Technische Universität
Kaiserslautern

o1o1seda o1o1oo

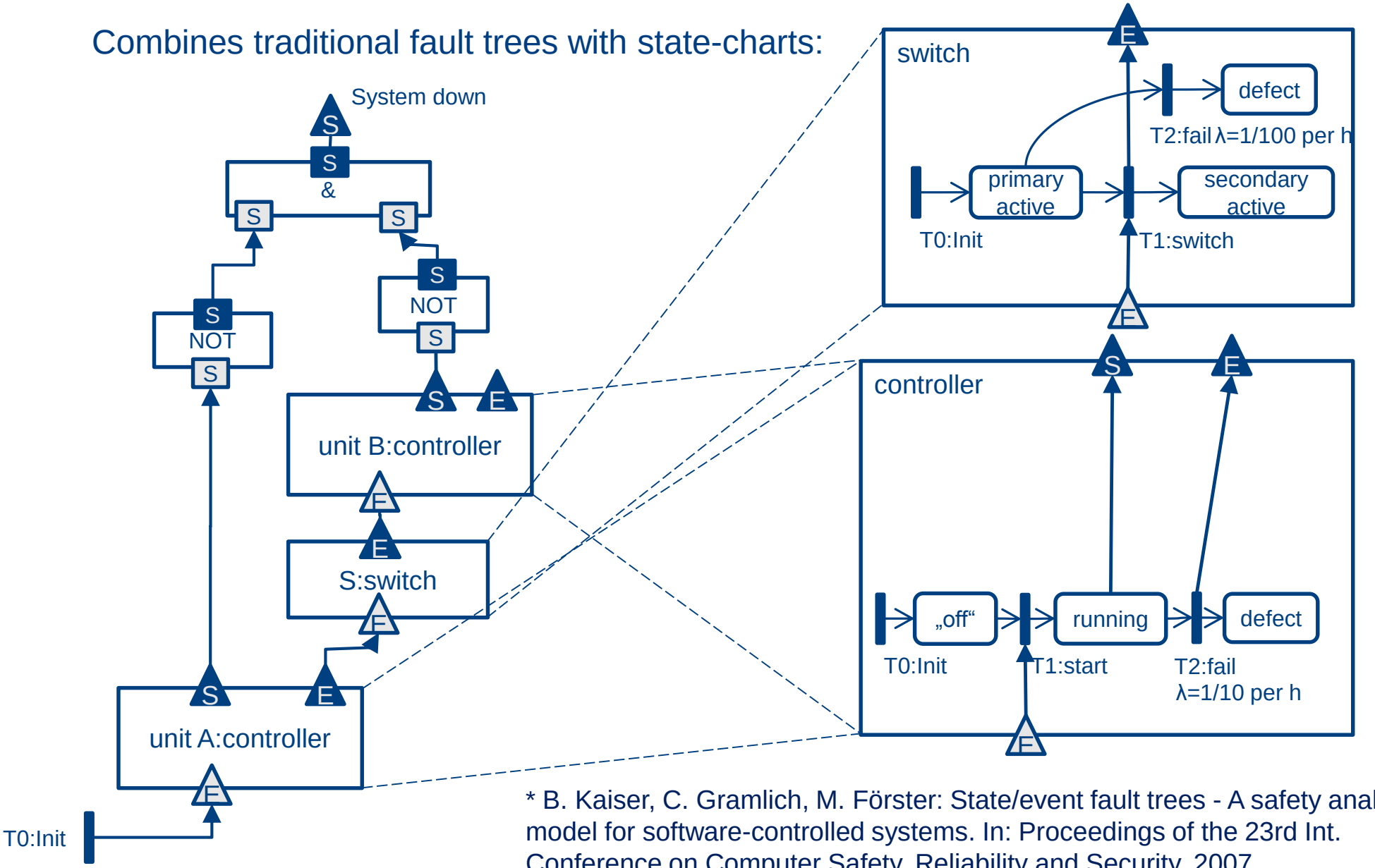
software engineering dependability

Efficient Reachability Graph Development for State/Event Fault Trees

Michael Roth, Agus Hartoyo and Peter Liggesmeyer
University of Kaiserslautern

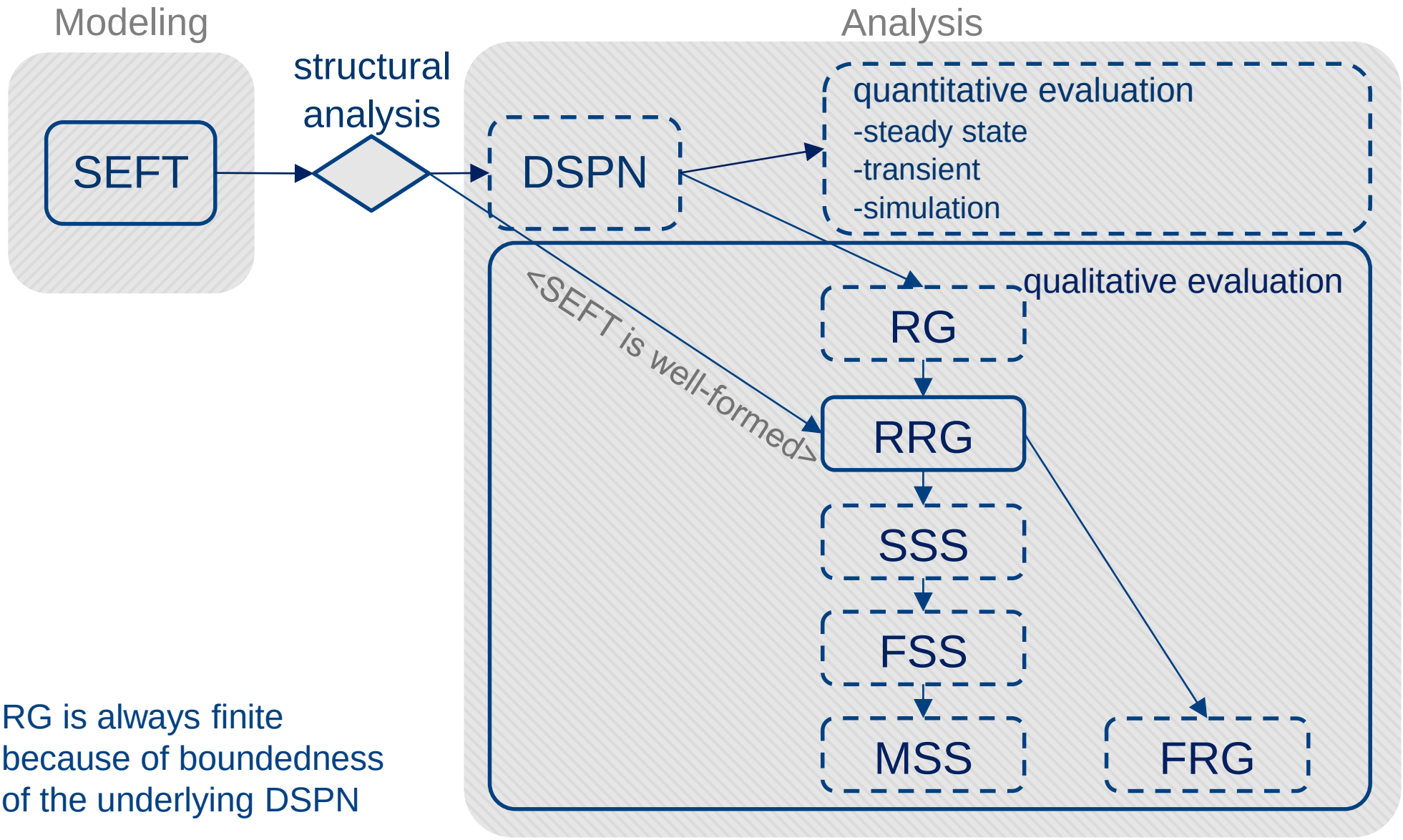
State/Event Fault Trees*

Combines traditional fault trees with state-charts:



* B. Kaiser, C. Gramlich, M. Förster: State/event fault trees - A safety analysis model for software-controlled systems. In: Proceedings of the 23rd Int. Conference on Computer Safety, Reliability and Security, 2007

