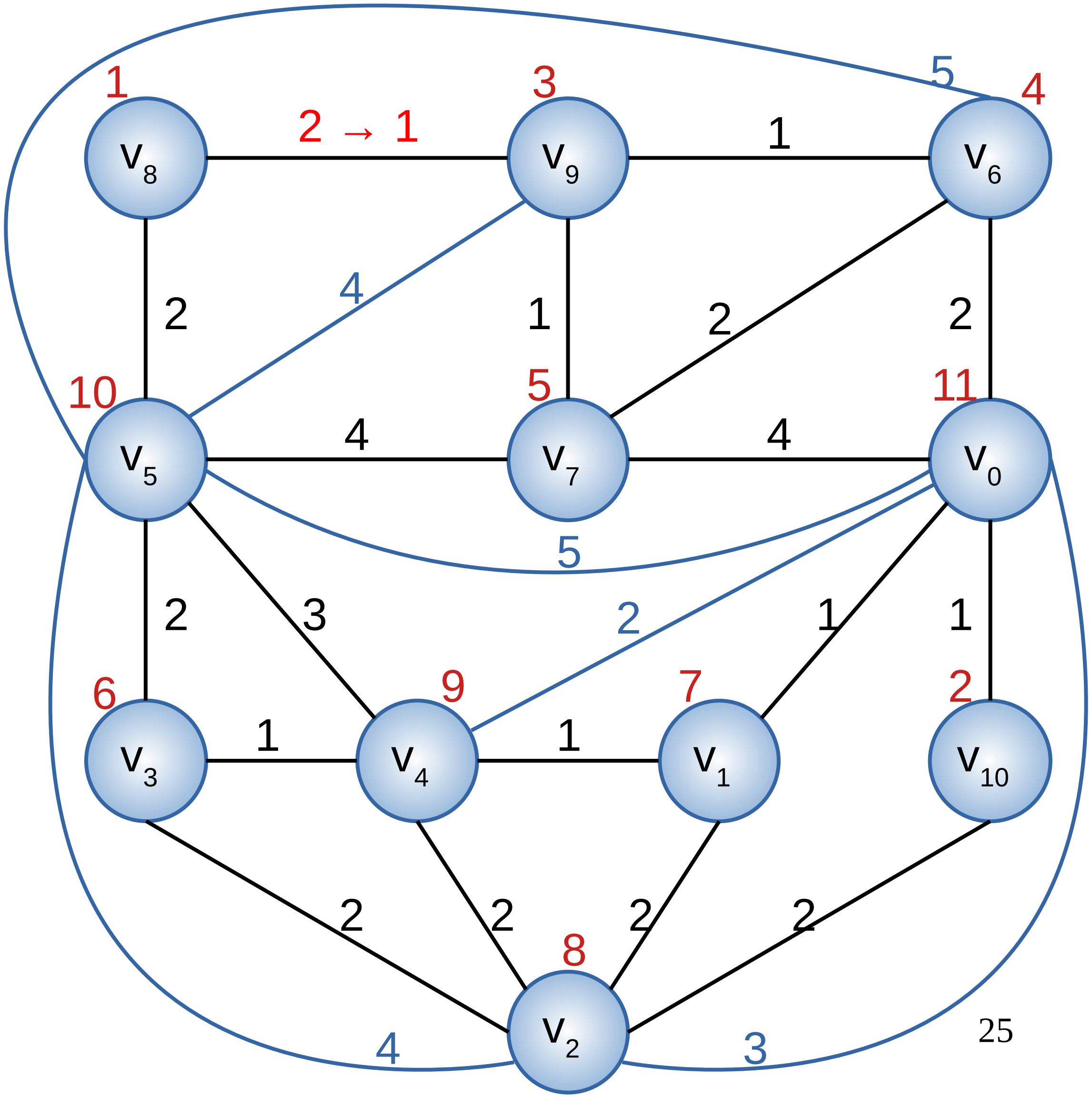
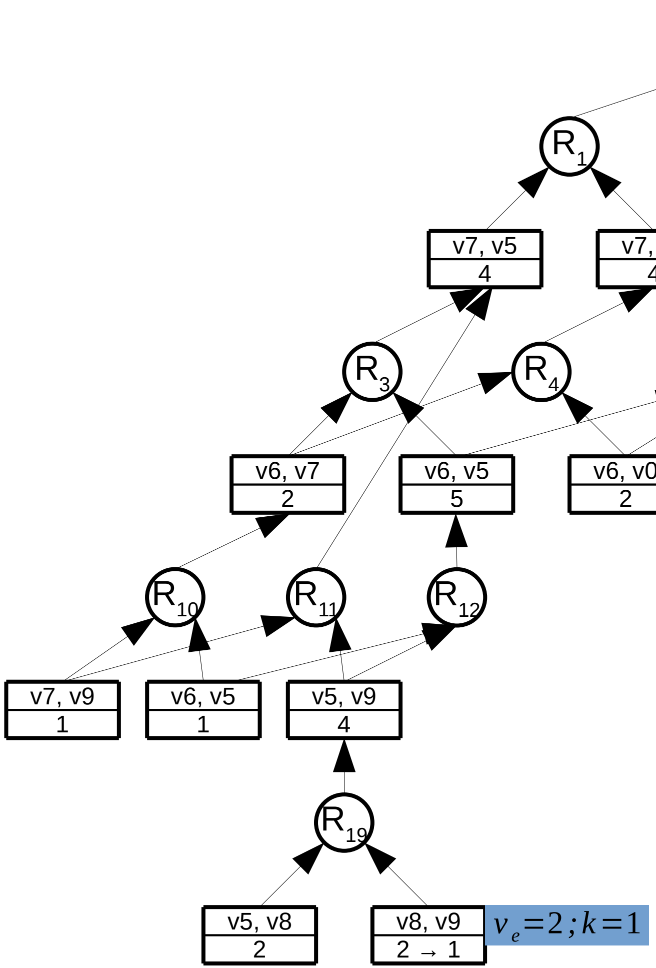
Weight Decrease

