KaDE: a Tool to Compile Kappa Rules into (Reduced) ODE Models*

Ferdinanda Camporesi[1,2], Jérôme Feret[1,2], and Kim Quyên Lý[1,2]

[1] INRIA
[2] Département d'informatique de l'ÉNS,
École normale supérieure, CNRS, PSL Research University,
75005 Paris, France
campores@di.ens.fr, feret@ens.fr, quyen@di.ens.fr

Abstract. Kappa is a formal language that can be used to model systems of biochemical interactions among proteins. It offers several semantics to describe the behaviour of Kappa models at different levels of abstraction. Each Kappa model is a set of context-free rewrite rules. One way to understand the semantics of a Kappa model is to read its rules as an implicit description of a (potentially infinite) reaction network. KaDE is interpreting this definition to compile Kappa models into reaction networks (or equivalently into sets of ordinary differential equations). KaDE uses a static analysis that identifies pairs of sites that are indistinguishable from the rules point of view, to infer backward and forward bisimulations, hence reducing the size of the underlying reaction networks without having to generate them explicitly. In this paper, we describe the main current functionalities of KaDE and we give some benchmarks on case studies.

1 The differential semantics of Kappa

Kappa [1] is a rule-based language which describes the behaviour of some agents that may be bound together on interaction sites. In applications to Systems Biology, agents usually abstract proteins and interaction sites specific regions on their amino acid chains. Mechanistic interactions among proteins are described by the means of rewriting rules. For instance, the rule on the left in Fig. 1 stipulates that two proteins may bind via their respective right and left sites. Graphically (we have used GKappa [2] to draw the rules), the shape of a protein implicitly denotes its type. The same way, sites in proteins are identified by their positions (left, right). Sites may also carry an internal state which stands for

* This material is based upon works partially sponsored by the Defense Advanced Research Projects Agency (DARPA) and the U. S. Army Research Office under grant number W911NF-14-1-0367, and by the ITMO Plan Cancer 2014. The views, opinions, and/or findings contained in this article are those of the authors and should not be interpreted as representing the official views or policies, either expressed or implied, of DARPA, the U. S. Department of Defense, or ITMO.

Fig. 1. Two rules. (left) Two proteins may bind. (right) The protein on the left may activate the right site of the protein on the right.

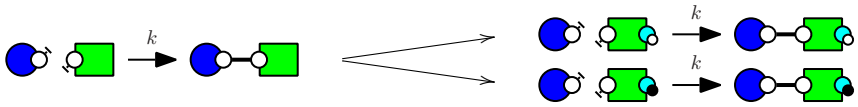


Fig. 2. From rules to reactions. The first rule in Fig. 1 is refined into two reactions according to whether or not right site of the right protein is phosphorylated.

an activation level (such as phosphorylation). In Fig. 1, the rule on the right stipulates that the bond between both proteins may activate the second one.

In a rule, the left hand side denotes some precondition, whereas the right hand side stands for a transformation. Some agents may miss some sites. This is the "Don't Care, Don't Write" convention [3]. The sites the state of which influences neither an interaction nor its kinetics are omitted. Each rule may be understood extensionally as a (finite or not) set of reactions, obtained by refining it according to its potential application contexts, until getting fully specified connected components. For instance the rule on the left in Fig. 1 may be applied with the protein on the right phosphorylated or not, as depicted in Fig. 2. In the differential semantics, rule applications preserve disconnectedness, unless specified explicitly. Thus, each connected component in the left hand side is refined separately. Agents may contain many sites and form arbitrary long chains. Thus Kappa models are usually highly combinatorial. A small number of rules may lead to a large (if not infinite) reaction networks [4,5].

