Benchmark on Reliability of Complex Discrete Systems

Emergency Power Supply of a Nuclear Power Plant

Analysis with a PyCATSHOO C++ knowledge base

The Teraflop++ knowledge base for PyCATSHOO
History

- PyCATSHOO : tool to manage hybrid complex systems
  - Hassane Chraibi : Dynamic reliability modeling and assessment with PyCATSHOO PSAM 2013
  - www.pycatshoo.org
- Knowledge base dedicated to electrical systems
- Written in C++
- Derived from another PyCATSHOO knowledge base (Teraflop) written in Python
- Teraflop was itself derived from a FIGARO knowledge base
- Simulation model more realistic than the BDMP ➔ new assumptions

The Teraflop++ knowledge base
Additional assumptions

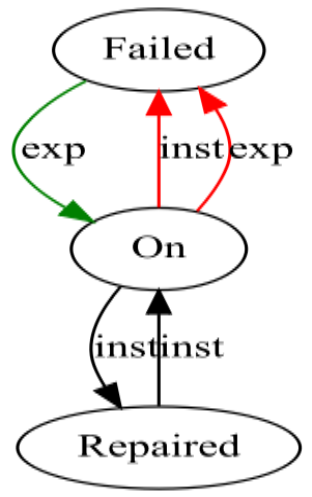
- Circuit breakers open in response to a short-circuit with a delay proportional to the distance to the source of the short-circuit
- When a line is no longer powered, its circuit breakers open
- Priorities between trains are implemented by the response time of the circuit breakers, and the power sources
- Batteries are instantly recharged each time they stop providing energy and there is power in one of the adjacent trains
- LHA and LHB bus bars are not powered during transition periods. Thus we have added a delay of 0.02 hours before the undesirable event is triggered

Flow propagation

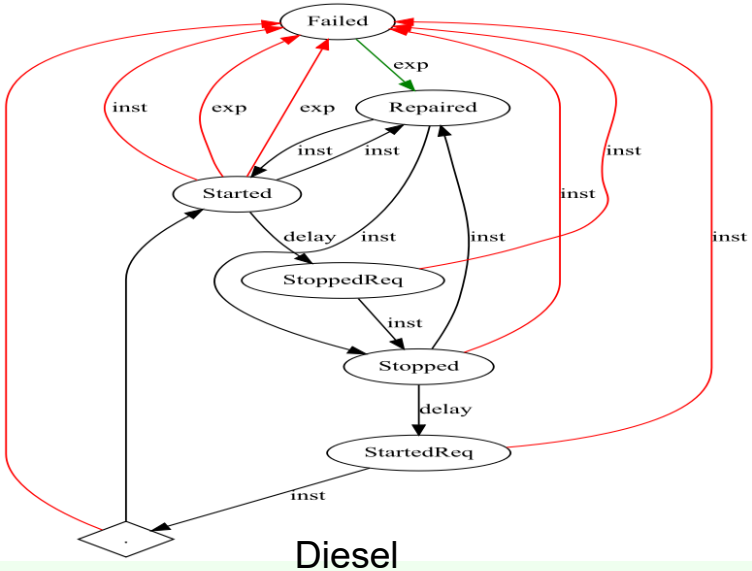
- Two flows are propagated:
  - Electrical flow:
    - Only downstream
    - No distinction between different voltages
  - Short-circuit flow:
    - Bi directional
- Global computation after each event:
  - Complete reset
  - Electrical flow propagation, starting from sources:
    - Downstream only
    - Stops in open, failed or shorted components
  - Short-circuit flow propagation:
    - Upstream and downstream, starting from the short-circuits
    - Stops in open components
    - The "distance" to a short-circuit is the number of components on the path to the short-circuit

Components behavior
3 kinds of state diagrams

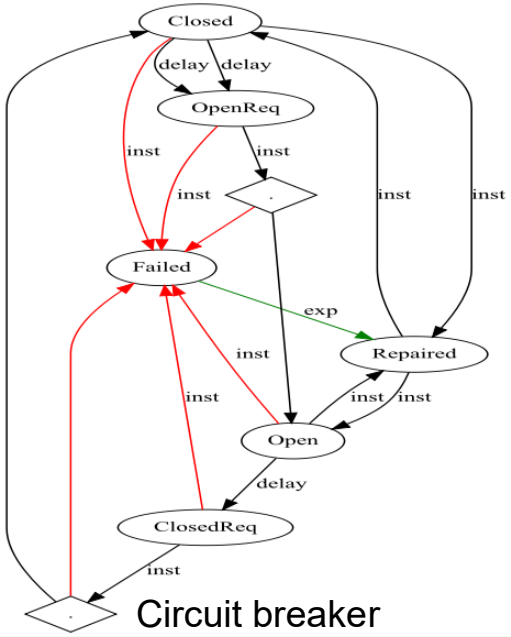
- Mechanical diagrams:



Simple

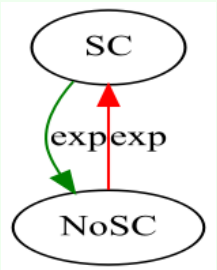


Diesel

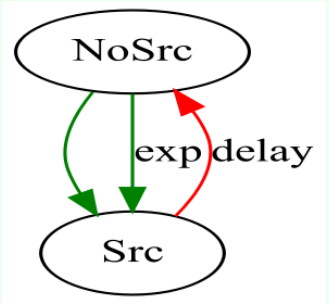


Circuit breaker

- Short-circuit diagram:



- Source diagram:



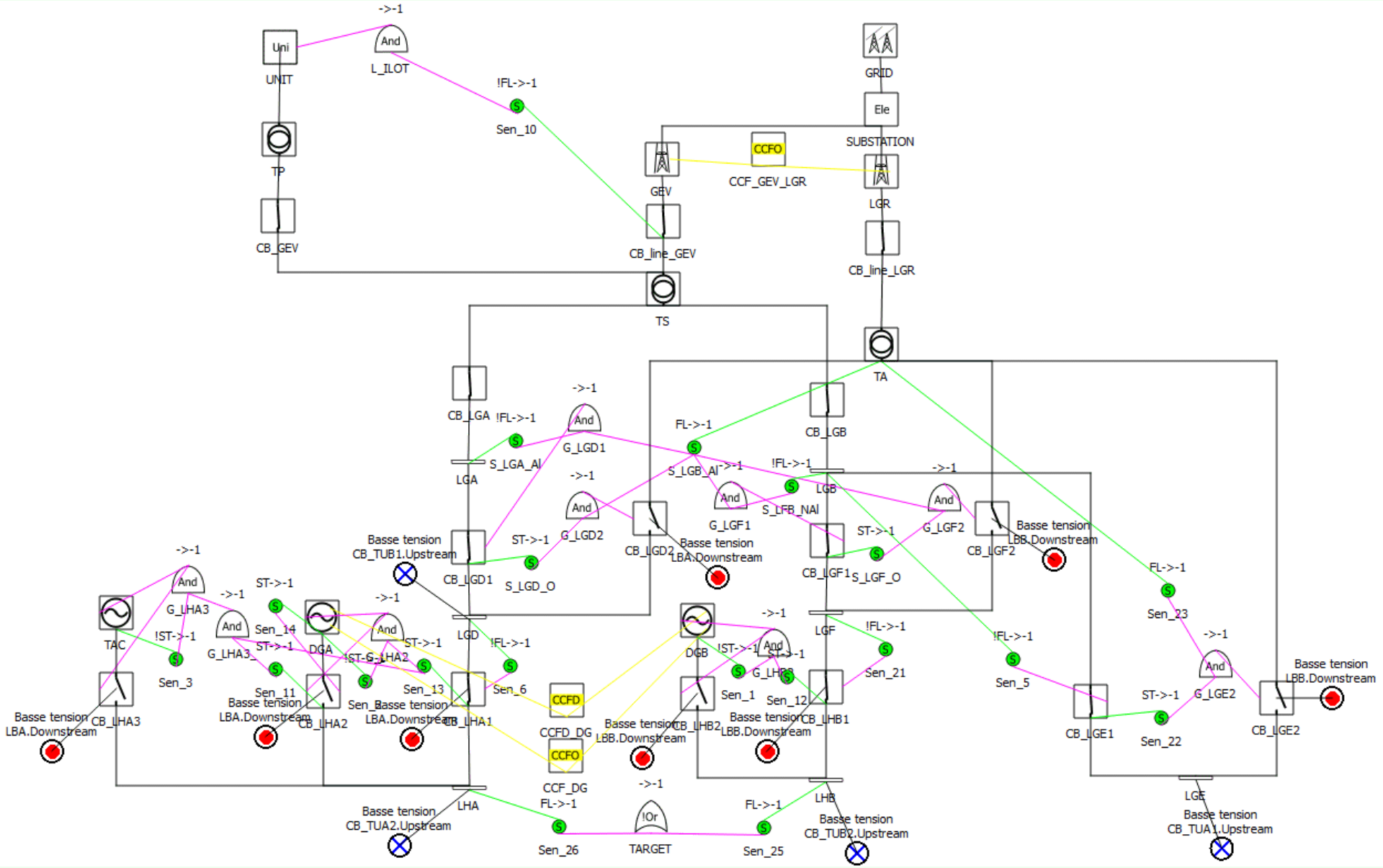
Instrumentation and control modeling

- Sensors:
  - Can detect:
    - The existence of power downstream the tested component,
    - The existence of a short-circuit downstream or upstream,
    - The state of the mechanical state diagram the tested component is in.
  - Emit a Boolean signal
  - Output signal can be negated 0←→1
- Logic gates:
  - N input signals from sensors or other logic gates
  - OR, AND or K/N logic gates
  - Output signal can be negated 0←→1
  - Can present a response time
- Active components (circuit breakers, diesel generators, batteries..):
  - Message box for start (closing) signal
  - Message box for stop (opening) signal
  - Optional default mechanical state (resumed when no signal received)

