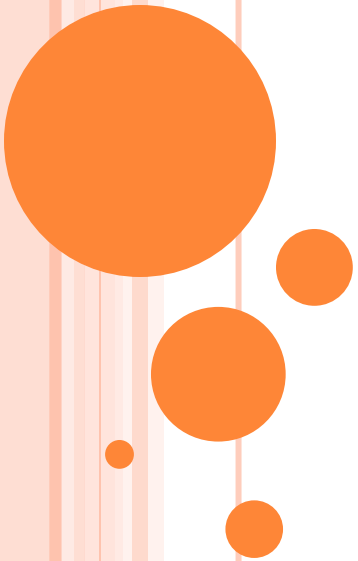


グラフの探索

深さ優先探索と幅優先探索

グラフ探索

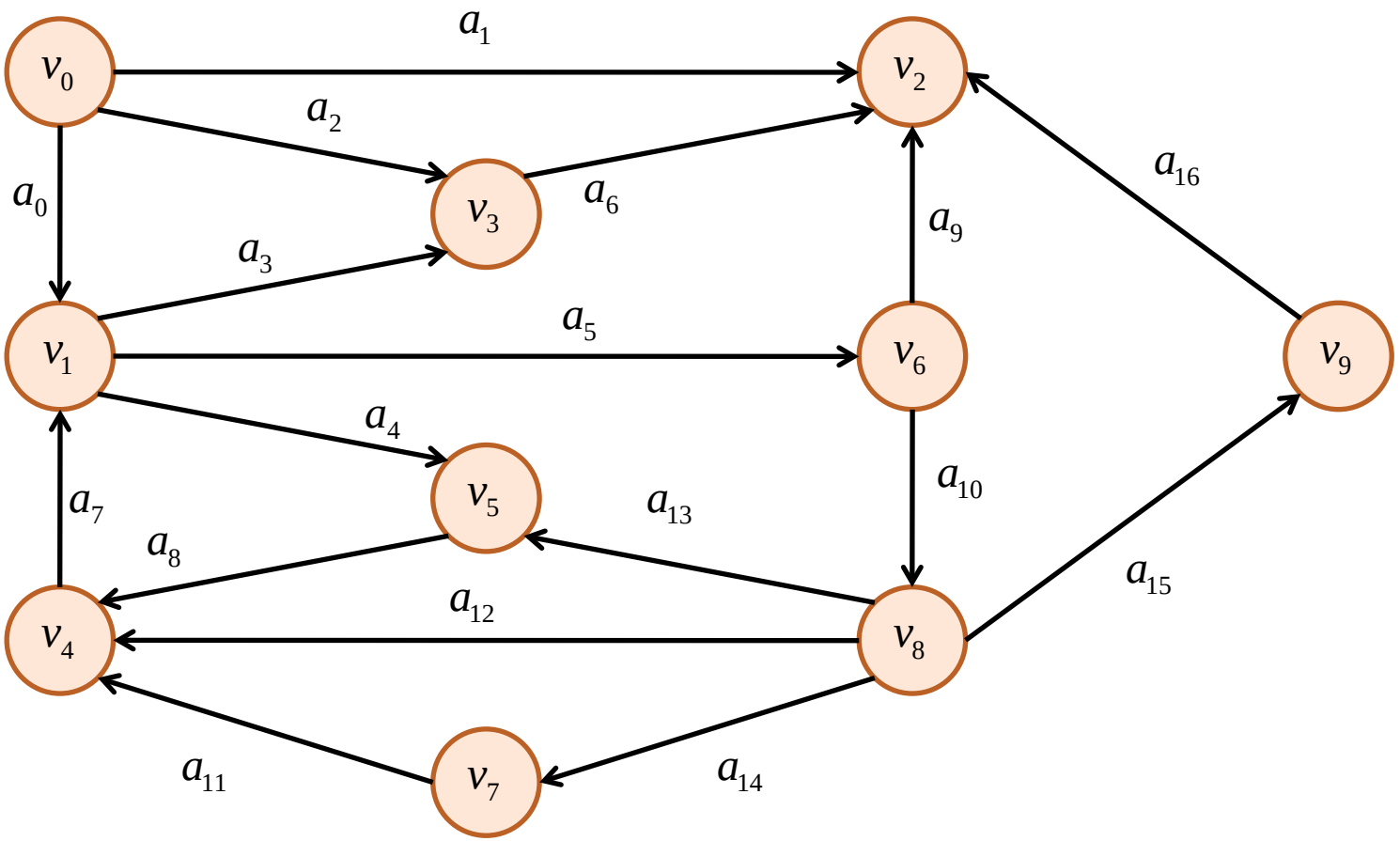
- 単純な探索
  - ある頂点に到達できるか
  - 到達できる頂点を列挙する
  - 二つのアルゴリズム
    - 深さ優先
    - 幅優先
- 単純でない探索：後述
  - 最小な木
  - 最短経路

