Repairing Boolean regulatory networks using Answer Set Programming

Alexandre Lemos[1], Pedro T. Monteiro[1], Inês Lynce[1]
[1]*INESC-ID/Instituto Superior Técnico, Universidade de Lisboa, Portugal*

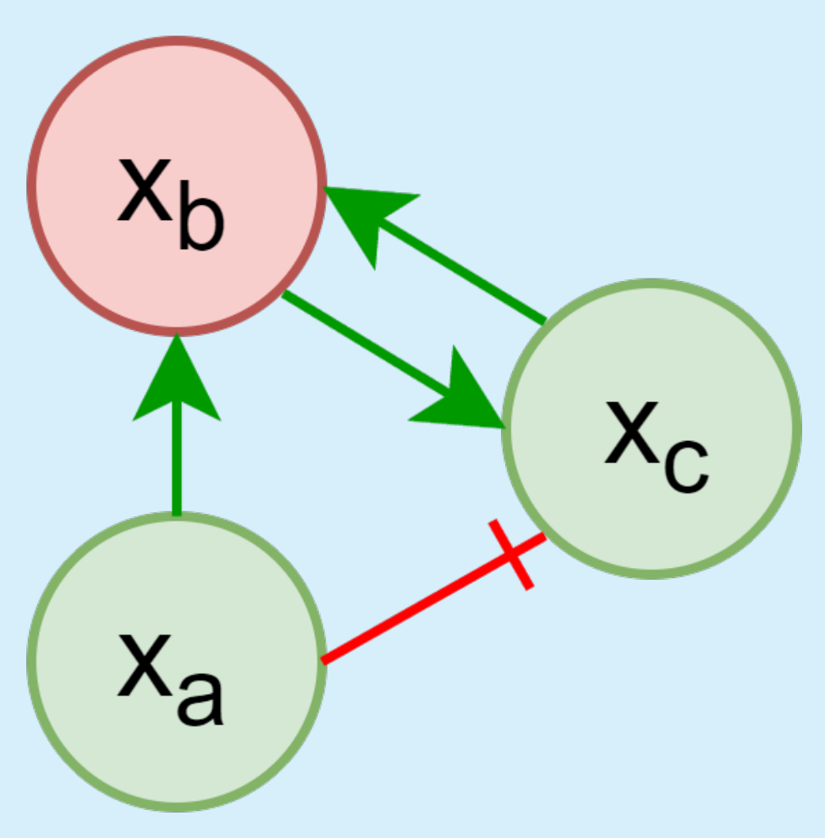
Introduction

Models of biological regulatory networks are increasingly used to formally describe and understand complex biological processes. Such models are often repaired whenever new observations become available, because the model cannot generate behaviors consistent with the new observations. However, the model repair procedure is often a manual process and therefore prone to errors.

Network Consistency

A biological regulatory networks is described using Boolean logical formalism [1], where:

- nodes denote biological components (*e.g.* `node(`x_a`)`);
- edges denote regulatory interactions between components (*e.g.* `edge(`x_a`,`x_b`)`);
- each component is associated with a Boolean variable representing its activity level (*e.g.* `obsvlabel(p,`x_a`,1)`);
- the evolution of each variable is defined by a Boolean logical function (*e.g.* `funcAnd(1,`x_b`)`) depending on the values of its regulators (*e.g.* `regulator(1,`x_a`)` and `regulator(1,`x_c`)`).



```
funcAnd(1,x_b).                node(x_b).                 node(temp(x_a)).
regulator(1,x_a).              node(x_a).                 obsvlabel(p,x_a,1).
regulator(1,x_c).              node(x_c).                 obsvlabel(p,x_c,1).
funcOr(2,x_c).                 edge(x_a,x_b).             obsvlabel(p,x_b,0).
regulator(2,x_b).              edge(x_c,x_b).             exp(p).
regulator(2,temp(x_a)).        edge(x_b,x_c).
funcNot(3,temp(x_a)).          edge(temp(x_a),x_c).
regulator(3,x_a).              edge(x_a,temp(x_a)).
```

Figure 1: Model inconsistent with the experimental profile on a steady state and the respective ASP encoding.

A model is consistent with a experimental profile if the functions associated with a node can explain the node's value.

ASP

Answer Set Programming (ASP) [2] is a form of declarative programming, similar to Prolog, that uses logic semantics to solve search problems. ASP has already been successfully applied to model biological networks [3, 4]. The proposed method was implemented using ASP. The complete encoding is available at `http://web.ist.utl.pt/~alexandre.lemos/rbnasp/`.