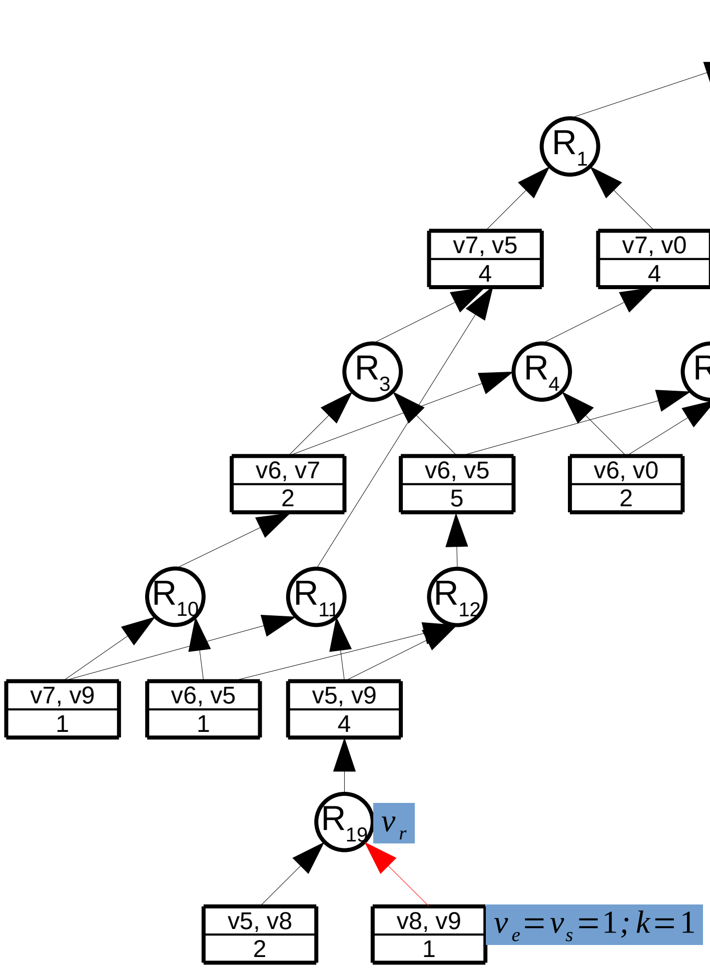
- Weight decrease ($v_8 v_9$) = 2 → 1
- Implies further weight changes
- Let’s use G* to find out which





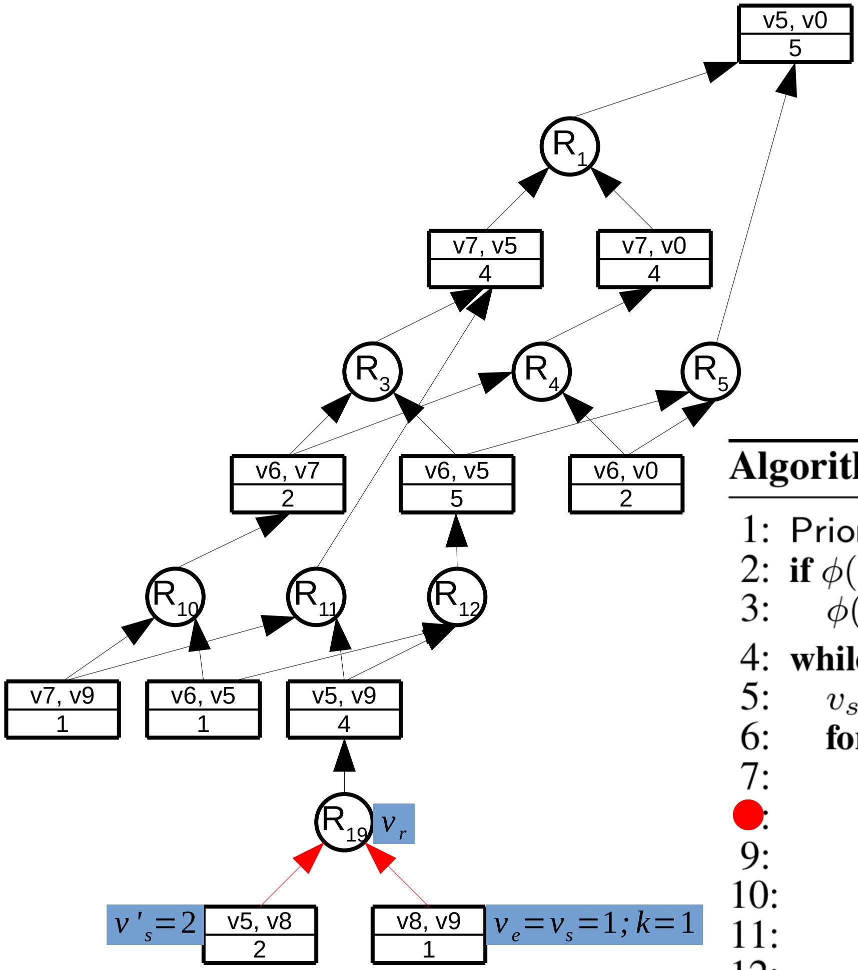
Algorithm 2 DCH$_{\text{scs}}$-WDec (G^*, G, e, k)

```
PriorityQueue $Q \leftarrow \emptyset$; $\phi(e, G) \leftarrow k$;
if $\phi(v_e, G^*) > k$ then
    $\phi(v_e, G^*) \leftarrow k$; $Q$.push$(v_e)$;
● while $Q \neq \emptyset$ do
    $v_s \leftarrow Q$.pop();
    for each $v_r \in$ nbr$^+(v_s)$ do
        $v'_s \leftarrow$ the other in-neighbor of $v_r$ except $v_s$;
        $v''_s \leftarrow$ nbr$^+(v_r)$;
        if $\phi(v''_s, G^*) > \phi(v_s, G^*) + \phi(v'_s, G^*)$ then
            $\phi(v''_s, G^*) \leftarrow \phi(v_s, G^*) + \phi(v'_s, G^*)$;
            if $v''_s \notin Q$ then
                $Q$.push$(v''_s)$;
```



