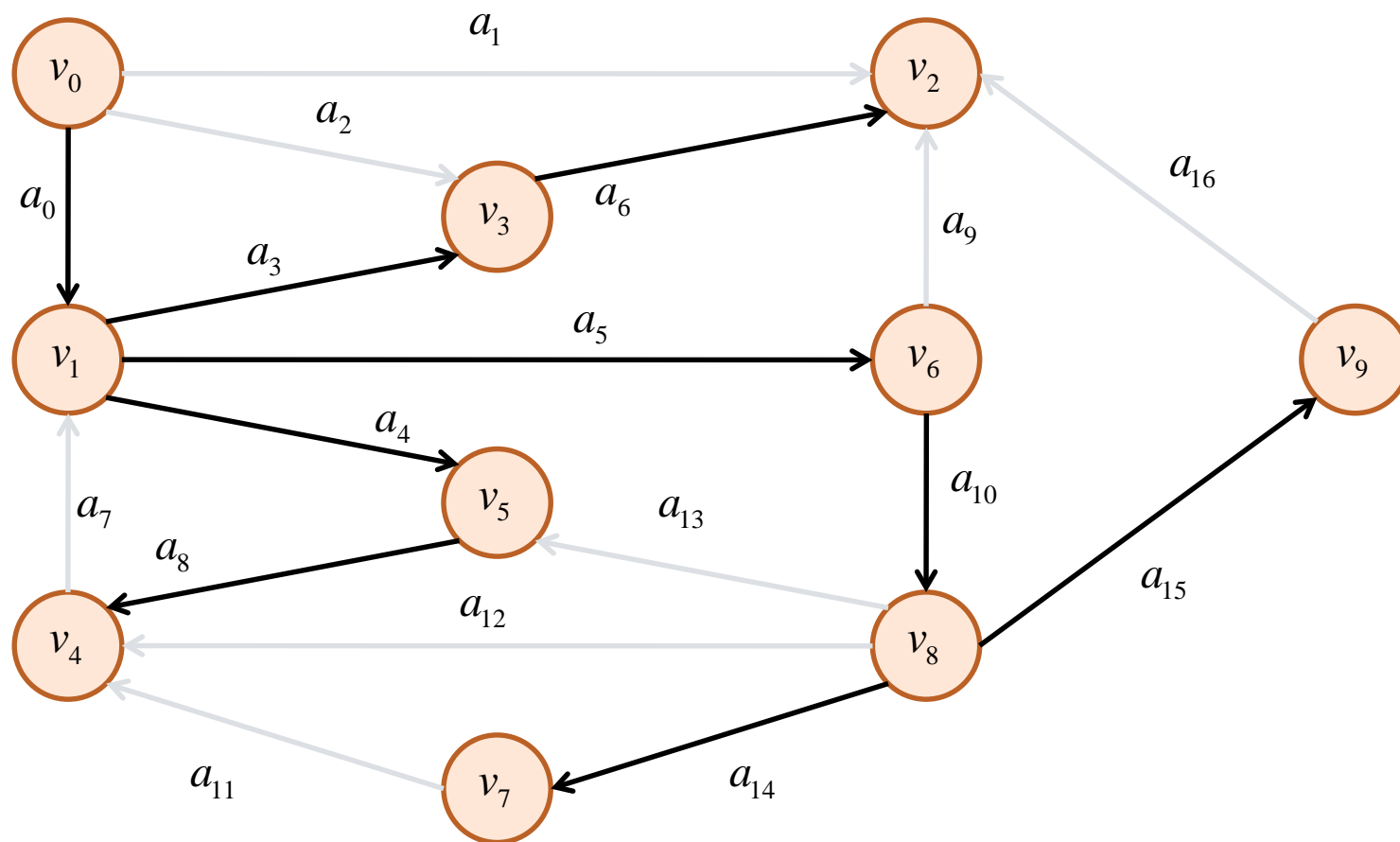
- 出発点を定める
- たどれる限り、弧をたどる
 - それ以上進めなくなるまで
 - 新たな点が無くなるまで
- 戻って、別の弧をたどる

- 結果としてできる木(spanning tree)は、深いものができる

深さ優先探索DFS (DEPTH-FIRST SEARCH)



結果



再帰的関数で表現

- L : 既にチェックした点のリスト
- v : 現在の頂点
- δ^+v : v を始点とする弧の集合

再帰的な探索



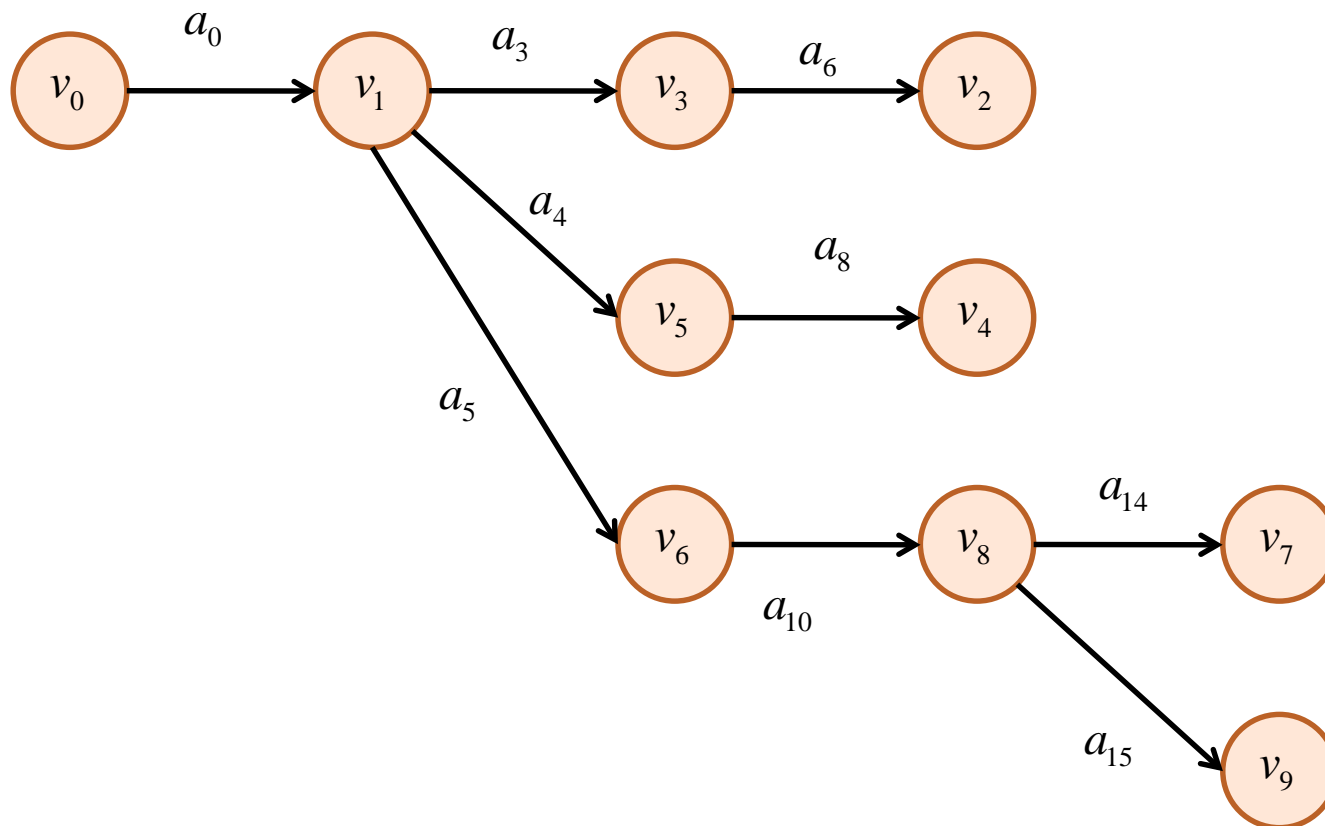
グラフを深い方向に探索

```
search(v, L) {  
    //vから出る全ての弧  
    forall( a ∈  $\delta^+v$  ) {  
        w =  $\partial^-a$  //反対側  
        if( w ∉ L ) {  
            L ← L ∪ {w}  
            search(w, L)  
        }  
    }  
}
```