深さ優先探索DFS (Depth-First Search)

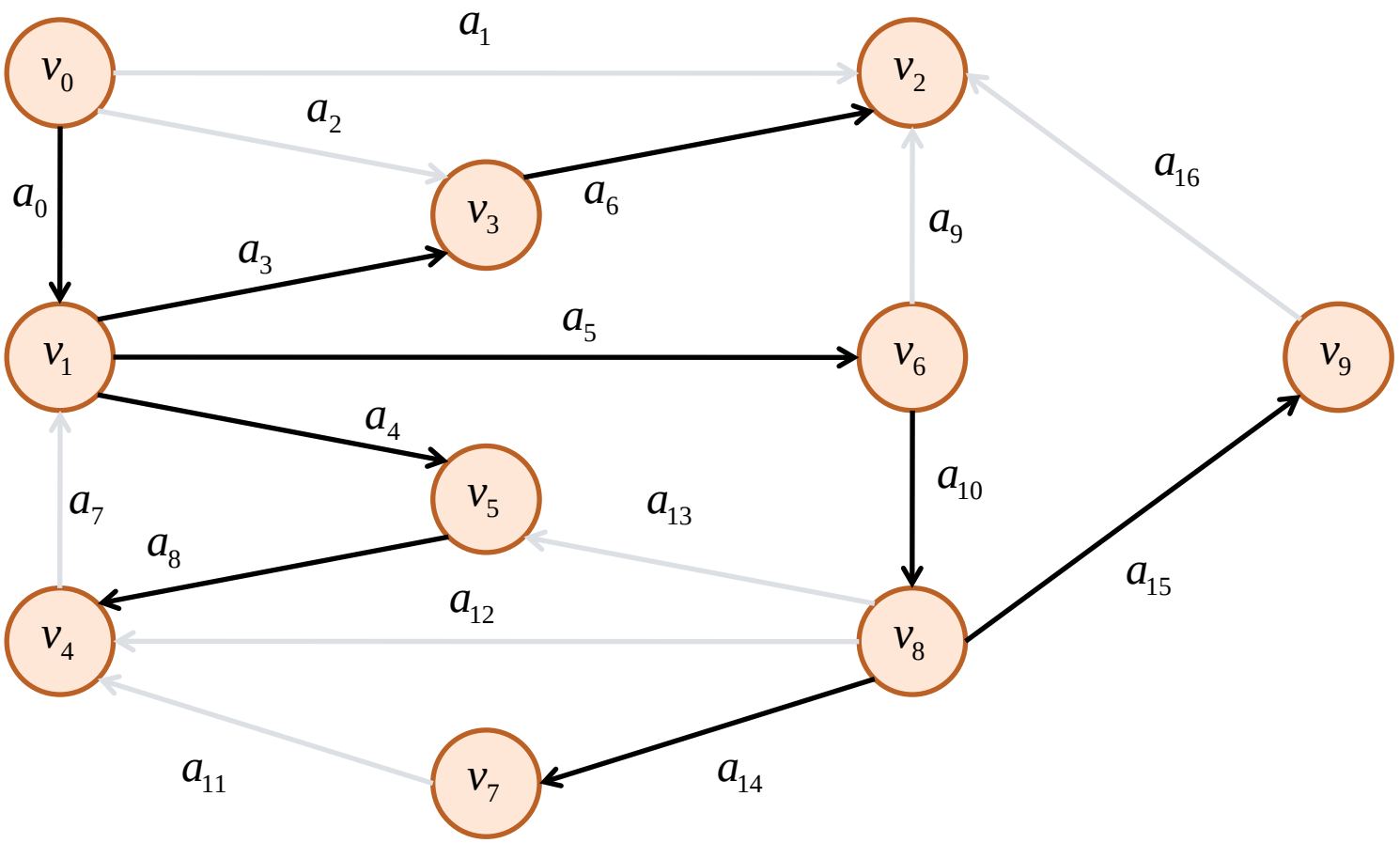
- 出発点を定める
- たどれる限り、弧をたどる
  - それ以上進めなくなるまで
  - 新たな点が無くなるまで
- 戻って、別の弧をたどる

- 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


v_0
v_1
v_2
v_3
v_4
v_5
v_6
v_7
v_8
v_9
a_1
a_2
a_0
a_3
a_6
a_9
a_{16}
a_5
a_4
a_7
a_8
a_{13}
a_{10}
a_{15}
a_{12}
a_{11}
a_{14}