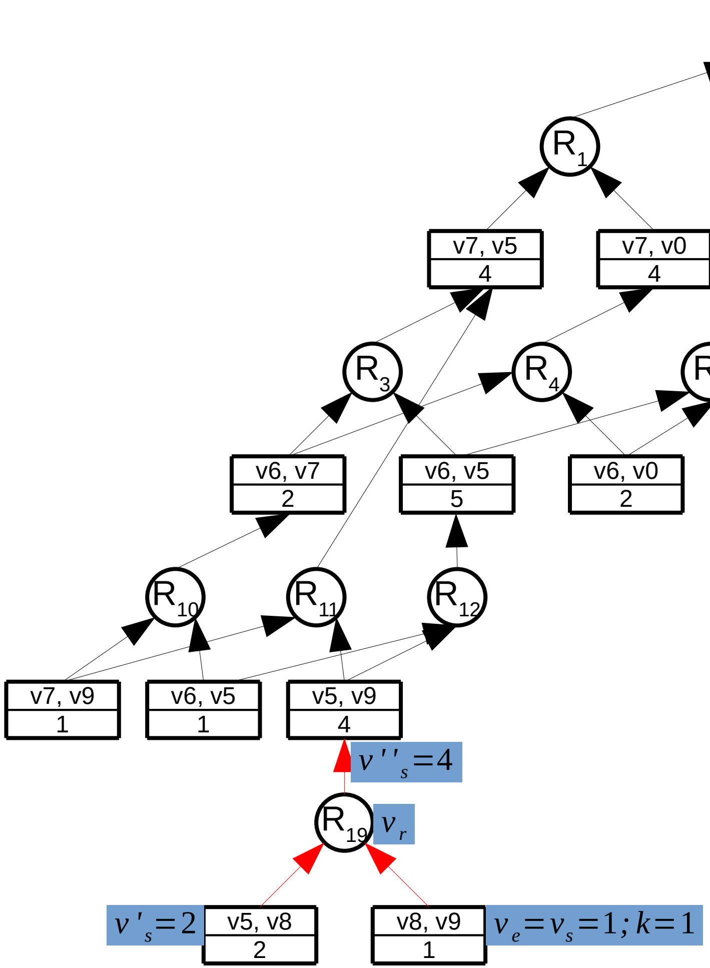
Algorithm 2 DCH$_{\text{scs}}$-WDec (G^*, G, e, k)

```
PriorityQueue $Q \leftarrow \emptyset$; $\phi(e, G) \leftarrow k$;
if $\phi(v_e, G^*) > k$ then
    $\phi(v_e, G^*) \leftarrow k$; $Q$.push$(v_e)$;
while $Q \neq \emptyset$ do
    $v_s \leftarrow Q$.pop();
    for each $v_r \in$ nbr$^+(v_s)$ do
        $v'_s \leftarrow$ the other in-neighbor of $v_r$ except $v_s$;
        $v''_s \leftarrow$ nbr$^+(v_r)$;
        if $\phi(v''_s, G^*) > \phi(v_s, G^*) + \phi(v'_s, G^*)$ then
            $\phi(v''_s, G^*) \leftarrow \phi(v_s, G^*) + \phi(v'_s, G^*)$;
            if $v''_s \notin Q$ then
                $Q$.push$(v''_s)$;
```



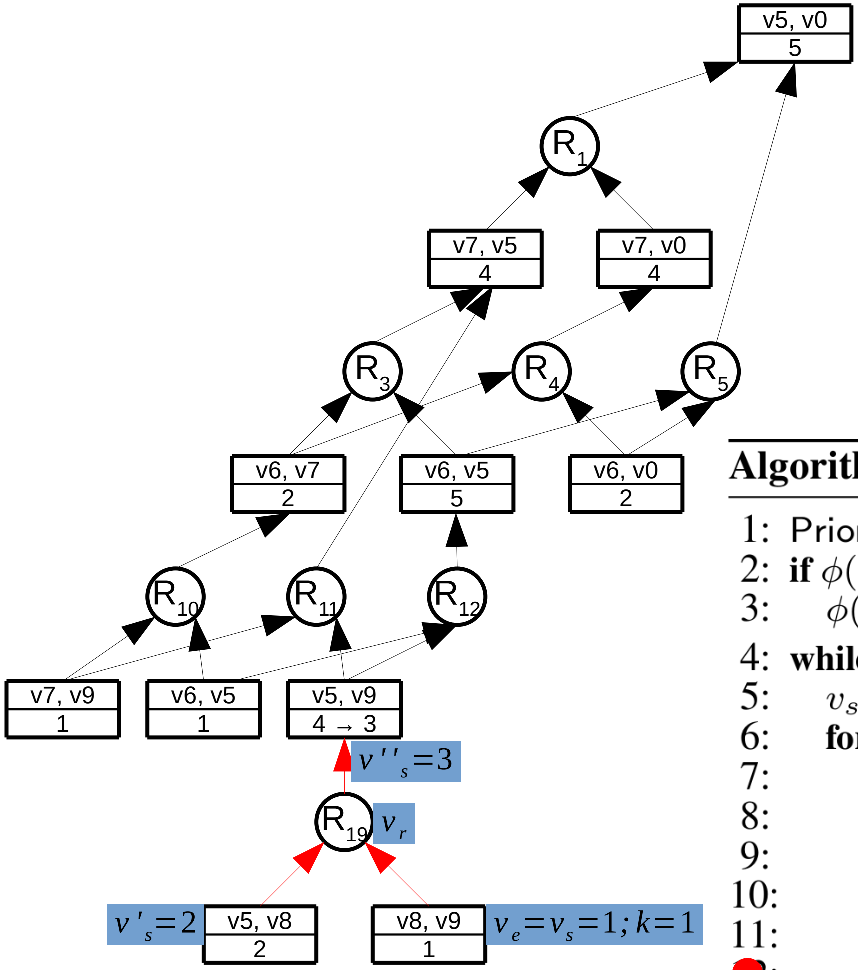
Algorithm 2 DCH$_{\mathsf{scs}}$-WDec (G^*, G, e, k)

```
PriorityQueue $Q \leftarrow \emptyset$; $\phi(e, G) \leftarrow k$;
if $\phi(v_e, G^*) > k$ then
    $\phi(v_e, G^*) \leftarrow k$; $Q$.push($v_e$);
while $Q \neq \emptyset$ do
    $v_s \leftarrow Q$.pop();
    for each $v_r \in \mathsf{nbr}^+(v_s)$ do
        $v'_s \leftarrow$ the other in-neighbor of $v_r$ except $v_s$;
●       $v''_s \leftarrow \mathsf{nbr}^+(v_r)$;
        if $\phi(v''_s, G^*) > \phi(v_s, G^*) + \phi(v'_s, G^*)$ then
            $\phi(v''_s, G^*) \leftarrow \phi(v_s, G^*) + \phi(v'_s, G^*)$;
            if $v''_s \notin Q$ then
                $Q$.push($v''_s$);
```