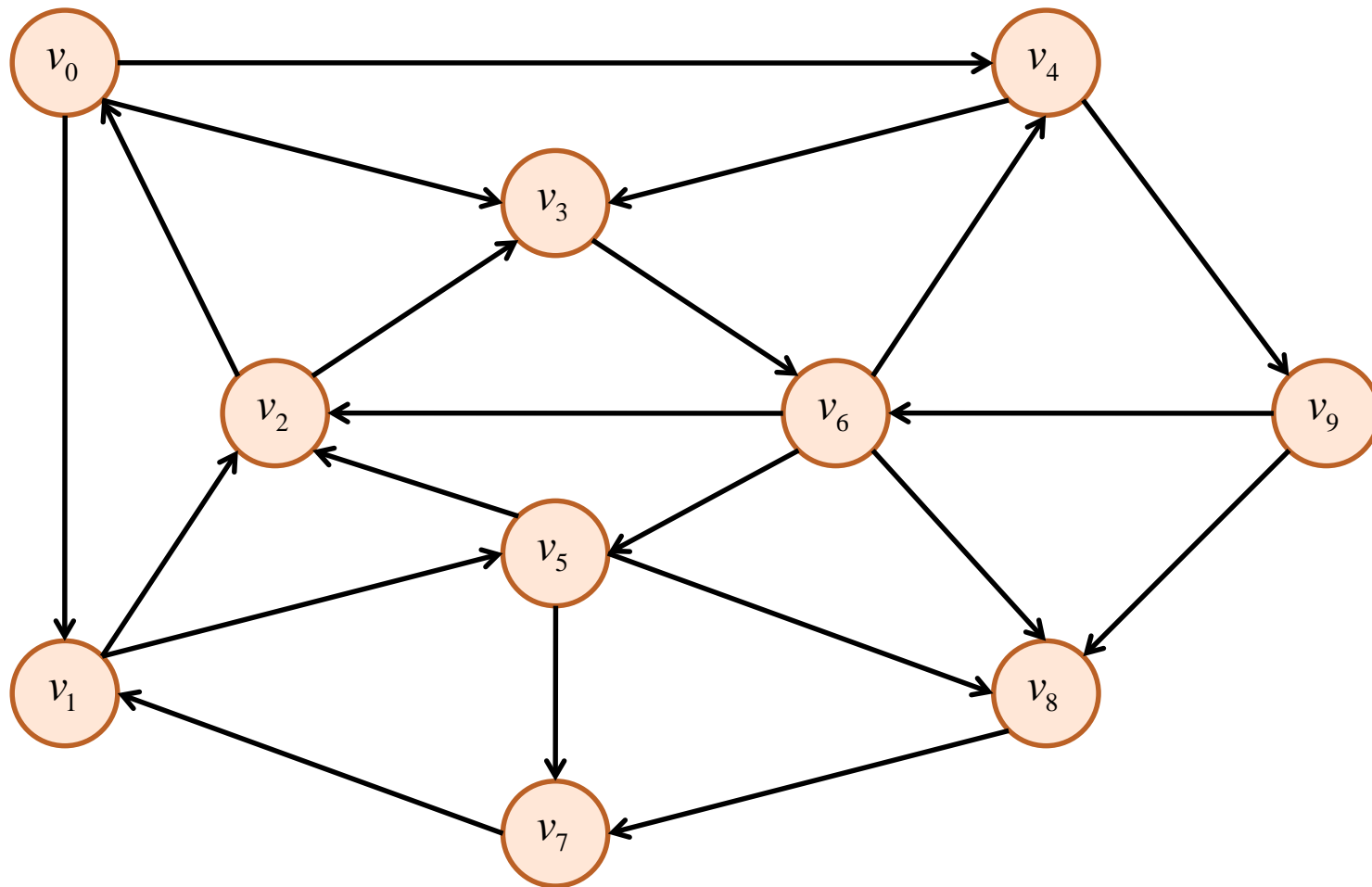
再帰呼び出しの様子

$$\begin{aligned} & (v_0, \{v_0\}) \rightarrow (v_1, \{v_0, v_1\}) \rightarrow (v_3, \{v_0, v_1, v_3\}) \rightarrow (v_2, \{v_0, v_1, v_2, v_3\}) \\ & \rightarrow (v_5, \{v_0, v_1, v_2, v_3, v_5\}) \rightarrow (v_4, \{v_0, v_1, v_2, v_3, v_4, v_5\}) \\ & \rightarrow (v_6, \{v_0, v_1, v_2, v_3, v_4, v_5, v_6\}) \rightarrow (v_8, \{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_8\}) \\ & \rightarrow (v_7, \{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}) \\ & \rightarrow (v_9, \{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}) \end{aligned}$$

探索の様子(結果としてのSPANNING TREE)