再帰的関数で表現

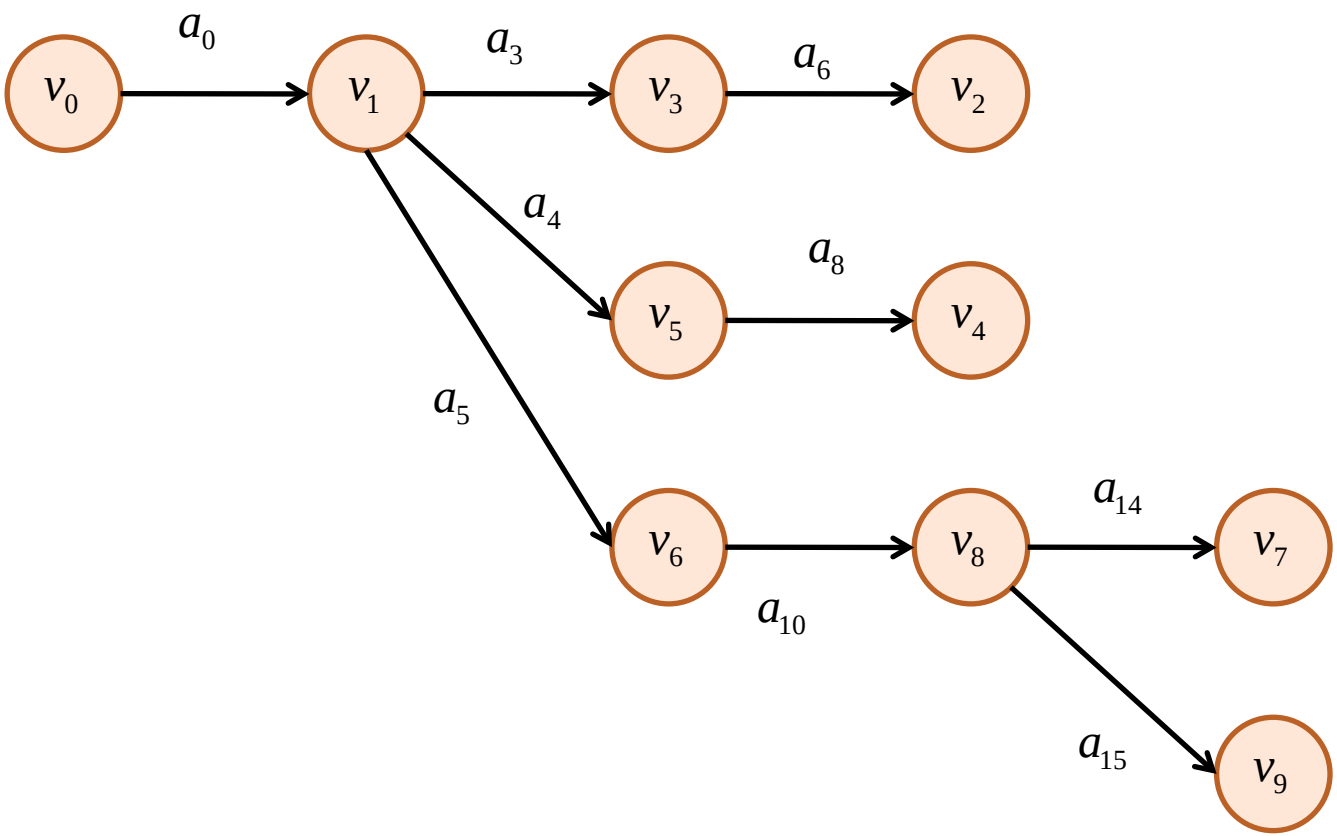
- L：既にチェックした点のリスト
- v：現在の頂点

再帰的な探索

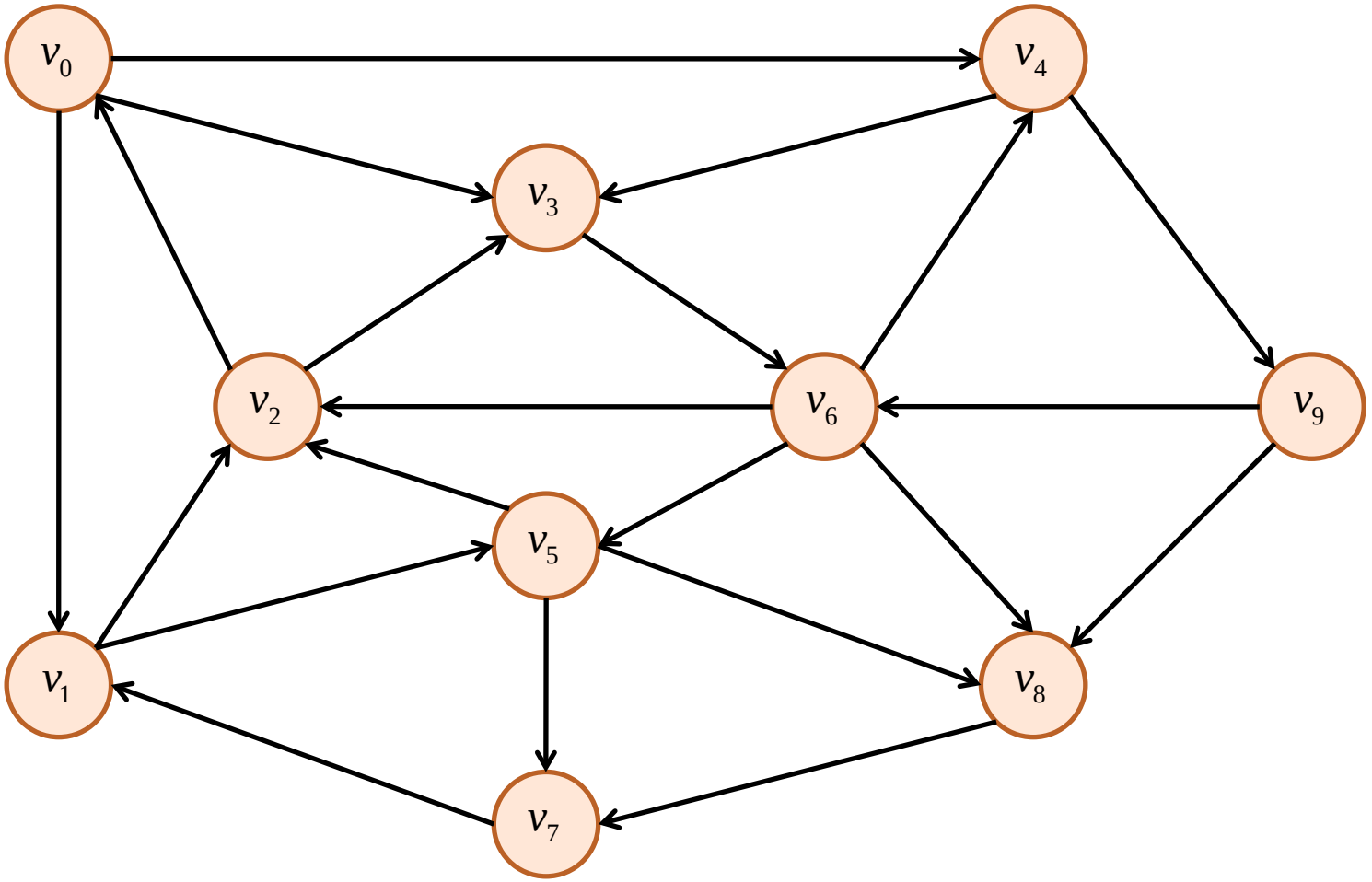
グラフを深い方向に探索

```
search(v, L){
    //v から出る全ての弧
    forall(a ∈ δ⁺v){
        w = ∂⁻a //反対側の頂点
        if(w ∉ L){
            L = L ∪ {w}
            search(w, L)
        }
    }
    //これ以上進めない
    return
}
```

