Where did I go wrong?
Explaining errors in process models

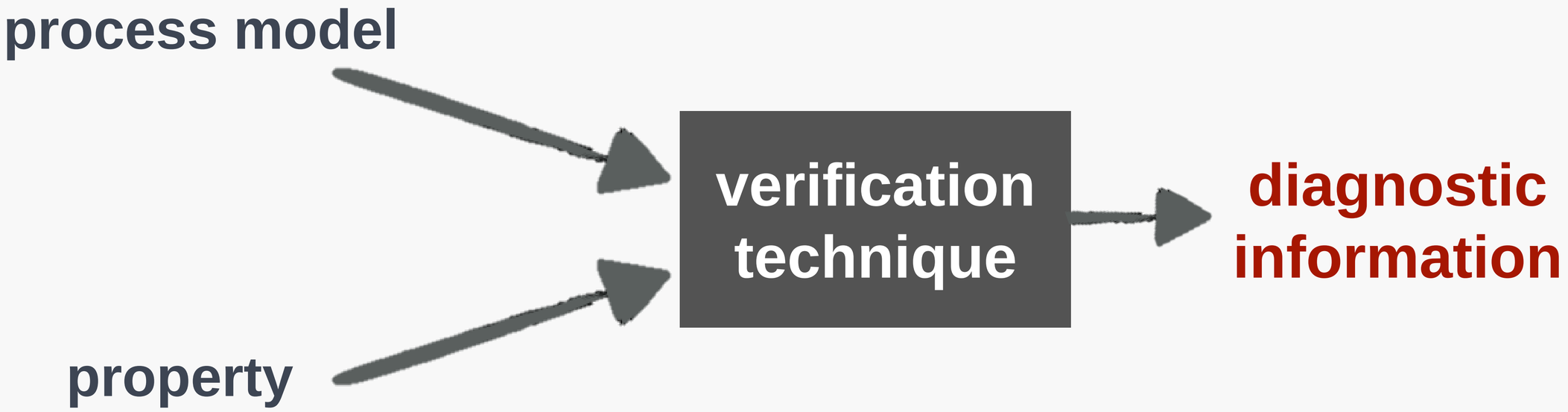
Niels Lohmann
@nlohmann

Dirk Fahland
@dfahland

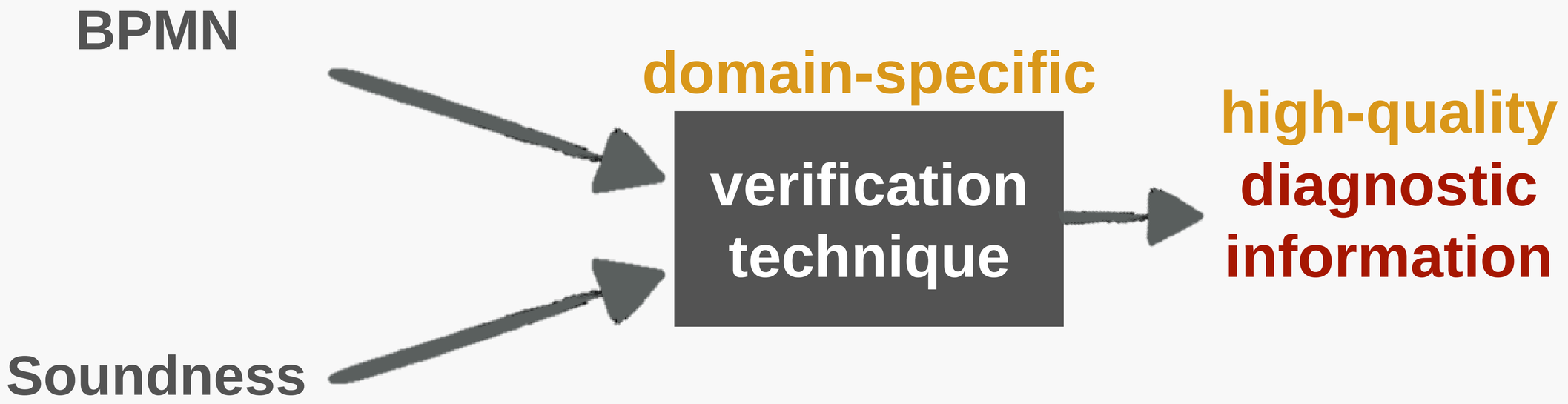
Universität Rostock
Traditio et Innovatio

TU/e
Technische Universiteit
Eindhoven
University of Technology
Where innovation starts