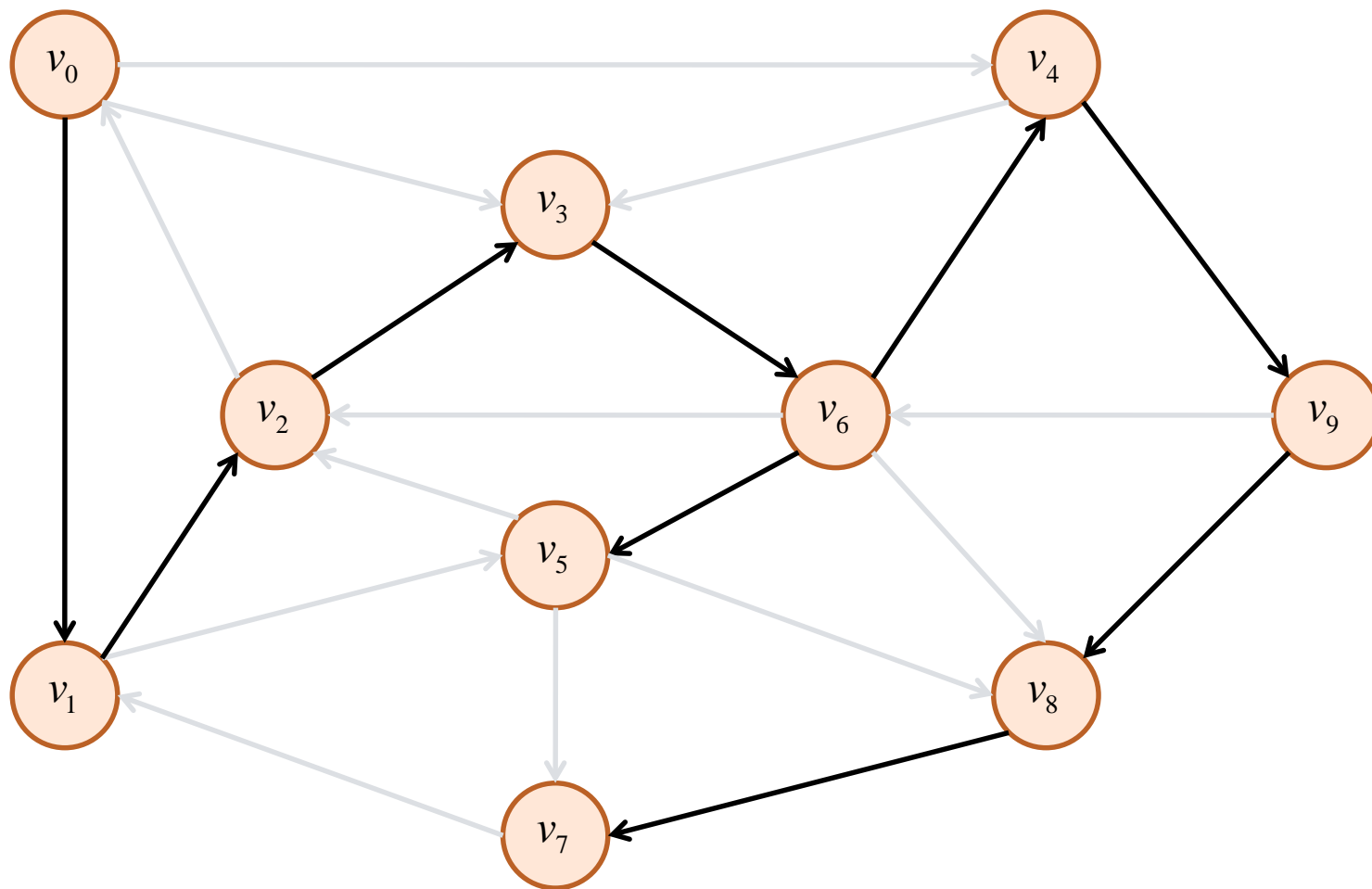
例2



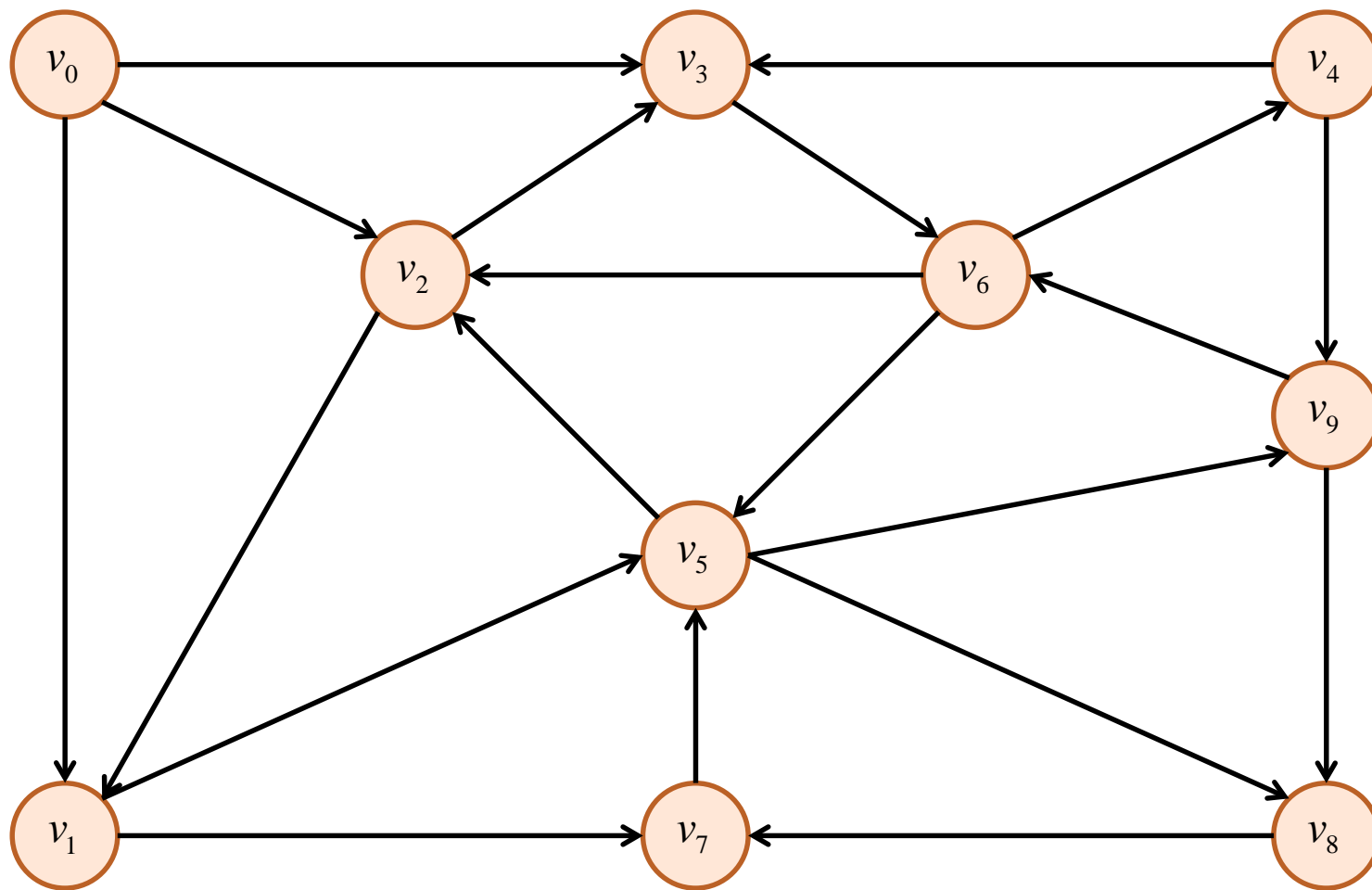
$$\begin{aligned}
 & (v_0, \{v_0\}) \rightarrow (v_1, \{v_0, v_1\}) \rightarrow (v_2, \{v_0, v_1, v_2\}) \rightarrow (v_3, \{v_0, v_1, v_2, v_3\}) \\
 & \rightarrow (v_6, \{v_0, v_1, v_2, v_3, v_6\}) \rightarrow (v_4, \{v_0, v_1, v_2, v_3, v_4, v_6\}) \rightarrow (v_9, \{v_0, v_1, v_2, v_3, v_4, v_6, v_9\})
 \end{aligned}$$