The ODE semantics is defined in the following way. Each connected component in a reaction denotes an instance of a bio-molecular species. For every bio-molecular species S, a reaction $R_1 + \ldots + R_m \longrightarrow P_1 + \ldots + P_n$ gives the following contribution to the derivative of the concentration of the species S:

$$\frac{\mathrm{d}[S]}{\mathrm{d}t} \stackrel{+}{=} \sum_r \gamma(r) \cdot [r, R] \cdot \Delta(R, S) \cdot [R_1] \cdot \ldots \cdot [R_m]$$

where: 1. $\gamma(r)$ is the corrected rate of the rule r (a fraction of the rate of the rule r is taken according to a convention defining how automorphisms are taken into account); 2. $[r, R]$ is the number of different ways to induce the reaction R from the rule r; 3. and $\Delta(R, S)$ is the difference between the number of occurrences of the species S in the sequence $P_1, \ldots, P_n$ and the one in the sequence $R_1, \ldots, R_m$. We use the symbol $\stackrel{+}{=}$ because we totalise the contribution for each reaction R.

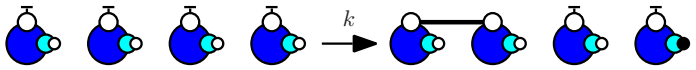
2 ODEs generation with KaDE

KaDE generates the differential semantics of Kappa models. In command-line mode, KaDE is called with a list of Kappa files and a list of options. A rudimentary graphical interface is available as well. The syntax of Kappa is described in

its reference manual [6]. KaDE generates output for the numerical integration tools Maple [7], Mathematica [8], Matlab [9], and Octave [10], and for the modelling standard languages DOTNET [11,12] and SBML [13]. DOTNET is the internal format of BioNetGen, we use it for compatibility with Erode [14], a tool to evaluate and reduce systems of ODEs. SBML output may be converted into LaTeX thanks to SBML2Latex [15]. SBML is also compatible with CellDesigner [16] which provides several tools dedicated to reaction networks.

The Kappa modelling platform extends the core Kappa language with tokens, algebraic expressions, and the possibility to allow the application of binary rules in unary contexts. Tokens are specific continuous variables which may be consumed and produced by rules according to user-specified stoichiometric coefficients. Kappa also supports arbitrary algebraic expressions both in rate parameters and in stoichiometric coefficients. These expressions may depend on the simulation time and on the concentration of some patterns in the current state of the system. They permit the encoding of kinetics laws beyond mass action. This feature is restricted to some specific backends. For instance, neither SBML, nor DOTNET cope with non-constant stoichiometric parameters. Lastly, a rule the left hand side of which is made of two connected components, may be provided two rates according to whether it is applied in a binary context (each connected component of the left hand side of the rule being embedded in two instances of bio-molecular species), or in a unary context (both connected components being embedded in the same instance of a bio-molecular species).

Some options let the end-user select the backend and change the name and the repository of the output file. Some other options tune the semantics of the model. It is also possible to truncate the ODES in order to ignore the bio-molecular species that would have more agents than a user-specified threshold. Three conventions exist for interpreting rate constants. In the following rule:



with the first convention (used by the simulator KaSim [17,6]), rates of rules are not corrected; with the second one (used by the simulator Simplx [3]), rates are divided by the number of automorphisms in the left hand side of rules (here 24); the third convention (used by the simulator NFSim [18]) accounts only for the permutations among the agents that are undistinguishable from a mechanistic point of view (here 2). The same issue occurs with reactions, where permutations among identical species are considered instead of automorphisms.

KaDE lets the end-user pick the convention for the rate constants of rules (in input files) and the one for the rate constant of reactions (in output networks). BioNetGen uses the third convention for rules and the first one for reactions. Lastly Erode takes the first convention for reaction rate constants in the differential setting and the second one in the stochastic one.

Some options tune the numerical integration parameters. This concerns the range for simulation time, the frequency of simulation plots, error tolerance parameters, and the size of integration steps. Moreover, the computation of the

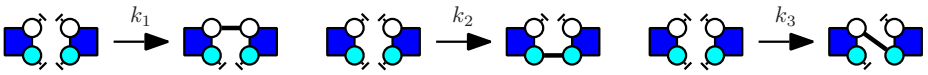


Fig. 3. Sites are equivalent, if the corrected rates of the third rule is twice the corrected rate of each other rule.

