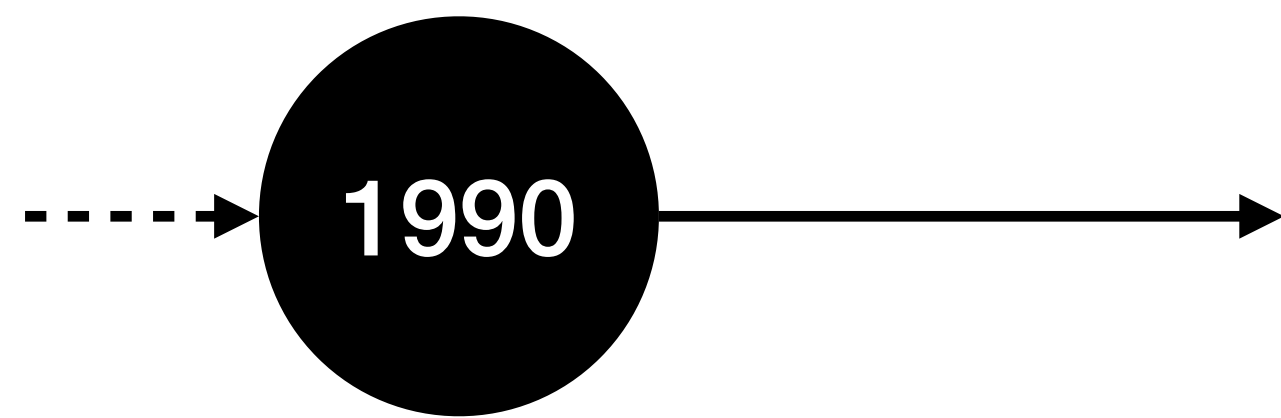
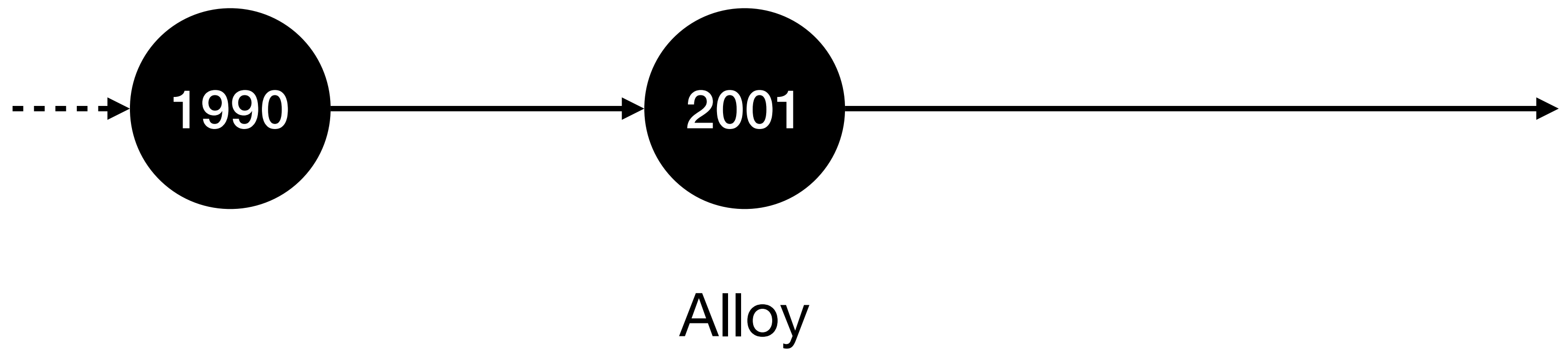


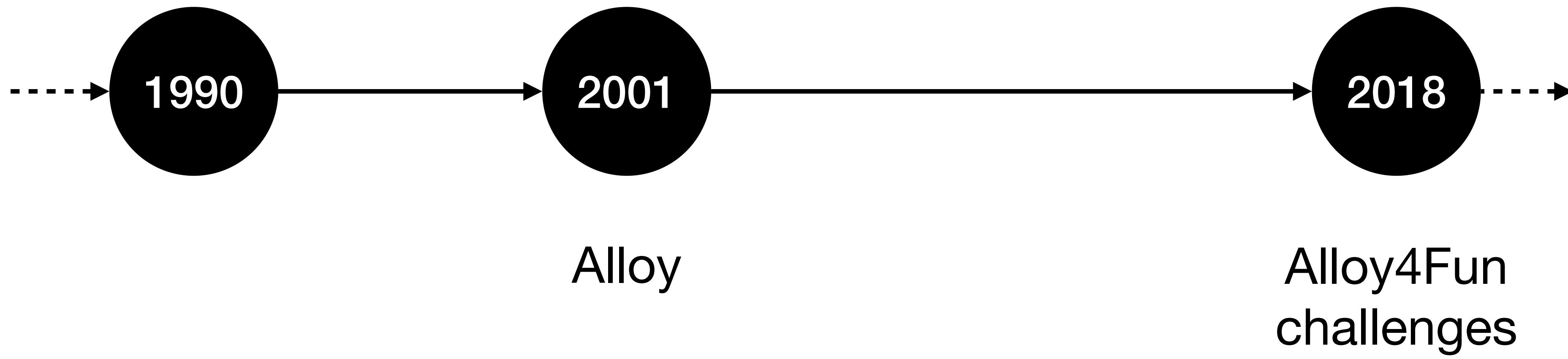
Teaching logic with specification challenges

Alcino Cunha & Nuno Macedo
INESC TEC & University of Minho, Portugal

Formal Methods @ UMinho









Formal Methods in Software Engineering — Métodos Formais em Engenharia de Software

Master in Informatics Engineering — Mestrado em Engenharia Informática

2025/26

This course unit in the first semester of the first year of the [Master in Informatics Engineering](#) at the [University of Minho](#) aims to introduce the use of Formal Methods in Software Engineering.

Students interested in deepening the knowledge acquired in this course unit may choose to attend the [Formal Programming Methods](#) specialization track in the second semester.