例2

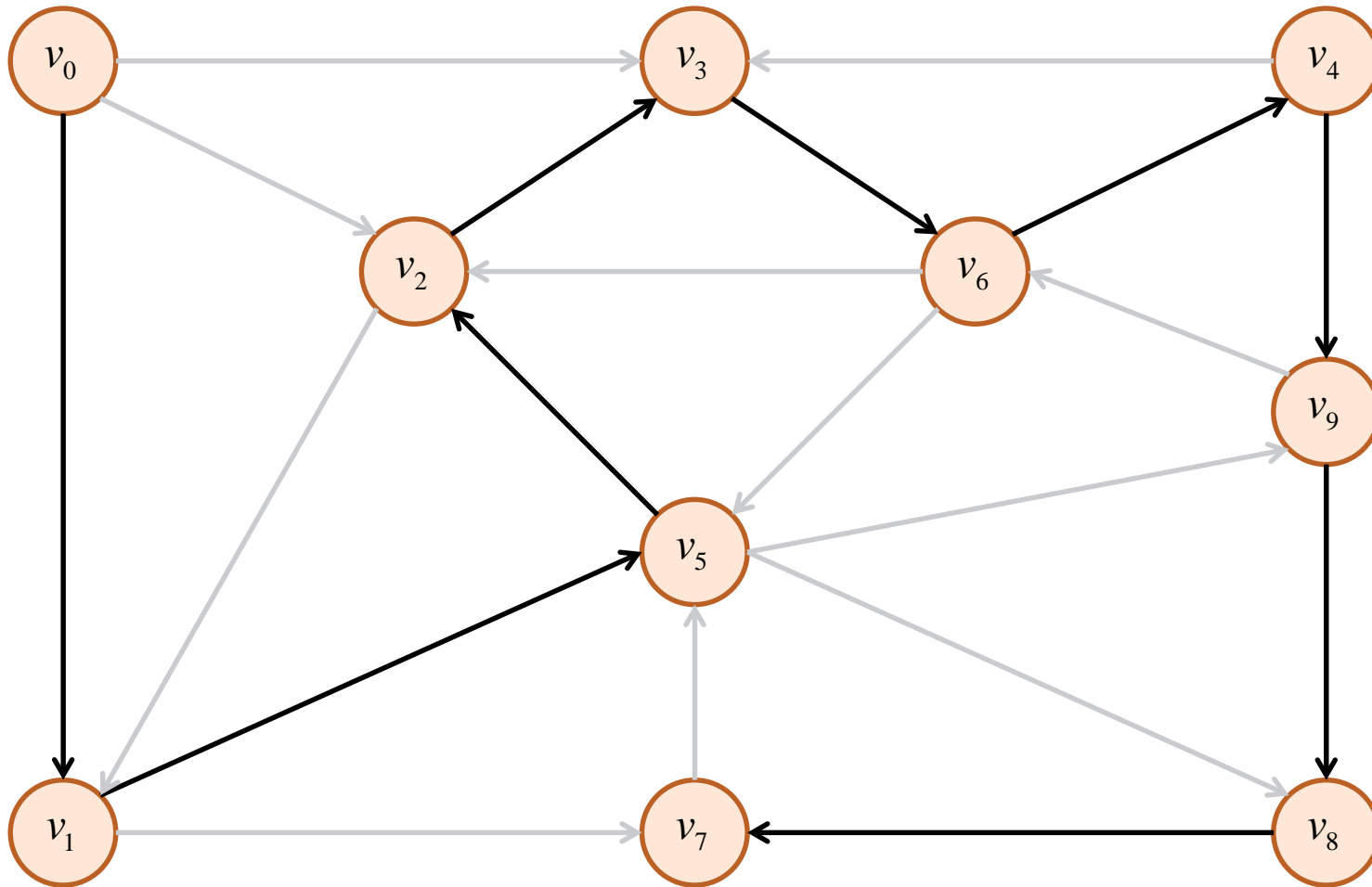
$$\begin{aligned}
 & \rightarrow (v_8, \{v_0, v_1, v_2, v_3, v_4, v_6, v_8, v_9\}) \rightarrow (v_7, \{v_0, v_1, v_2, v_3, v_4, v_6, v_7, v_8, v_9\}) \\
 & \rightarrow (v_5, \{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\})
 \end{aligned}$$



例3



例3:結果



非再帰的関数で表現

- L : 既にチェックした点のリスト
- v_0 : 探索の始点
- Q : 後で探索すべき弧のスタック

```
L = {v_0}
Q ← δ+v0//push
while(Q ≠ ∅) {
  a = Q.pop()
  w = δ-a
  if ( w ∉ L ) {
    L ← L ∪ {w}
    U:wとLの要素を結ぶ弧
    Q ← δ+w \ U
  }
}
```