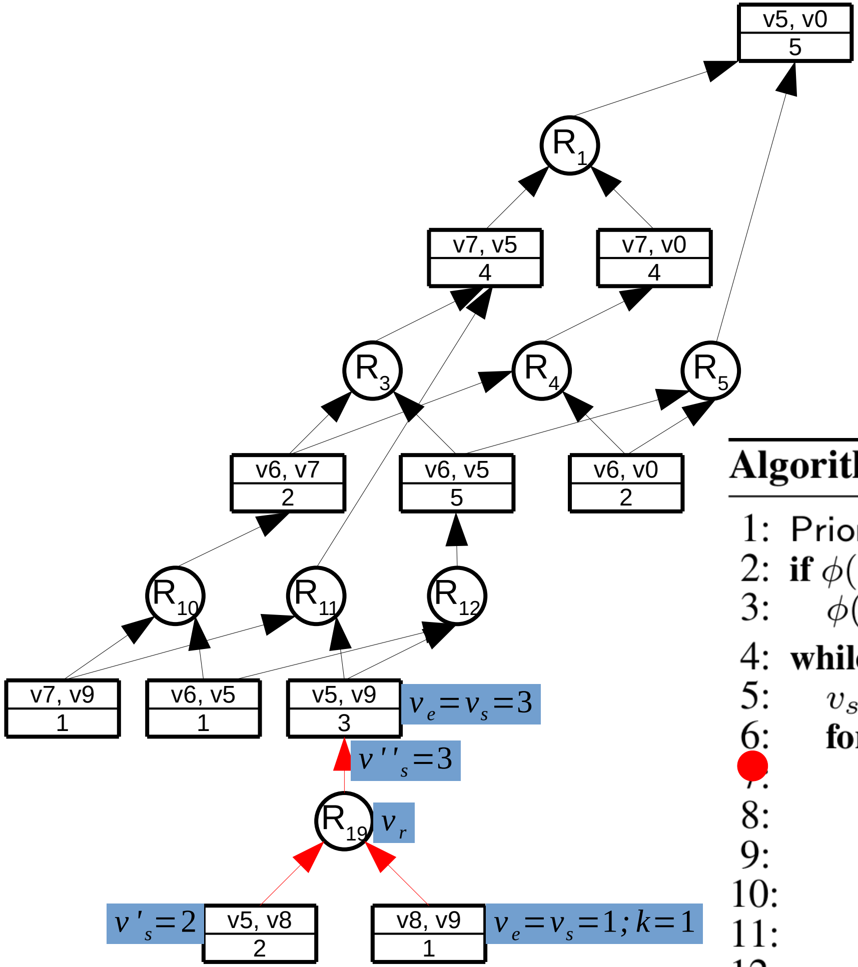
Algorithm 2 DCH$_{\text{scs}}$-WDec (G^*, G, e, k)

```
PriorityQueue $Q \leftarrow \emptyset$; $\phi(e, G) \leftarrow k$;
if $\phi(v_e, G^*) > k$ then
    $\phi(v_e, G^*) \leftarrow k$; $Q$.push$(v_e)$;
while $Q \neq \emptyset$ do
    $v_s \leftarrow Q$.pop$()$;
    for each $v_r \in$ nbr$^+(v_s)$ do
        $v'_s \leftarrow$ the other in-neighbor of $v_r$ except $v_s$;
        $v''_s \leftarrow$ nbr$^+(v_r)$;
        if $\phi(v''_s, G^*) > \phi(v_s, G^*) + \phi(v'_s, G^*)$ then
            $\phi(v''_s, G^*) \leftarrow \phi(v_s, G^*) + \phi(v'_s, G^*)$;
            if $v''_s \notin Q$ then
                $Q$.push$(v''_s)$;
```