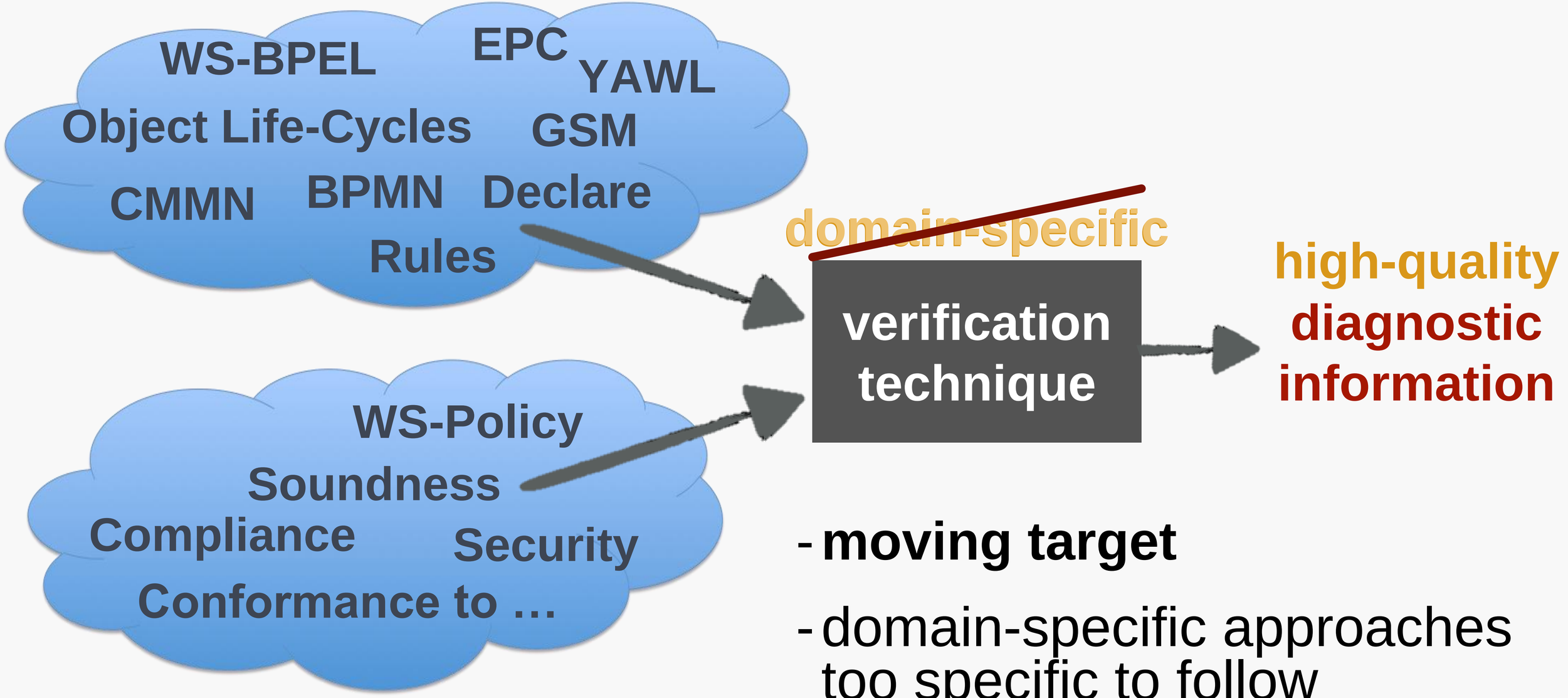
Verification of processes and services



Verification of processes and services

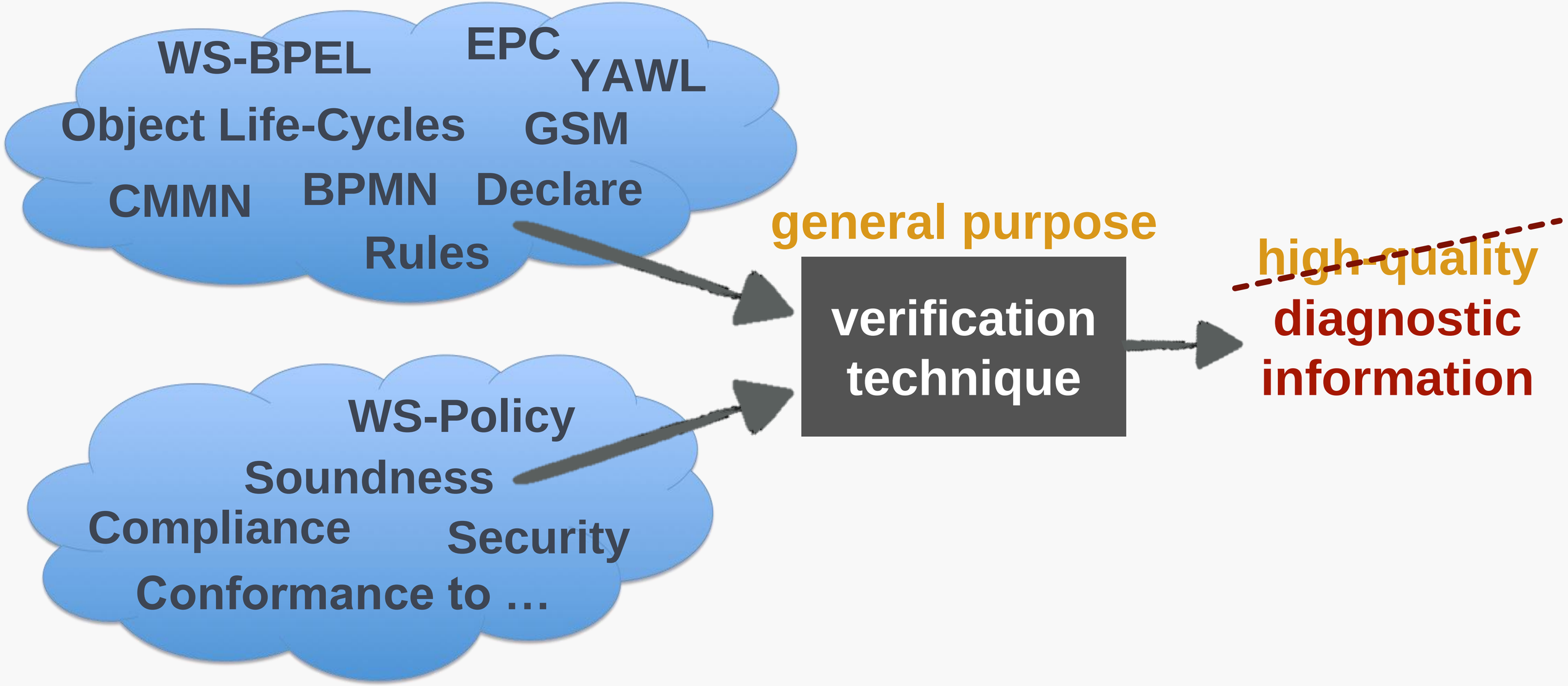


Verification of processes and services

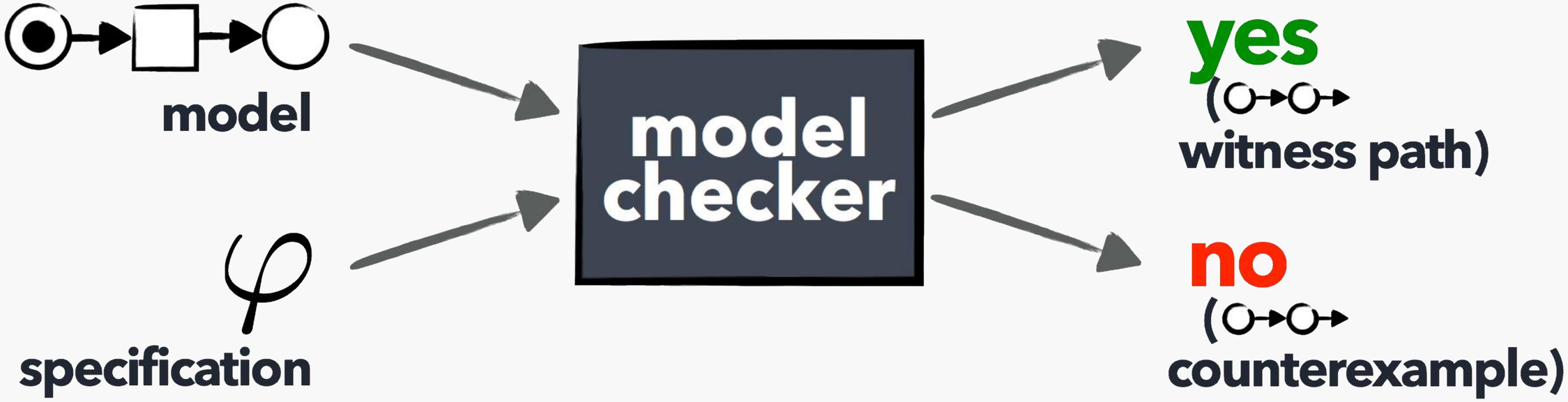


- **moving target**
- domain-specific approaches too specific to follow

Verification of processes and services



Model checking



general purpose verification approach:

1. formalize model and specification*
2. push a button

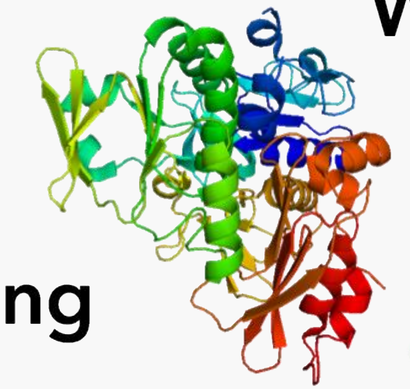
* can be hidden from the user

Effectiveness and efficiency

- model checking works in reality
- successful applications in many domains

WebSphere software

BPMN™



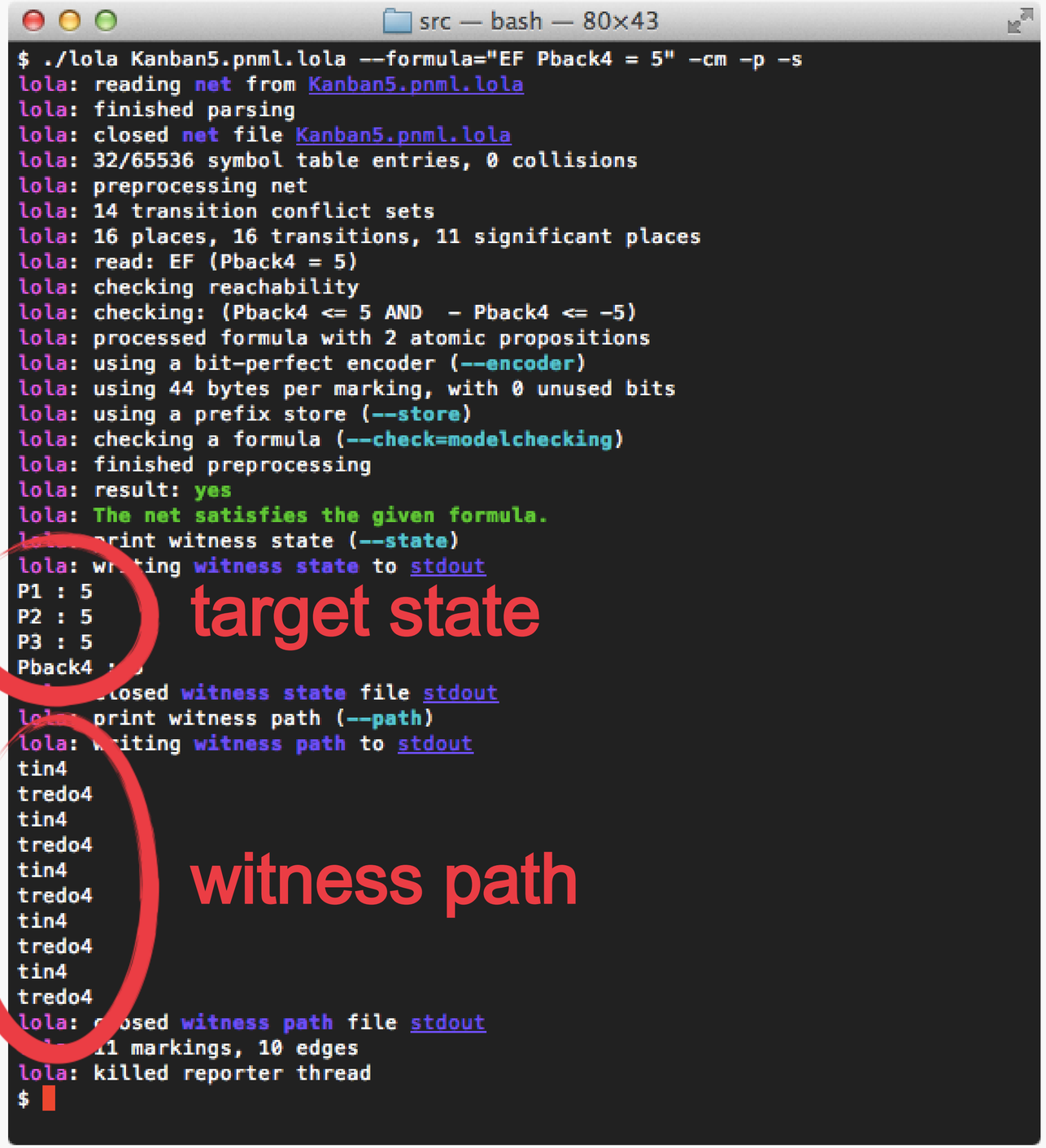
WS-BPEL

AI planning

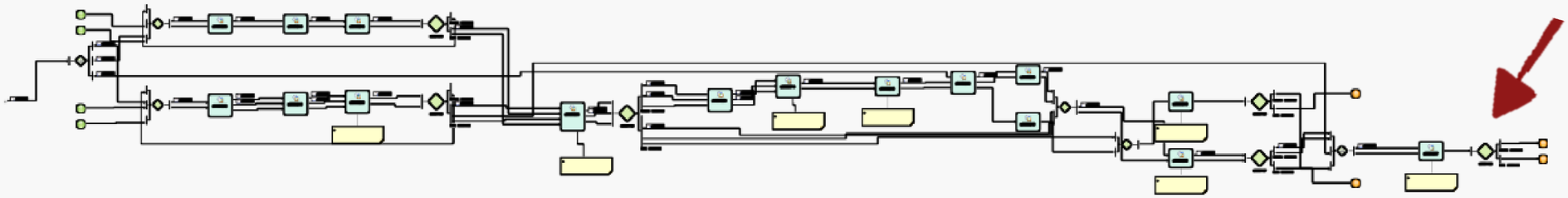
- **very fast**: “verify while you model”