例1の場合

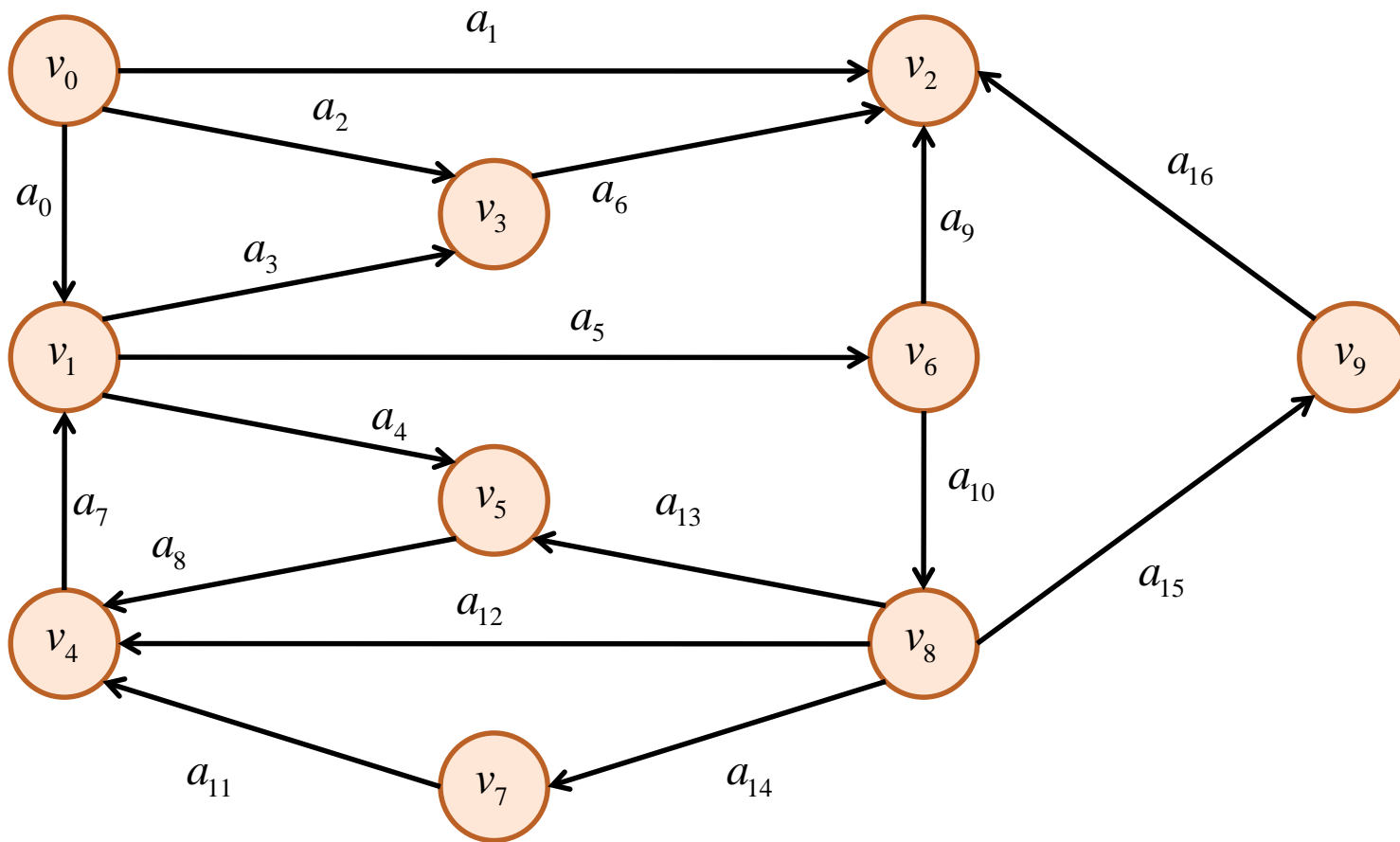
L	Q
$\{v_0\}$	$[a_0, a_1, a_2]$
$\{v_0, v_1\}$	$[a_3, a_4, a_5, a_1, a_2]$
$\{v_0, v_1, v_3\}$	$[a_6, a_4, a_5, a_1, a_2]$
$\{v_0, v_1, v_2, v_3\}$	$[a_4, a_5, a_1, a_2]$
$\{v_0, v_1, v_2, v_3, v_5\}$	$[a_8, a_5, a_1, a_2]$
$\{v_0, v_1, v_2, v_3, v_4, v_5\}$	$[a_5, a_1, a_2]$
$\{v_0, v_1, v_2, v_3, v_4, v_5, v_6\}$	$[a_{10}, a_1, a_2]$
$\{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_8\}$	$[a_{14}, a_{15}, a_1, a_2]$
$\{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}$	$[a_{15}, a_1, a_2]$
$\{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$	$[a_1, a_2]$
$\{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$	$[a_2]$
$\{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$	$[\]$

幅優先探索(BREADTH-FIRST SEARCH)