Jacobian may be disabled/enabled. It is also possible to warn numerical solvers that concentrations shall remain nonnegative.

Comparison with other tools. Both BioNetGen and Kappa can convert rules into reactions. BioNetGen supports compartmentalisation unlike Kappa. In BioNetGen, equivalent sites can be specified. In contrast, KaDE detects them automatically. BioNetGen does not support tokens.

3 Equivalent sites

Some sites may have exactly the same capabilities of interaction. This may be used to generate more compact systems of ODEs, by partitioning the set of bio-molecular species up to permutation of equivalent sites [19,20,21].

Consider the rules in Fig. 3. Each rule may be obtained from one another by swapping pairs of sites in agents: we say that these sites are equivalent. Equivalent sites may be used to induce forward and backward bisimulations over the stochastic and the differential semantics of Kappa [22,23,19,20,21].

Let us consider two sites x and y in a given kind of agent. A *set of rules* is symmetric with respect to the sites x and y if the corrected rates of every two rules that may be obtained one from the other by permuting the sites x and y in some agents, are inversely proportional to their numbers of automorphisms. The same way, a *valuation* from bio-molecular species to real numbers is symmetric with respect to the sites x and y if the images of every two bio-molecular species that can be obtained one from the other by permuting the sites x and y in some agents, are inversely proportional to their numbers of automorphisms. Lastly an *expression* over bio-molecular species is symmetric with respect to the sites x and y if and only if it takes the same values for every two symmetric valuations.

Whenever the set of rules and the initial state of the model are symmetric with respect to two sites, ignoring the difference among these sites in each bio-molecular species induces a backward bisimulation (i.e. the state of the system remains symmetric at every time [24,19]). Whenever the set of rules and each algebraic expression in rates or in stoichiometric coefficients are symmetric, ignoring the difference between these sites induces a forward bisimulation (we can define the ODEs directly over the equivalence-classes of species [24,19]).

KaDE may be parameterised for detecting the forward and backward bisimulations that are induced by pairs of equivalent sites. Then, it generates the corresponding reduced ODEs without relying on the initial reaction network.

Comparison with other tools. In BioNetGen [11,12] pairs of equivalent sites may be user-specified. In KaDE, equivalent sites are inferred automatically. The

expressive power of equivalent sites in BioNetGen and in KaDE are similar. Yet in BioNetGen equivalent sites must be equivalent in the rules, in the algebraic expressions, and in the initial state, whereas KaDE may exploit pairs of sites that are equivalent in the rules and in the algebraic expressions, but not necessarily in the initial state (forward bisimulation), or that are equivalent in the rules and in the initial state, but not necessarily in the algebraic expressions (backward bisimulation). Moreover, the kinetics conventions are a bit different. As a consequence, some models require more rules to be described in Kappa and some others require more rules to be described in BioNetGen (more details are provided in Supplementary Information [25]). From a combinatorial point of view, BioNetGen reasons on agents with multiple occurrences of equivalent sites, which may make the detection of embeddings exponentially costly (with respect to the number of agents). In constrast, KaDE quotients the set of bio-molecular species on the fly: it reasons on rigid site graphs for which the detection of embeddings is at worst quadratic [26,27].