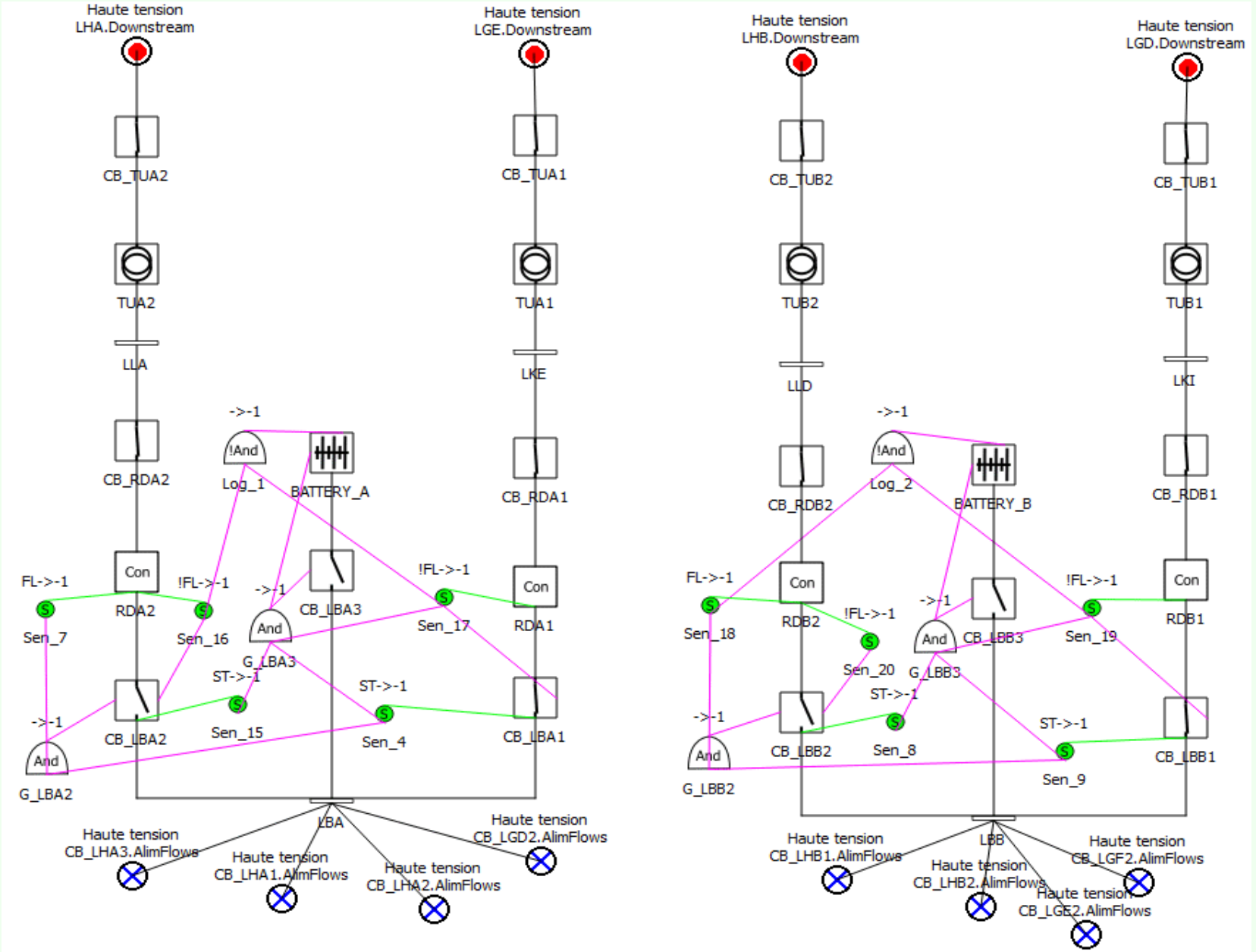
Common cause failures

- CCF in operation: 2 options:
  - Only if all components are running
  - If at least 1 component is running (option chosen here)
- CCF on demand:
  - Checked on each component demand
  - No immediate impact on unsolicited components
  - Remains faulty during a given period (we chose 1 hour)
  - While it's faulty, impossible to operate any other component
- Each component is repaired individually
  - Significant difference with the BDMP model

High voltage model



Low voltage model



Monte Carlo Analysis

- Computations performed on an Intel Core i5 3.2 GHz server running Windows 7

Nb of simulations	CPU time
10^6	24 mn
10^7	4 h

  - Unreliability at 10000h: $1.2\ 10^{-5}$
  - Asymptotic unavailability (10^6 h sequences): $2\ 10^{-7}$
- Difference with BDMP results:
  - Model complexity:
    - Low voltage circuit breakers are more often operated
  - CCF behavior:
    - Repair mode is different
- Computation time:
  - Non-Markovian model
  - Sequences much longer than BDMP sequences

Main sequences

- Example of sequences

- 120 sequences leading to target (on 10^7 simulations)
  - 1 sequence met 3 times
  - 6 sequences met 2 times
  - 105 sequences met once

Number of sequences	Succession of failures
3	CCF_GEV_LGR.frun
	UNIT.housefrun
	DGA.frunl
	TAC.frunl
	BATTERY_A.out
	DGB.fruns
2	CCF_GEV_LGR.frun
	UNIT.housefrun
	DGB.fruns
	BATTERY_B.out
	DGA.frunl
	TAC.frunl
2	GRID.frun
	Unit.housefdem
	CCFD_DG.fdem
	BATTERY_B.out
	TAC.frunl
2	GRID.frun
	Unit.housefdem
	CCF_DG.frun
	BATTERY_B.out
	TAC.frunl