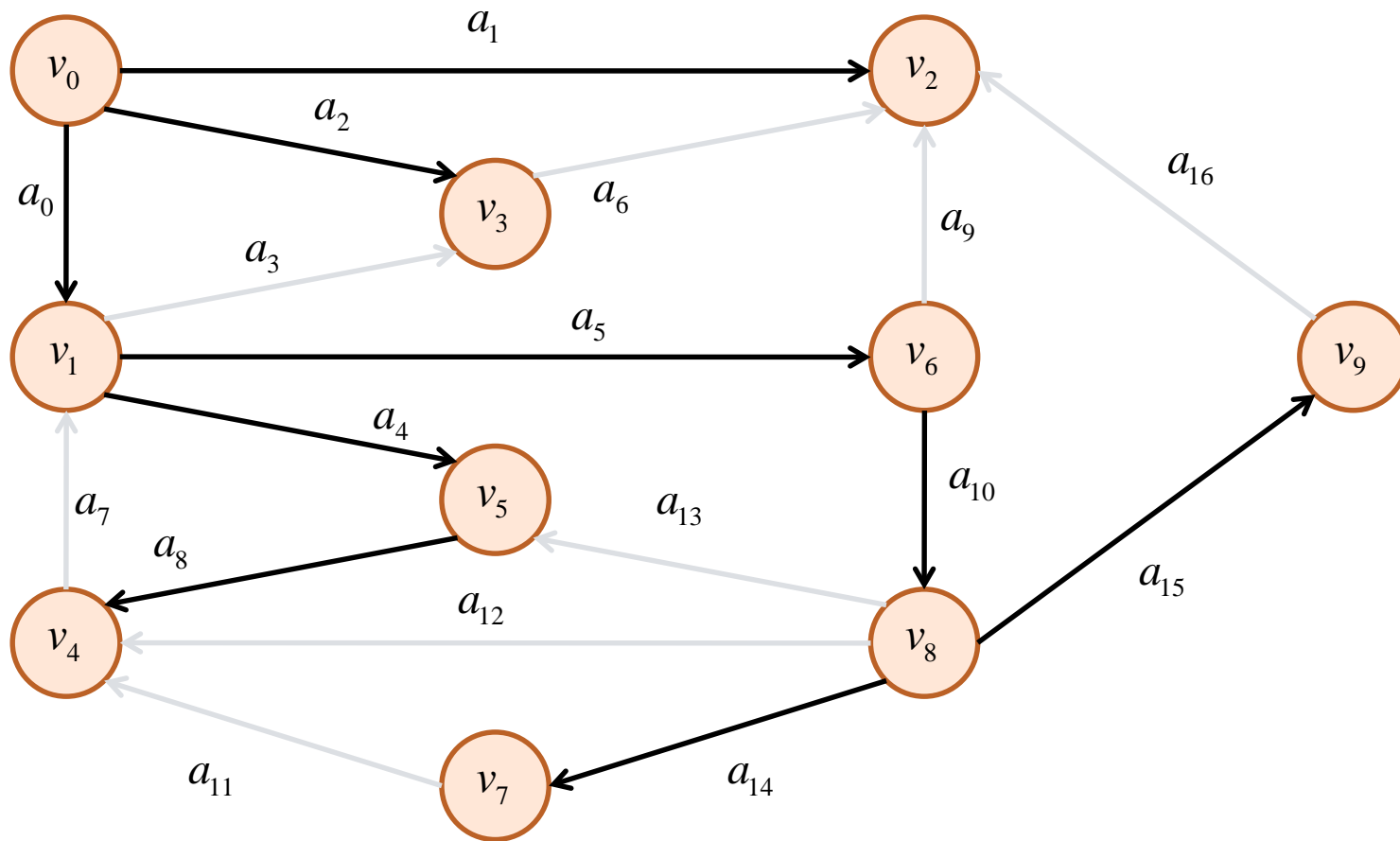
- 出発点を定める
- 出発点に直接繋がっている点に印を付ける
- 印を付けた点に直接繋がっている点に印を付ける

- 結果としてできる木(spanning tree)は、幅の広いものができる

幅優先探索BFS (BREADTH-FIRST SEARCH)