Syllabus

- Logics for formal specification
 - Propositional logic (PL)
 - First-order logic (FOL)
 - Relational logic (RL)
 - Linear temporal logic (LTL)

Alloy

```
sig User {  
    follows : set User  
}  
sig Influencer extends User {}
```

```
sig User {  
    follows : set User  
}  
sig Influencer extends User {}  
  
fact {  
    // Influencers are followed by everyone else  
    all u : Influencer | follows.u = User - u  
}
```

```
sig User {  
    follows : set User  
}  
sig Influencer extends User {}  
  
fact {  
    // Influencers are followed by everyone else  
    all u : Influencer | follows.u = User - u  
}  
  
run {  
    some Influencer  
}
```

/Users/alcino/ (SocialNetwork) Run Example

New Open Reload Save Execute Show Viz Txt Table Tree Theme Magic Layout Evaluator New Projection: none

```
sig User {
  follows : set User
}
sig Influencer extends User {}

fact {
  // Influencers are followed by everyone else
  all u : Influencer | follows.u = User - u
}

run Example {
  some Influencer
}
```

```
graph TD
  User1 -- follows --> User0
  User1 -- follows --> Influencer
  User0 -- follows --> Influencer
  User1 -- follows --> User1
```

Executing "Run Example"
Actual scopes: 3 User, 3 Influencer
Solver=minisat Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetr
127 vars. 15 primary vars. 191 clauses. 4ms.
Instance found. Predicate is consistent. 10ms.

Line 11, Column 12 [modified]

```
sig User {
  follows : set User
}
sig Influencer extends User {}

fact {
  // Influencers are followed by everyone else
  all u : Influencer | follows.u = User - u
}

run Example {
  some Influencer
}
```

Executing "Run Example"
Actual scopes: 3 User, 3 Influencer
Solver=minisat Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetr
127 vars. 15 primary vars. 191 clauses. 4ms.
Instance found. Predicate is consistent. 10ms.

```
graph TD
  User1[User1] -- follows --> User0[User0]
  User0 -- follows --> Influencer{{Influencer}}
  User1 -- follows --> Influencer
```

Line 11, Column 12 [modified]

dl.acm.org

ACM DIGITAL LIBRARY | acm Association for Computing Machinery | Premium Edition | acm open Provided by Massachusetts Institute of... | Sign in | Register

Journals | Magazines | Proceedings | Books | SIGs | Conferences | More | Search ACM Digital Library | Advanced Search

Conference | Proceedings | Upcoming Events | Authors | Affiliations | Award Winners

Home > Conferences > FSE > Proceedings > FSE 2016 > Lightweight specification and analysis of dynamic systems with rich configurations

RESEARCH-ARTICLE | FREE ACCESS | X | in | reddit | f | envelope

Lightweight specification and analysis of dynamic systems with rich configurations

Authors: Nuno Macedo, Julien Brunel, David Chemouil, Alcino Cunha, Denis Kuperberg | Authors Info & Claims

FSE 2016: Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering
Pages 373 - 383 • <https://doi.org/10.1145/2950290.2950318>

Published: 01 November 2016 | Publication History | Check for updates

48 335

PDF | eReader

Abstract

Model-checking is increasingly popular in the early phases of the software development process. To establish the correctness of a software design one must usually verify both *structural* and *behavioral* (or temporal) properties. Unfortunately, most specification languages, and accompanying model-

```
sig User {  
    var follows : set User  
}  
sig Influencer extends User {}
```

```
sig User {  
    var follows : set User  
}  
sig Influencer extends User {}  
  
pred follow[u,v : User] {  
    v not in u.follows and u != v  
    follows' = follows + u->v  
}
```

```
sig User {  
    var follows : set User  
}  
sig Influencer extends User {}  
  
pred follow[u,v : User] {  
    v not in u.follows and u != v  
    follows' = follows + u->v  
}  
pred stutter { follows' = follows }  
  
fact {  
    no follows  
    always (stutter or some u,v : User | follow[u,v])  
}
```

```
sig User {  
    var follows : set User  
}  
sig Influencer extends User {}  
  
pred follow[u,v : User] {  
    v not in u.follows and u != v  
    follows' = follows + u->v  
}  
pred stutter { follows' = follows }  
  
fact {  
    no follows  
    always (stutter or some u,v : User | follow[u,v])  
}  
  
run Example {  
    some Influencer  
} for exactly 3 User
```

New Open Reload Save Execute Show

```
sig User {
  var follows : set User
}
sig Influencer extends User {}

pred follow[u,v : User] {
  v not in u.follows and u !=
  follows' = follows + u->v
}

pred stutter {
  follows' = follows
}

fact {
  no follows
  always (stutter or some u,v : User | follow[u,v])
}

run Example {
  some Influencer
} for exactly 3 User
```

Line 22, Column 15 [modified]

(SocialNetwork) Run Example for exactly 3 User

Viz Txt Table Tree Theme Magic Layout Evaluator New Config New Trace New Init New Fork Projection: none

Influencer User0 User1

Influencer User0 User1

Executing "Run Example for exactly 3 User"
Actual scopes: exactly 3 User, 3 Influencer
Solver=minisat Steps=1..10 Bitwidth=4 MaxSeq=4 SkolemDep
1..1 steps. 96 vars. 13 primary vars. 181 clauses. 2ms.
Instance found. Predicate is consistent. 3ms.

```
sig User {
  var follows : set User
}
sig Influencer extends User {}

pred follow[u,v : User] {
  v not in u.follows and u !=
  follows' = follows + u->v
}

pred stutter {
  follows' = follows
}

fact {
  no follows
  always (stutter or some u,v : User | follow[u,v])
}

run Example {
  some Influencer
} for exactly 3 User
```

(SocialNetwork) Run Example for exactly 3 User

Viz Txt Table Tree Theme Magic Layout Evaluator New Config New Trace New Init New Fork Projection: none

Executing "Run Example for exactly 3 User"
Actual scopes: exactly 3 User, 3 Influencer
Solver=minisat Steps=1..10 Bitwidth=4 MaxSeq=4 SkolemDep
1..1 steps. 96 vars. 13 primary vars. 181 clauses. 2ms.
Instance found. Predicate is consistent. 3ms.