Monte Carlo analysis

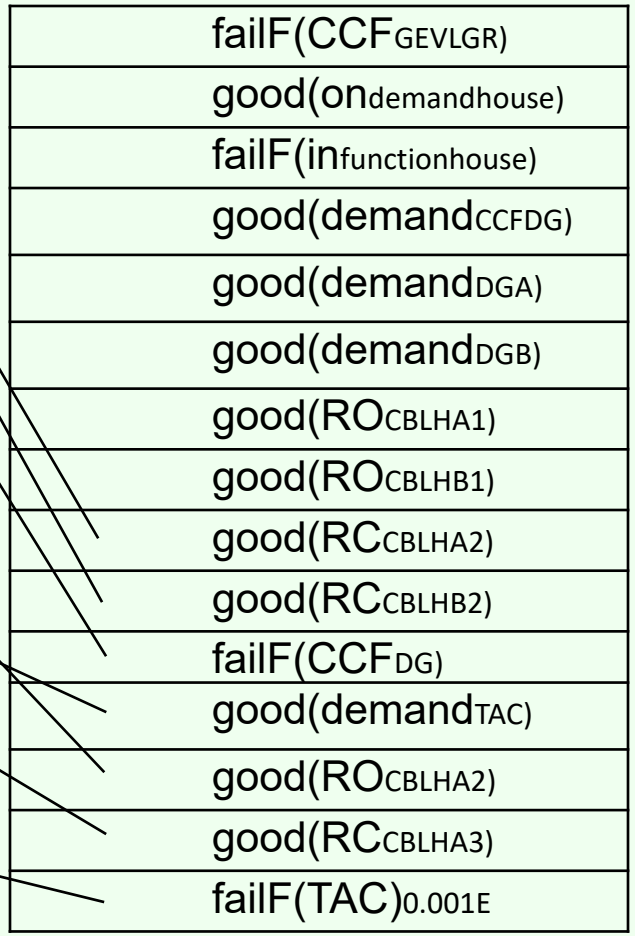
Low voltage reconfiguration

CCF_GEV_LGR.OK->NOK	NOK
GEV.PassCCF->F LGR.PassCCF->F	Failed Failed
CCF_GEV_LGR.NOK->OK	OK
UNIT.On->Ilot/F	ilot
UNIT.Ilot->F	Failed
CB_LBA1.Clos->OpenReq CB_LBB1.Clos->OpenReq	OpenReq OpenReq
CB_LBA1.OpenReq->Open/F CB_LBB1.OpenReq->Open/F	Open Open
BATTERY_A.Off->OnReq BATTERY_B.Off->OnReq	OnReq OnReq
BATTERY_A.OnReq->On/F BATTERY_B.OnReq->On/F	On On
CB_LBA3.Open->ClosReq CB_LBB3.Open->ClosReq	ClosedReq ClosedReq
CB_LBA3.ClosReq->Clos/F CB_LBB3.ClosReq->Clos/F	Closed Closed
CB_LGE1.Clos->OpenReq	OpenReq
CB_LGE1.OpenReq->Open/F	Open
CB_LHA1.Clos->OpenReq CB_LHB1.Clos->OpenReq	OpenReq OpenReq
CB_LHA1.OpenReq->Open/F CB_LHB1.OpenReq->Open/F	Open Open
DGA.Stop->StarReq DGB.Stop->StarReq	StartedReq StartedReq
CCFD_DG.NT->OK/NOK	OK
DGA.StarReq->Star/F DGB.StarReq->Star/F	Started Started

failF(CCF_{GEVLGR})
good($on_{demandhouse}$)
failF($in_{functionhouse}$)
good($demand_{CCFDG}$)
good($demand_{DGA}$)
good($demand_{DGB}$)
good(RO_{CBLHA1})
good(RO_{CBLHB1})
good(RC_{CBLHA2})
good(RC_{CBLHB2})
failF(CCF_{DG})
good($demand_{TAC}$)
good(RO_{CBLHA2})
good(RC_{CBLHA3})
failF($TAC_{0.001E}$)

Monte Carlo analysis