Erode [14] is a tool for lumping systems of ODEs. In particular, it offers some primitives to discover the best forward bisimulation (resp. the best uniform backward bisimulation) induced by an equivalence relation over the bio-molecular species of a reaction network [28,29]. Erode can capture more forward bisimulations than KaDE since equivalent sites can induce only a particular kind of equivalence relations over species. Erode and KaDE are incomparable on backward bisimulations: on the first hand, KaDE focuses on equivalence among sites, but on the second hand, Erode focuses on uniform bisimulation which means that it cannot assign weights to bio-molecular species. For instance, Erode cannot express the backward bisimulation that gathers every kind of dimer in the example of Fig. 3 since the dimer made of a protein bound on its top site to the bottom site of another protein is twice abundant as the dimer made of two proteins bound together on their top sites (whenever the initial state and the rate constants are such that sites x and y are equivalent). As far as computation cost is concerned, Erode works on a fully expanded description of the system (either a reaction networks, or an ODE system), which may be impossible to compute for large models. KaDE discovers equivalent sites directly on the set of rules. Another difference is that KaDE applies on uninterpreted parameters (KaDE reductions remain valid if the value of rate parameters is modified) whereas Erode can compute bisimulations only over fully instantiated networks.

On fully instantiated networks, KaDE and Erode may be combined. Firstly, KaDE may quickly detect equivalent sites and generate reduced networks accordingly. Then Erode may look for further reductions. When focusing on forward bisimulation, Erode also provides a proof that final reductions are optimal.

4 Benchmarks

We test our framework on three families of models. These examples have been chosen to test for the time efficiency of model reduction tools under various conditions about the ratio between the number of Kappa rules with respect to

model	sites	rules	species		reactions	
			original	reduced	original	reduced
kinase/phosphatase	n	$6n$	$2+4^n$	$2+\binom{n+3}{3}$	$6n4^{n-1}$	$2n\binom{n+2}{2}$
multiple phosphorylation	n	$n2^n$	2^n	$n+1$	$n2^n$	$2n$
mult. phosphoryl. with counter	n	$2n^2$	2^n	$n+1$	$n2^n$	$2n$

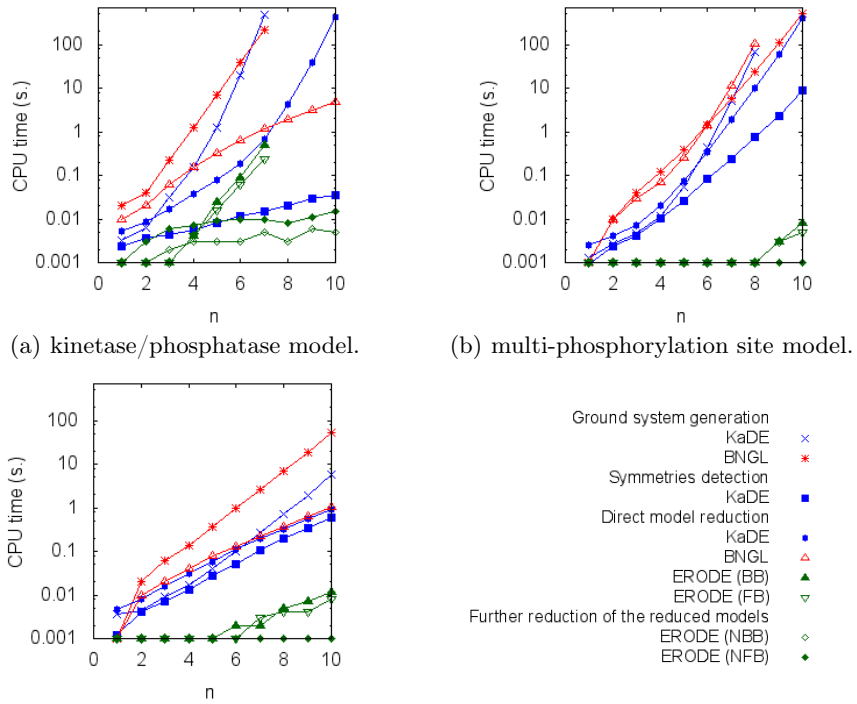
Fig. 4. Key attributes of our models with respect to the parameter n.

the number of reactions and about the ratio between different configurations for the bio-molecular species and the number of equivalence classes of configurations. In KaDE, the computation time for generating the networks (or the ODEs) depends mainly on the number of rules and the number of equivalence classes of configurations for the bio-molecular species. More examples, including most of the BioNetGen test suite, are provided in Supplementary Information [25].

