Algorithm 1 SeedSubTrees(Γ)

```
1: for comp ∈ {Components} do
2:    level = [ ]; {dynamic array of failed components at subTree level}
3:    nFailed = (0,0,...,0); {counts failed components}
4:    BFHist = {Array of linked lists };
      {breadth first history of subTrees, array is indexed by component,
      stores parent in linked list for each component}
5:
6:    if (NotEmpty(Γ_comp)) then
7:       append comp to level;
8:       nFailed[comp] = 1;
9:       append | to BFHist[comp]; {signifies comp has failed}
10:      AddSubTreeLevel(level, nFailed, BFHist, 1);
11:   end if
12: end for
```

Algorithm 2 AddSubTreeLevel(*level*, *nFailed*, *BFHist*, *subTreeRate*, *rootC*)

1: $nextLevelPossibilities = \bigtimes_{i=1}^{|level|} \mathcal{P}(\Gamma_{level[i]})$;
{Cartesian Product (Power Sets (Ordered Set))}
2: **for** $oneNextLevelPossibility \in nextLevelPossibilities$ **do**
3: $nextLevel = oneNextLevelPossibility$;
4:
5: **for** $parentC \in level$ **do**
6: **for** $childC \in \Gamma_{parentC}$ **do**
7: **if** $childC \in nextLevel$ **then**
8: **if** $nFailed[childC] ==$ Redundancy($childC$) **then**
9: goto line 3; {invalid subtree, requires more comps than total}
10: **end if**
11: $nFailed[childC] = nFailed[childC] + 1$;
12: append | to $BFHist[comp]$; {signifies comp has failed}
13: $subTreeRate = subTreeRate * \phi_{parentC,\ childC}$;
{update rate with ϕ}
14: **else**
15: append $parentC$ to $BFHist[childC]$; {signifies comp has not failed}
16: **end if**
17: **end for**
18: **end for**
19:
20: **if** Children Added **then**
21: AddSubTreeLevel(*level*, *nFailed*, *BFHist*, *subTreeRate*, *rootC*);
{subTree can be grown further}
22: **else**
23: ProcessRates(*nFailed*, *BFHist*, *subTreeRate*, *rootC*);
{subTree is stunted}
24: **end if**
25: **end for**

Algorithm 3 ProcessRates(*nFailed*, *BFHist*, *subTreeRate*, *rootC*)

1: **for** $x \in Q$ **do**
2: $y = x + nFailed$;
3: $env =$ Environment(x);
4: **if** Valid State(y) **then**
5: $n =$ Redundancy($rootC$) $- x[rootC]$;
6: $rootFailureRate = n * \lambda_{rootC,\, env}$;
7: $complementRate = 1$; {complement rate of comps that could have failed}
8: **for** $comp \in \{Components\}$ **do**
9: $compsAvailable =$ Redundancy($comp$) $- x[comp]$;
10: **for** $parentC \in BFHist[comp]$ **do**
11: **if** $parentC ==$ | **then**
12: $compsAvailable = compsAvailable - 1$;
13: **else if** $compsAvailable > 0$ **then**
14: $complementRate = complementRate * (1 - \phi_{parentC,\, comp})$;
15: **end if**
16: **end for**
17: **end for**
18: **end if**
19: $Q(x, y) = rootFailureRate * subTreeRate * complementRate$;
20: **end for**