Line 22, Column 15 [modified]

```

sig User {
  var follows : set User
}
sig Influencer extends User {}

pred follow[u,v : User] {
  v not in u.follows and u !=
  follows' = follows + u->v
}

pred stutter {
  follows' = follows
}

fact {
  no follows
  always (stutter or some u,v : User | follow[u,v])
}

run Example {
  some Influencer
} for exactly 3 User

```

(SocialNetwork) Run Example for exactly 3 User

Projection: none

follows: 1

follows: 2

Executing "Run Example for exactly 3 User"

Actual scopes: exactly 3 User, 3 Influencer
 Solver=minisat Steps=1..10 Bitwidth=4 MaxSeq=4 SkolemDep
 1..1 steps. 96 vars. 13 primary vars. 181 clauses. 2ms.
Instance found. **Predicate** is consistent. 3ms.

Line 22, Column 15 [modified]

/Users/alcino/Talks/2026 FMTea/SocialNetwork.als

New Open Reload Save Execute Show

```
}
sig Influencer extends User {}

pred follow[u,v : User] {
  v not in u.follows and u != v
  follows' = follows + u->v
}

pred stutter {
  follows' = follows
}

fact {
  no follows
  always (stutter or some u,v : User | follow[u,v])
}

run Example {
  some Influencer
} for exactly 3 User

check NoSelfFollows {
  always (no follows & iden)
} for 3 but 1.. steps
```

Executing "Run Example for exactly 3 User"
Actual scopes: exactly 3 User, 3 Influencer
Solver=minisat Steps=1..10 Bitwidth=4 MaxSeq=4 SkolemDep
1..1 steps. 96 vars. 13 primary vars. 181 clauses. 2ms.
Instance found. Predicate is consistent. 3ms.

Option Solver changed to electrodnuxmv
Executing "Run Example for exactly 3 User"
Actual scopes: exactly 3 User, 3 Influencer
Solver=electrod.nuxmv Steps=1..10 Bitwidth=4 MaxSeq=4 SkolemDep
No translation information available. 0ms.
Instance found. Predicate is consistent. 178ms.

Executing "Check NoSelfFollows for 3 but 1.. steps"
Actual scopes: 3 User, 3 Influencer
Solver=electrod.nuxmv Steps=1..2147483647 Bitwidth=4 MaxSeq=4 SkolemDep
No translation information available. 0ms.
No counterexample found. Assertion may be valid. 53ms.

Line 26, Column 22 [modified]

Alloy4Fun

dl.acm.org

ACM DIGITAL LIBRARY | Association for Computing Machinery | Premium Edition | acm open | Provided by Massachusetts Institute of... | Sign in | Register

Journals | Magazines | Proceedings | Books | SIGs | Conferences | More | Search ACM Digital Library | Advanced Search

Conference | Proceedings | Upcoming Events | Authors | Affiliations | Award Winners

Home > Conferences > PASTE > Proceedings > PASTE '13 > Increasing human-tool interaction via the web

RESEARCH-ARTICLE | FREE ACCESS | X | in | reddit | f | mail

Increasing human-tool interaction via the web

Authors: Thomas Ball, Peli de Halleux, Nikhil Swamy, Daan Leijen | [Authors Info & Claims](#)

PASTE '13: Proceedings of the 11th ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering
Pages 49 - 52 • <https://doi.org/10.1145/2462029.2462031>

Published: 20 June 2013 [Publication History](#) | Check for updates

3 122 | PDF | eReader

Abstract

Software tools researchers can accelerate their ability to learn by exposing tools to users via web technologies, allowing them to observe and test the interactions between humans and tools. At Microsoft Research, we have developed a web service (<http://www.rise4fun.com/>) for such a purpose that is available for community use.

Feedback

This puzzle is an interactive Coding Duel. Can you write code that matches a secret implementation? [Help](#)

```
using System;

public class Player {
    public static int Puzzle(int x) {
        return x;
    }
}
```




Ask Pex!

Done. 3 interesting inputs found. [How does Pex work?](#)

Save

Permalink

Pex found 2 differences between your puzzle method and the secret implementation. Improve your code, so that it matches the other implementation, and 'Ask Pex!' again.