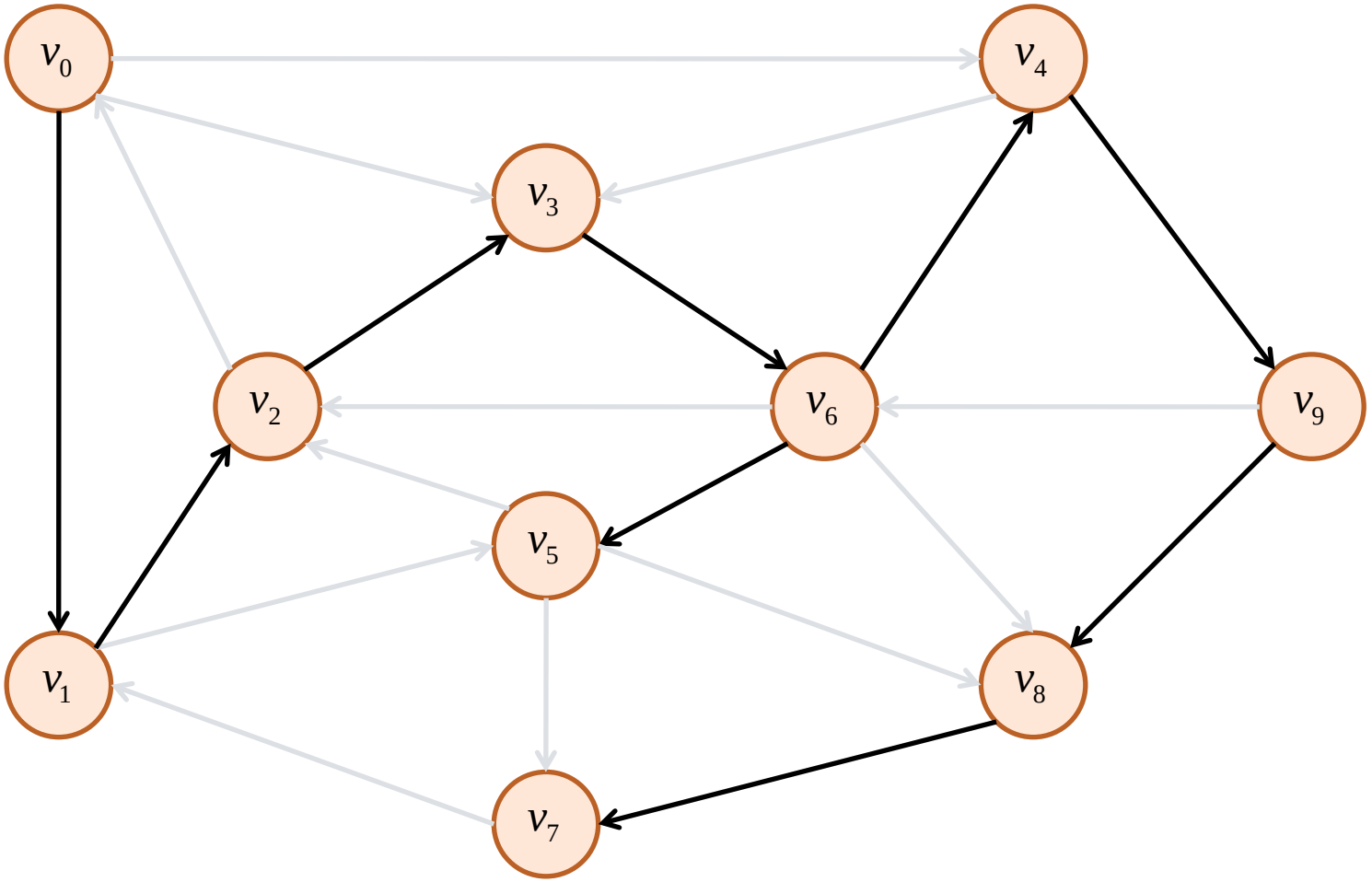
探索の様子(結果としてのSPANNING TREE)



例2



例2



再帰的関数で表現(終点指定)

- L : 既にチェックした点のリスト
- v : 現在の頂点
- d : 終点
- f : 終点に到達したら真

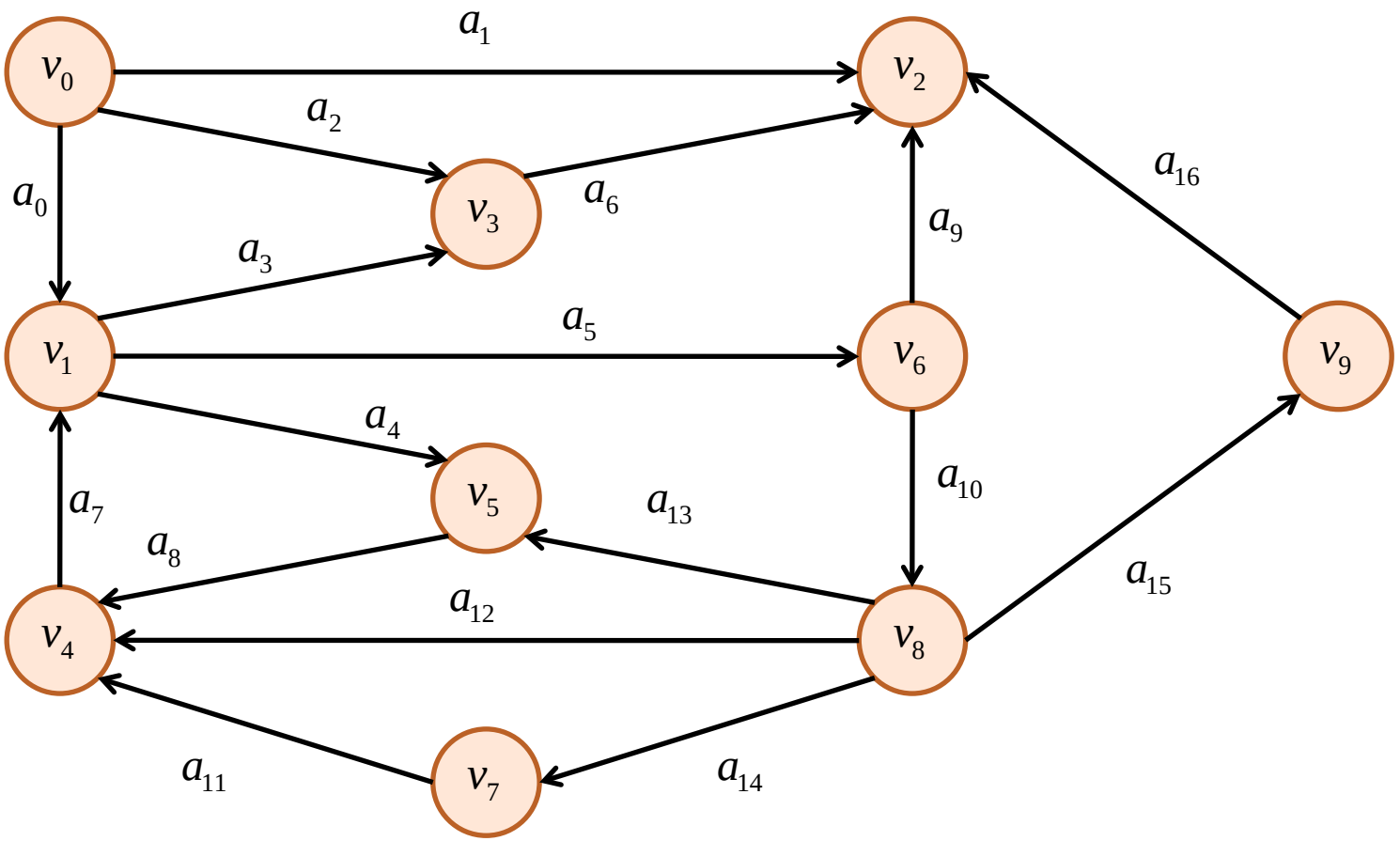
```
search(v, L, d){
    //vから出る全ての弧
    forall(a ∈ δ⁺v){
        w = ∂⁻a //反対側の頂点
        if(w ∉ L ∧ ! f ){
            L = L ∪ {w}
            if(w == d ){
                f = true
                return
            }
            search(w, L, d)
        }
    }
    //これ以上進めない
    return
}
```