結果



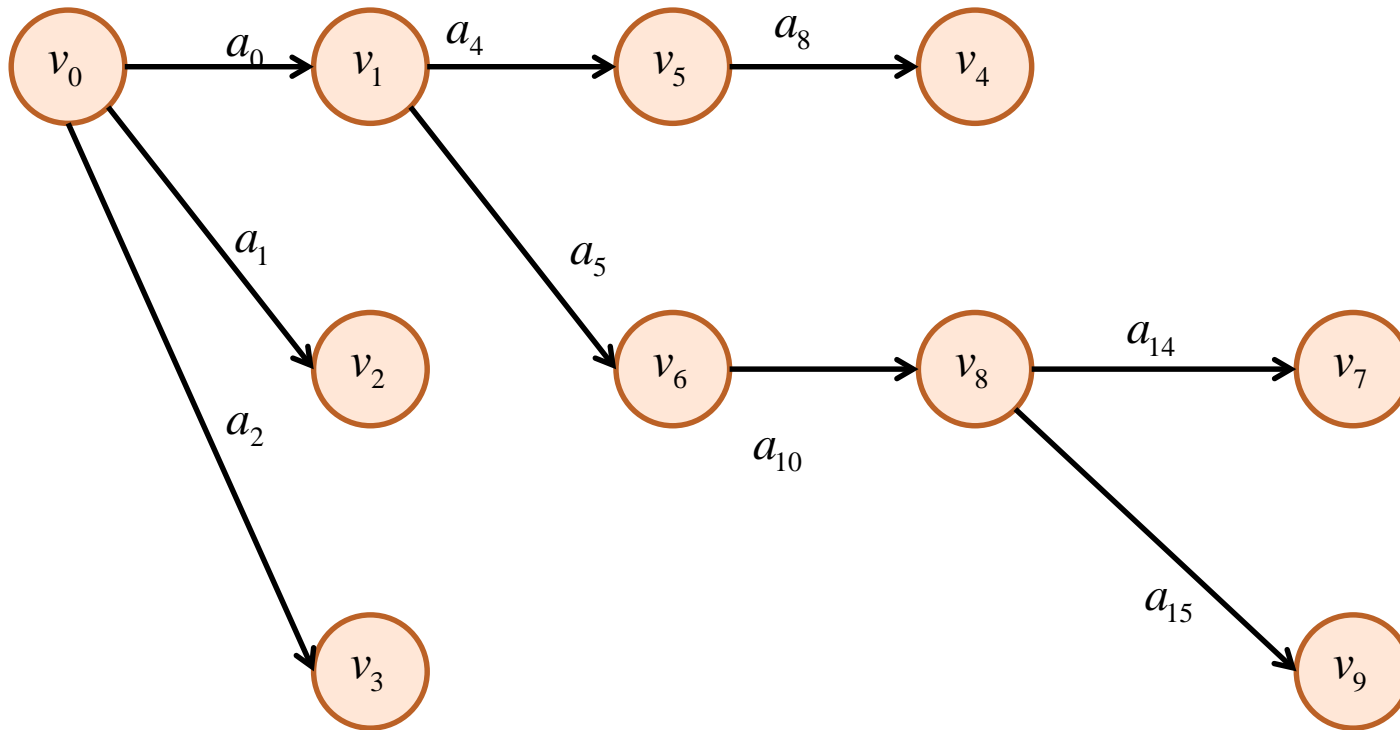
幅優先探索

- L : すでにチェックした点のリスト: 初期 $L = \phi$
- Q : 調査すべき点のキュー: 初期 $Q = [r]$

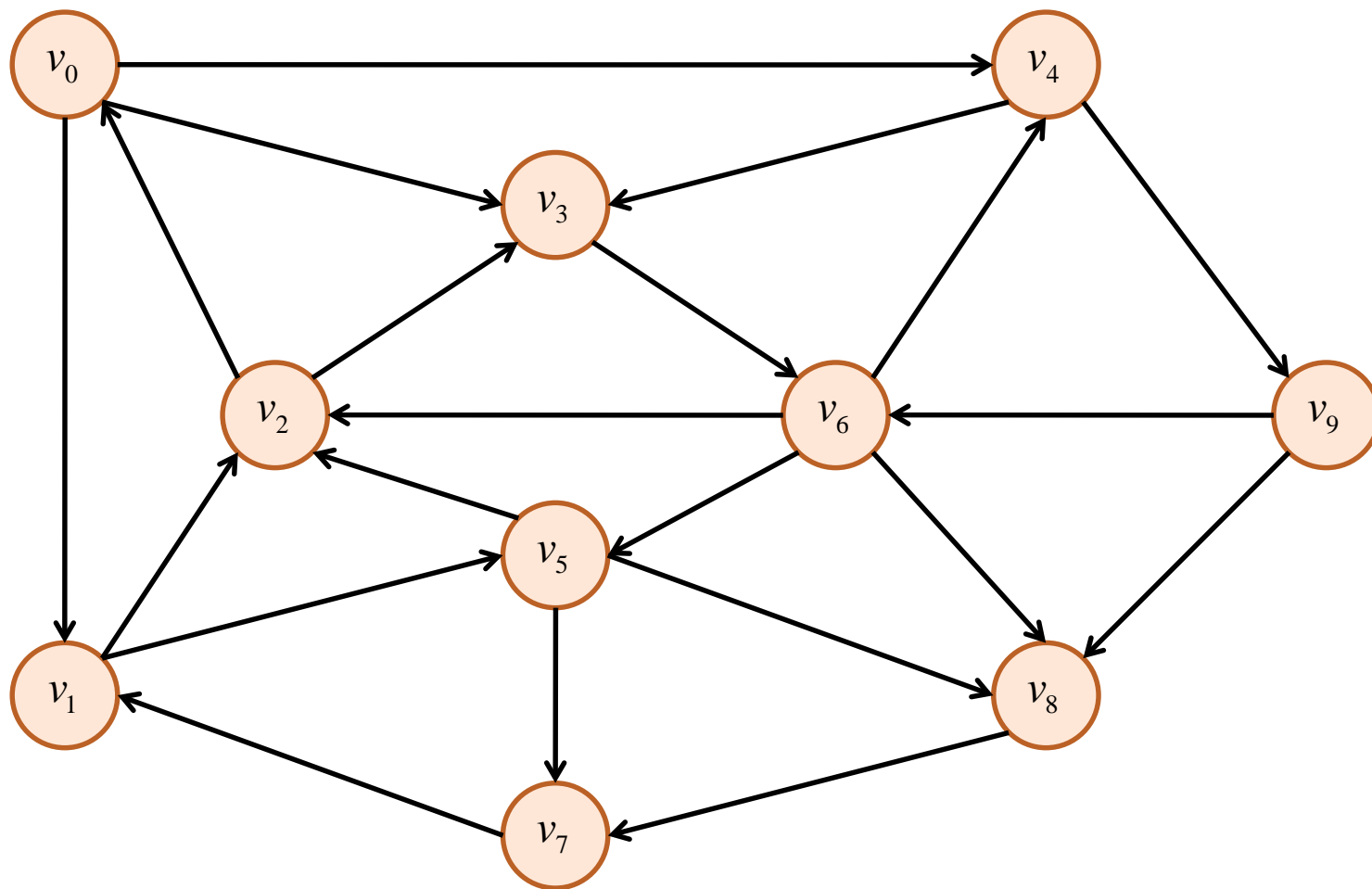
```
L =  $\emptyset$ 
Q = [r]
while(Q  $\neq \emptyset$ ) {
    v = Q.poke//先頭
    forall( a  $\in \delta^+v$  ) {
        w =  $\partial^-a$ 
        if( w  $\notin L$  && w  $\notin Q$  ) {
            Q  $\leftarrow$  w
        }
    }
    L  $\leftarrow$  L  $\cup$  {v}
}
```

現在の頂点	L	Q
	\emptyset	$[v_0]$
v_0	$\{v_0\}$	$[v_1, v_2, v_3]$
v_1	$\{v_0, v_1\}$	$[v_2, v_3, v_5, v_6]$
v_2	$\{v_0, v_1, v_2\}$	$[v_3, v_5, v_6]$
v_3	$\{v_0, v_1, v_2, v_3\}$	$[v_5, v_6]$
v_5	$\{v_0, v_1, v_2, v_3, v_5\}$	$[v_6, v_4]$
v_6	$\{v_0, v_1, v_2, v_3, v_5, v_6\}$	$[v_4, v_8]$
v_4	$\{v_0, v_1, v_2, v_3, v_4, v_5, v_6\}$	$[v_8]$
v_8	$\{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_8\}$	$[v_7, v_9]$
v_7	$\{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}$	$[v_9]$
v_9	$\{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$	\emptyset