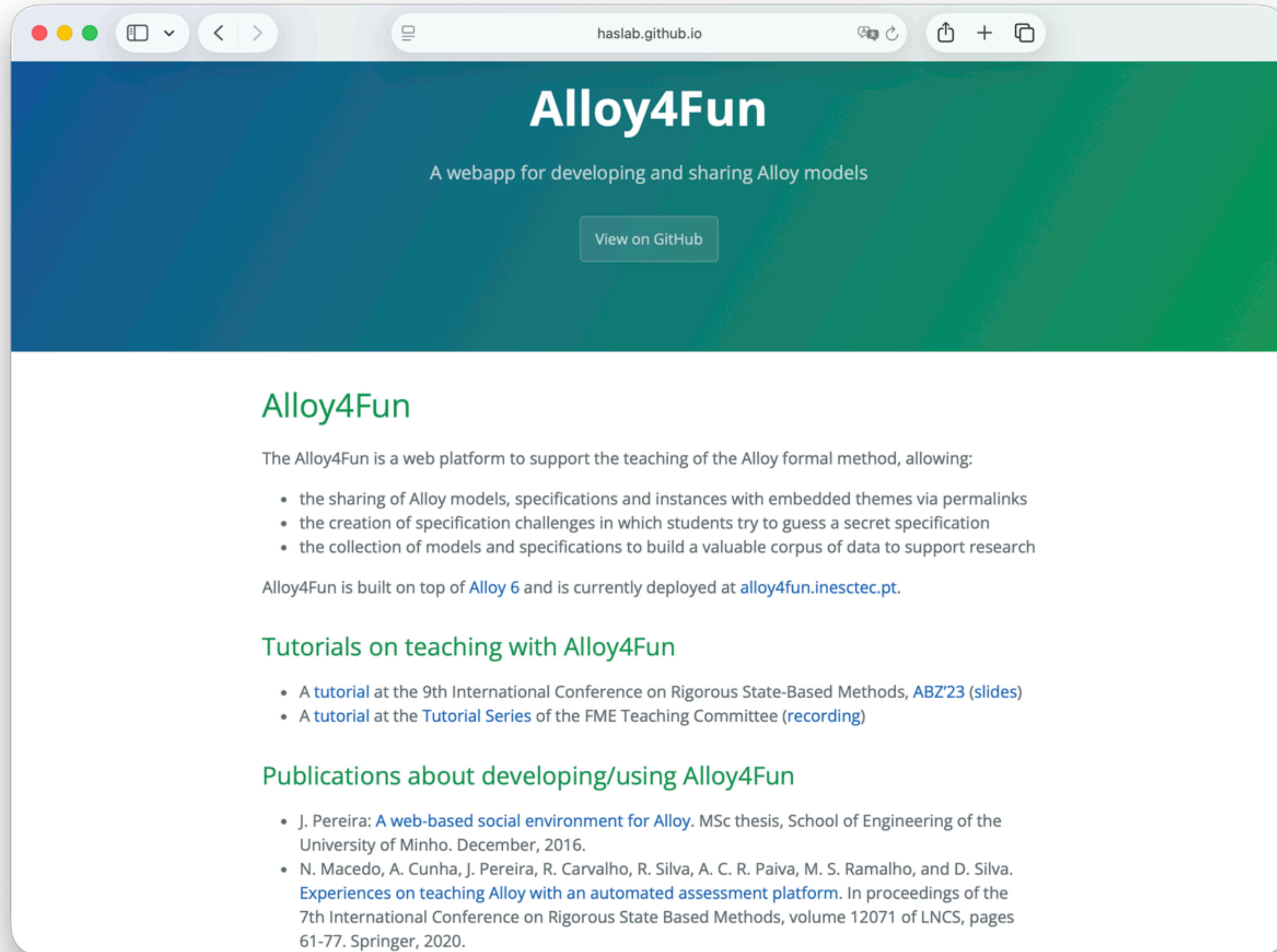
	x	your result	secret implementation result	Output/Exception	Error Message
	0	0	1	Mismatch	Your puzzle method produced the wrong result.
	1	1	1		
	7	7	5040	Mismatch	Your puzzle method produced the wrong result.

Features inspired by Rise4Fun

- Easy sharing of Alloy models via permalinks
- Automatic data-collection
- Support for specification duels (aka challenges)

New features

- Graphical depiction of instances
- Easy sharing of instances via permalinks (preserving layout and theme)
- More refined data-collection mechanism
- Fully anonymous interaction



Alloy4Fun

A webapp for developing and sharing Alloy models

[View on GitHub](#)

Alloy4Fun

The Alloy4Fun is a web platform to support the teaching of the Alloy formal method, allowing:

- the sharing of Alloy models, specifications and instances with embedded themes via permalinks
- the creation of specification challenges in which students try to guess a secret specification
- the collection of models and specifications to build a valuable corpus of data to support research

Alloy4Fun is built on top of [Alloy 6](#) and is currently deployed at alloy4fun.inesctec.pt.

Tutorials on teaching with Alloy4Fun

- A [tutorial](#) at the 9th International Conference on Rigorous State-Based Methods, [ABZ'23 \(slides\)](#)
- A [tutorial](#) at the [Tutorial Series](#) of the FME Teaching Committee ([recording](#))

Publications about developing/using Alloy4Fun

- J. Pereira: [A web-based social environment for Alloy](#). MSc thesis, School of Engineering of the University of Minho. December, 2016.
- N. Macedo, A. Cunha, J. Pereira, R. Carvalho, R. Silva, A. C. R. Paiva, M. S. Ramalho, and D. Silva. [Experiences on teaching Alloy with an automated assessment platform](#). In proceedings of the 7th International Conference on Rigorous State Based Methods, volume 12071 of LNCS, pages 61-77. Springer, 2020.

View PDF Download full issue

Reading Assistant

Experiences on teaching alloy with an automated assessment platform

Nuno Macedo, Alcino Cunha, José Pereira, Renato Carvalho, Ricardo Silva, Ana C.R. Paiva, Miguel Sozinho Ramalho, Daniel Silva

Add to Mendeley Share Cite

https://doi.org/10.1016/j.scico.2021.102690 Under an Elsevier user license Open archive

- Highlights
Alloy is a specification language taught in several graduate formal methods courses.
Alloy4Fun is a web application for online editing and sharing of Alloy models.

Part of special issue
ABZ 2020 Rigorous State Based Methods
Edited by Dominique Mery, Raschke Alexander
View special issue

- Recommended articles
A comprehensive analysis of a pseudo-brittle fracture at the...
Influence of platform design and power take-off characteristics o...
Metal-zeolite catalysts for the removal of pharmaceutical...

- Outline
Highlights
Abstract
Keywords
1. Introduction
2. Alloy4Fun overview
3. Formalizing Alloy4Fun with Alloy
4. Writing and mining challenges
5. Experiences on teaching with Alloy4...
6. Concluding remarks and future work
CRediT authorship contribution state...
Declaration of Competing Interest
Acknowledgements
References
Show full outline

Show 3 more articles FEEDBACK



Not Secure — alloy4fun.inesctec.pt



Alloy4Fun

```
1 sig User {
2   follows : set User
3 }
4 sig Influencer extends User {}
5
6 fact {
7   // Influencers are followed by everyone else
8   all u : Influencer | follows.u = User - u
9 }
10
11 run {
12   some Influencer
13 }
```



Execute



Share model



Statistics



Derivations

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).

