**Supplementary file of “Livenees Analysis and Deadlock Controller Design for Flexible Assembly Systems Using Petri Nets”**

Algorithm 1 (compute all A-paths and A-chains)

Input: APNMR (*N*, *M*0);

Output: Ψ, the set of all A-paths, and *K*, the set of all A-chains;

1. Let τ = {α = *pt* | *p* ∈ *PA* and *t* = *p*• ∩ *T*} and Ψ = τ;
2. **for each** α = *pt* ∈ τ
3. **for each** α′ = *p*1′*t*1′…*pn*′*tn*′ ∈ Ψ
4. **if** *t* ∈ •*p*1′
5. Let α′′ = *pt p*1′*t*1′…*pn*′*tn*′ and add α′′ into Ψ;
6. **end**
7. **if** *tn*′ ∈•*p*
8. Let α′′ = *p*1′*t*1′…*pn*′*tn*′*pt* and add α′′ into Ψ;
9. **end**
10. **end**
11. **end**
12. Let Κ = Ψ;
13. **for each** α = *p*1*t*1…*pktk* ∈ Ψ //Ψ is the set of all A-paths
14. **for each** α′ = *p*1′*t*1′…*pn*′*tn*′ **∈** Κ
15. **if** (*r*)*tk* = ℜ(*p*1′)
16. Let α1 = αα′ = *p*1*t*1…*pktkp*1′*t*1′…*pn*′*tn*′ and add α1 into Κ;
17. **end**
18. **end**
19. **end**
20. output Ψ and Κ;

*The computational complexity of Algorithm 1*: let *n* be the number of activity places, and hence |τ| = *n*. Note that there are at most A-paths in the system, that is, |Ψ| ≤ . Thus, the complexity of lines 2−11 is *O*(). On the other hand, since there are at most A-chains, the complexity of lines 13−19 is *O*(()()). Therefore, the complexity of Algorithm 1 can be considered as *O*().

Algorithm 2 (compute all A-circuits)

Input: APNMR (*N*, *M*0);

Output: Ξ, the set of all A-circuits;

1. According to Algorithm 1, obtain K, the set of all A-chains;
2. let Ξ = Κ;
3. **for each** θ = *p*1*t*1…*pktk* ∈ Κ
4. **if** (*r*)*tk ≠* ℜ(*p*1)
5. Delete θ from Ξ;
6. **end**
7. **end**
8. output Ξ;

*The computational complexity of Algorithm 2*: The analysis shows that there are most A-chains, hence, the complexity of Algorithm 2 is *O*(), where *n* denotes the number of activity places.

Algorithm 3 (compute all closed Ω-structures)

Input: APNMR (*N*, *M*0);

Output: Ξ, the set of all A-circuits;

1. According to Algorithm 1, obtain Ψ (the set of all A-paths) and *K* (the set of all A-chains);
2. Let Ε = ∅;
3. **for each** A-chainα = *p*1*t*1…*pktk*∈ Κ
4. **for each** A-path α′ = *p*1′*t*1′…*pn*′*tn*′ ∈ Ψ
5. **if** ℘(α) ∩ ℘(α′) = ∅ and *tk* = *tn*′
6. let *v* = (α, α′) and add *v* into Ε;
7. **end**
8. **end**
9. **end**
10. let Τ = Ε;
11. **for each** *v-*structure *v* = (α, α′) ∈ Ε
12. **for each** Ω-structure *w* = α1α1′α2α2′…α*k*α*k*′ ∈ Τ
13. **if** ℜ(ϑ(α′)) = ℜ(ϑ(α1))
14. Let *w′* = αα′α1α1′α2α2′…α*k*α*k*′ and add *w′* into Τ;
15. **end**
16. **if** ℜ(ϑ(α*k*′)) = ℜ(ϑ(α))
17. Let *w′* = α1α1′α2α2′…α*k*α*k*′αα′ and add *w′* into Τ;
18. **end**
19. **end**
20. let Θ = ∅;
21. **for each** *w*=α1α1′α2α2′…α*k*α*k*′=(Π1, Π2)∈Τ //Π1 ={α*i*, *i*∈*Zk*}, Π2 = {α*i*′,*i*∈*Zk*}
22. **if** ℘(Π1) ∩ ℘(Π2) = ∅ and ℜ(ϑ(α1)) = ℜ(ϑ(α*k*′))
23. Add *w* into Θ;
24. **end**
25. **end**
26. output Θ;

*The computational complexity of Algorithm 3*: since the number of A-paths (resp. A-chains) is less than (resp. ), then the number of *v-structures* is less than ][] and the complexity of lines 3−9 is *O*(][]). Each Ω-structure is constructed by several *v*-structures, thus there are at most Ω-structures. Therefore, the complexity of Algorithm 3 is *O*().