# What is a cell cycle checkpoint ? The TotemBioNet answer

Déborah Boyenval, Gilles Bernot, Hélène Collavizza, and Jean-Paul Comet

University Côte d'Azur, I3S laboratory, UMR CNRS 7271, CS 40121,
06903 Sophia Antipolis Cedex, France
{firstname.lastname}@univ-cotedazur.fr

**Abstract.** TotemBioNet is a new software platform to assist the design of qualitative regulatory network models by combining "genetically modified Hoare logic", temporal logic model checking and optimized enumeration techniques. TotemBioNet is particularly efficient to manage parameter identification, the most critical step of formal modelling. It is also remarkably flexible and efficient to check properties in order to explore biological assumptions. To illustrate this efficacy, we address the classical example of the cell cycle, where the passage from one phase to the next one, called *checkpoint*, is crucial but is usually a rather fuzzy informal concept. The cyclic behaviour of the cell cycle is specified by temporal logic and the order of individual events inside each phase is explored thanks to quantifiers introduced in Hoare logic. This way, TotemBioNet rapidly suggests a sensible formalization of the notion of checkpoint.

**Keywords:** Regulatory network · discrete modelling · parameter identification · Hoare logic · temporal logic · model-checking · cell cycle.

## 1 Formal Methods for Thomas Regulatory Networks

In the 70's, *qualitative models* based on discrete mathematics [10,17] have proved useful to understand the main causalities that govern observed phenotypes [19,18], and the multivalued framework of René Thomas and Houssine Snoussi has become a classic for biological regulatory networks. It gained new power with the introduction of formal methods in the early 2000s [4], concomitantly with [5] for signalling networks.

A qualitative model is an influence graph where the important actors for the biological question, as well as their interactions, have been inventoried on the basis of biological knowledge. Formally, the graph covers a huge set of different discrete models because the strengths of combined activations and inhibitions are unknown. They are encoded by a set of discrete *parameters* [15], which we need to identify: Any biological knowledge reduces the number of possible parameter values, by rejecting the parameterizations that do not comply. Parameter identification is the most difficult part of the modelling activity.

Simulations rapidly reach their limit, because of the non-determinism of trajectories and of a high number of parameterizations: A finite number of simulations cannot establish general properties. Formal methods solve this difficulty.

There are software platforms based on Thomas' semantics: some study the *invariants* of a given model using Petri net tools [12] and some others perform *model checking* algorithms inspired from program verification techniques [1,11]. *Temporal logics* allow to check very general properties, including those universally quantified on traces, and SMBioNet [4], based on CTL, has made possible to *exhaustively* treat *sets of models* (sets of parameterizations) by optimized enumeration algorithms. Nevertheless, to find the *exhaustive* set of possible parameterizations, enumeration asks for exponentially growing computation time w.r.t. the size and connectivity of the regulatory network.

On another note, biological experiments directly request a set of traces in the model, and the "genetically modified Hoare logic" [3] is much more effective than temporal logic for this task. Instead of commonly enumerating parameterizations as in many approaches [14], [16] and [9], it produces a set of constraints on the parameters that characterizes those in which these traces exist. The program Hoare-fol [13] efficiently computes these constraints.

`TotemBioNet` stands out by combining all these approaches in such a way that modellers converge *rapidly* toward the exhaustive set of parameterizations satisfying all *formalized* biological knowledge.

Finding the set of parameterizations compatible with current knowledge does not end the modelling activity. It remains to use the model to study the *biological question*... which, most of the time, is *not yet formalized*. Each possible explanation constitutes an hypothesis, and formalizing the latter is necessary to, at least, check its *consistency*: if the set of compatible parameterizations is empty then the hypothesis is inconsistent. `TotemBioNet`'s ability to repeatedly question the model about its diverse properties, and to obtain quick answers, allows for a fast convergence towards a formalisation of the biological question.

## 2 The platform TotemBioNet

`TotemBioNet` supports two variants of temporal logics: CTL and a dedicated *fair-path* CTL, needed for certain reachability properties in the Thomas framework. Indeed, the quantifiers $A$ and $E$ in CTL often induce artifactual results because they consider *unfair* paths that cross an infinite number of times a given state but never fire a possible transition from this state. So, universally quantified CTL formulas to study, for instance, attraction basins become unfairly false. Fair-path CTL quantifiers $A$ and $E$ simply ignore unfair paths [13]. `TotemBioNet` automatically translates fair-path CTL formulas into (more complex) CTL formulas, allowing it to benefit from usual CTL model checking algorithms.

Besides, `TotemBioNet` Hoare triples contain: a pre-condition describing the possible initial states of a given biological experiment, a path, and an observed post-condition. The path abstracts the curves obtained from experimental observations: according to thresholds setting, a threshold crossing of a variable $v$ along the curve is written $v+$ if $v$ increases, or $v-$ if $v$ decreases. When experimental conditions are not precise enough to know which variable passes its threshold first, *existential quantifiers* can express this uncertainty: $\exists(v_1+; v_2- \ , \ v_2-; v_1+)$

means that $v_1$ has increased and $v_2$ has decreased, but in an unknown order. Also, universal quantifiers permit to abstract together a collection of similar experimental observations. Genetically modified Hoare logic extends classical Hoare logic by formalising under which conditions on the parameters each $v+$ or $v-$ of the path can occur. Then, the usual *weakest pre-condition* is the constraint on the parameters that makes the abstract path possible [3].

The inputs of TotemBioNet are: an influence graph, any knowledge on the parameter values, and properties on the dynamics of the system expressed using CTL, fair-path CTL, or Hoare triples. TotemBioNet integrates *Hoare-fol* [8] and an extended version of *SMBioNet* [4]. First, *Hoare-fol* computes and simplifies the weakest pre-condition *wp* w.r.t. genetically modified Hoare logic [3]. Then, the enumeration process of TotemBioNet is based on that of *SMBioNet*, which exploits self-influences and the *Snoussi constraint* (more resources cannot reduce the expression level) to greatly reduce the enumeration complexity.