CB_LHA2.Open->ClosReq CB_LHB2.Open->ClosReq	ClosedReq ClosedReq
CB_LHA2.ClosReq->Clos/F CB_LHB2.ClosReq->Clos/F	Closed Closed
Low voltage reconfiguration…	
CCFD_DG.OK->NT	NotTested
CCF_DG.OK->NOK	NOK
DGA.StarCCF->F DGB.StarCCF->F	Failed Failed
Low voltage reconfiguration…	
CCF_DG.NOK->OK	OK
CB_LHA2.Clos->OpenReq CB_LHB2.Clos->OpenReq	OpenReq OpenReq
CB_LHA2.OpenReq->Open/F CB_LHB2.OpenReq->Open/F	Open Open
TAC.Stop->StarReq	StartedReq
TAC.StarReq->Star/F	Started
CB_LHA3.Open->ClosReq	ClosedReq
CB_LHA3.ClosReq->Clos/F	Closed
CB_LBA2.Open->ClosReq	ClosedReq
Low voltage reconfiguration…	
BATTERY_B.Src->NoSrc	NoSrc
TAC.frunl	Failed
Low voltage reconfiguration…	
CB_LHA3.Clos->OpenReq	OpenReq
CB_LHA3.OpenReq->Open/F	Open
L_CIBLE.OK->Req	Requested
L_CIBLE.Req->OK/Def	OK



Conclusion

- PyCATSHOO allows a modeling of the 6.6kV test case
  - Few modeling limits
  - Computation time limits

- PyCATSHOO encourages a very detailed modeling
  - Fewer assumptions to justify
  - More complex model
  - Longer computation time
  - Validation of simplifications made in other models

- PyCATSHOO offers tools for accelerating calculations
  - MPI : parallel processing
  - Importance sampling
  - Sequential Monte Carlo

} Phd dissertation Thomas Galtier