Diagnosis

- in case of error: outputs target state and produce a **witness path**
- describes how target state can be reached
- operational semantics: can be **simulated**

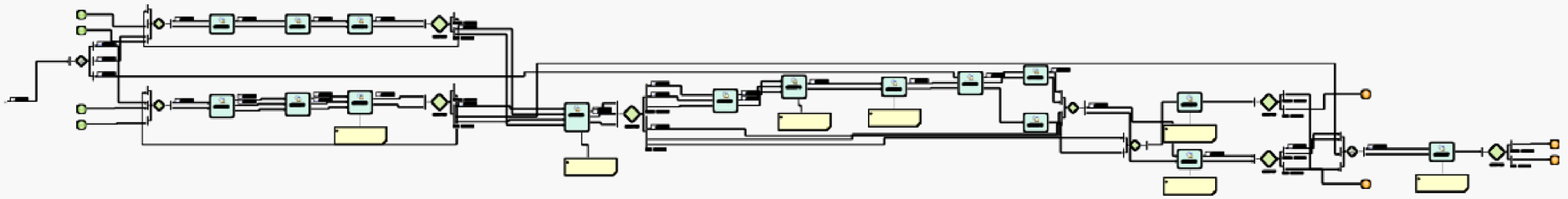


Diagnosis: the bad



- paths can become **very long**
- length correlates with size of the model
- reports all events equally: disregarding **importance**

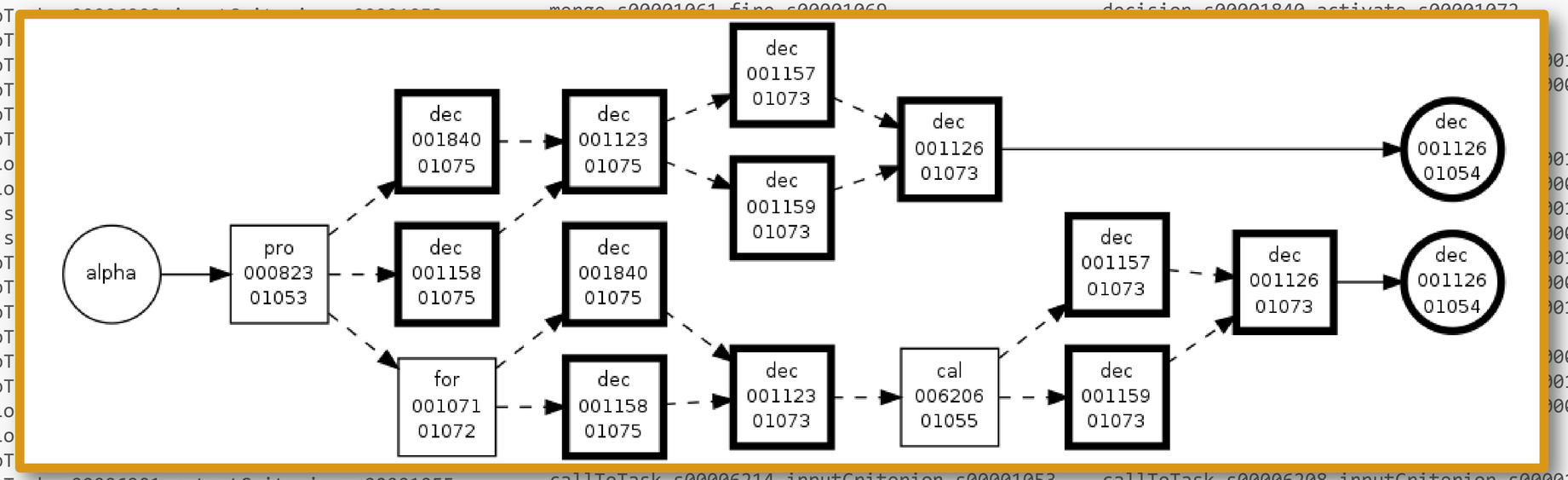
This talk: better diagnosis



PATH

process.s00000823##s00006200.inputCriterion.s00001053
fork.s00001071.activate.s00001072
fork.s00001071.fire.s00001078
merge.s00001061.activate.s00001065
merge.s00001061.fire.s00001069
callToT
callToT
callToT
callToT
callToT
callToT
decisio
decisio
merge.s
merge.s
callToT
callToT
callToT
callToT
callToT
callToT
callToT
decisio
decisio
callToT
callToTask.s00006201.outputCriterion.s00001055
decision.s00001123.activate.s00001072
decision.s00001123.fire.s00001075
merge.s00001161.activate.s00001064
merge.s00001161.fire.s00001069
callToTask.s00006208.inputCriterion.s00001053
callToTask.s00006208.outputCriterion.s00001055

decision.s00001157.activate.s00001072
decision.s00001157.fire.s00001073
fork.s00001071.fire.s00001073
merge.s00001061.activate.s00001064
join.s00001163.activate.s00001062



callToTask.s00006214.inputCriterion.s00001053
callToTask.s00006202.inputCriterion.s00001053
callToTask.s00006202.outputCriterion.s00001055
callToTask.s00006211.inputCriterion.s00001053
callToTask.s00006211.outputCriterion.s00001055
callToTask.s00006209.inputCriterion.s00001053
callToTask.s00006209.outputCriterion.s00001055

decision.s00001158.activate.s00001072
decision.s00001158.fire.s00001075
callToTask.s00006214.outputCriterion.s00001055
callToTask.s00006213.inputCriterion.s00001053
callToTask.s00006213.outputCriterion.s00001055

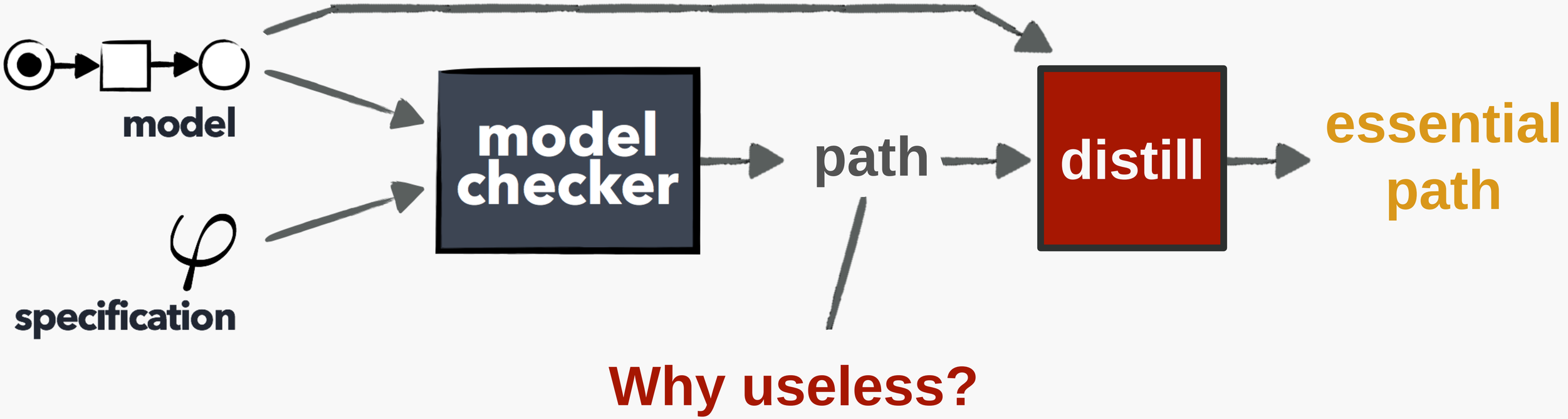
callToTask.s00006208.inputCriterion.s00001053
callToTask.s00006208.outputCriterion.s00001055
decision.s00001157.activate.s00001072
decision.s00001157.fire.s00001073
join.s00001163.activate.s00001062
callToTask.s00006212.inputCriterion.s00001053
callToTask.s00006212.outputCriterion.s00001055

merge.s00001162.activate.s00001064
merge.s00001162.fire.s00001069
callToTask.s00006210.inputCriterion.s000
callToTask.s00006210.outputCriterion.s00
decision.s00001159.activate.s00001072
decision.s00001159.fire.s00001073
join.s00001163.activate.s00001065
join.s00001163.fire.s00001069
callToTask.s00006207.inputCriterion.s000
callToTask.s00006207.outputCriterion.s00
decision.s00001126.activate.s00001072
decision.s00001126.fire.s00001073