The first family involves a kinase, a phosphatase, and a target protein. The target protein has n sites (n is left as a parameter). The kinase may bind and unbind to each non-phosphorylated site of the target protein. The kinase may phosphorylate a site when releasing it. Conversely, the phosphatase may bind and unbind to each phosphorylated site of the target protein. The phosphatase may also dephosphorylate a site when releasing it. We assume that every site has the same mechanistic properties and that the rate of reactions does not depend on the state of the other sites in the target protein.

The second and third families of models are inspired by the protein Kai. This protein plays a crucial role in the control of the circadian clock oscillations. We consider a protein with n sites (n is left as a parameter) which may each be phosphorylated, or not. The kinase and the phosphatase are not described explicitly. We assume that the rate constants of phosphoralylation (resp. dephosphorylation) of a site in a protein depend on the number of sites that are already phosphorylated in this protein. In the third family of models, a trick suggested by Pierre Boutillier is used to reduce drastically the number of rules that are required to describe the models. We use a fictitious site that is bound to a chain of fictitious proteins the length of which encodes the number of phosphorylated sites. When a site is phosphorylated, a new protein is inserted in the chain and removed when a site gets dephosphorylated. Thus the phosphorylation level of a protein can be checked by looking at the length of this chain, without having to enumerate the different combinations fot the sites that are phosphorylated.

In Fig. 4, we give the number of rules, species and reactions, for each family of models for the parameter n ranging from 1 to 10, as well as the number of reactions and species when equivalent sites are considered. In Fig. 5, we compare the computation time to generate the original and the reduced networks with BioNetGen and KaDE. The generation of reduced models with KaDE (which does not require explicit annotation of equivalent sites) is much faster than the one of the unreduced networks. KaDE and BioNetGen generate exactly the same reduced networks. Lastly, we apply the fast version of Erode of the bisimulation inference algorithm [29] on the original networks and the complete version on the reduced ones [28]. But we found not further reduction this way. In [25], we observe as good results on the BioNetGen test suite.



(a) kinetase/phosphatase model.

(b) multi-phosphorylation site model.

(c) multi-phosphorylation site model with counter.

(d) legend.

Fig. 5. Comparison between the time performances of KaDE, BioNetGen, and Erode, on a MacBookPro with a `2,8 GHz Intel Core i7` CPU and a `16 Go 1600 MHz DDR3` memory and with a 10 minutes time-out.

References

1. Danos, V., Laneve, C.: Formal molecular biology. TCS **325**(1) (2004) 69–110
2. Feret, J.: Gkappa: a library to generate site graphs with graphviz.
3. Danos, V., Feret, J., Fontana, W., Krivine, J.: Scalable simulation of cellular signaling networks. In Shao, Z., ed.: Proc. APLAS'07. Volume 4807 of LNCS., Springer (2007) 139–157
4. Feret, J., Danos, V., Krivine, J., Harmer, R., Fontana, W.: Internal coarse-graining of molecular systems. PNAS (2009)
5. Danos, V., Feret, J., Fontana, W., Harmer, R., Krivine, J.: Abstracting the differential semantics of rule-based models: exact and automated model reduction. In Jouannaud, J.P., ed.: Proc. LICS'10, IEEE Computer Society (2010) 362–381
6. Boutillier, P., Feret, J., Krivine, J., Q., Kim Lý.: Kasim development homepage
7. Monagan, M.B., Geddes, K.O., Heal, K.M., Labahn, G., Vorkoetter, S.M., McCarron, J., DeMarco, P.: Maple 10 Programming Guide. Maplesoft (2005)
8. Wolfram Research, I.: Mathematica. Wolfram Research, Inc. (2017)
9. MATLAB: version 9.2. The MathWorks Inc., Natick, Massachusetts (2017)
10. Eaton, J.W., Bateman, D., Hauberg, S., Wehbring, R.: GNU Octave version 4.0.0 manual: a high-level interactive language for numerical computations. Free Software Foundation (2015)