TotemBioNet enumerates all parameterizations that satisfy *wp*: *i)* if $wp \equiv False$, the enumeration process stops, *ii)* if *wp* is a conjunction of atoms of the form $(K_{v_i} \leq s_i)$ or $\neg(K_{v_i} \leq s_i)$ where $K_{v_i}$ is a parameter and $s_i$ a threshold for variable $v_i$, then the enumeration domains of $K_{v_i}$ are reduced, and *iii)* if *wp* contains disjunctions, the validity of *wp* is checked on the fly. This considerably reduces the search space of all possible parameterizations and TotemBioNet generates, for each remaining parameterization, one input file for the model checker *NuSMV* [6]. This file contains the conjunction of temporal formulas and an automaton which encodes the state transition graph for the current parameterization. TotemBioNet also offers *environment variables* used to freeze some variables according to an experimental environment (by the way, it also reduces the number of parameters).

TotemBioNet, see `https://gitlab.com/totembionet/totembionet`, comes with many examples that illustrate the combination of CTL properties, fair-path CTL properties and Hoare triples. TotemBioNet allows one to describe the influence graph with *yEd* graph editor (`https://www.yworks.com/products/yed`). A typical session consists in building the influence graph using *yEd*, in automatically generating the corresponding text file and then in adding temporal properties and Hoare triples in concrete syntax. TotemBioNet generates an output file (possibly in `csv` format) which contains all parameterizations, labeled with "OK" when the dynamic properties are verified, and if not with all the properties which are not satisfied. The global TotemBioNet process is illustrated in Figure 1.

## 3 TotemBioNet Use Case: a Simplified Cell Cycle Model

The cell cycle is a series of events leading to correct duplication of DNA of a cell (synthesis or S phase) and its division into two genetically identical daugther cells (mitosis or M phase). Gap phases G1 and G2 lie respectively before S and M. Progression through the cell cycle is driven by Cyclins/Cyclin-dependant kinases complexes (Cyc/Cdks) and their inhibitors. A 5-variables cell-cycle model has been designed in [2] where the variables $sk$, $a$ and $b$ are the main Cyc/Cdks

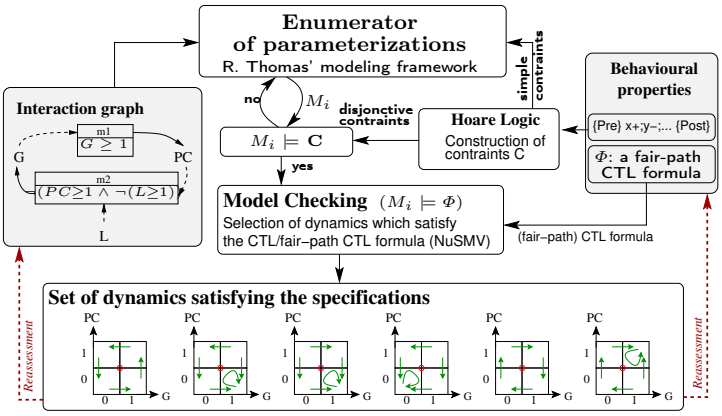



Fig. 1: `TotemBioNet`' processing flow

involved in the mammalian cell cycle and *en* and *ep* their inhibitors. The interaction graph and its variables are detailed in Appendix A. Moreover, the succession of phases, G1, S, G2 and M, has been described in [2] *via* the Hoare triple $\mathbf{H_{init}} : \left\{ G1_{init} \right\} \begin{array}{c} sk+;\, sk+;\, en-; \\ a+;\, sk-;\, sk-;\, b+; \\ a-;\, ep+; \\ en+;\, b-;\, ep-; \end{array} \left\{ G1_{init} \right\}$ where the pre- and post-condition $G1_{init}$ specifies the state $sk = 0, ep = 0, a = 0, b = 0, en = 1$, G1 is the blue subsequence, S the red one, G2 the dark gray one and M the green one.

Our main question is: *Is this small model powerful enough to represent checkpoints?* First, notice that the model assumes constant infusion of growth factors and consequently its dynamics must *always* be cyclic. The ability of `TotemBioNet` to mix several formal approaches allows us to combine $H_{init}$ and this property specified using fair-path CTL: $\varphi_{cyclic} \equiv G1_{init} \Rightarrow AX(AF(G1_{init}))$ (where $AXAF$ means a strict future). `TotemBioNet` results are synthetized in the first line of Table 1: from the 100800 parameterizations satisfying Snoussi constraint, 676 of them satisfy the weakest precondition calculated from $H_{init}$. Then 609 out of the 676 also validate $\varphi_{cyclic}$. Notice the great efficiency of Hoare Logic: The use of the equivalent CTL formula $\varphi_{init}$ (see Appendix B) instead of the $H_{init}$ Hoare triple in the first experiment of the table would drastically increase the computation time: from 6.1s to 18.5min for the same result.

Another question to understand checkpoint is the order of transitions inside phases. [2] suggests that some transitions inside a phase may admit permutations: all transitions except the first one for G1 and S, all transitions for G2 and none for M. $H_{forall}$ encodes the 12 possible paths owing to the *Forall* quantifier,

$$\mathbf{H_{forall}} : \left\{ G1_{init} \right\} \begin{array}{l} sk+; \quad Forall((sk+; en-), (en-; sk+)); \\ a+; \quad Forall((sk-; sk-; b+), (sk-; b+; sk-), (b+; sk-; sk-)); \\ \qquad Forall((ep+; a-), (a-; ep+)); \\ \qquad en+; b-; ep-; \end{array} \left\{ G1_{init} \right\}$$

and surprisingly `TotemBioNet` returns the same 609 parameterizations!

| **Exp** | Hoare triple | $\lvert H_m \rvert$ | Temporal logic formula | $\lvert S_m \rvert$ | Computation Time (s) |
|---|---|---|---|---|---|
| **1** | $H_{init}$ | 676 | $\varphi_{cyclic}$ | 609 | 6.1 |
| **2** | $H_{forall}$ | 676 | $\varphi_{cyclic}$ | 609 | 6.1 |
| **3** | $H_{perm}$ | 0 | $\varphi_{cyclic}$ | 0 | 0.24 |
| **4** | $H_{permG1}$ | 260 | $\varphi_{cyclic}$ | 240 | 2.4 |
| **5** | $H_{permG1}$ | 260 | $\varphi_{cyclic} \wedge \varphi_{G2/M} \wedge \varphi_{G1/S}$ | **28** | 2.9 |

Table 1: Formal properties of a simplified 5-variables cell cycle model: $H_m$ is the set of models satisfying Hoare and Snoussi constraints. $S_m$ is the set of selected models after model-checking of a temporal logic formula on each element of $H_m$. (Performed on an Intel Core i7-8650U processor, 1.90GHz, 8 cores.)