Privacy Policy: Alloy4Fun logs all user interactions, including code and options, which may be analyzed and



Not Secure — alloy4fun.inesctec.pt



Alloy4Fun

```
1 sig User {
2   follows : set User
3 }
4 sig Influencer extends User {}
5
6 fact {
7   // Influencers are followed by everyone else
8   all u : Influencer | follows.u = User - u
9 }
10
11 run {
12   some Influencer
13 }
```



Execute



Share model



Statistics



Derivations

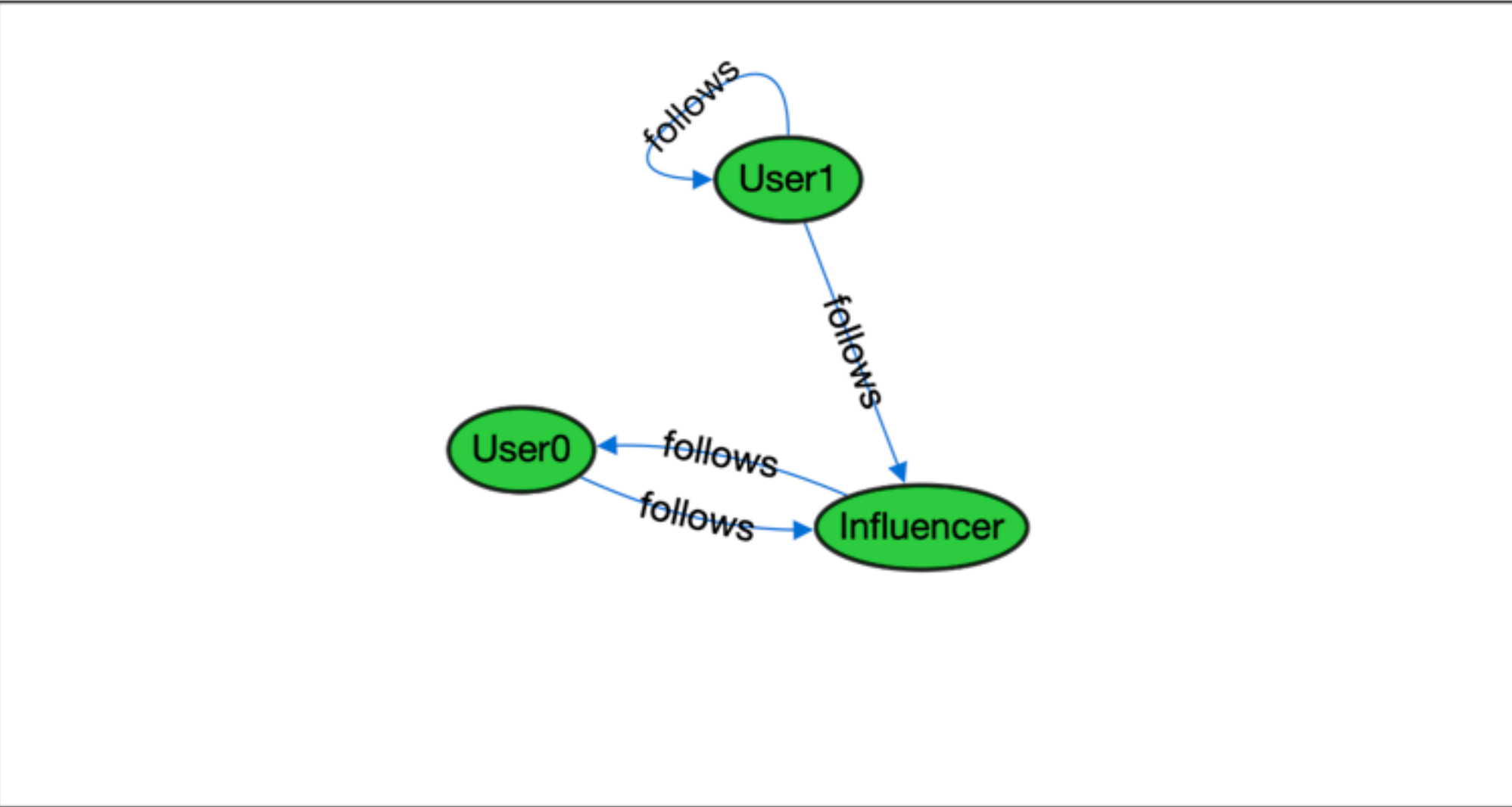
public link

<http://alloy4fun.inesctec.pt/KqoeLamCR4jRz3QbD>



Copy to clipboard

Instance found. run 1 is consistent.