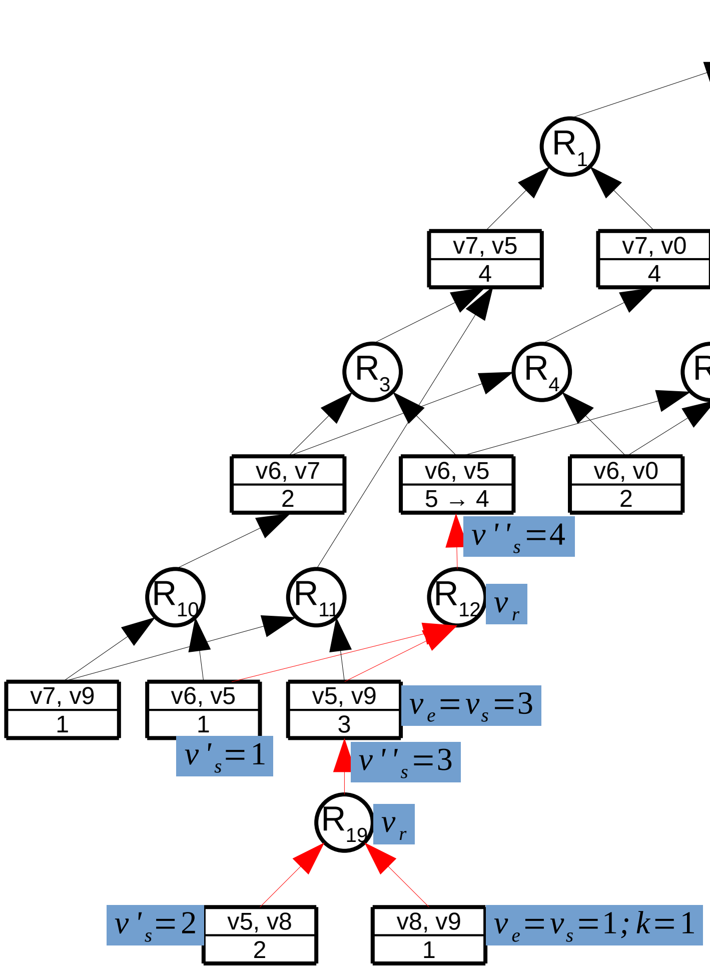
Algorithm 2 DCH$_{\text{scs}}$-WDec (G^*, G, e, k)

```
PriorityQueue $Q \leftarrow \emptyset$; $\phi(e, G) \leftarrow k$;
if $\phi(v_e, G^*) > k$ then
    $\phi(v_e, G^*) \leftarrow k$; $Q$.push$(v_e)$;
while $Q \neq \emptyset$ do
    $v_s \leftarrow Q$.pop();
    for each $v_r \in$ nbr$^+(v_s)$ do
        $v'_s \leftarrow$ the other in-neighbor of $v_r$ except $v_s$;
        $v''_s \leftarrow$ nbr$^+(v_r)$;
        if $\phi(v''_s, G^*) > \phi(v_s, G^*) + \phi(v'_s, G^*)$ then
            $\phi(v''_s, G^*) \leftarrow \phi(v_s, G^*) + \phi(v'_s, G^*)$;
            if $v''_s \notin Q$ then
                $Q$.push$(v''_s)$;
```



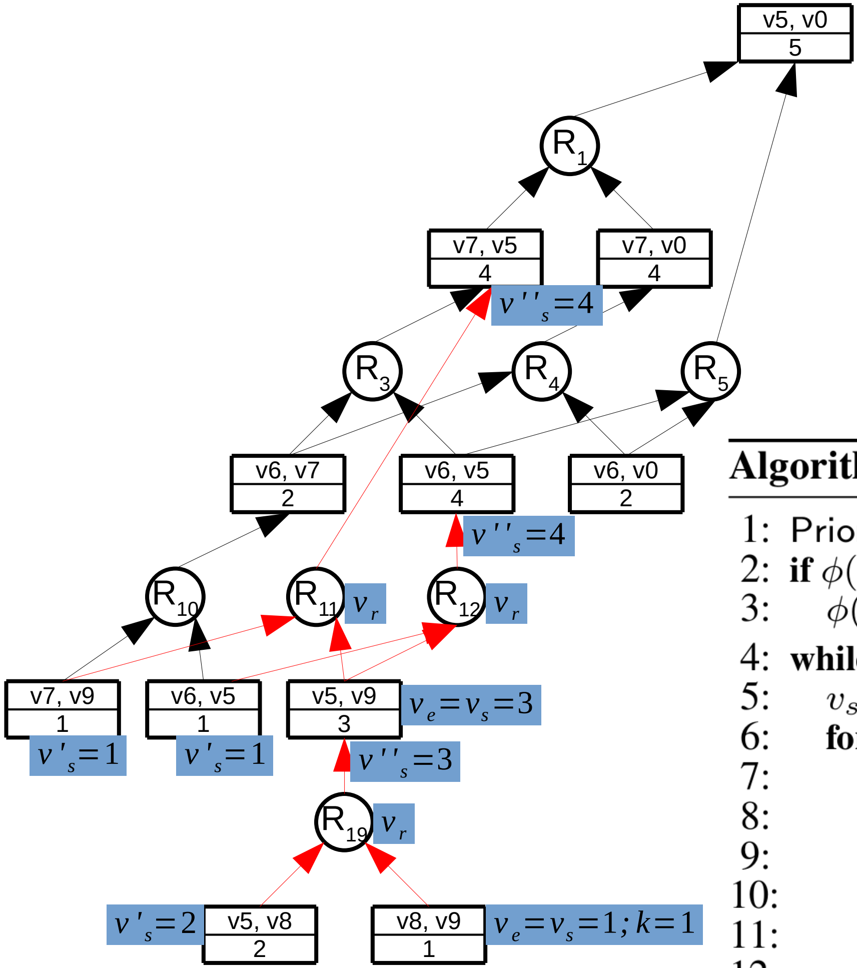
Algorithm 2 DCH$_{\text{scs}}$-WDec (G^*, G, e, k)

```
PriorityQueue $Q \leftarrow \emptyset$; $\phi(e, G) \leftarrow k$;
if $\phi(v_e, G^*) > k$ then
    $\phi(v_e, G^*) \leftarrow k$; $Q$.push$(v_e)$;
while $Q \neq \emptyset$ do
    $v_s \leftarrow Q$.pop();
    for each $v_r \in$ nbr$^+(v_s)$ do
        $v'_s \leftarrow$ the other in-neighbor of $v_r$ except $v_s$;
        $v''_s \leftarrow$ nbr$^+(v_r)$;
        if $\phi(v''_s, G^*) > \phi(v_s, G^*) + \phi(v'_s, G^*)$ then
            $\phi(v''_s, G^*) \leftarrow \phi(v_s, G^*) + \phi(v'_s, G^*)$;
            if $v''_s \notin Q$ then
                $Q$.push$(v''_s)$;
```