This suggests that a *phase* could be simply a bag of transitions that can be performed in arbitrary order. We check this idea with the Hoare triple $H_{perm}$ (Appendix C) and `TotemBioNet` returns a unsatisfiable weakest precondition. So, let us check individually for G1, S and M. $H_{permG1}$ asks for all permutations in G1, whereas S, G2 and M are the same as in $H_{forall}$ (Appendix D): `TotemBioNet` returns 240 parameterizations. For S and for M no parameterization is selected. We conclude that the order of transitions suggested in [2] is constrained within S and M but not within G1 and G2.

Now, having a better idea of what goes on within a phase, it appears that a *checkpoint* between two phases, $p_1$ and $p_2$ should ensure that none of the possible *first* transitions of $p_2$ can be performed before one of the transitions of $p_1$. The most biologically important and the most studied checkpoints are G2/M and G1/S. They can be formalized using Hoare logic but the CTL formula is simpler if we remark that the first state of a phase is unique, whatever the order of the previous phases: $\varphi_{G2/M} \equiv \begin{pmatrix} sk = 0 \\ \wedge\ ep = 0 \\ \wedge\ a = 1 \\ \wedge\ b = 1 \\ \wedge\ en = 0 \end{pmatrix} \Rightarrow \neg \begin{pmatrix} EX(\mathbf{en} = \mathbf{1} \wedge EX(a = 0 \wedge EX(ep = 1))) \\ \vee\ EX(\mathbf{en} = \mathbf{1} \wedge EX(ep = 1 \wedge EX(a = 0))) \\ \vee\ EX(a = 0 \wedge EX(\mathbf{en} = \mathbf{1} \wedge EX(ep = 1))) \\ \vee\ EX(ep = 1 \wedge EX(\mathbf{en} = \mathbf{1} \wedge EX(a = 0))) \end{pmatrix}$.
Similarly $\varphi_{G1/S}$ is given in Appendix E, and `TotemBioNet` returns 28 parameterizations satisfying $\varphi_{G2/M}$ and $\varphi_{G1/S}$ in addition to $H_{permG1}$ and $\varphi_{cyclic}$. Thus, *checkpoints can be captured in a purely discrete framework*. The biologically less studied checkpoints, S/G2 and M/G1, have been also formalized (Appendix F). No parameterisation is selected suggesting that the current model is not detailed enough to satisfy S/G2 or M/G1 checkpoint, as defined.

## 4 Conclusion

`TotemBioNet` combines in an optimized manner Hoare logic and (different variants of) temporal logic. Two of our current works are to facilitate an incremen-

tal analysis of models, as well as to provide a more user-friendly interface with jupyter notebook as BioCHAM [7] and CoLoMoTo [11] do. `TotemBioNet` aims at offering a growing palette of formal methods to the modellers, so that each biological knowledge can be formalized according to the most suited one. Thanks to the versatility and efficacy of `TotemBioNet`, the general properties of qualitative Thomas models can be rapidly checked during their design. We showed on the small cell cycle model initially specified with a Hoare triple in [2] how the main properties of the phases can be explored, leading to a proper formalization of the notions of phase and checkpoint.

**Acknowledgements.** We are grateful to all contributors/users: M. Folschette (Hoare-fol), S. Ndèye and E. Gallésio (`antlr4` parser and installation scripts), L. Gibart (beta tests on big models). We are also indebted to A. Richard for *SMBioNet* and the constructive proof of translation from *fair-path CTL* to *CTL*. This work also benefited from fruitful collaborations and discussions with J. Behaegel and F. Delaunay.

## References

1. Batt, G., Bergamini, D., de Jong, H., Garavel, H., Mateescu, R.: Model checking genetic regulatory networks using GNA and CADP. In: Graf, S., Mounier, L. (eds.) Model Checking Software. pp. 158–163. Springer Berlin Heidelberg (2004)
2. Behaegel, J., Comet, J.P., Bernot, G., Cornillon, E., Delaunay, F.: A hybrid model of cell cycle in mammals. J. Bioinformatics Comput. Biol. **14**(1), 1640001 (2016)
3. Bernot, G., Comet, J.P., Khalis, Z., Richard, A., Roux, O.F.: A genetically modified Hoare logic. Theoretical Computer Science **765**, 145–157 (2019)
4. Bernot, G., Comet, J.P., Richard, A., Guespin, J.: Application of formal methods to biological regulatory networks: extending Thomas' asynchronous logical approach with temporal logic. J.Theor.Biol **229**(3), 339 – 347 (2004)
5. Chabrier, N., Fages, F.: Symbolic model checking of biochemical networks. In: Priami, C. (ed.) CMSB'2003. pp. 149–162. LNCS 2602, Springer-Verlag (2003)
6. Cimatti, A. *et al.*: NuSMV 2: An opensource tool for symbolic model checking. In: CAV '02. pp. 359–364 (2002)
7. Fages, F., Soliman, S.: On robustness computation and optimization in BIOCHAM-4. In: CMSB 2018. LNCS, vol. 11095 (2018)
8. Folschette, M.: The Hoare-fol tool. Tech. rep., Univ. Lille & CNRS UMR 9189 (2019), https://hal.archives-ouvertes.fr/hal-02409801
9. Guziolowski, C., Videla, S., Eduati, F., Thiele, S., Cokelaer, T., Siegel, A., Saez-Rodriguez, J.: Exhaustively characterizing feasible logic models of a signaling network using answer set programming. Bioinformatics **30** (07 2013)
10. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. J.Theor.Biol **22**(3), 437–67 (1969)
11. Naldi, A. *et al.*: The CoLoMoTo interactive notebook. Frontiers in Physiology **9**, 680 (2018)
12. Remy, E., Ruet, P., Mendoza, L., Thieffry, D., Chaouiya, C.: From logical regulatory graphs to standard Petri nets: Dynamical roles and functionality of feedback circuits. In: Transactions on Computational Systems Biology VII. pp. 56–72 (2006)