Previous instance

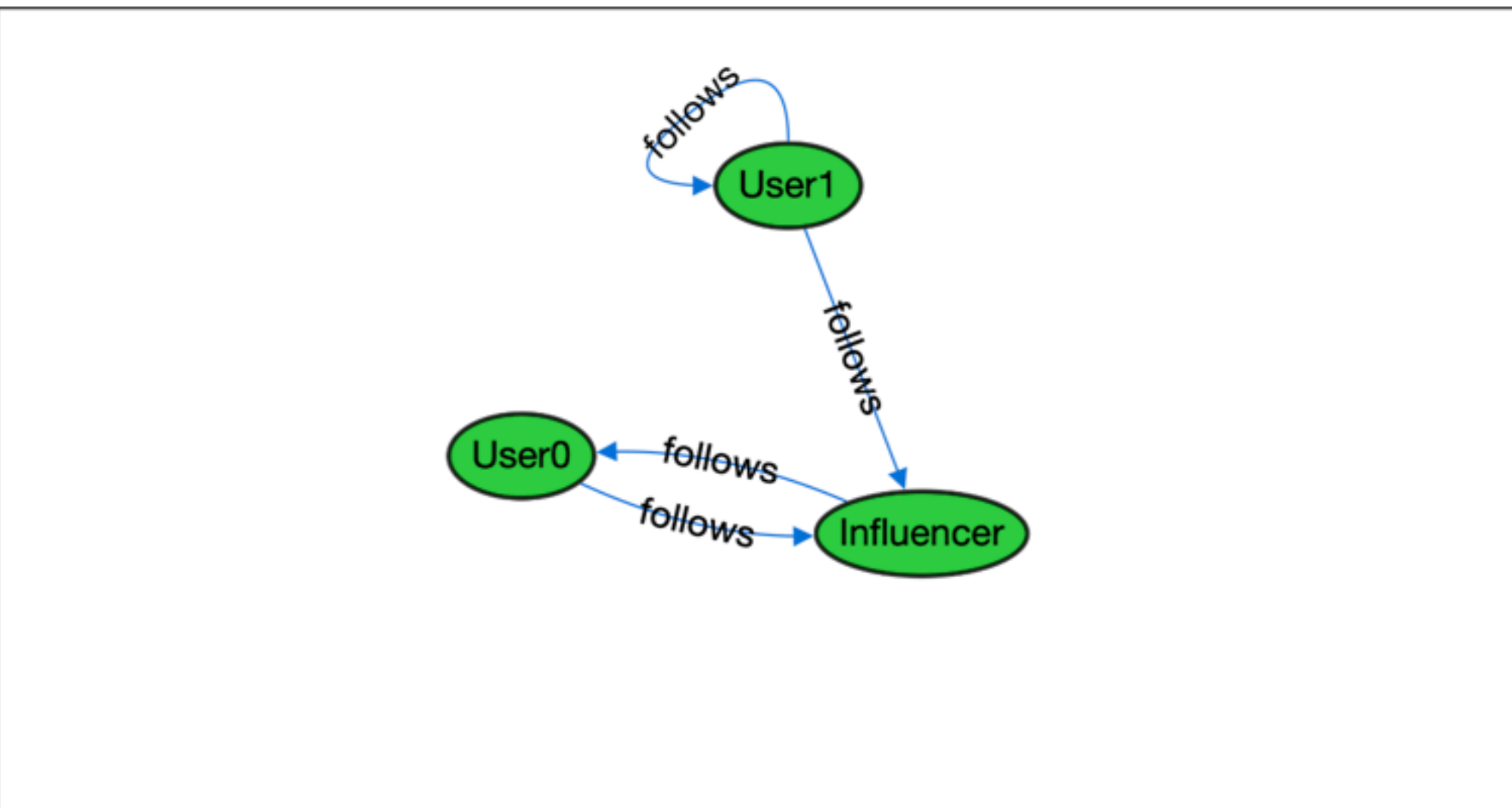

Next instance


Share instance

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).

Privacy Policy: Alloy4Fun logs all user interactions, including code and options, which may be analyzed and made public for research purposes. Do not share confidential information. However, Alloy4Fun does not collect any personally identifiable information.

Instance found. run 1 is consistent.



Previous instance Next instance

Share instance

instance link

<http://alloy4fun.inesctec.pt/yFzRjyxjpvEmCaJZD>

Copy to clipboard

Specification challenges

```
sig User {  
    follows : set User  
}  
sig Influencer extends User {}  
  
//SECRET  
fact {  
    // Influencers are followed by everyone else  
    all u : Influencer | follows.u = User - u  
}  
  
run {  
    some Influencer  
}
```

```
1 sig User {
2   follows : set User
3 }
4 sig Influencer extends User {}
5
6 //SECRET
7 fact {
8   // Influencers are followed by everyone else
9   all u : Influencer | follows.u = User - u
10 }
11
12 run {
13   some Influencer
14 }
```



Execute



Share model



Statistics



Derivations

public link

<http://alloy4fun.inesctec.pt/ADagtDQB4fDgmrNQC>

Copy to clipboard

private link

<http://alloy4fun.inesctec.pt/BSnGATog6bzKWvq4K>

Copy to clipboard



Not Secure — alloy4fun.inesctec.pt



Alloy4Fun

```
1 sig User {
2   follows : set User
3 }
4 sig Influencer extends User {}
5
6
7
8 run {
9   some Influencer
10 }
```



Execute



Share model



Statistics



Derivations

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).





Privacy Policy: Alloy4Fun logs all user interactions, including code and options, which may be analyzed and

```
sig User {
    follows : set User
}
sig Influencer extends User {}

pred Spec {
    // Influencers are followed by everyone else
}
//SECRET
pred Oracle {
    all u : Influencer | follows.u = User - u
}
//SECRET
check Spec {
    Spec iff Oracle
} for 4
```


Not Secure — alloy4fun.inesctec.pt

```
9 }
10
11 //SECRET
12 pred Oracle {
13     all u : Influencer | follows.u = User - u
14 }
15
16 //SECRET
17 check Spec {
18     Spec iff Oracle
19 } for 4
```

 *Execute*  *Share model*  *Statistics*  *Derivations*


public link

<http://alloy4fun.inesctec.pt/YmiNBZgsE5ADk6riq>

 *Copy to clipboard*

private link

<http://alloy4fun.inesctec.pt/H7QoF2z2WXz8N9wcu>

 *Copy to clipboard*

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).

Privacy Policy: Alloy4Fun logs all user interactions, including code and options, which may be analyzed and



Not Secure — alloy4fun.inesctec.pt



Alloy4Fun

```
1 sig User {
2   follows : set User
3 }
4 sig Influencer extends User {}
5
6 pred Spec {
7   // Influencers are followed by everyone else
8
9 }
10
11
```



Execute



Share model



Statistics




Derivations

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).

Privacy Policy: Alloy4Fun logs all user interactions, including code and options, which may be analyzed and

Alloy4Fun

```
1 sig User {  
2   follows : set User  
3 }  
4 sig Influencer extends User {}  
5  
6 pred Spec {  
7   // Influencers are followed by everyone else  
8   all u : Influencer | follows.u = User  
9 }  
10  
11
```

-  Execute
-  Share model
-  Statistics
-  Derivations

Counter-example found. check Spec is invalid.





Not Secure — alloy4fun.inesctec.pt



Alloy4Fun

```
1 sig User {
2   follows : set User
3 }
4 sig Influencer extends User {}
5
6 pred Spec {
7   // Influencers are followed by everyone else
8   all i : Influencer, u : User | i != u iff u in follows.i
9 }
10
11
```



Execute



Share model



Statistics



Derivations

No counter-example found. check Spec may be valid.

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).



