```
// classifica i lati di un grafo
dfs-schema(GRAPH G, NODE u, int &time, int[ ] dt, int[ ] ft)
    // time: contatore
    // dt:   tempo di scoperta
    // ft:   tempo di fine
    esamina il nodo u (caso pre-visita)

    time++  // incremento il contatore
    dt[u] ← time  // lo memorizzo nel vettore di scoperta

    // effettuo una visita in profondità
    foreach v ∈ G.adj(u) do
        { esamina l'arco (u, v) (qualsiasi) }  // qui si sviluppa la logica dell'algoritmo
        if dt[v] == 0 then // non ho ancora esaminato il nodo
            { esamina l'arco (u, v) (albero) }
            dfs-schema(g,v)  // effettuo la chiamata ricorsiva
        else if dt[u] < dt[v] and ft[v] == 0 then
            // se raggiungo un mio discendente e non ho ancora terminato la mia visita,
               allora ho trovato un arco all'indietro
            { esamina l'arco (u, v) (indietro) }
        if dt[u] < dt[v] and ft[v] ≠ 0 then
            // se raggiungo un mio discendente e ho terminato la mia visita, allora ho
               trovato un arco in avanti
            { esamina l'arco (u, v) (avanti) }
        else
            // l'ultimo caso rimanente
            { esamina l'arco (u, v) (attraversamento) }

    { visita il nodo u (post-visita) }

    time++ // aggiorno il contatore
    ft[u] ← time  // lo memorizzo nel vettore di fine
```