13. Richard, A.: Fair paths in CTL (2008), personnal communication, available at https://gitlab.com/totembionet/totembionet
14. Schwab, J., Kühlwein, S., Ikonomi, N., Kühl, M., Kestler, H.: Concepts in boolean network modeling: What do they all mean? Computational and Structural Biotechnology Journal **18** (03 2020)
15. Snoussi, E.: Qualitative dynamics of a piecewise-linear differential equations : a discrete mapping approach. Dynamics and stability of Systems **4**, 189–207 (1989)
16. Streck, A., Thobe, K., Siebert, H.: Comparative statistical analysis of qualitative parametrization set (09 2015)
17. Thomas, R.: Boolean formalization of genetic control circuits. J.Theor.Biol **42**(3), 563–585 (1973)
18. Thomas, R.: Logical analysis of systems comprising feedback loops. J. Theor. Biol. **73**(4), 631–56 (1978)
19. Thomas, R., Gathoye, A., Lambert, L.: A complex control circuit. Regulation of immunity in temperate bacteriophages. Eur. J. Biochem. **71**(1), 211–27 (1976)

## Appendix A: Static description of the cell cycle model

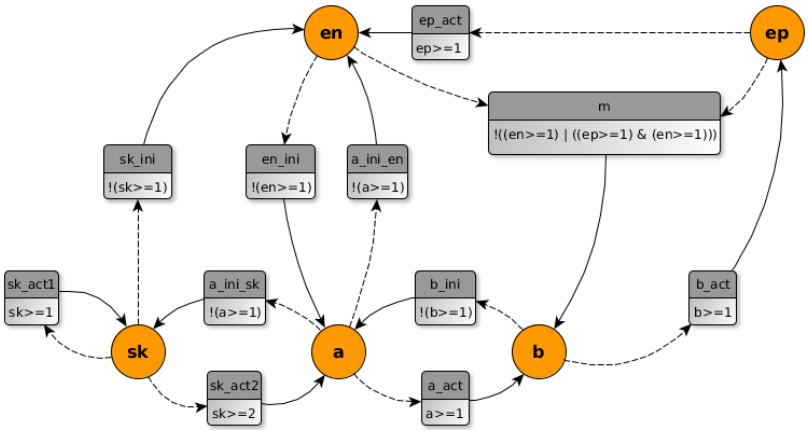



Fig. 2: **A 5-variable interaction graph of the mammalian cell cycle, from [2].** Progression through the cell cycle is driven by 2 types of genetic entities: complexes of Cyclins/Cyclin-dependant kinases (Cyc/Cdks) and their inhibitors known as *ennemies*. The 5 variables of the graph represent these entities, in orange. *sk* is the abstraction of both complexes CycE/Cdk2 and CycH/Cdk7, known as *starting kinases*. *a* and *b* respectively represent CycA/Cdk1 and CycB/Cdk1. *en* is the abstraction of the main Cyc/Cdks ennemies: the anaphase-promoting complex APC/Cdh1, cyclin-kinase inhibitors $p21$ and $p27$, and Wee1 protein. The variable *ep* is the anaphase-promoting complex APC/Cdc20, which is a Cyc/Cdks ennemy involved in mitosis exit and so-called *exit protein*. Regulations between variables are described in [2]. This interaction graph was designed using the tool *yEd* (`www.yworks.com/products/yed`).

## Appendix B: Equivalent specification of $H_{init}$ using a Fair CTL formula

In the first experiment, the cell cycle is specified by the $H_{init}$ Hoare triple. Here, the cell cycle is specified by the $\varphi_{init}$ CTL formula depicting the $H_{init}$ path.

$$\varphi_{init} \equiv \left( \begin{array}{l} (sk = 0 \wedge ep = 0 \wedge a = 0 \wedge b = 0 \wedge en = 1) -> EX(\mathbf{sk = 1} \wedge ep = 0 \wedge a = 0 \wedge b = 0 \wedge en = 1) \\ \wedge (EX(\mathbf{sk = 2} \wedge ep = 0 \wedge a = 0 \wedge b = 0 \wedge en = 1) \wedge (EX(sk = 2 \wedge ep = 0 \wedge a = 0 \wedge b = 0 \wedge \mathbf{en = 0}) \\ \wedge (EX(sk = 2 \wedge ep = 0 \wedge \mathbf{a = 1} \wedge b = 0 \wedge en = 0) \wedge (EX(\mathbf{sk = 1} \wedge ep = 0 \wedge a = 1 \wedge b = 0 \wedge en = 0) \\ \wedge (EX(\mathbf{sk = 0} \wedge ep = 0 \wedge a = 1 \wedge b = 0 \wedge en = 0) \wedge (EX(sk = 0 \wedge ep = 0 \wedge a = 1 \wedge \mathbf{b = 1} \wedge en = 0) \\ \wedge (EX(sk = 0 \wedge ep = 0 \wedge \mathbf{a = 0} \wedge b = 1 \wedge en = 0) \wedge (EX(sk = 0 \wedge \mathbf{ep = 1} \wedge a = 0 \wedge b = 1 \wedge en = 0) \\ \wedge (EX(sk = 0 \wedge ep = 1 \wedge a = 0 \wedge b = 1 \wedge \mathbf{en = 1}) \wedge (EX(sk = 0 \wedge ep = 1 \wedge a = 0 \wedge \mathbf{b = 0} \wedge en = 1) \\ \wedge (EX(sk = 0 \wedge \mathbf{ep = 0} \wedge a = 0 \wedge b = 0 \wedge en = 1)))))))))))) \end{array} \right)$$

## Appendix C: Specification of $H_{perm}$

This Hoare triple encodes a cell cycle in which phases are described by all permutations of their respective transitions. G1 is specified in blue, S in red, G2 in grey and M in green.