11. Blinov, M., Faeder, J.R., Goldstein, B., Hlavacek, W.S.: Bionetgen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. Bioinformatics (Oxford, England) **20**(17) (November 2004) 3289–3291
12. Faeder, J.R., Blinov, M.L., Hlavacek, W.S.: Rule-based modeling of biochemical systems with bionetgen. Methods Mol Biol **500** (2009) 113–67
13. Hucka, M., Bergmann, F.T., Hoops, S., Keating, S.M., Sahle, S., Schaff, J.C., Smith, L.P., Wilkinson, D.J.: The systems biology markup language (sbml): Language specification for level 3 version 1 core (2010)
14. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: ERODE: A tool for the evaluation and reduction of ordinary differential equations. In Legay, A., Margaria, T., eds.: Proc. TACAS'17. (2017) 310–328
15. Dräger, A., Planatscher, H., Wouamba, D.M., Schröder, A., Hucka, M., Endler, L., Golebiewski, M., Müller, W., Zell, A.: SBML2LaTeX: Conversion of SBML files into human-readable reports. Bioinformatics **25**(11) (April 2009) 1455–1456
16. Funahashi, A., Matsuoka, Y., Jouraku, A., Morohashi, M., Kikuchi, N., Kitano, H.: Celldesigner 3.5: A versatile modeling tool for biochemical networks. Proc. of the IEEE **96** (2008)
17. Boutillier, P., Ehrhard, T., Krivine, J.: Incremental update for graph rewriting. In Yang, H., ed.: Proc. ESOP'17. Volume 10201 of LNCS., Springer (2017) 201–228
18. Sneddon, M.W., Faeder, J.R., Emonet, T.: Efficient modeling, simulation and coarse-graining of biological complexity with nfsim. Nat. meth. **8** (2011) 177–183
19. Camporesi, F., Feret, J.: Formal reduction for rule-based models. ENTCS **276** (2011) 29 – 59 Proc. MFPS XXVII.
20. Camporesi, F., Feret, J., Koeppl, H., Petrov, T.: Combining model reductions. ENTCS **265** (2010) 73 – 96 Proc. MFPS XXVI.
21. Feret, J.: An algebraic approach for inferring and using symmetries in rule-based models. ENTCS **316** (2015) 45 – 65 Proc. SASB'14.
22. Buchholz, P.: Bisimulation relations for weighted automata. Theor. Comput. Sci. **393(1-3)** (2008) 109–123
23. Feret, J., Koeppl, H., Petrov, T.: Stochastic fragments: A framework for the exact reduction of the stochastic semantics of rule-based models. International Journal of Software and Informatics **7**(4) (2013) 527 – 604
24. Buchholz, P.: Exact and ordinary lumpability in finite Markov chains. Journal of Applied Probability **31(1)** (1994) 59–75
25. Camporesi, F., Feret, J., Lý, K.Q.: KaDE: a tool to compile kappa rules into (reduced) ode models: Supplementary information
26. Petrov, T., Feret, J., Koeppl, H.: Reconstructing species-based dynamics from reduced stochastic rule-based models. In Laroque, C., Himmelspach, J., Pasupathy, R., Rose, O., Uhrmacher, A.M., eds.: Proc. WSC'12, WSC (2012)
27. Oury, N., Pedersen, M., Petersen, R.L.: Canonical labelling of site graphs. In Petre, I., ed.: Proc. CompMod'13. Volume 116 of EPTCS. (2013) 13–28
28. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Forward and backward bisimulations for chemical reaction networks. In Aceto, L., de Frutos-Escrig, D., eds.: Proc. CONCUR'15. Volume 42 of LIPIcs., Schloss Dagstuhl (2015) 226–239
29. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Efficient syntax-driven lumping of differential equations. In Chechik, M., Raskin, J., eds.: Proc. TACAS'16. Volume 9636 of LNCS., Springer (2016) 93–111