幅優先探索(Breadth-First Search)

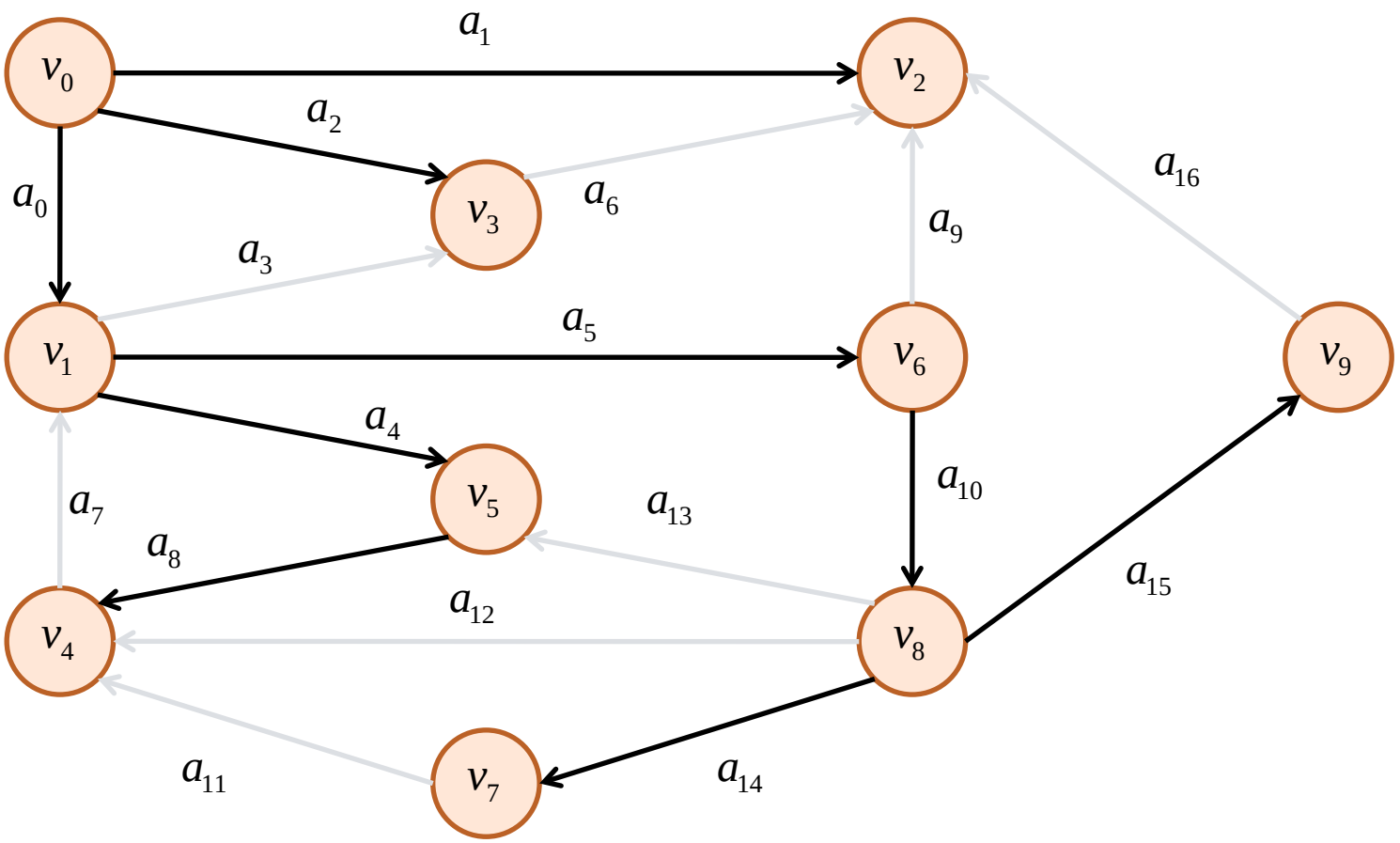
- 出発点を定める
- 出発点に直接繋がっている点に印を付ける
- 印を付けた点に直接繋がっている点に印を付ける