$$\mathbf{H_{perm}} : \left\{ G1_{init} \right\} \quad \begin{array}{c} Forall((sk+; sk+; en-), (sk+; en-; sk+), (en-; sk+; sk+)); \\ Forall((a+; sk-; sk-; b+), (a+; sk-; b+; sk-), \\ (a+; b+; sk-; sk-), (sk-; a+; sk-; b+), \\ (sk-; a+; b+; sk-), (b+; a+; sk-; sk-), \\ (sk-; sk-; a+; b+), (sk-; b+; a+; sk-), \\ (b+; sk-; a+; sk-), (sk-; sk-; b+; a+), \\ (sk-; b+; sk-; a+), (b+; sk-; sk-; a+)) \\ Forall[(ep+; a-), (a-; ep+)]; \\ Forall((en+; b-; ep-), (en+; ep-; b-), \\ (ep-; b-; en+), (ep-; en+; b-), \\ (b-; en+; ep-), (b-; ep-; en+)); \end{array} \quad \left\{ G1_{init} \right\}$$

## Appendix D: Specification of $H_{permG1}$

This Hoare triple describes the cell cycle in which G1 in addition to G2 allows all permutations of its transitions.

$$\mathbf{H_{permG1}} : \left\{ G1_{init} \right\} \quad \begin{array}{c} Forall((sk+; sk+; en-), \\ (sk+; en-; sk+), (en-; sk+; sk+)); \\ a+; \\ Forall((sk-; sk-; b+), (sk-; b+; sk-), \\ (b+; sk-; sk-)); \\ Forall((ep+; a-), (a-; ep+)); \\ en+; b-; ep-; \end{array} \quad \left\{ G1_{init} \right\}$$

## Appendix E: Specification of $\varphi_{G1/S}$ with CTL

The premise $G1_{init}$ of the formula $\varphi_{G1/S}$ is the precondition of the Hoare triple $H_{init}$ defined in [2]. It defines the initial state of G1. The first transition of S, $a+$, must not occur before any G1 transition. Thus 9 paths must not exist starting from the first G1 state encoded in premise.
The notation $EX(a = 1 \wedge EX(sk = 1 \wedge EX(en = 0 \wedge EX(sk = 2))$ is a CTL version of the Hoare path: $a+; sk+; en-; sk+$.

$$\varphi_{G1/S} \equiv \Big( G1_{init} \Big) \Rightarrow \neg \begin{pmatrix} EX(\mathbf{a{=}1} \wedge EX(sk = 1 \wedge EX(en = 0 \wedge EX(sk = 2)))) \\ \vee\ EX(sk = 1 \wedge EX(\mathbf{a{=}1} \wedge EX(en = 0 \wedge EX(sk = 2)))) \\ \vee\ EX(sk = 1 \wedge EX(en = 0 \wedge EX(\mathbf{a{=}1} \wedge EX(sk = 2)))) \\ \vee\ EX(\mathbf{a{=}1} \wedge EX(en = 0 \wedge EX(sk = 1 \wedge EX(sk = 2)))) \\ \vee\ EX(sk = 1 \wedge EX(\mathbf{a{=}1} \wedge EX(sk = 1 \wedge EX(en = 0)))) \\ \vee\ EX(sk = 1 \wedge EX(sk = 2 \wedge EX(\mathbf{a{=}1} \wedge EX(en = 0)))) \\ \vee\ EX(\mathbf{a{=}1} \wedge EX(sk = 1 \wedge EX(sk = 2 \wedge EX(en = 0)))) \\ \vee\ EX(en = 0 \wedge EX(\mathbf{a{=}1} \wedge EX(sk = 1 \wedge EX(sk = 2)))) \\ \vee\ EX(en = 0 \wedge EX(sk = 1 \wedge EX(\mathbf{a{=}1} \wedge EX(sk = 2)))) \end{pmatrix}$$

## Appendix F: Specification and checking of S/G2 and M/G1 checkpoints with CTL

The premise of $\varphi_{S/G2}$ formula (see below) encodes the first state of S. $a-$ and $ep+$ are the 2 possible first events of G2 according to $H_{permG1}$. They must not occur before completion of S events. Thus 21 paths must not exist starting from the state in premise. $\varphi_{S/G2}$ is then defined as:

$$\begin{pmatrix} sk{=}2 \\ \wedge\ ep{=}0 \\ \wedge\ a{=}0 \\ \wedge\ b{=}0 \\ \wedge\ en{=}0 \end{pmatrix} \Rightarrow \neg \begin{pmatrix} EX(a{=}1 \wedge EX(sk{=}1 \wedge EX(sk{=}0 \wedge EX(\mathbf{a{=}0} \wedge EX(b{=}1 \wedge EX(ep{=}1)))))) \\ \vee\ EX(a{=}1 \wedge EX(sk{=}1 \wedge EX(\mathbf{a{=}0} \wedge EX(sk{=}0 \wedge EX(b{=}1 \wedge EX(ep{=}1)))))) \\ \vee\ EX(a{=}1 \wedge EX(\mathbf{a{=}0} \wedge EX(sk{=}1 \wedge EX(sk{=}0 \wedge EX(b{=}1 \wedge EX(ep{=}1)))))) \\ \vee\ EX(a{=}1 \wedge EX(b{=}1 \wedge EX(sk{=}1 \wedge EX(\mathbf{a{=}0} \wedge EX(sk{=}0 \wedge EX(ep{=}1)))))) \\ \vee\ EX(a{=}1 \wedge EX(b{=}1 \wedge EX(\mathbf{a{=}0} \wedge EX(sk{=}1 \wedge EX(sk{=}0 \wedge EX(ep{=}1)))))) \\ \vee\ EX(a{=}1 \wedge EX(\mathbf{a{=}0} \wedge EX(b{=}1 \wedge EX(sk{=}1 \wedge EX(sk{=}0 \wedge EX(ep{=}1)))))) \\ \vee\ EX(a{=}1 \wedge EX(sk{=}1 \wedge EX(b{=}1 \wedge EX(\mathbf{a{=}0} \wedge EX(sk{=}0 \wedge EX(ep{=}1)))))) \\ \vee\ EX(a{=}1 \wedge EX(sk{=}1 \wedge EX(\mathbf{a{=}0} \wedge EX(b{=}1 \wedge EX(sk{=}0 \wedge EX(ep{=}1)))))) \\ \vee\ EX(a{=}1 \wedge EX(\mathbf{a{=}0} \wedge EX(sk{=}1 \wedge EX(b{=}1 \wedge EX(sk{=}0 \wedge EX(ep{=}1)))))) \\ \vee\ EX(a{=}1 \wedge EX(sk{=}1 \wedge EX(sk{=}0 \wedge EX(\mathbf{ep{=}1} \wedge EX(b{=}1 \wedge EX(a{=}0)))))) \\ \vee\ EX(a{=}1 \wedge EX(sk{=}1 \wedge EX(\mathbf{ep{=}1} \wedge EX(sk{=}0 \wedge EX(b{=}1 \wedge EX(a{=}0)))))) \\ \vee\ EX(a{=}1 \wedge EX(\mathbf{ep{=}1} \wedge EX(sk{=}1 \wedge EX(sk{=}0 \wedge EX(b{=}1 \wedge EX(a{=}0)))))) \\ \vee\ EX(\mathbf{ep{=}1} \wedge EX(a{=}1 \wedge EX(sk{=}1 \wedge EX(sk{=}0 \wedge EX(b{=}1 \wedge EX(a{=}0)))))) \\ \vee\ EX(a{=}1 \wedge EX(b{=}1 \wedge EX(sk{=}1 \wedge EX(\mathbf{ep{=}1} \wedge EX(sk{=}0 \wedge EX(a{=}0)))))) \\ \vee\ EX(a{=}1 \wedge EX(b{=}1 \wedge EX(\mathbf{ep{=}1} \wedge EX(sk{=}1 \wedge EX(sk{=}0 \wedge EX(a{=}0)))))) \\ \vee\ EX(a{=}1 \wedge EX(\mathbf{ep{=}1} \wedge EX(b{=}1 \wedge EX(sk{=}1 \wedge EX(sk{=}0 \wedge EX(a{=}0)))))) \\ \vee\ EX(\mathbf{ep{=}1} \wedge EX(a{=}1 \wedge EX(b{=}1 \wedge EX(sk{=}1 \wedge EX(sk{=}0 \wedge EX(a{=}0)))))) \\ \vee\ EX(a{=}1 \wedge EX(sk{=}1 \wedge EX(b{=}1 \wedge EX(\mathbf{ep{=}1} \wedge EX(sk{=}0 \wedge EX(a{=}0)))))) \\ \vee\ EX(a{=}1 \wedge EX(sk{=}1 \wedge EX(\mathbf{ep{=}1} \wedge EX(b{=}1 \wedge EX(sk{=}0 \wedge EX(a{=}0)))))) \\ \vee\ EX(a{=}1 \wedge EX(\mathbf{ep{=}1} \wedge EX(sk{=}1 \wedge EX(b{=}1 \wedge EX(sk{=}0 \wedge EX(a{=}0)))))) \\ \vee\ EX(\mathbf{ep{=}1} \wedge EX(a{=}1 \wedge EX(sk{=}1 \wedge EX(b{=}1 \wedge EX(sk{=}0 \wedge EX(a{=}0)))))) \end{pmatrix}$$

Similarly, the premise of $\varphi_{M/G1}$ formula (see below) encodes the first state of M. $sk+$ and $en-$ are the 2 possible first events of G1 according to $H_{permG1}$. Thus the 8 paths enabling these events to occur before completion of M events must not exist, starting from the state in premise. $\varphi_{M/G1}$ is then defined as:

$$\begin{pmatrix} sk{=}0 \\ \wedge\ ep{=}1 \\ \wedge\ a{=}0 \\ \wedge\ b{=}1 \\ \wedge\ en{=}0 \end{pmatrix} \Rightarrow \neg \begin{pmatrix} EX(en{=}1 \wedge EX(b{=}0 \wedge EX(\mathbf{sk{=}1} \wedge EX(ep{=}0 \wedge EX(sk{=}2 \wedge EX(en{=}0)))))) \\ \vee\ EX(en{=}1 \wedge EX(\mathbf{sk{=}1} \wedge EX(b{=}0 \wedge EX(ep{=}0 \wedge EX(sk{=}2 \wedge EX(en{=}0)))))) \\ \vee\ EX(\mathbf{sk{=}1} \wedge EX(en{=}1 \wedge EX(b{=}0 \wedge EX(ep{=}0 \wedge EX(sk{=}2 \wedge EX(en{=}0)))))) \\ \vee\ EX(en{=}1 \wedge EX(b{=}0 \wedge EX(\mathbf{sk{=}1} \wedge EX(ep{=}0 \wedge EX(en{=}0 \wedge EX(sk{=}2)))))) \\ \vee\ EX(en{=}1 \wedge EX(\mathbf{sk{=}1} \wedge EX(b{=}0 \wedge EX(ep{=}0 \wedge EX(en{=}0 \wedge EX(sk{=}2)))))) \\ \vee\ EX(\mathbf{sk{=}1} \wedge EX(en{=}1 \wedge EX(b{=}0 \wedge EX(ep{=}0 \wedge EX(en{=}0 \wedge EX(sk{=}2)))))) \\ \vee\ EX(en{=}1 \wedge EX(b{=}0 \wedge EX(\mathbf{en{=}0} \wedge EX(ep{=}0 \wedge EX(sk{=}1 \wedge EX(sk{=}2)))))) \\ \vee\ EX(en{=}1 \wedge EX(\mathbf{en{=}0} \wedge EX(b{=}0 \wedge EX(ep{=}0 \wedge EX(sk{=}1 \wedge EX(sk{=}2)))))) \end{pmatrix}$$

`TotemBioNet` extracts no model (Table 2) for each of these checkpoints, from which we conclude that the model is not precise enough to capture them.

| **Exp** | Hoare triple | $\lvert H_m \rvert$ | Temporal logic formula | $\lvert S_m \rvert$ | Computation Time (s) |
|---|---|---|---|---|---|
| **6** | $H_{permG1}$ | 260 | $\varphi_{cyclic} \wedge \varphi_{G2/M} \wedge \varphi_{G1/S} \wedge \varphi_{S/G2}$ | 0 | 3.5 |
| **7** | $H_{permG1}$ | 260 | $\varphi_{cyclic} \wedge \varphi_{G2/M} \wedge \varphi_{G1/S} \wedge \varphi_{M/G1}$ | 0 | 3.2 |