STATE

decision.s00001126.output.s00001054 : 2

This talk: better diagnosis



Reasons for useless paths

detours

depth-first search

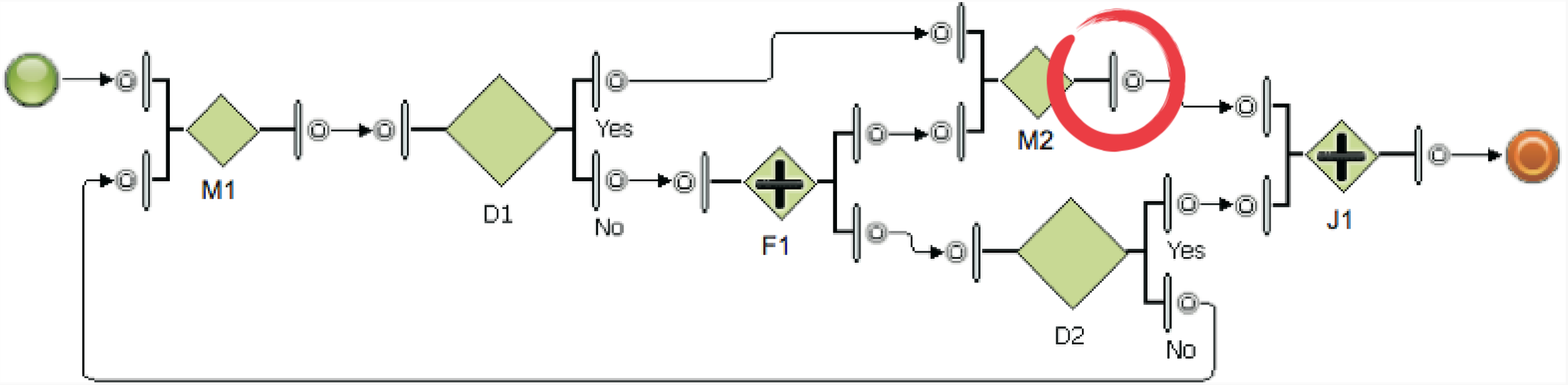
indisputable parts

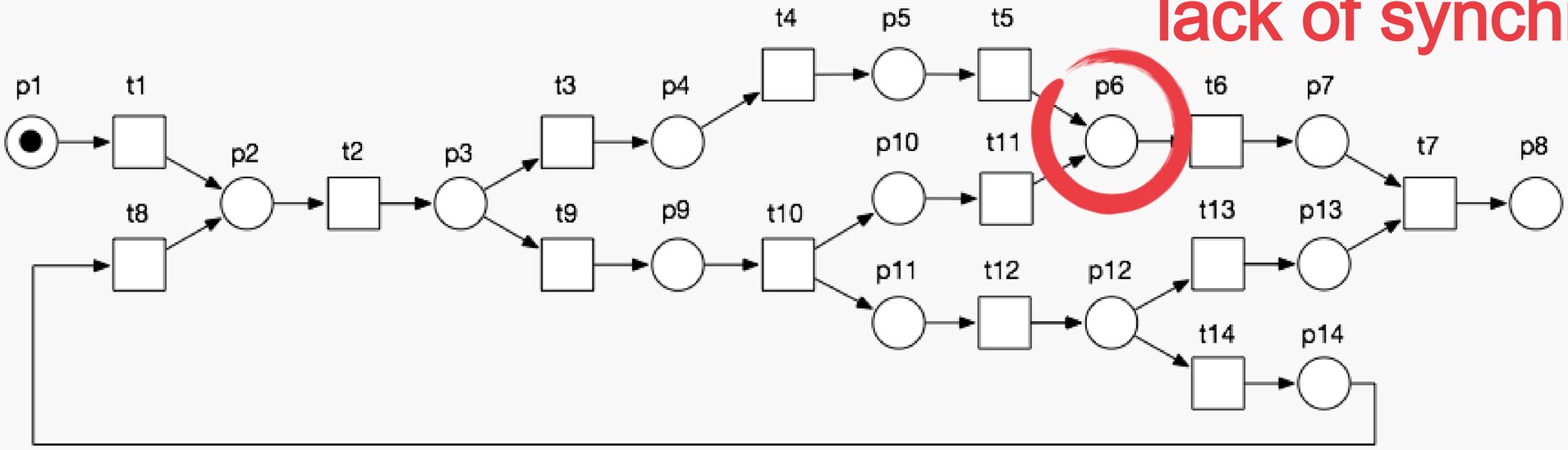
bootstrapping

interleavings

concurrency

Running example

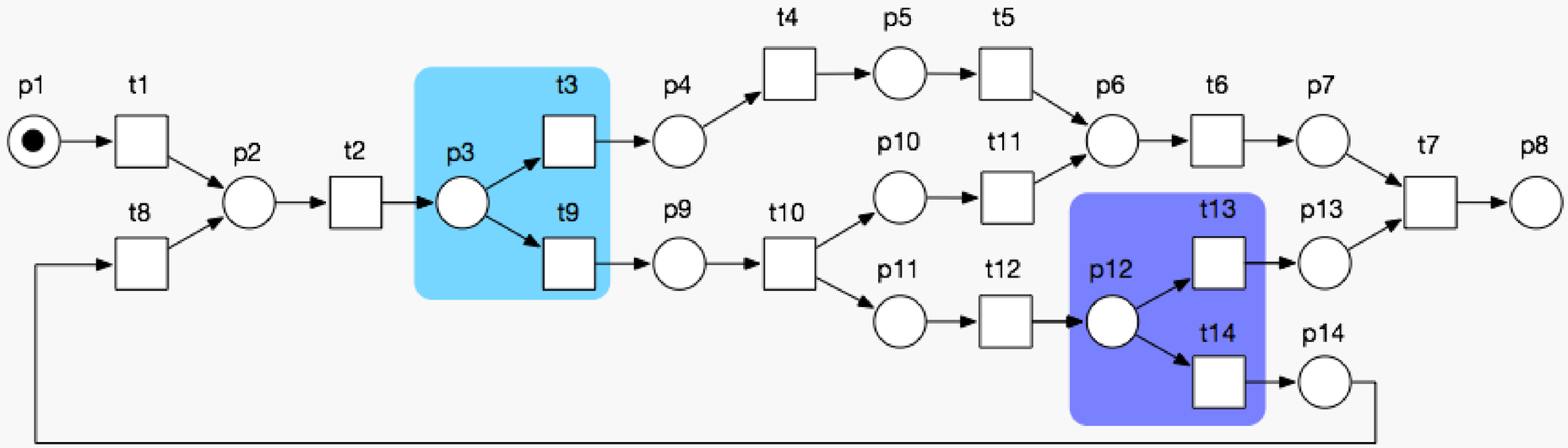




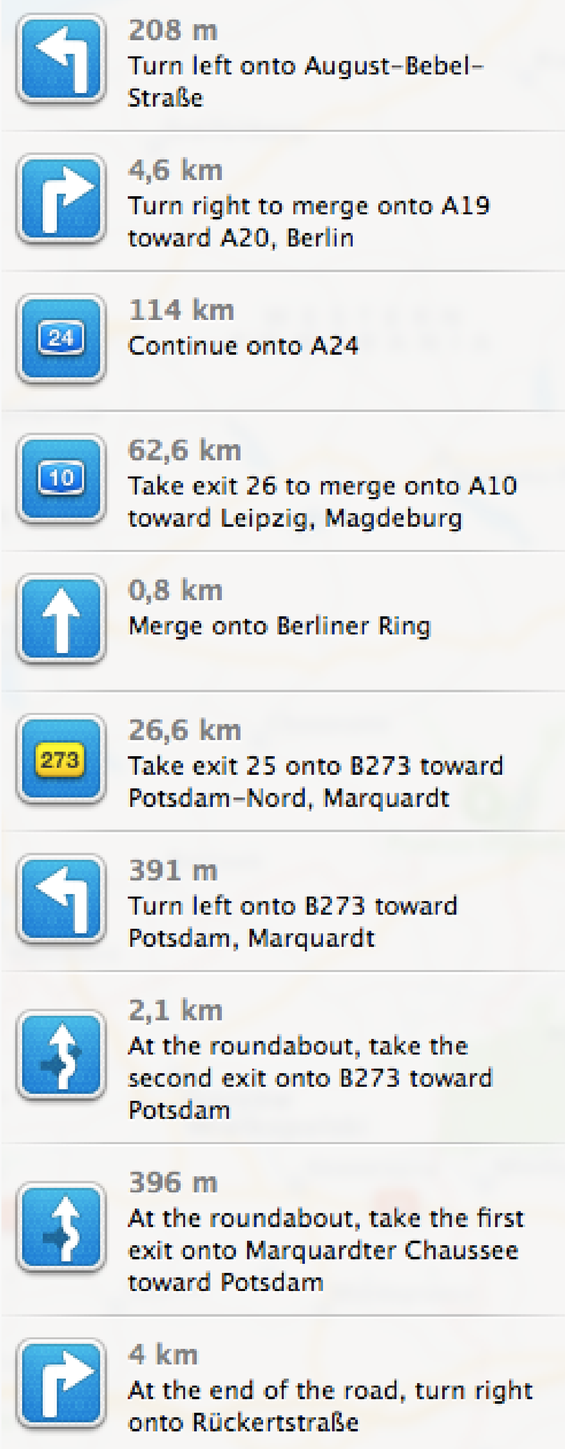
lack of synchronization

Reduction: obvious parts

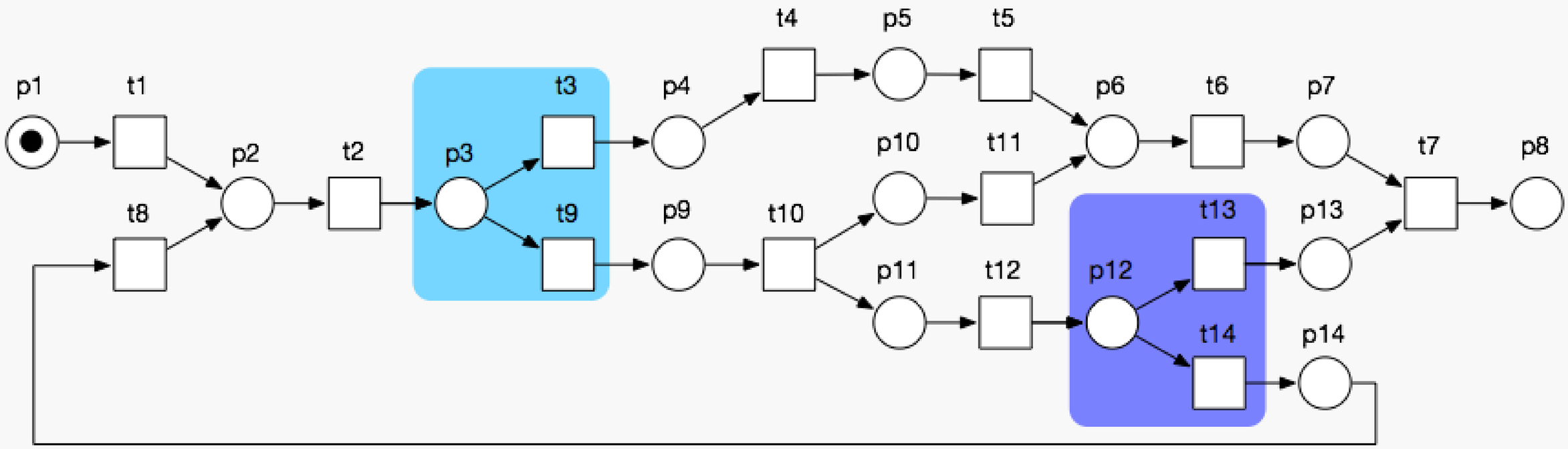
- classify transitions
- only report points of **alternative** continuations*

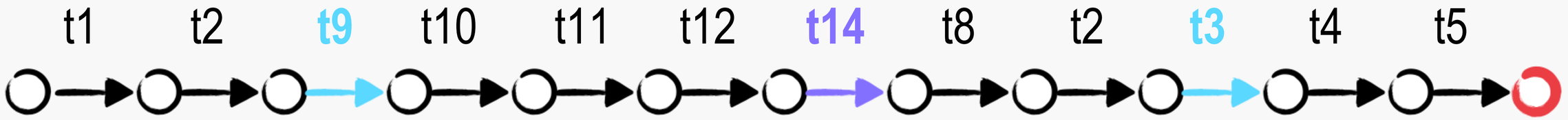


* XOR-gateways, events, exceptions, …
  assume **progress** of flow



Reduction: obvious parts





“down” “down” “up”

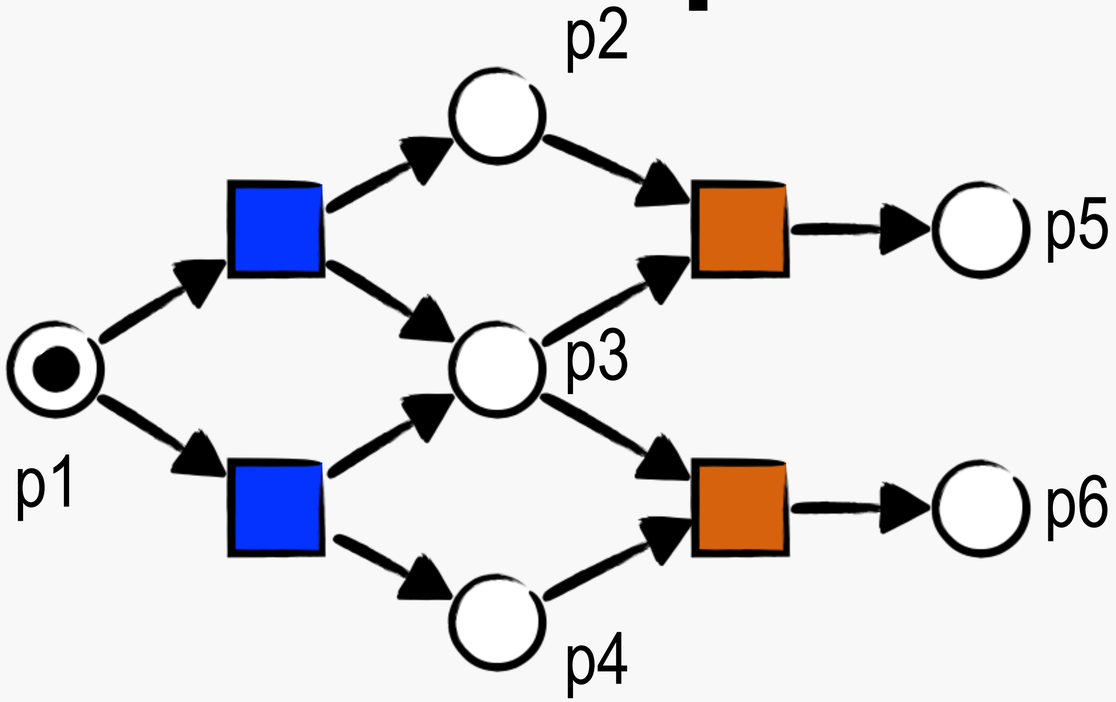
Non-obvious “core” of a path ≈ 10-25%

library	A	B1	B2	B3	C
avg. path length before / after	17.51 / 1.83	17.52 / 2.11	16.06 / 1.54	20.34 / 1.67	13.40 / 2.30
max. path length before / after	53 / 8	66 / 7	56 / 6	54 / 5	21 / 3