Not Secure — alloy4fun.inesctec.pt



Alloy4Fun

```
1 sig User {
2   follows : set User,
3   sees : set Photo,
4   posts : set Photo,
5   suggested : set User
6 }
7
8 sig Influencer extends User {}
9
10 sig Photo {
11   date : one Day
12 }
13 sig Ad extends Photo {}
14
15 sig Day {}
16 // Specify the following properties.
17 // You can check their correctness with the different commands and
18 // when specifying each property you can assume all the previous ones to be true.
19
20 pred inv1 {
21   // Every image is posted by one user.
22 }
23
24
25
26 pred inv2 {
27   // Users cannot follow themselves.
28 }
29
30
31
32 pred inv3 {
33   // Users can see ads posted by everyone,
34   // but only see non ads posted by followed users.
35 }
36
```

Command :



Execute



Share model



Statistics



Derivations

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. [More](#)



Not Secure — alloy4fun.inesctec.pt



Alloy4Fun

```
13 var sig File {
14     var link : lone File
15 }
16 var sig Trash in File {}
17 var sig Protected in File {}
18
19 // initially the trash is empty and there are no protected file
20 pred prop1 {
21
22 }
23
24 // initially there are no files, but some are immediately created
25 pred prop2 {
26
27 }
28
29 // there is always some file in the system
30 pred prop3 {
31
32 }
33
34 // some file will eventually be sent to the trash
35 pred prop4 {
36
37 }
38
39 // some file will eventually be deleted
40 pred prop5 {
41
42 }
43
44 // whenever a file is sent to the trash, it remains in there forever
45 pred prop6 {
46
47 }
48 }
```

Command :



Execute



Share model



Statistics



Derivations

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. [More](#)



Not Secure — alloy4fun.inesctec.pt



Alloy4Fun

```
1 module network
2
3 some sig Node {
4   adj : set Node,
5 }
6 sig Router in Node {}
7
8
9
10 sig Message {}
11
12 /* Assume the existence of the following events, and that
13 only one may happen at each state:
14 pred send[n : Node, m : Message] {...}
15 pred receive[n : Node, m : Message] {...}
16 pred stutter {...} */
17
18
19 pred temp1 {
20   // nothing will ever happen
21 }
22
23
24 pred temp2 {
25   // any received message must have been sent before
26 }
27
28
29
30 pred temp3 {
31   // nodes eventually stop sending messages
32 }
33 }
```

Command :



Execute



Share model



Statistics



Derivations

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. [More](#)



Not Secure — alloy4fun.inesctec.pt



Alloy4Fun

```
1 module network
2
3 sig Message {}
4 enum State { Active, Inactive }
5 some sig Node {
6   adj : set Node,
7   var inbox : set Message,
8   var state : one State
9 }
10 sig Router in Node {}
11
12
13
14 pred inv {
15   all n : Node | n.state = Inactive implies no n.inbox
16 }
17 /* Assume that only one event may happen at each state. */
18
19
20 pred receive[n : Node, m : Message] {
21   // add the message to the inbox if active
22 }
23 }
24
25 pred send[n : Node, m : Message] {
26   // send a message to the adjacent nodes if active
27 }
28 }
29
```

Command :



Execute



Share model



Statistics



Derivations

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. [More](#)

Helping students with hints



Not Secure — alloy4fun.inesctec.pt



Alloy4Fun

```
1 sig User { follows : set User }
2 sig Influencer extends User {}
3
4 pred Spec { // Influencers are followed by everyone else
5   all u : Influencer | follows.u = User
6 }
7
```