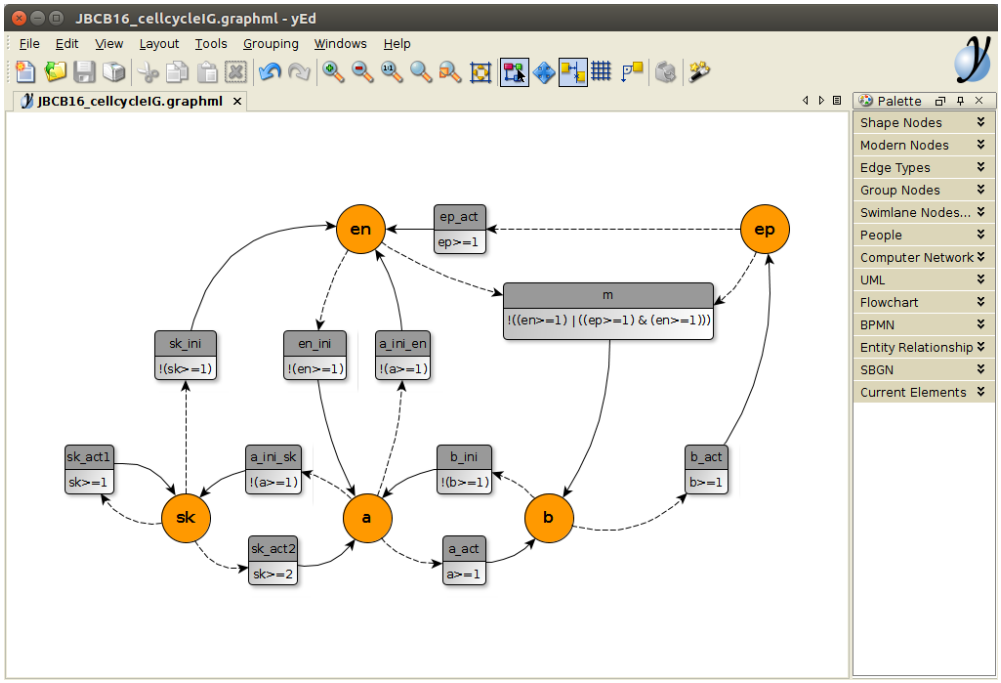
Table 2: **Verification of S/G2 and M/G1 checkpoints.** $H_m$ is the set of models satisfying Hoare and Snoussi constraints. $S_m$ is the set of selected models after model-checking of a temporal logic formula on each element of $H_m$.

# A step-by-step user guide of `TotemBioNet` applied to the tool paper's use case

This scenario is based on the $5^{th}$ experiment of use case, tool paper's table 1. `TotemBioNet` and its user manual are available at `https://gitlab.com/totembionet/totembionet`. Associated benchmarks are available at `http://www.i3s.unice.fr/~comet/DOCUMENTS/BenchTotemBioNet.zip`.

## Step 1: Build the interaction graph with yEd

The yEd graph editor can be downloaded at : `https://www.yworks.com/products/yed`.




JBCB16_cellcycleIG.graphml

## Step 2: Generate static part of TotemBioNet's .smb input file

```
$ totembionet -yed ./JBCB16_cellcycleIG.graphml
```

JBCB16_cellcycleIGFromYed.smb

```
VAR
sk = 0 .. 2;
a = 0 .. 1;
b = 0 .. 1;
ep = 0 .. 1;
en = 0 .. 1;

REG
sk_act1 [sk>=1] => sk;
a_ini_sk [!(a>=1)] => sk;
sk_act2 [sk>=2] => a;
b_ini [!(b>=1)] => a;
en_ini [!(en>=1)] => a;
a_act [a>=1] => b;
m[!((en>=1)|((en>=1)&(ep>=1)))]=>b;
b_act [(b>=1)] => ep;
sk_ini [!(sk>=1)] => en;
a_ini_en [!(a>=1)] => en;
ep_act [ep>=1] => en;
```

The tool automatically returns variables' upper bounds equal to their outgoing degree. Each variable's upper bound must be manually decreased when several regulation thresholds are equal.

## Step 3: Add properties into TotemBioNet's .smb input file

5_Experiment.smb

Content of JBCB16_cellcycleIGFromYed.smb

+

```
#######
HOARE
PRE : {sk=0,ep=0,a=0,b=0,en=1}

TRACE:
Forall((sk+; sk+; en-),
(sk+; en-; sk+),
(en-; sk+; sk+));
a+; sk-; sk-; b+;
Forall((a-;ep+), (ep+;a-));
en+; b-; ep-;

POST: {sk=0,ep=0,a=0,b=0,en=1}

#######
FAIRCTL
phicyclic=((sk=0 & ep=0 & a=0 & b=0 & en =1)->
AX(AF(sk=0 & ep=0 & a=0 & b=0 & en =1)));

###
CTL
phiG2M=((sk=0 & ep=0 & a=1 & b=1 & en=0) ->
(!( (EX((en=1) &  EX((a=0) & EX((ep=1) ))))
|  (EX((en=1) &  EX((ep=1) & EX((a=0) ))))
| (EX((a=0) &  EX((en=1) & EX((ep=1) ))))
| (EX((ep=1) &  EX((en=1) & EX((a=0) )))))));

phiG1S=((sk=0 & ep=0 & a=0 & b=0 & en=1) ->
(!((EX((a=1) & EX((sk=1) & EX((en=0) & EX((sk=2))))))
| (EX((sk=1) & EX((a=1) & EX((en=0) & EX((sk=2))))))
| (EX((sk=1) & EX((en=0) & EX((a=1) & EX((sk=2))))))
| (EX((a=1) & EX((en=0) & EX((sk=1) & EX((sk=2))))))
| (EX((sk=1) & EX((a=1) & EX((sk=2) & EX((en=0))))))
| (EX((sk=1) & EX((sk=2) & EX((a=1) & EX((en=0))))))
| (EX((a=1) & EX((sk=1) & EX((sk=2) & EX((en=0))))))
| (EX((en=0) & EX((a=1) & EX((sk=1) & EX((sk=2))))))
| (EX((en=0) & EX((sk=1) & EX((a=1) & EX((sk=2)))))))));
```