sum of path lengths before / after	1699 / 178	1419 / 171	1349 / 129	1688 / 139	134 / 23
reduction	89.52 %	87.95 %	90.44 %	91.77 %	82.84 %

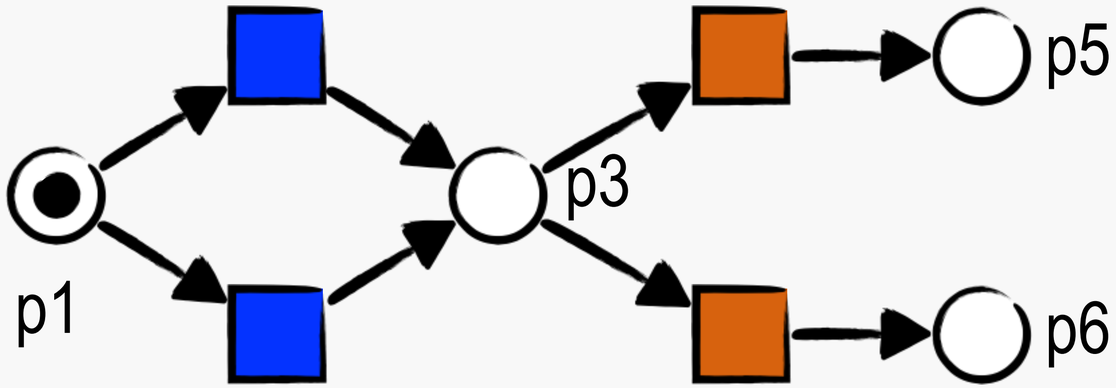
library	A	B1	B2	B3	C
avg. path length before / after	30.83 / 3.17	10.47 / 0.66	12.16 / 0.68	11.50 / 0.59	51.00 / 7.57
max. path length before / after	89 / 13	52 / 7	100 / 8	103 / 14	120 / 17
sum of path lengths before / after	1079 / 111	1047 / 66	1459 / 82	1507 / 77	357 / 53
reduction	89.71 %	93.70 %	94.38 %	94.89 %	85.15 %

library	A	B1	B2	B3	C
avg. path length before / after	12.06 / 2.79	13.82 / 2.55	18.13 / 2.33	14.27 / 2.55	11.27 / 2.33
max. path length before / after	44 / 7	70 / 7	95 / 7	95 / 7	27 / 3
sum of path lengths before / after	19699 / 4557	5707 / 1054	13835 / 1777	17494 / 3130	169 / 35
reduction	76.87 %	81.53 %	87.16 %	82.11 %	79.29 %

Reduction: spurious decisions



genuine decision



spurious decision
= irrelevant for outcome

- can be found by **model checking**
- results: **50%-80% spurious**, occasionally no reduction (timeout)