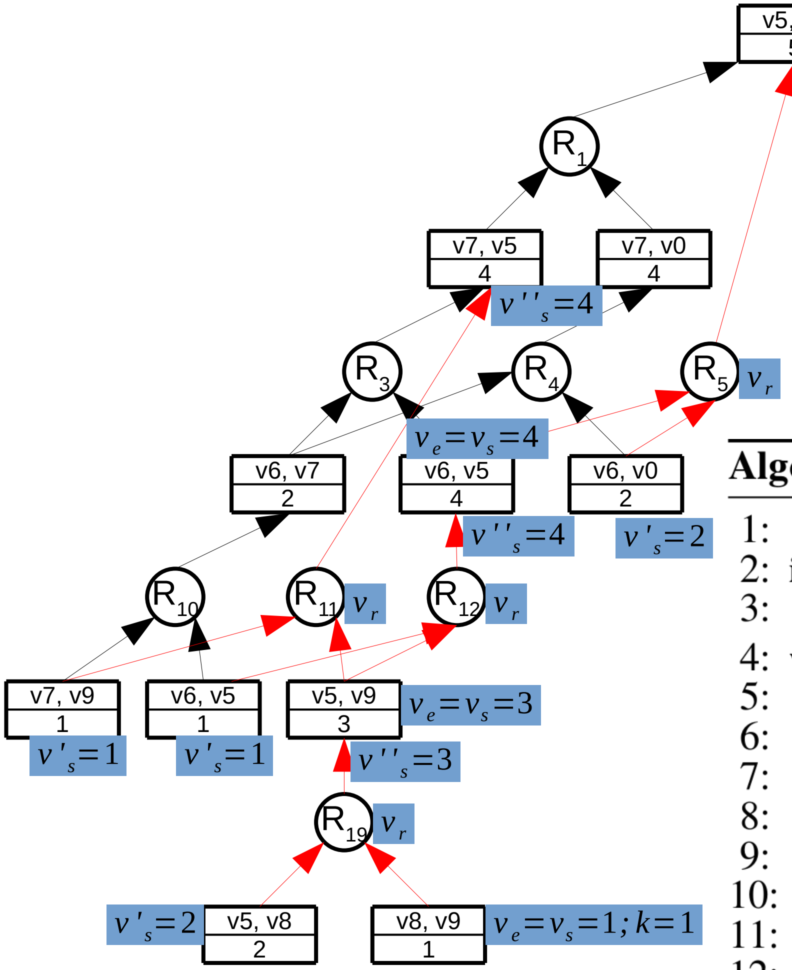
Algorithm 2 DCH$_{\text{scs}}$-WDec (G^*, G, e, k)

```
PriorityQueue $Q \leftarrow \emptyset$; $\phi(e, G) \leftarrow k$;
if $\phi(v_e, G^*) > k$ then
    $\phi(v_e, G^*) \leftarrow k$; $Q$.push$(v_e)$;
while $Q \neq \emptyset$ do
    $v_s \leftarrow Q$.pop();
    for each $v_r \in$ nbr$^+(v_s)$ do
        $v'_s \leftarrow$ the other in-neighbor of $v_r$ except $v_s$;
        $v''_s \leftarrow$ nbr$^+(v_r)$;
        if $\phi(v''_s, G^*) > \phi(v_s, G^*) + \phi(v'_s, G^*)$ then
            $\phi(v''_s, G^*) \leftarrow \phi(v_s, G^*) + \phi(v'_s, G^*)$;
            if $v''_s \notin Q$ then
                $Q$.push$(v''_s)$;
```



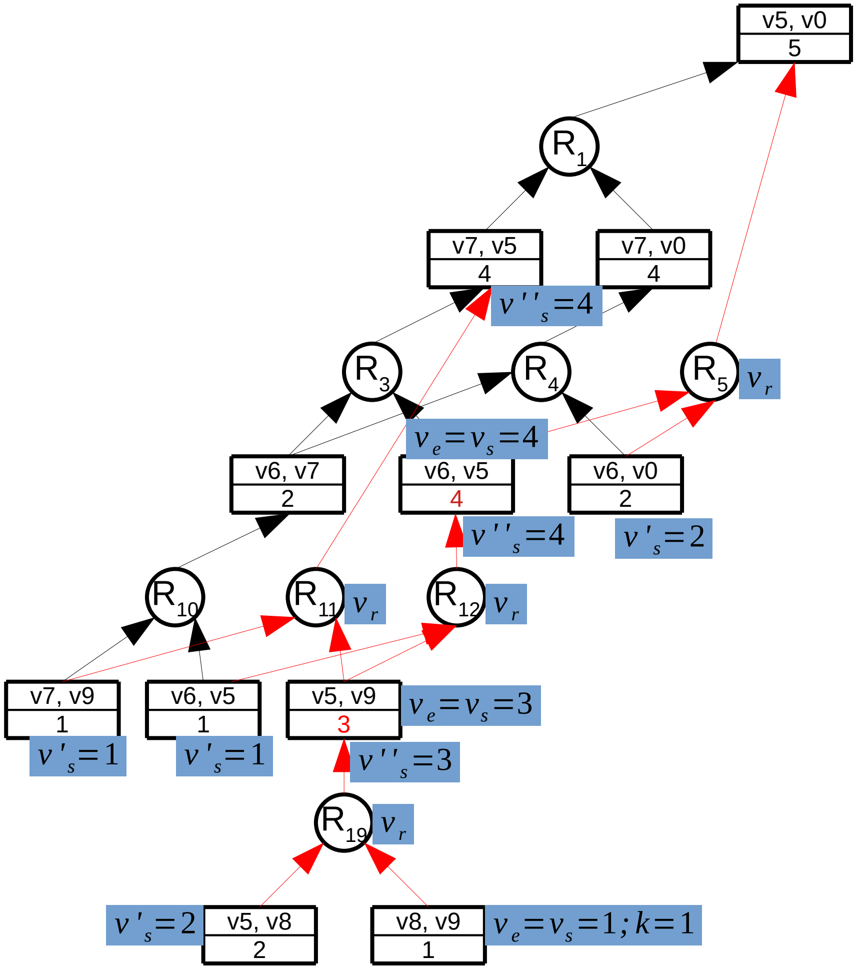
Algorithm 2 DCH$_{\text{scs}}$-WDec (G^*, G, e, k)