State/Event Fault Tree Analysis



RG is always finite because of boundedness of the underlying DSPN

Direct RRG development

Dynamic part of the SEFT:*

1a) Build complete state set (Cartesian product): A x S x B

1b) Filter out invalid states (via unreachable state pairs)

→ **Valid state set**

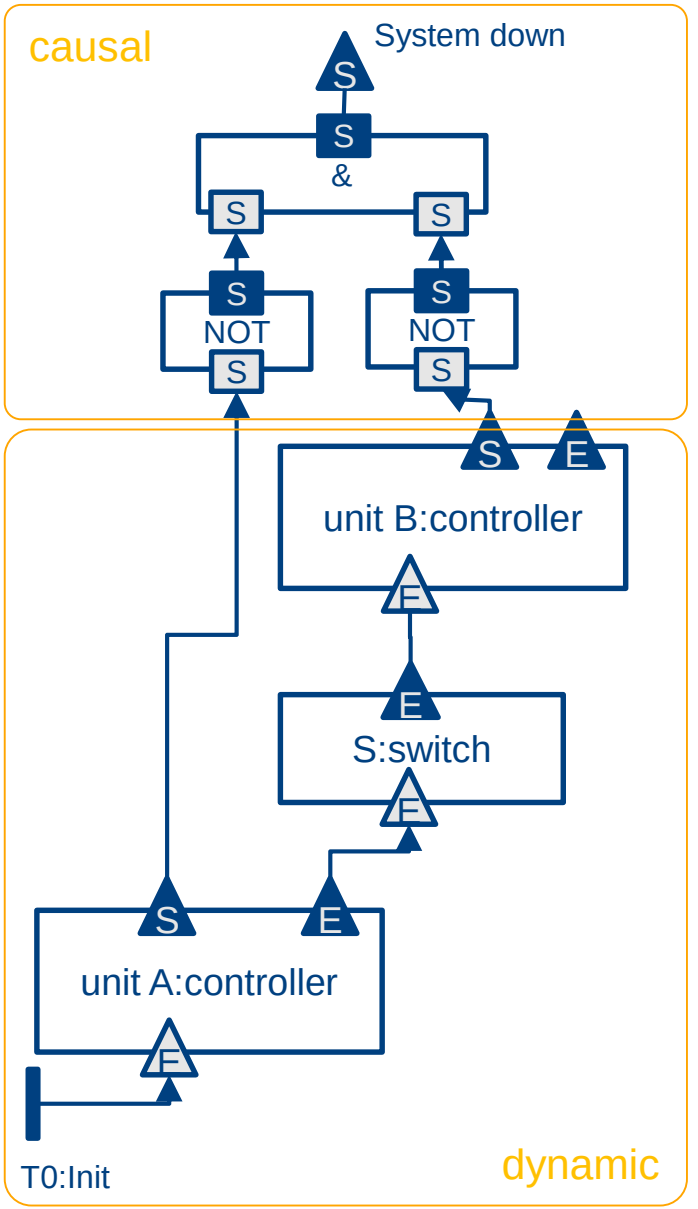
2) Insert state transitions (link pattern table)

3) Reachability analysis (ensures a valid reachability graph)

→ **Reduced Reachability Graph**

Causal part of the SEFT:*

4) Search hazardous reachability states



* M. Förster, B. Kaiser: Increased efficiency in the quantitative evaluation of State/Event Fault Trees. In: Proceedings of the 12th IFAC Symposium on Information Control Problems in Manufacturing, Elsevier Science, 2006

Well-formed SEFTs

Dynamic part of the SEFT:*

1a) Build complete state set (Cartesian product): A x S x B

1b) Filter out invalid states (via unreachable state pairs)

→ possible in case of **direct** triggered event relations

→ **Valid state set**

2) Insert state transitions (link pattern table)

3) Reachability analysis (ensures a valid reachability graph)