Reasons for useless paths

detours

depth-first search

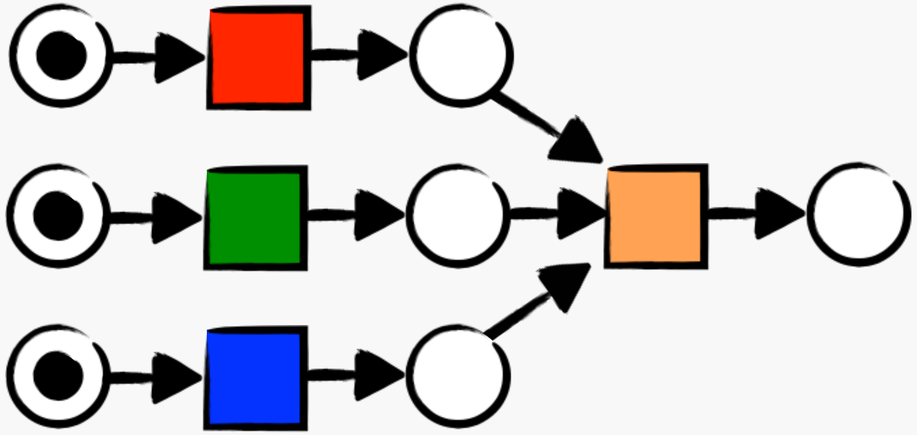
indisputable parts

bootstrapping

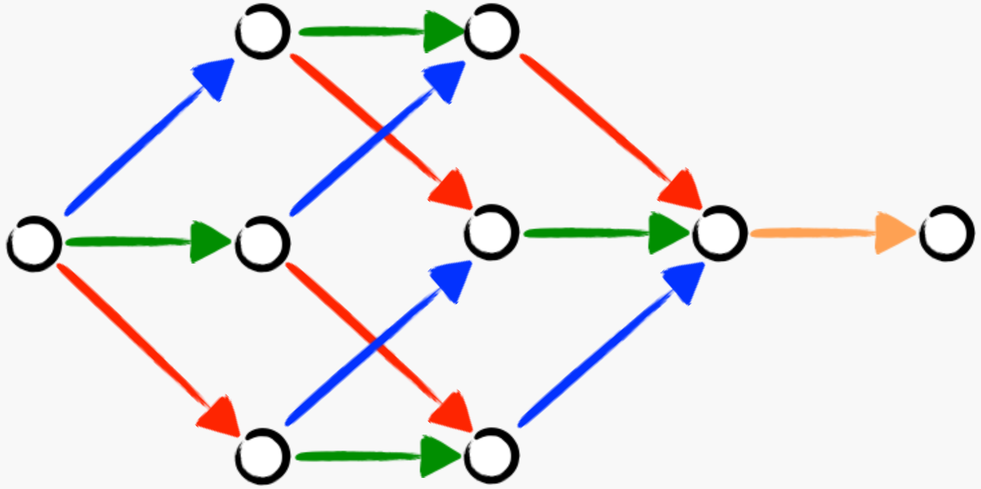
interleavings

concurrency

Reduction: unorder steps



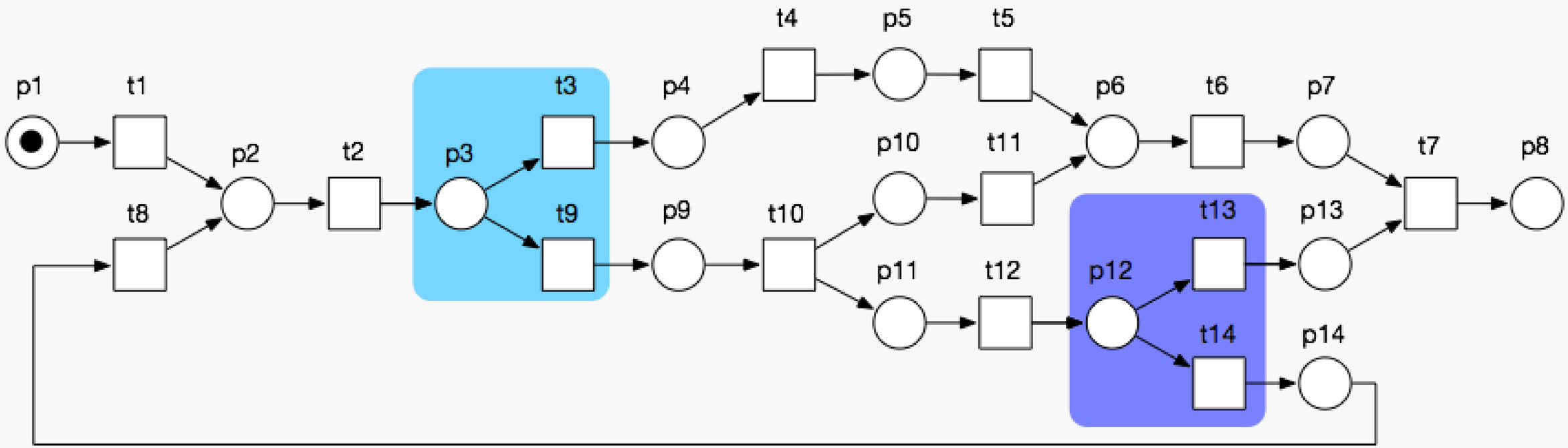
independent steps

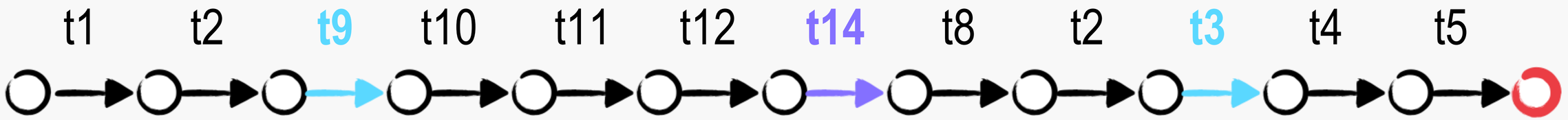


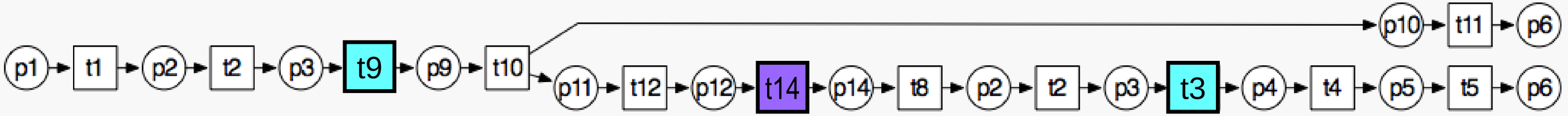
many paths to same goal state
order of steps **irrelevant**

- idea: **show independence of steps**
  (→ partially ordered runs)
- makes synchronization points (milestones) explicit