- 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


v_0
v_1
v_2
v_3
v_4
v_5
v_6
v_7
v_8
v_9
a_0
a_1
a_2
a_3
a_4
a_5
a_6
a_7
a_8
a_9
a_{10}
a_{11}
a_{12}
a_{13}
a_{14}
a_{15}
a_{16}

幅優先探索

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

$L = \varnothing$

$Q = \{r\}$

while($Q \neq \varnothing$){

　　v = Qの先頭の取り出し

　　forall($a \in \delta^{+} v$){

　　　　$w = \partial^{-} a$

　　　　if($w \notin L$ && $w \notin Q$){

　　　　　　Qへwを追加

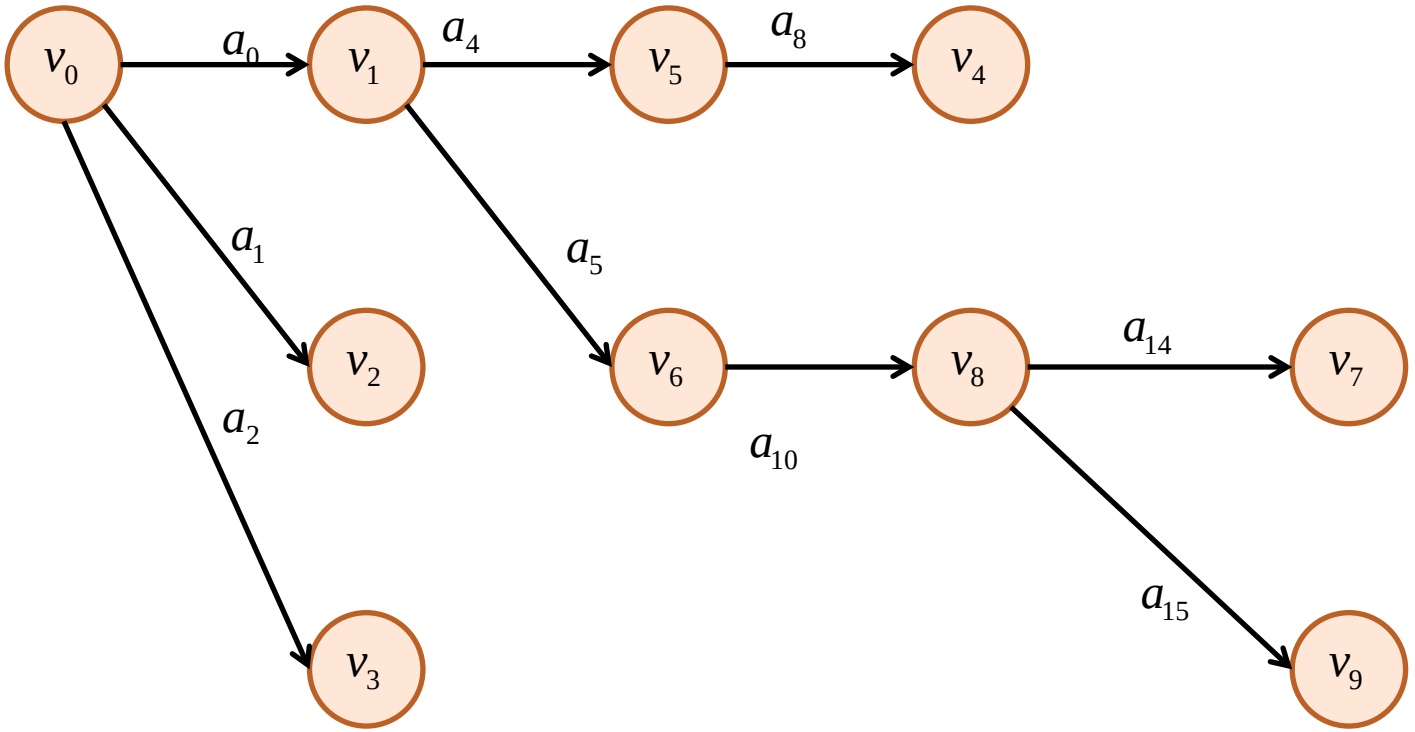
　　　　}

　　}

　　$L \leftarrow L \cup \{v\}$

}

探索の様子(結果としてのSPANNING TREE)