Repair Operations

Four basic types of repairs to the logical functions, which can then be further combined to produce more detailed revisions:

1. **Repair g** - Change a Boolean function (AND to OR, NOT to ID);
2. **Repair n** - Negates a Boolean function;
3. **Repair e** - Removes a regulator (never removes the last regulator, i.e., component cannot become an input);
4. **Repair i** - Negates a regulator (not previously negated).

Example

The model in Figure 1, can be repaired by applying two repairs of type n, *i.e.* first negating the function `funcAnd(1,`x_b`)` and then the function `funcOr(2,`x_c`)`.
It is also possible, for example, to negate regulators: `regulator(2,`x_b`)`, `regulator(1,`x_c`)`, `regulator(1,`x_a`)` and removing `regulator(2,`$temp(x_c)$`)` as shown in Figure 2.

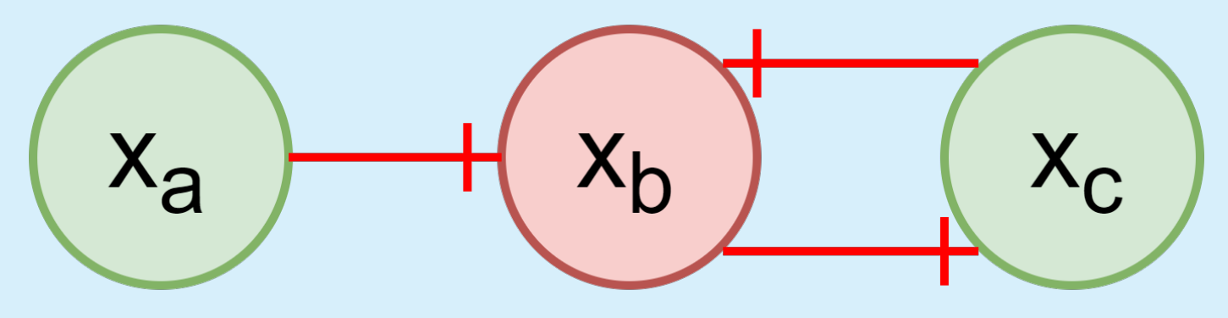


Figure 2: Model after negating regulators x_b, x_c, x_a and removing $temp(x_c)$.

Function Coverage

The number of possible Boolean functions that can be used to repair a function will increase with the number of regulators influencing a given component, following the expression 2^{2^n}, where n is the number of arguments of the function.
When considering a function with two arguments, by combining repairs e, i and g, one can achieve a total of twelve of the sixteen functions. The functions XOR (exclusive OR), XNOR (equivalence), true and false are not achievable by any combination of these repairs.

	$A \wedge B$	$A \wedge \neg B$	$\neg A \wedge \neg B$	$\neg A \vee B$	$A \vee B$	$A \vee \neg B$	$\neg A \vee \neg B$	B	A	$\neg B$	$\neg A$
repair	g	g,i	i	g	g	g, i	g, i	e	e,g	e,i	e

Table 1: Possible repairs for the function ¬ A ∧ B and which repairs are used to achieve them.

Method Validation

The implementation was tested using two regulatory networks with the following size:

	Nodes (input)	Not Func	Other Func
E-coli	1915 (34)	1327	1881
Candida	6410 (71)	10774	6339

Table 2: Size of the networks used for testing.

Conclusion

The proposed ASP-based method to repair Boolean networks is capable of repairing functions with any number of regulators. The repair operations are able to find a feasible solution to all real biological networks tested.
Repair n finds always a cardinality minimal solution. Repair e exceeds memory limits for bigger networks.

References

[1] R. Thomas, "Boolean formalization of genetic control circuits," *Journal of Theoretical Biology*, vol. 42, no. 3, pp. 563 – 585, 1973.

[2] M. Gebser, R. Kaminski, B. Kaufmann, and T. Schaub, "Answer set solving in practice," ser. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan and Claypool Publishers, 2012.

[3] M. Gebser, C. Guziolowski, M. Ivanchev, T. Schaub, A. Siegel, S. Thiele, and P. Veber, "Repair and prediction (under inconsistency) in large biological networks with answer set programming." in *Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010, Toronto, Ontario, Canada, May 9-13*, 2010.

[4] N. Mobilia, A. Rocca, S. Chorlton, E. Fanchon, and L. Trilling, "Logical modeling and analysis of regulatory genetic networks in a nonmonotonic framework," in *IWBBIO*, ser. LNCS, vol. 9043, 2015, pp. 599–612.

Acknowledgements

This work was partially supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2013. Alexandre Lemos was supported by FCT project grant EXCL/EEI-ESS/0257/2012. Pedro T. Monteiro was supported by FCT grant IF/01333/2013.