→ **Reduced Reachability Graph**

Causal part of the SEFT:*

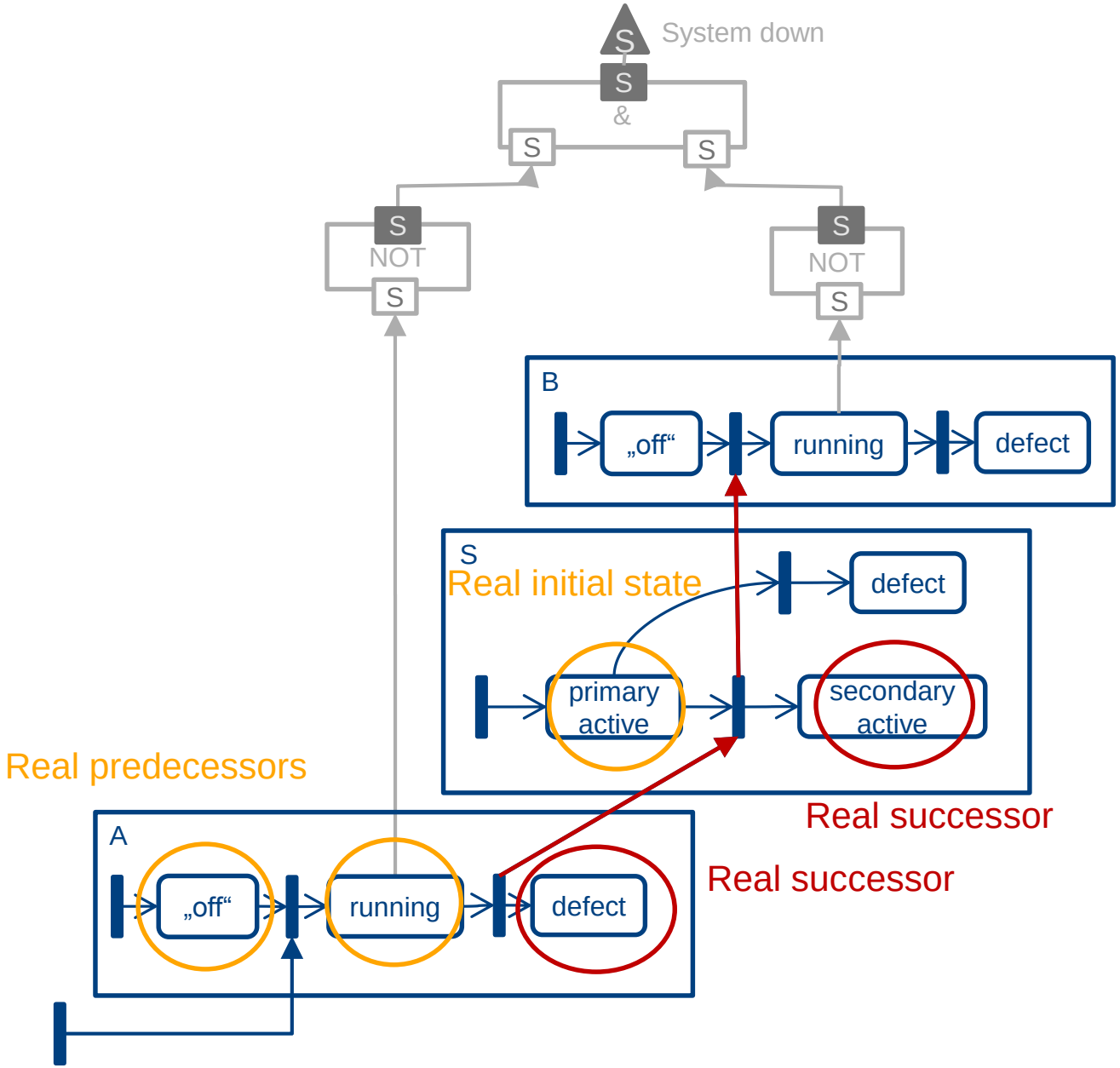
4) Search hazardous reachability states

→ possible in case of **"Boolean" state-based** gates

↓

Well-formed SEFTs

Dynamic model: Unreachable state pairs

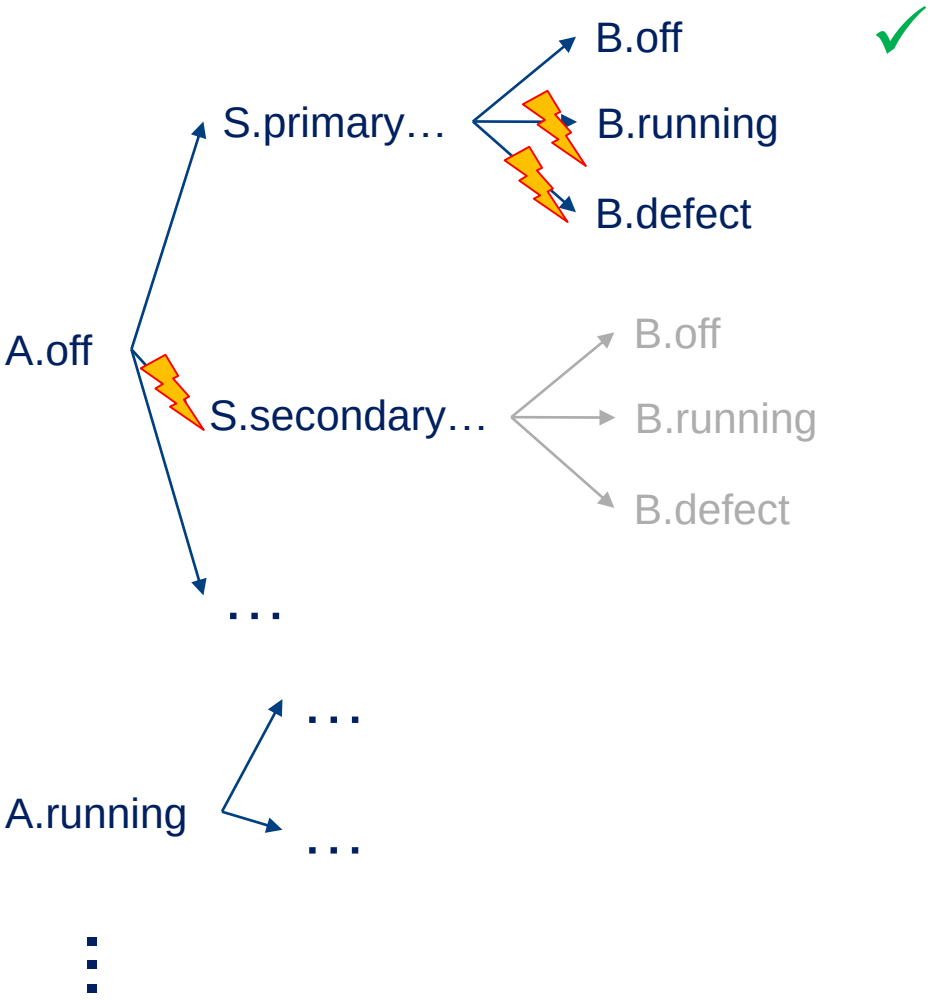


- A.off; S.secondary active
- A.running; S.secondary …
- A.defect; S.primary active
- S.defect; B.running
- S.defect; B.defect
- S.primary …; B.running
- S.primary …; B.defect
- S.secondary active; B.off

- A.off; B.running
- A.off; B.defect
- A.running; B.running
- A.running; B.defect