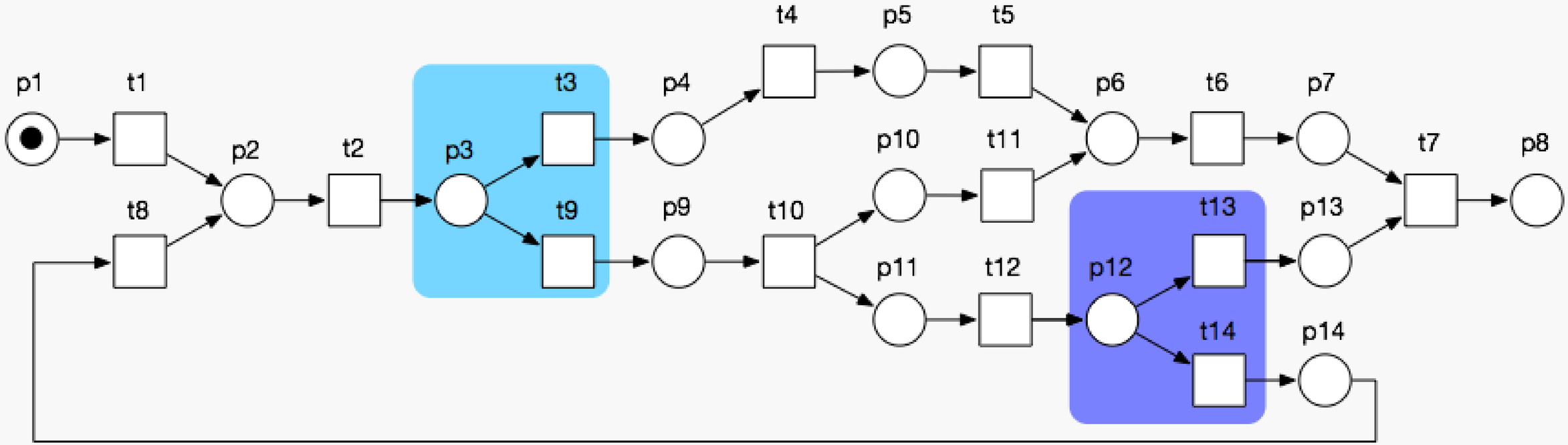
Reduction: unorder steps

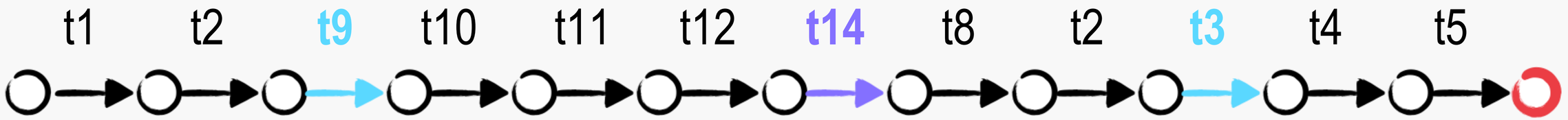


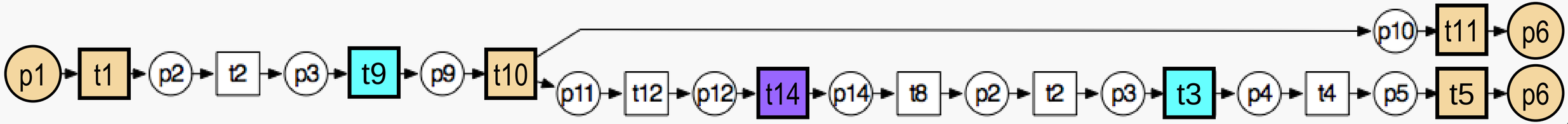




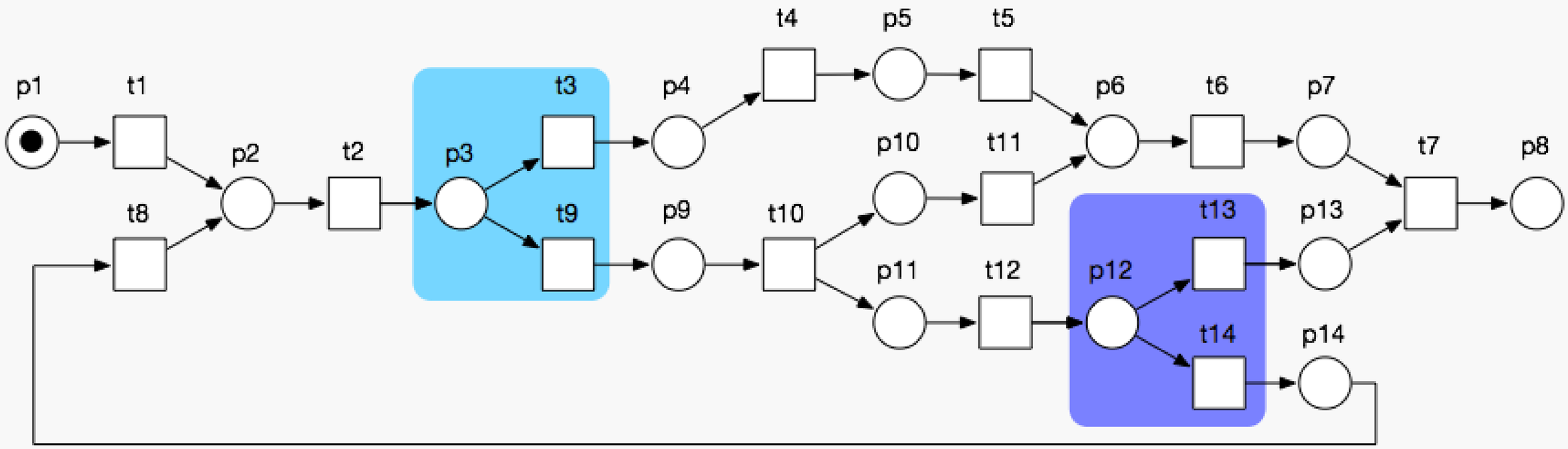
More aid: preserve reference points

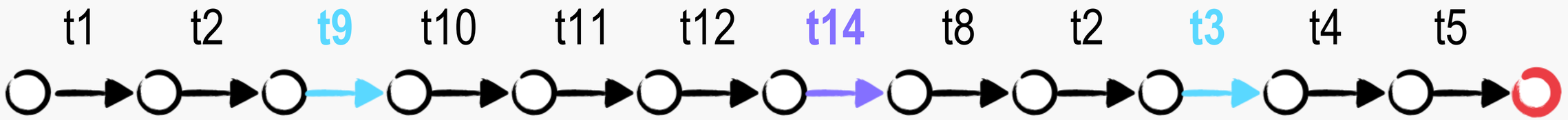


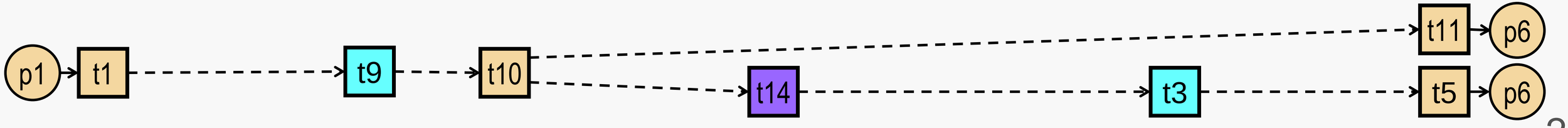




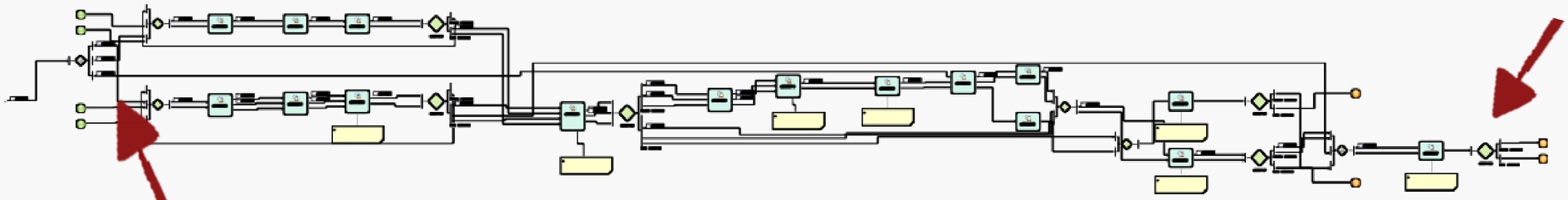
Final: remove obvious/spurious parts



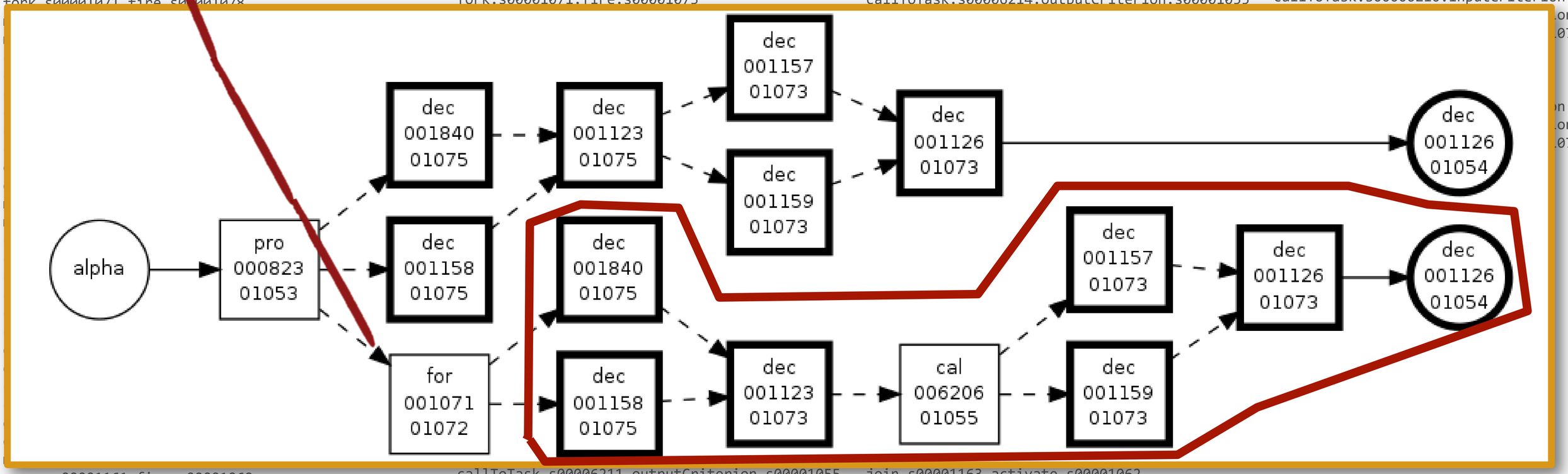




Essential path: find source of error



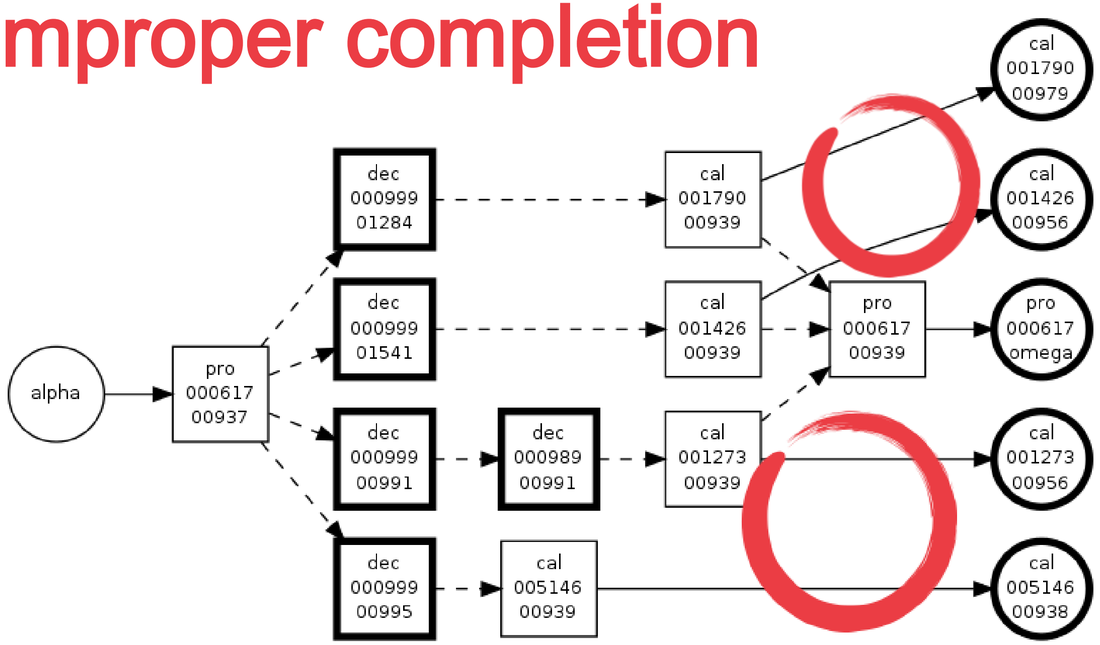
PATH
process.s00000823##s[illegible]0006200.inputCriterion.s00001053
fork.s00001071.activate.s00001072
fork.s00001071.fire.s00001078
decision.s00001157.activate.s00001072
decision.s00001157.fire.s00001073
fork.s00001071.fire.s00001073
decision.s00001158.activate.s00001072
decision.s00001158.fire.s00001075
callToTask.s00006214.outputCriterion.s00001055
merge.s00001162.activate.s00001064
merge.s00001162.fire.s00001069
callToTask.s00006210.inputCriterion.s000
on.s00
072