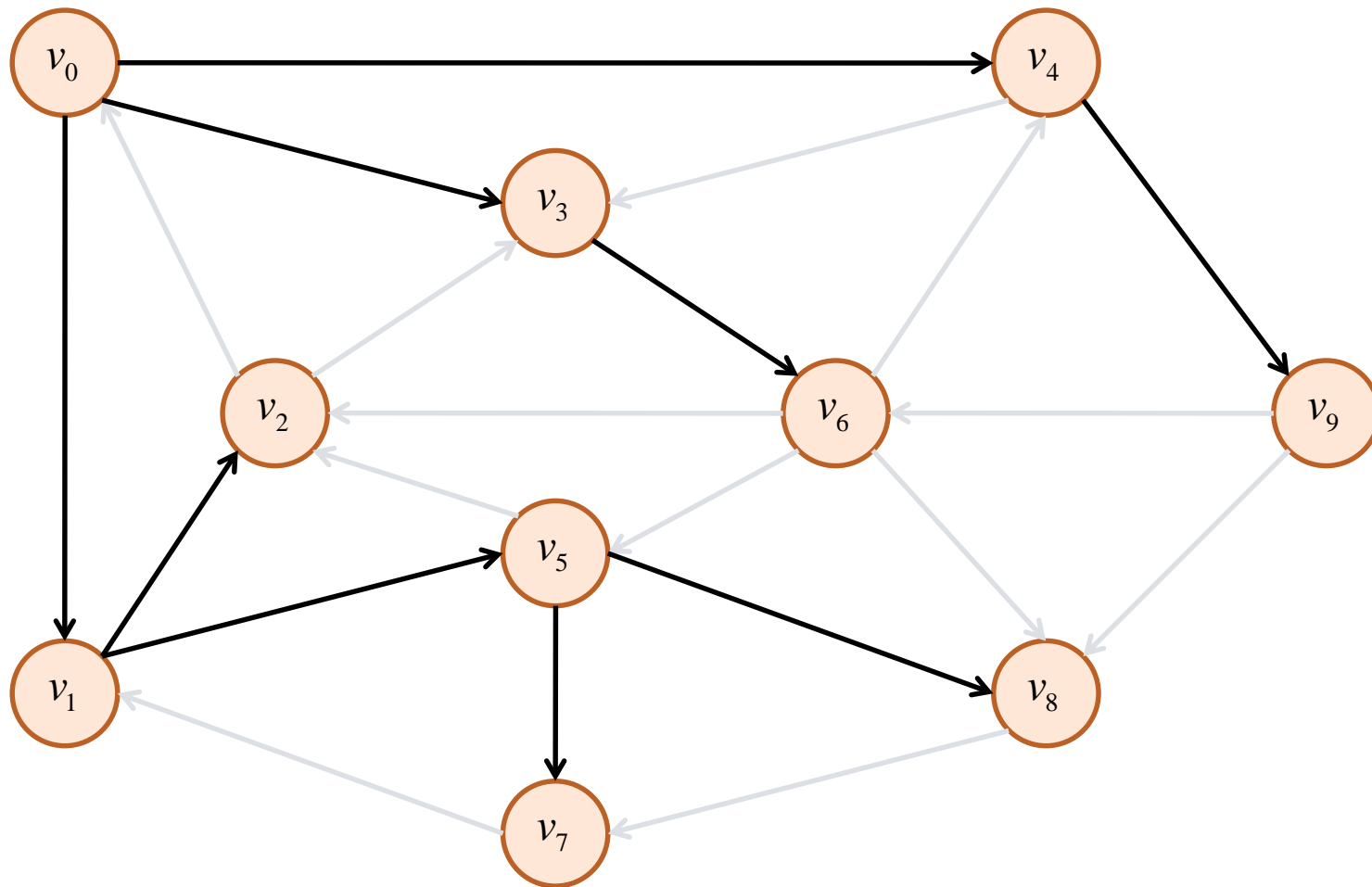
探索の様子(結果としてのSPANNING TREE)



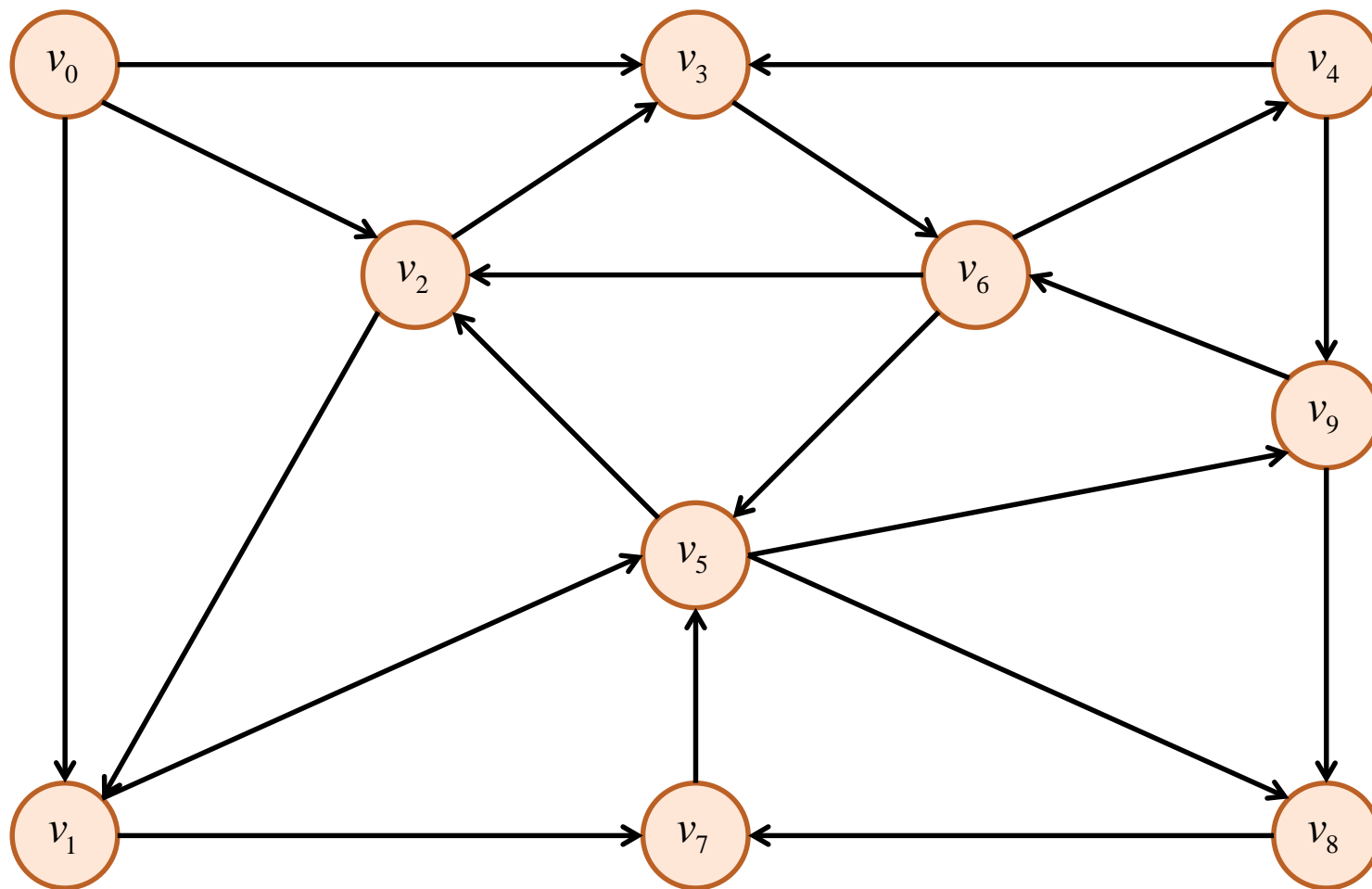
例2



例2



例3



例3

