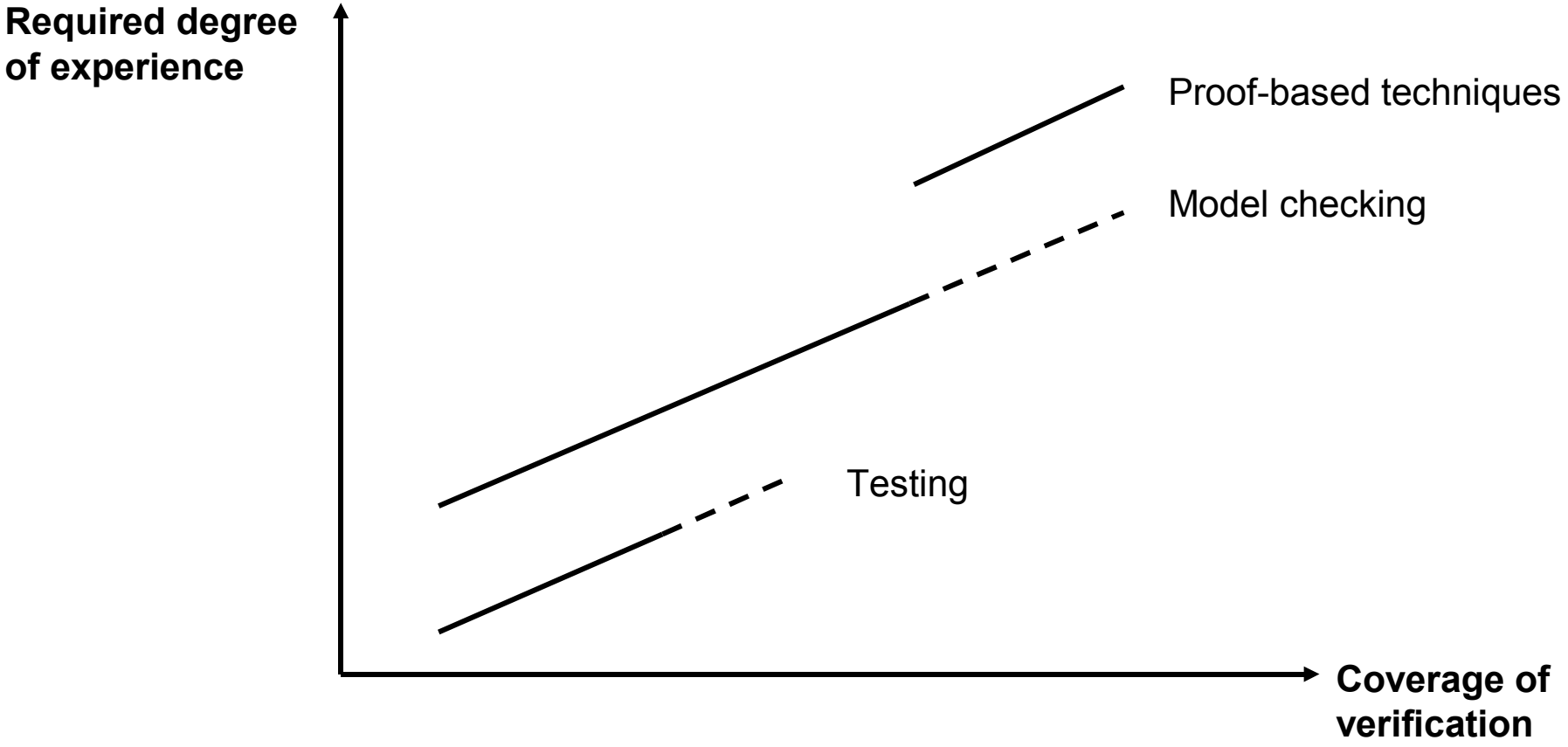
IEEE 1394 (FireWire) Workshop

A Simple Verification of the Tree Identify Protocol with SMV

Viktor Schuppan
Viktor.Schuppan@inf.ethz.ch

Armin Biere
Armin.Biere@inf.ethz.ch

Model Checking between Testing and Theorem Proving



Model Checking with SMV
- Process -

1. Build model as state machine in SMV input language.
2. Give specification as CTL formula.
3. Check specification with SMV.
4. Refine model and specification, check again.

Model Checking with SMV
- Tool -

- BDD-based symbolic model checker
- Developed by Ken McMillan
- Several variants available, e.g. Bwolen Yang, NuSMV, Cadence
- Hardware-oriented input language
- Synchronous or interleaving execution
- No continuous real time model / specification
- Communication by shared variables

Model - Node

- Basic building block
- Contains entire state machine
- All states of Tree Id Protocol are implemented
- State T3 refined, state S0 added
- Directly use line-states for communication
- Time-out and force-root are modeled with counters
- Resolution of root contention: nodes choose paths of different length

Model - Configuration

Properties of the model involving several nodes, e.g.:

- Sound interconnection of nodes:
  - node[i].port[j] = (k, l) -> node[k].port[l] = (i, j)
  - reachability of nodes
  - no cycles
- Initial configuration
- Different paths are eventually chosen in root contention

Model - Variants

	# 1	# 2	# 3
interleaved execution	✔	✘	✘
force root non-determinism	✔	✔	✔
configuration non-determinism	✘	✘	✔
number of nodes non-determinism	✘	✘	?

Specification - Part 1

1. A leader is eventually chosen.

   AF (AG node[0].root | ...)

2. Only one leader is chosen.

   AF AG ((node[0].root -> !node[1].root & ...) &

   (node[1].root -> !node[0].root & ...) ...)

Specification - Part 2

3. Every node reaches state S0.
4. All roles are finally determined.
5. All links are finally idle.
6. No timeout.
7. No known problems.
8. *Configuration dependent.*
9. Force root takes effect.
10. Once a leader is chosen it doesn't change.

Results

- Synchronous execution:
  all properties are verified.

- Interleaving execution:
  a well known timing issue shows up
  (described e.g. by Simons and Stoelinga):
  the protocol may fail if nodes can have
  processing time > ROOT_CONTEND_FAST

Results - Data

	Run time [s]	Bytes allocated [MBytes]	# states reachable	# states	
10 det.	3	29	2^38	2^276	# 2
10 frn.	409	429	2^50	2^276	
20 det.	15	99	2^80	2^551	
20 frn.	-	-	-	-	
3 det.	64	272	2^13	2^122	# 3
3 frn.	65	273	2^17	2^122	
3 cfn.	69	293	2^28	2^122	
3 n.	2852	463	2^32	2^122	
5 det.	292	554	?	?	
5 frn.	274	554	?	?	
5 cfn.	-	-	-	-	
5 n.	-	-	-		

Cfg: PIII 850, 1,5 GB RAM, Linux 2.2.18

Evaluation

- First author only recently started PhD
- Experience in software engineering, not in model checking
- ~ 2 weeks of introductory reading
- First prototype completed in about one week
- Refinement process started; problem: turn around time
- Model is easier to come up with than specification
- Experience required in formulation of model and selection and operation of tool to keep run times low
- Problem: research versions provide limited features

Conclusion

Model checking proved effective:

- The first model was developed quickly
- Verification was straight forward
- Extension based on first model are easily possible

Limitations:

- Experience is needed for verification of larger model:
  formulation
  execution
- Limited scalability for larger number of nodes