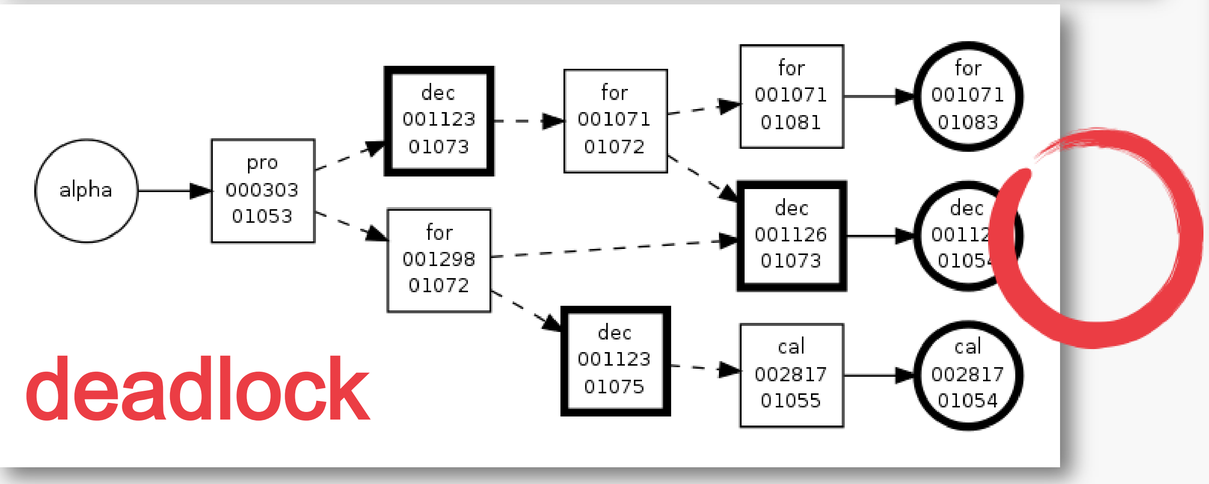
merge.s00001161.fire.s00001069
callToTask.s00006208.inputCriterion.s00001053
callToTask.s00006208.outputCriterion.s00001055
callToTask.s00006211.outputCriterion.s00001055
callToTask.s00006209.inputCriterion.s00001053
callToTask.s00006209.outputCriterion.s00001055
join.s00001163.activate.s00001062
callToTask.s00006212.inputCriterion.s00001053
callToTask.s00006212.outputCriterion.s00001055

Results: typical reduced paths

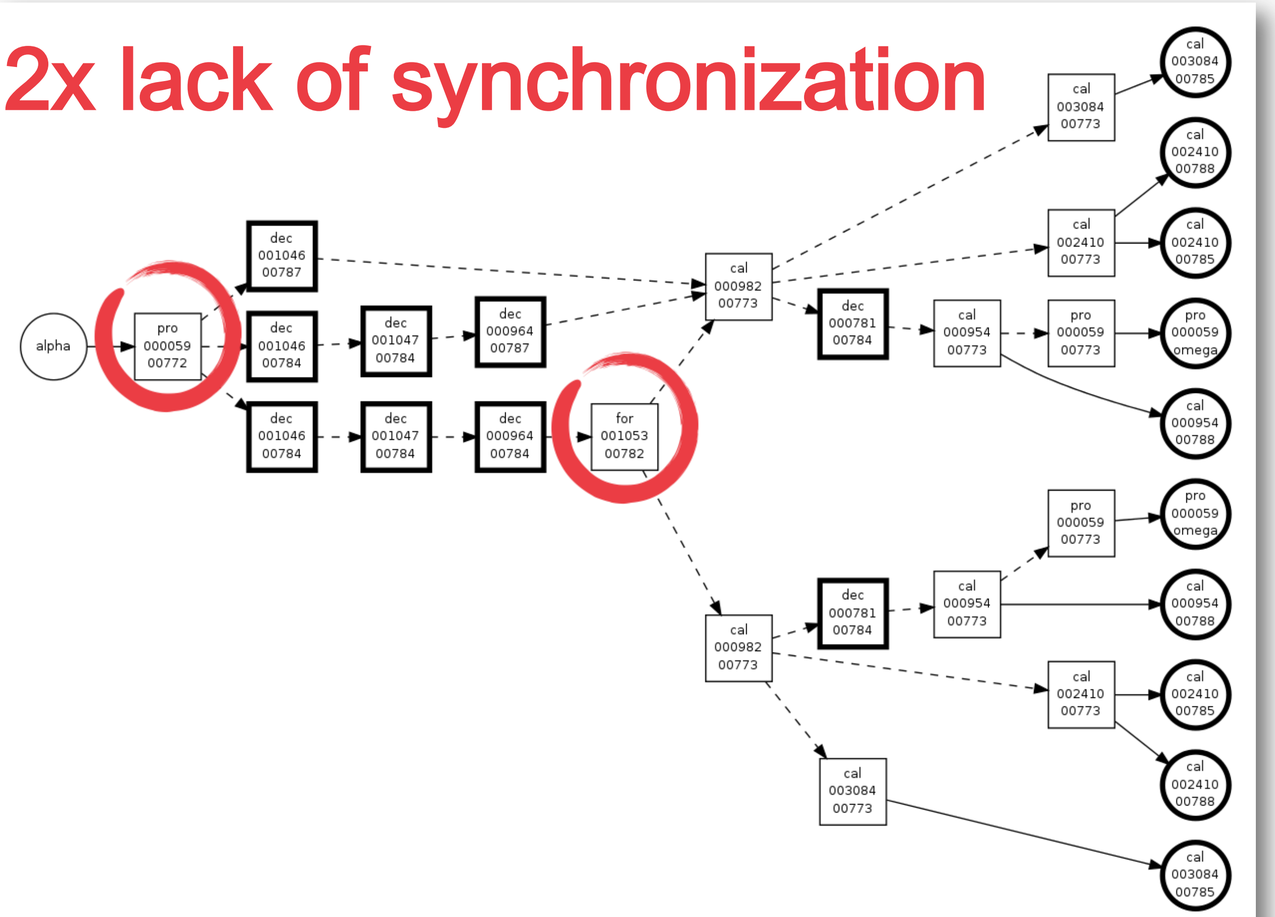
improper completion



deadlock



2x lack of synchronization

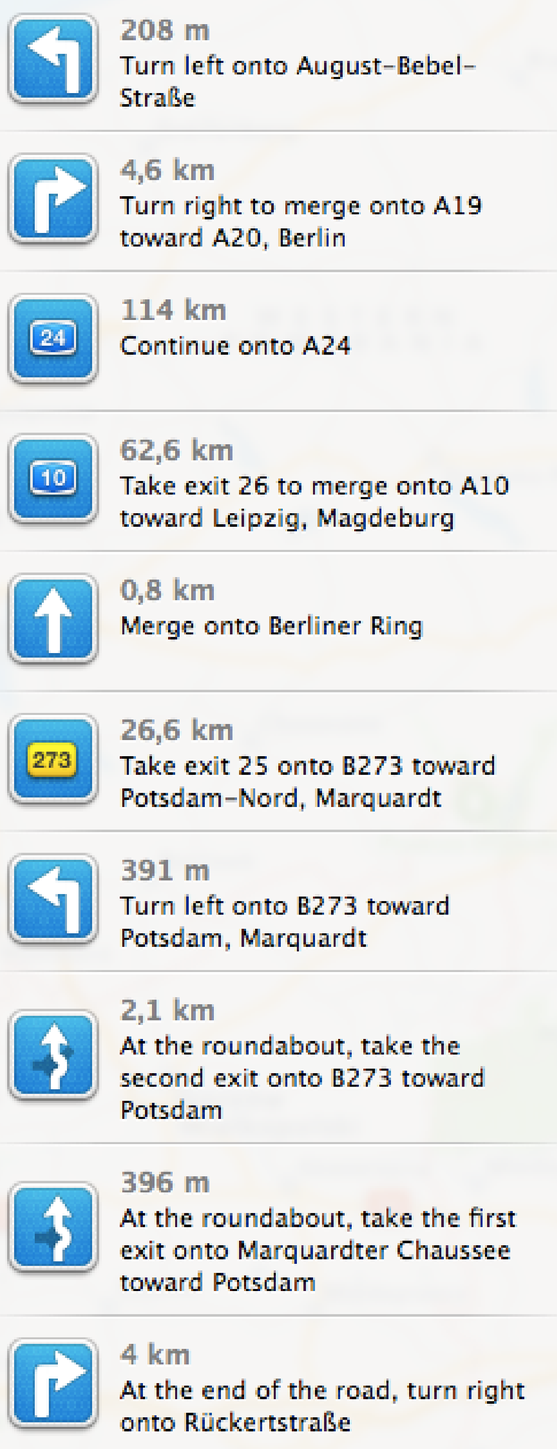


Summary

- general purpose verification **more user friendly**
- paths → **partial order** of important **decisions**
- applicable to any verification goal
- keep **reference points** to aid diagnosis

Next steps

- error localization vs. explanation
- detect useless cycles
- How should a good diagnosis for $problem look like?



Where did I go wrong?
Explaining errors in process models

Niels Lohmann

Universität Rostock
Traditio et Innovatio