ALGORITHM 2: *MinOverlayScheduling*(*G*, *P*)

```
1: Initialize $min\_overlay \leftarrow +\infty$
2: Initialize $PASS \leftarrow MinBufferScheduling(G)$
3: Calculate current buffer usage, $buf\_mem$, of $PASS$
4: $code\_mem \leftarrow C_p - buf\_mem$
5: repeat
6:    $<V,R> \leftarrow RegionAssignment(G, PASS, code\_mem)$ /*Actor to region assignment*/
7:    if $\sum_{r \in R} C_r \le code\_mem$ then
8:       $<V,S> \leftarrow Segmentation(G, PASS, <V,R>)$ /*Actor to segment assignment*/
9:       $cur\_overlay \leftarrow calCodeOverlay(G, PASS, <V,R>, <V,S>)$ /*Overlay overhead*/
10:       if $cur\_overlay < min\_overlay$ then
11:          $min\_overlay \leftarrow overlay$
12:          $solution \leftarrow clone(G, PASS, <V,R>, <V,S>)$
13:       end if
14:    end if
15: until $collapseTwoExecs(PASS) = false$ /*Evolve from Min. Buf to Min. Switch*/
16: return $solution$
```