Dynamic model: Valid state detection

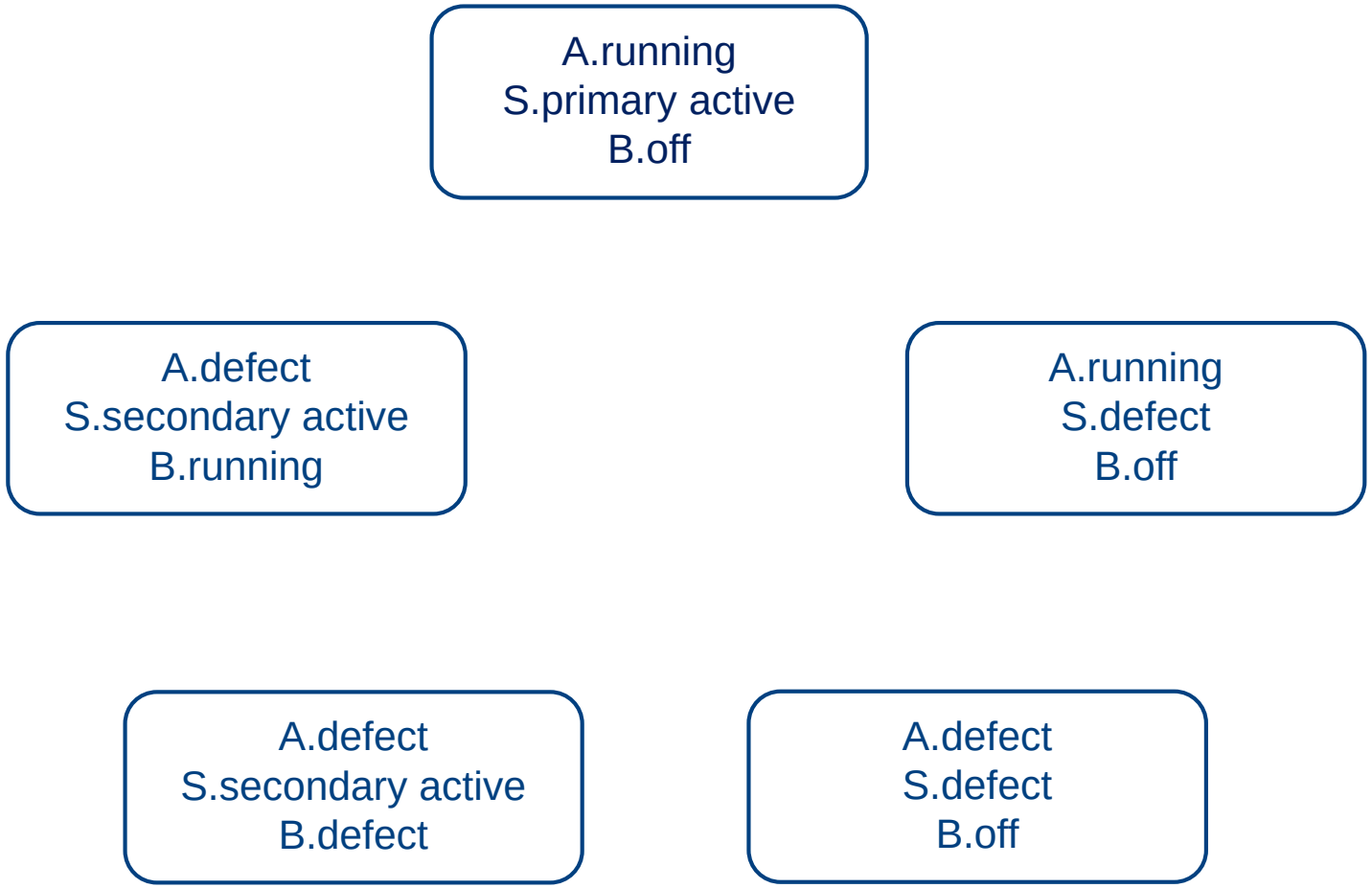
Valid state set (A x S x B):



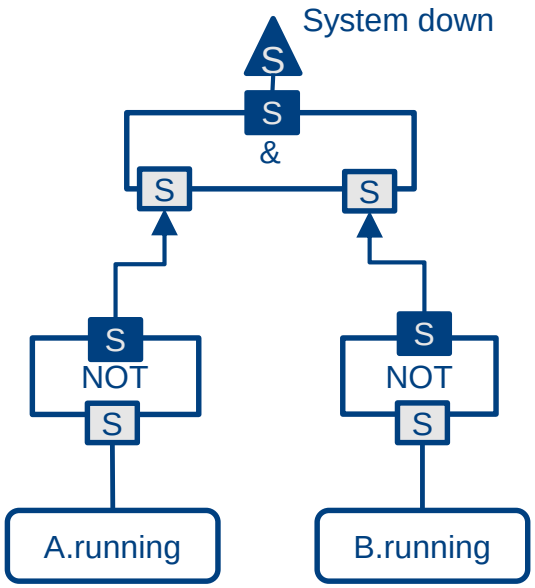
Unreachable pairs:

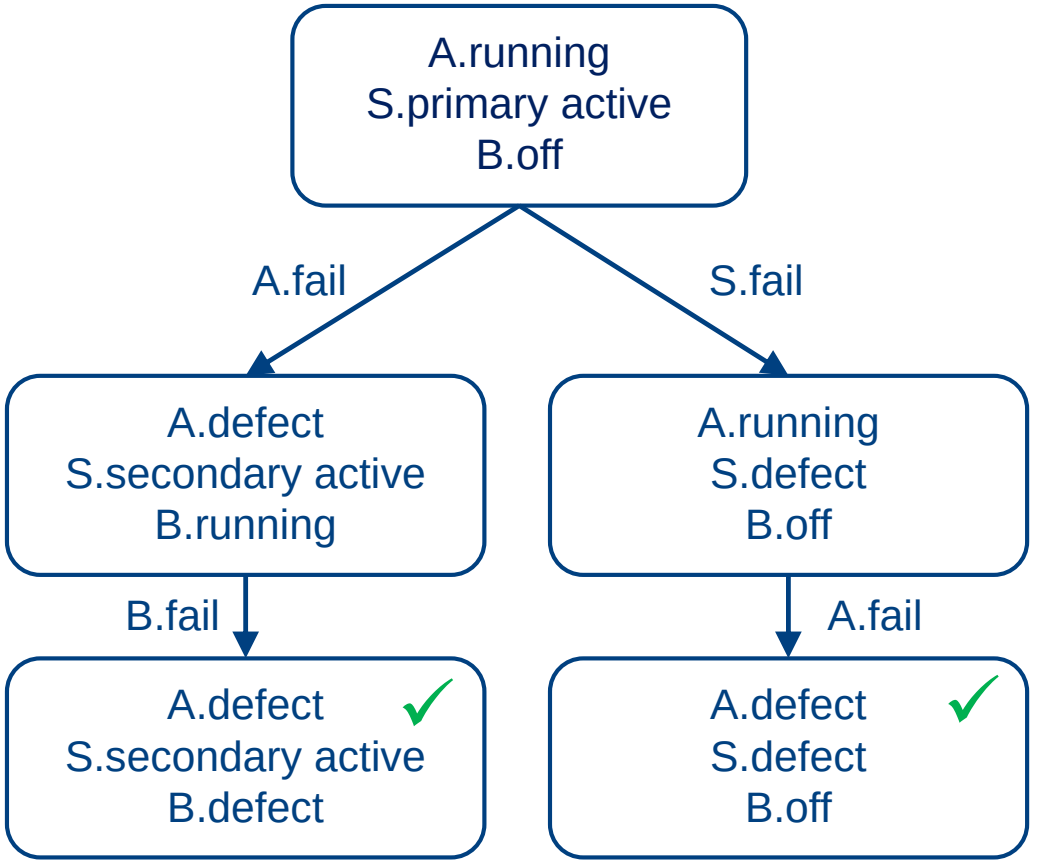
- A.off; S.secondary active
- A.running; S.secondary …
- A.defect; S.primary active
- S.defect; B.running
- S.defect; B.defect
- S.primary …; B.running
- S.primary …; B.defect
- S.secondary active; B.off
- A.off; B.running
- A.off; B.defect
- A.running; B.running
- A.running; B.defect

Valid State Space



RRG development





A.fail → B.fail v S.fail → A.fail

Evaluation

		CM	CM³	FA	FA⁴	HDS	HDS²
	Components	4	12	3	12	4	12
	states	13	39	7	28	9	18
	events	19	57	10	40	9	18
state set	Cartesian product	60	216000	9	6561	81	6561
	direct method	35	42875	9	6561	22	484
	traditional method	138	439482	49	125325	248	61504
vanishing states	direct method	0	0	0	0	0	0
	traditional method	103	396607	40	118764	224	61020
generation time	direct method	<10ms	14min	<1ms	13s	5ms	7s
	traditional method	47ms	~6h	16ms	21min	31ms	11min
	memory consumption	~25%	~10%	~18%	~5%	~9%	~1%

FA: Kaiser et al.: State/event fault trees - A safety analysis model for software-controlled systems. 2007

HDS: Yong, Dugan: Modular solution of dynamic multi-phase systems. IEEE Transactions on Reliability, Vol 53, 2004

Conclusion

Summary

- Qualitative analysis process of SEFTs
- Well formed SEFTs
- Direct RG development
- Significant improvement of the computation time/memory allocation

Perspectives

- Loosen the restrictions of well-formed SEFTs
- Use the direct method also in case of quantitative analysis