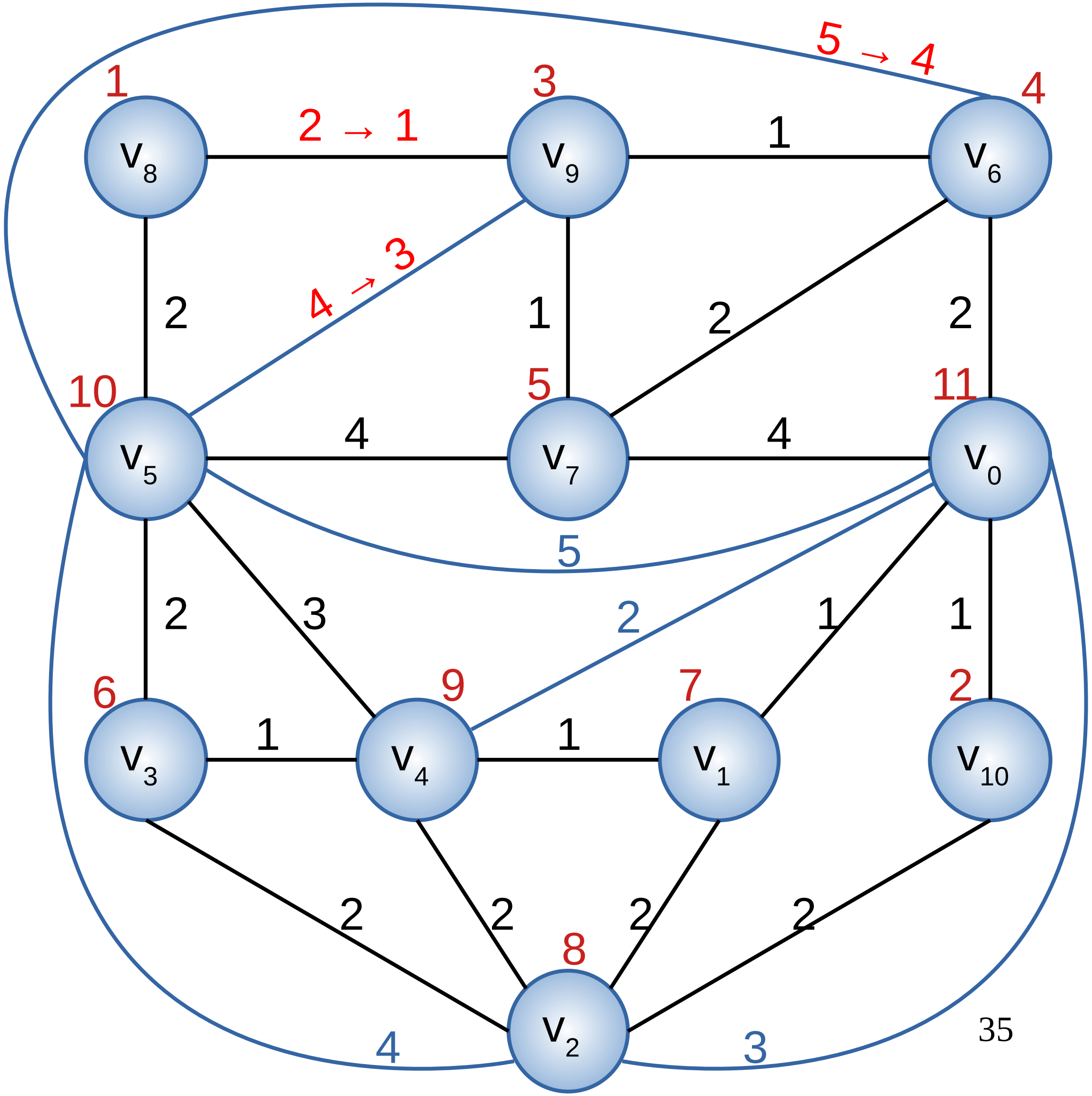
```
PriorityQueue $Q \leftarrow \emptyset$; $\phi(e, G) \leftarrow k$;
if $\phi(v_e, G^*) > k$ then
    $\phi(v_e, G^*) \leftarrow k$; $Q$.push($v_e$);
while $Q \neq \emptyset$ do
    $v_s \leftarrow Q$.pop();
    for each $v_r \in$ nbr$^+(v_s)$ do
        $v'_s \leftarrow$ the other in-neighbor of $v_r$ except $v_s$;
        $v''_s \leftarrow$ nbr$^+(v_r)$;
        if $\phi(v''_s, G^*) > \phi(v_s, G^*) + \phi(v'_s, G^*)$ then
            $\phi(v''_s, G^*) \leftarrow \phi(v_s, G^*) + \phi(v'_s, G^*)$;
            if $v''_s \notin Q$ then
                $Q$.push($v''_s$);
```



Algorithm 2 DCH$_{\text{scs}}$-WDec (G^*, G, e, k)