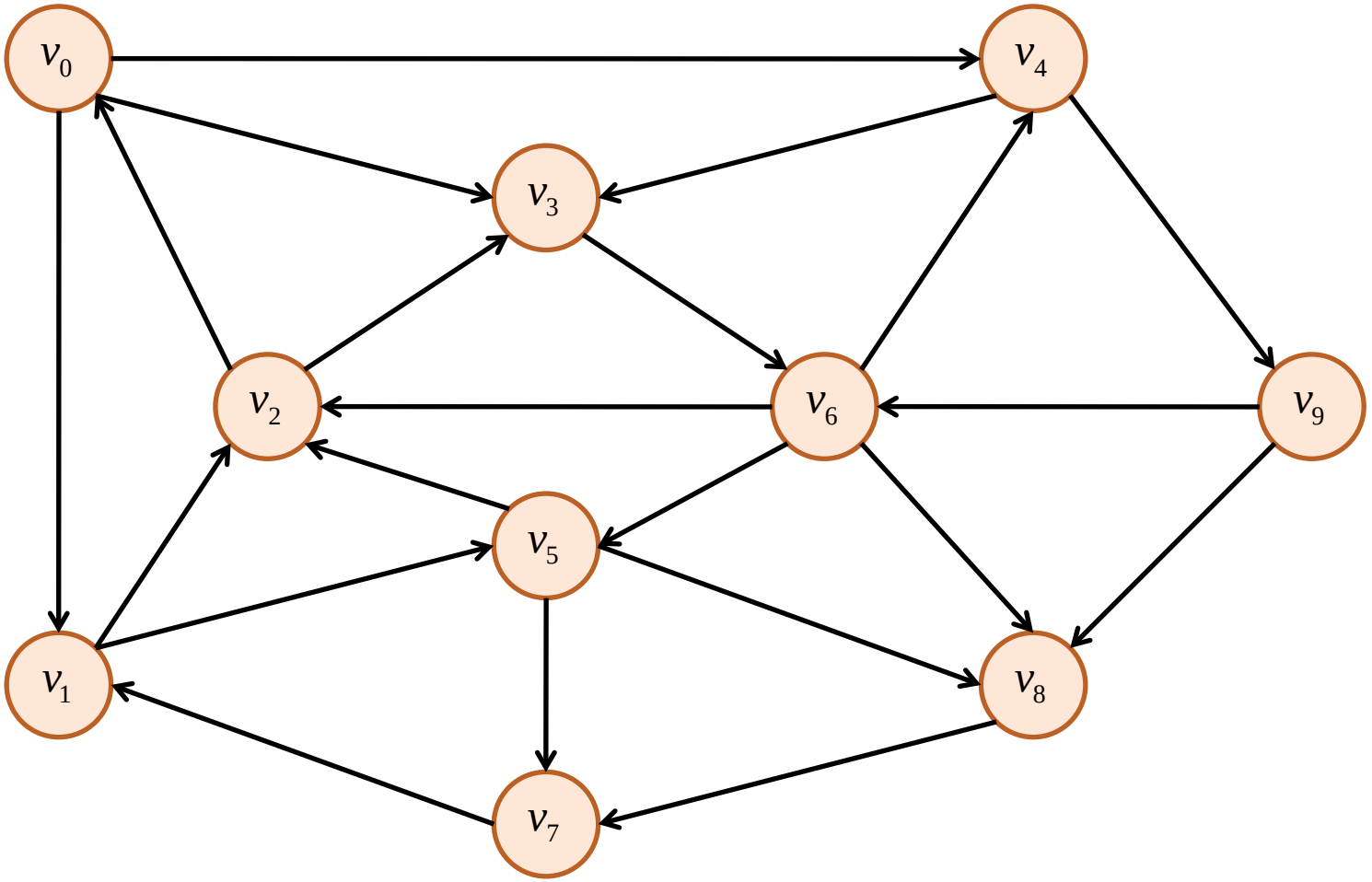


註:キュー(QUEUE、待ち行列)

- 先入れ先出し (FIFO、First In First Out)のリスト構造
- java.util.concurrent.ConcurrentLinkedQueueクラス
- add(E e) 要素eを追加
- poll() 先頭の要素を取り出す

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

例2



例2