## Step 4: Call TotemBioNet with .smb input file

```
$ totembionet -csv ./5_Experiment.smb
```

```
./hoare/5_Experiment.out
./5_Experiment.out
./5_Experiment.csv
```

## Step 5: Content of TotemBioNet output files

The weakest precondition ($wp$) calculated from the Hoare triple is in:

./hoare/5_Experiment.out

```
(((sk=0) & ((ep=0) & ((a=0) & ((b=0) & (en=1)))))
& ((((K_sk:a_ini_sk>0) & ((K_sk:sk_act1:a_ini_sk>1)
& ((K_en:a_ini_en<1) & ((K_a:sk_act2:b_ini:en_ini>0)
& ((K_sk:sk_act1<2) & ((K_sk:sk_act1<1)
& ((K_b:a_act:m>0) & (((K_a:en_ini<1)
& ((K_ep:b_act>0) & ((K_en:sk_ini:a_ini_en:ep_act>0)
& ((K_b<1) & (K_ep<1))))) & ((K_ep:b_act>0)
& ((K_a:en_ini<1) & ((K_en:sk_ini:a_ini_en:ep_act>0)
& ((K_b<1) & (K_ep<1)))))))))))) & ((K_sk:a_ini_sk>0)
& ((K_en:a_ini_en<1) & ((K_sk:sk_act1:a_ini_sk>1)
& ((K_a:sk_act2:b_ini:en_ini>0) & ((K_sk:sk_act1<2)
& ((K_sk:sk_act1<1) & ((K_b:a_act:m>0)
& (((K_a:en_ini<1) & ((K_ep:b_act>0)
& ((K_en:sk_ini:a_ini_en:ep_act>0) & ((K_b<1)
& (K_ep<1))))) & ((K_ep:b_act>0) & ((K_a:en_ini<1)
& ((K_en:sk_ini:a_ini_en:ep_act>0) & ((K_b<1)
& (K_ep<1)))))))))))))) & ((K_en:sk_ini:a_ini_en<1)
& ((K_sk:a_ini_sk>0) & ((K_sk:sk_act1:a_ini_sk>1)
& ((K_a:sk_act2:b_ini:en_ini>0) & ((K_sk:sk_act1<2)
& ((K_sk:sk_act1<1) & ((K_b:a_act:m>0)
& (((K_a:en_ini<1) & ((K_ep:b_act>0)
& ((K_en:sk_ini:a_ini_en:ep_act>0)
& ((K_b<1) & (K_ep<1))))) & ((K_ep:b_act>0)
& ((K_a:en_ini<1) & ((K_en:sk_ini:a_ini_en:ep_act>0)
& ((K_b<1) & (K_ep<1)))))))))))))))
```

The conjunctive or disjunctive form of *wp* and results of model-checking are in:

`./5_Experiment.out`

```
Hoare result is a conjunction of conditions on K.
100540 models have been removed.

############ Result of model checking ############
# Total number of models 260
# Selected models: 28
# Computation time: 2s999ms
```

Selected (OK) and rejected (KO) parameterizations are saved in `./5_Experiment.csv`. The first line contains the complete list of *K parameters*. For example `K_sk:a_ini_sk:sk_act1` is the parameter of *sk* variable that denotes the presence of two resources: `a_ini_sk` and `sk_act1`. 28 parameterizations are selected in the $5^{th}$ experiment, so the file contains 28 *OK models* and 232 *KO models*.

`./5_Experiment.csv`: OK model section

| n° | valid | K_sk | K_sk:a_ini_sk | K_sk:sk_act1 | K_sk:a_ini_sk:sk_act1 | K_a | K_a:b_ini | K_a:en_ini | K_a:sk_act2 | K_a:b_ini:en_ini | K_a:en_ini:sk_act2 | K_a:b_ini:sk_act2 | K_a:b_ini:en_ini:sk_act2 | K_b | K_b:a_act | K_b:m | K_b:a_act:m | K_ep | K_ep:b_act | K_en | K_en:a_ini_en | K_en:ep_act | K_en:sk_ini | K_en:a_ini_en:ep_act | K_en:ep_act:sk_ini | K_en:a_ini_en:sk_ini | K_en:a_ini_en:ep_act:sk_ini |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 2 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 3 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 4 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 5 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 6 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 7 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 8 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 9 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 10 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 11 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 12 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 13 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 14 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 15 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 16 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 17 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 18 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 19 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 20 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 21 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 22 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 23 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 24 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 25 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 26 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 27 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 28 | OK | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |

The last column of `./5_Experiment.csv`, named *Explanation*, is filled in only for KO parameterizations. It provides an explanation of model rejection by giving the unsatisfied formulas. Only 5 out of the 232 *KO models* are shown below:

`./5_Experiment.csv`: extract of KO model section

| n° | valid | K_sk | K_sk:a_ini_sk | K_sk:sk_act1 | K_sk:a_ini_sk:sk_act1 | K_a | K_a:b_ini | K_a:en_ini | K_a:sk_act2 | K_a:b_ini:en_ini | K_a:en_ini:sk_act2 | K_a:b_ini:sk_act2 | K_a:b_ini:en_ini:sk_act2 | K_b | K_b:a_act | K_b:m | K_b:a_act:m | K_ep | K_ep:b_act | K_en | K_en:a_ini_en | K_en:ep_act | K_en:sk_ini | K_en:a_ini_en:ep_act | K_en:ep_act:sk_ini | K_en:a_ini_en:sk_ini | K_en:a_ini_en:ep_act:sk_ini |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 29 | KO | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 47 | KO | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 105 | KO | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 109 | KO | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 135 | KO | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |

`./5_Experiment.csv`: extract of KO model explanations

| n° | Explanation |
|---|---|
| 29 | [phiG1S] |
| 47 | [phicyclic] |
| 105 | [phiG2M] |
| 109 | [phiG2M, phiG1S] |
| 135 | [phicyclic, phiG2M, phiG1S] |