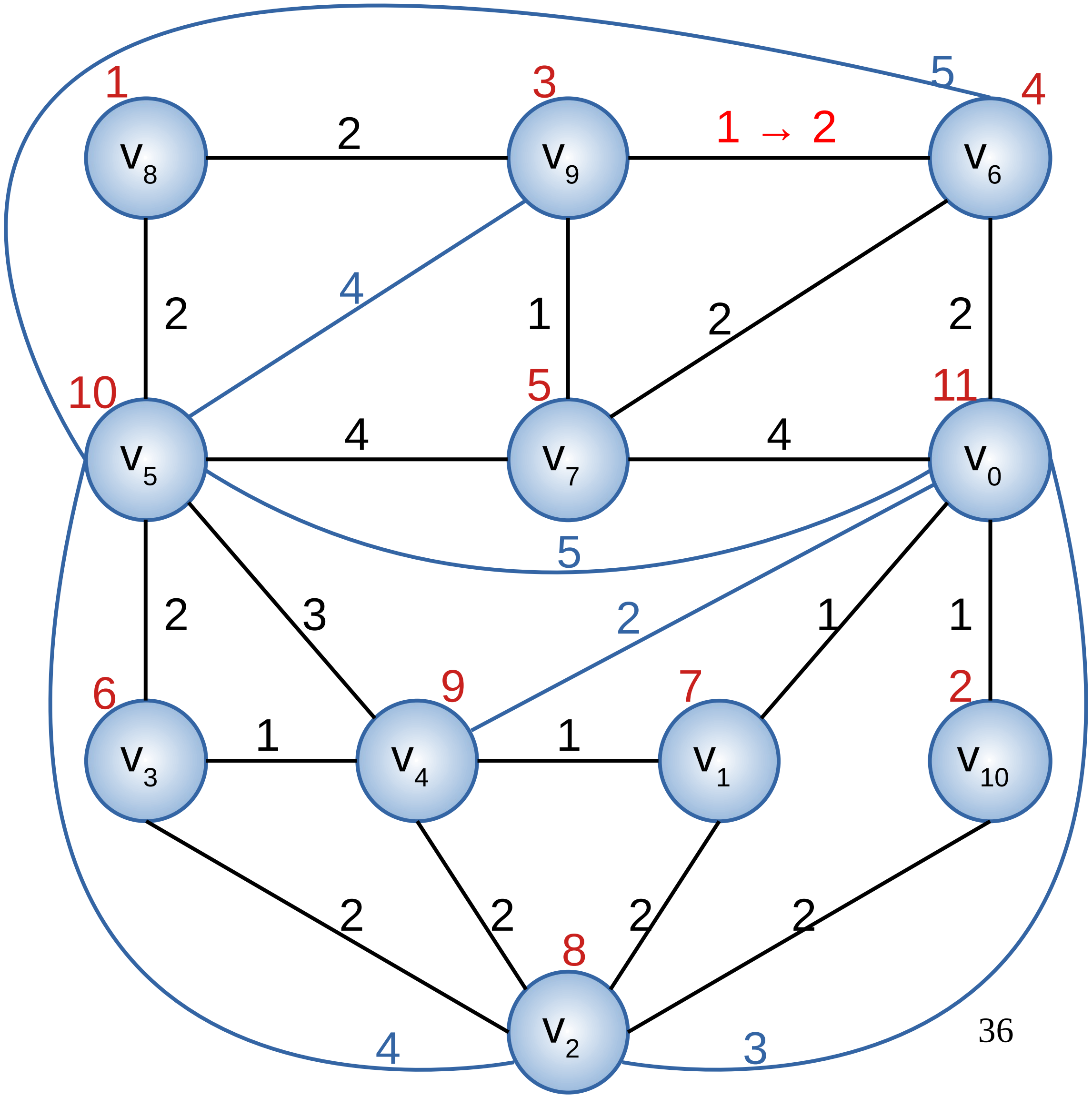
```
PriorityQueue $Q \leftarrow \emptyset$; $\phi(e, G) \leftarrow k$;
**if** $\phi(v_e, G^*) > k$ **then**
    $\phi(v_e, G^*) \leftarrow k$; $Q$.push$(v_e)$;
**while** $Q \neq \emptyset$ **do**
    $v_s \leftarrow Q$.pop();
    **for each** $v_r \in$ nbr$^+(v_s)$ **do**
        $v'_s \leftarrow$ the other in-neighbor of $v_r$ except $v_s$;
        $v''_s \leftarrow$ nbr$^+(v_r)$;
        **if** $\phi(v''_s, G^*) > \phi(v_s, G^*) + \phi(v'_s, G^*)$ **then**
            $\phi(v''_s, G^*) \leftarrow \phi(v_s, G^*) + \phi(v'_s, G^*)$;
            **if** $v''_s \notin Q$ **then**
                $Q$.push$(v''_s)$;
```


v5, v0
5
R1
v7, v5
4
v7, v0
4
v''s=4
R3
R4
R5
vr
ve=vs=4
v6, v7
2
v6, v5
4
v6, v0
2
v''s=4
v's=2
R10
R11
vr
R12
vr
v7, v9
1
v6, v5
1
v5, v9
3
ve=vs=3
v's=1
v's=1
v''s=3
R19
vr
v's=2
v5, v8
2
v8, v9
1
ve=vs=1;k=1


v8
v9
v6
v5
v7
v0
v3
v4
v1
v10
v2
1
3
4
10
5
11
6
9
7
2
8
2 → 1
5 → 4
4 → 3
1
2
1
2
2
4
4
5
2
3
2
1
1
1
1
2
2
2
2
4
3

Weight Increase

- Weight decrease ($v_9 v_6$) = 2 → 1
- Implies further weight changes
- Let's use G* to find out which



1
v_8
2
3
v_9
1 → 2
5
4
v_6
2
4
1
2
2
10
v_5
4
5
v_7
4
11
v_0
5
2
3
2
1
1
6
v_3
1
9
v_4
1
7
v_1
2
v_{10}
2
2
2
2
8
v_2
4
3