Execute



Share model



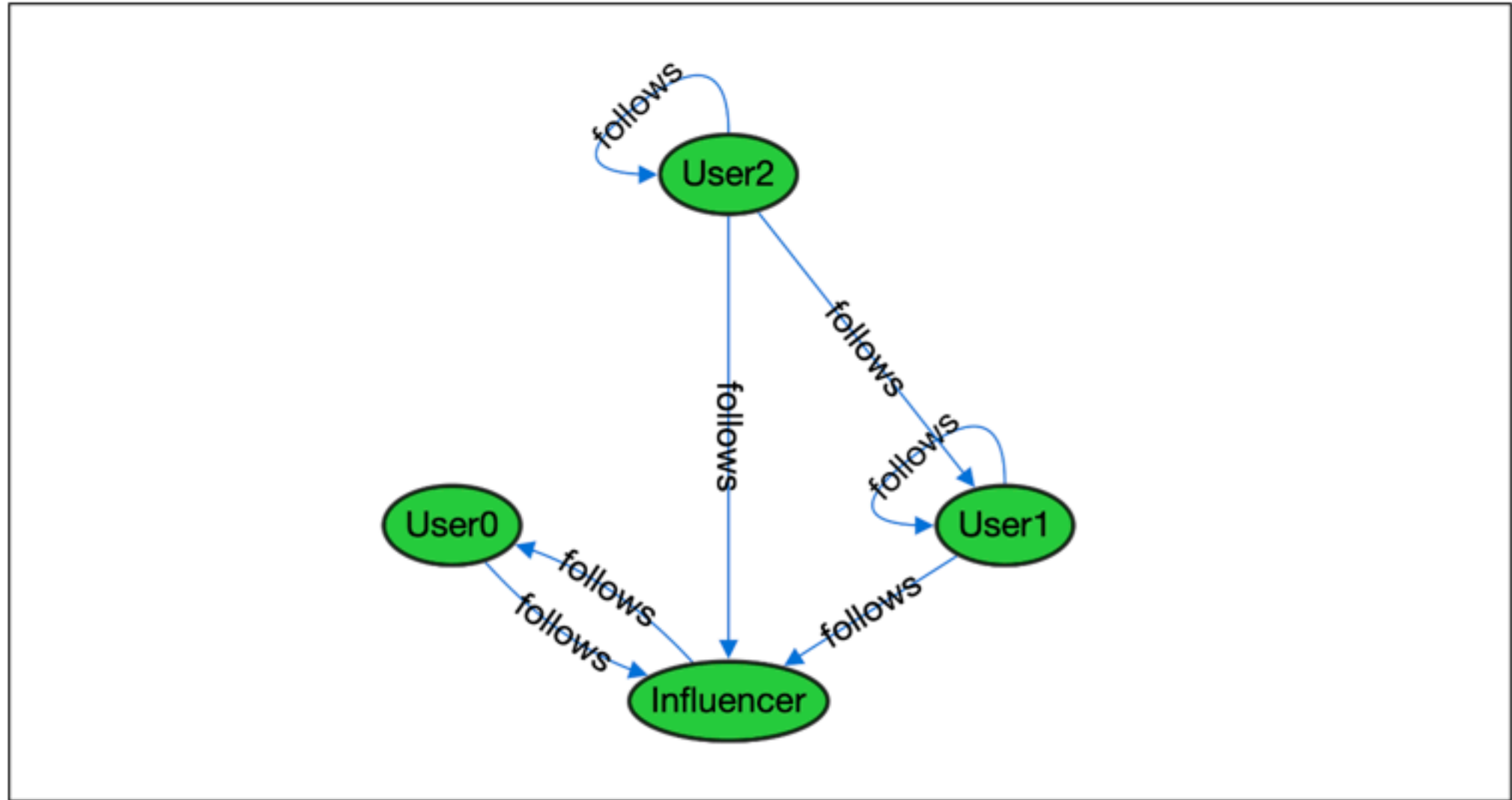
Statistics



Derivations

Counter-example found. check Spec is invalid.

Counter-example found. check Spec is invalid.



⏪
Previous instance

⏩
Next instance

🔗
Share instance

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).

Privacy Policy: Alloy4Fun logs all user interactions, including code and options, which may be analyzed and made public for research purposes. Do not share confidential information. However, Alloy4Fun does not collect any personally identifiable information.

Not Secure — alloy4fun.inesctec.pt

Counter-example found. check Spec is invalid.

```
graph TD; User2((User2)) -- follows --> User0((User0)); User2 -- follows --> User1((User1)); User2 -- follows --> Influencer{{Influencer}}; User1 -- follows --> User2; User1 -- follows --> Influencer; User0 -- follows --> Influencer; User2 -- follows --> User2; User1 -- follows --> User1;
```

Previous instance

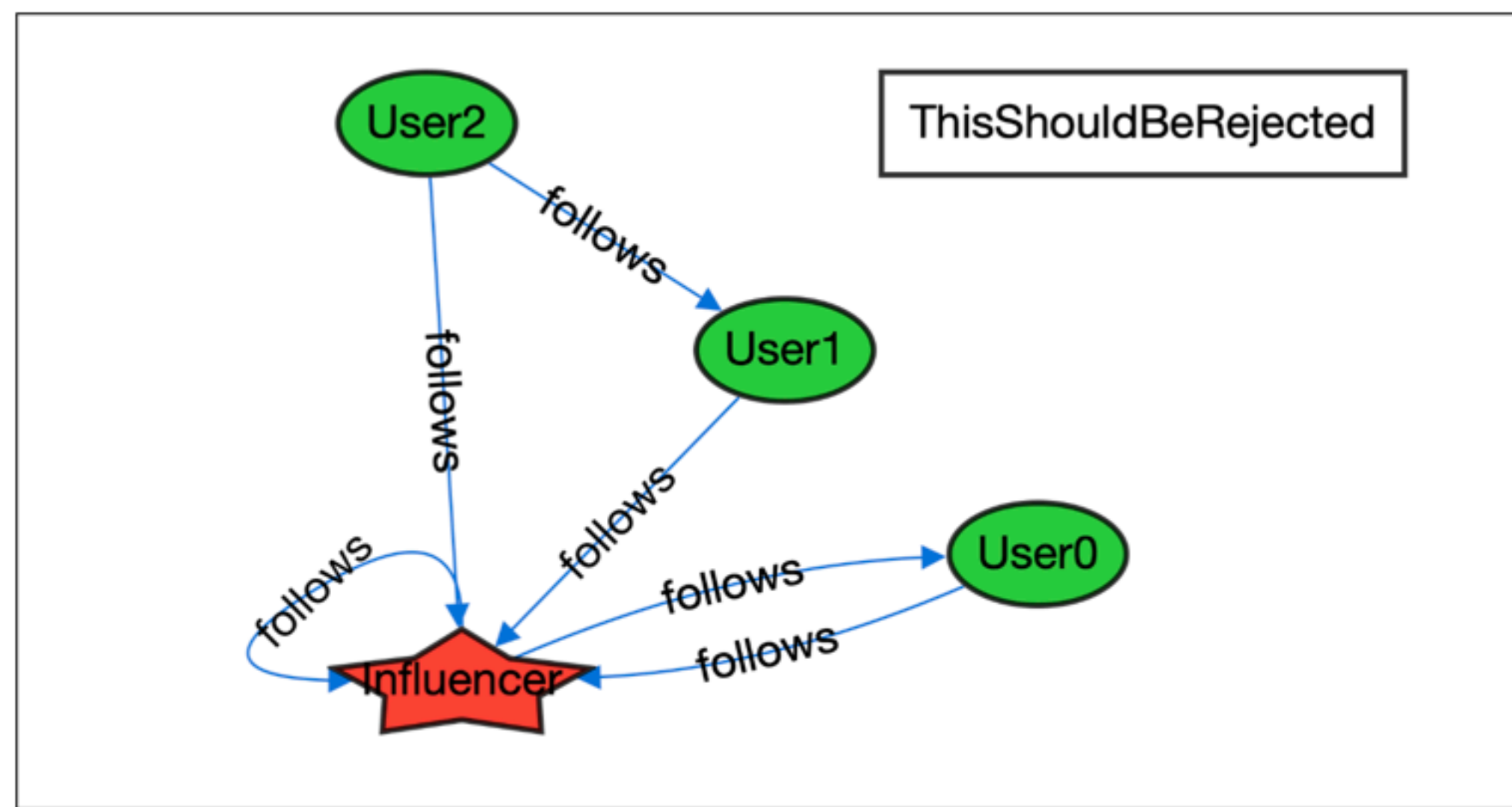
Next instance

Share instance

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).

Privacy Policy: Alloy4Fun logs all user interactions, including code and options, which may be analyzed and made public for research purposes. Do not share confidential information. However, Alloy4Fun does not collect any personally identifiable information.

Counter-example found. check Spec is invalid.

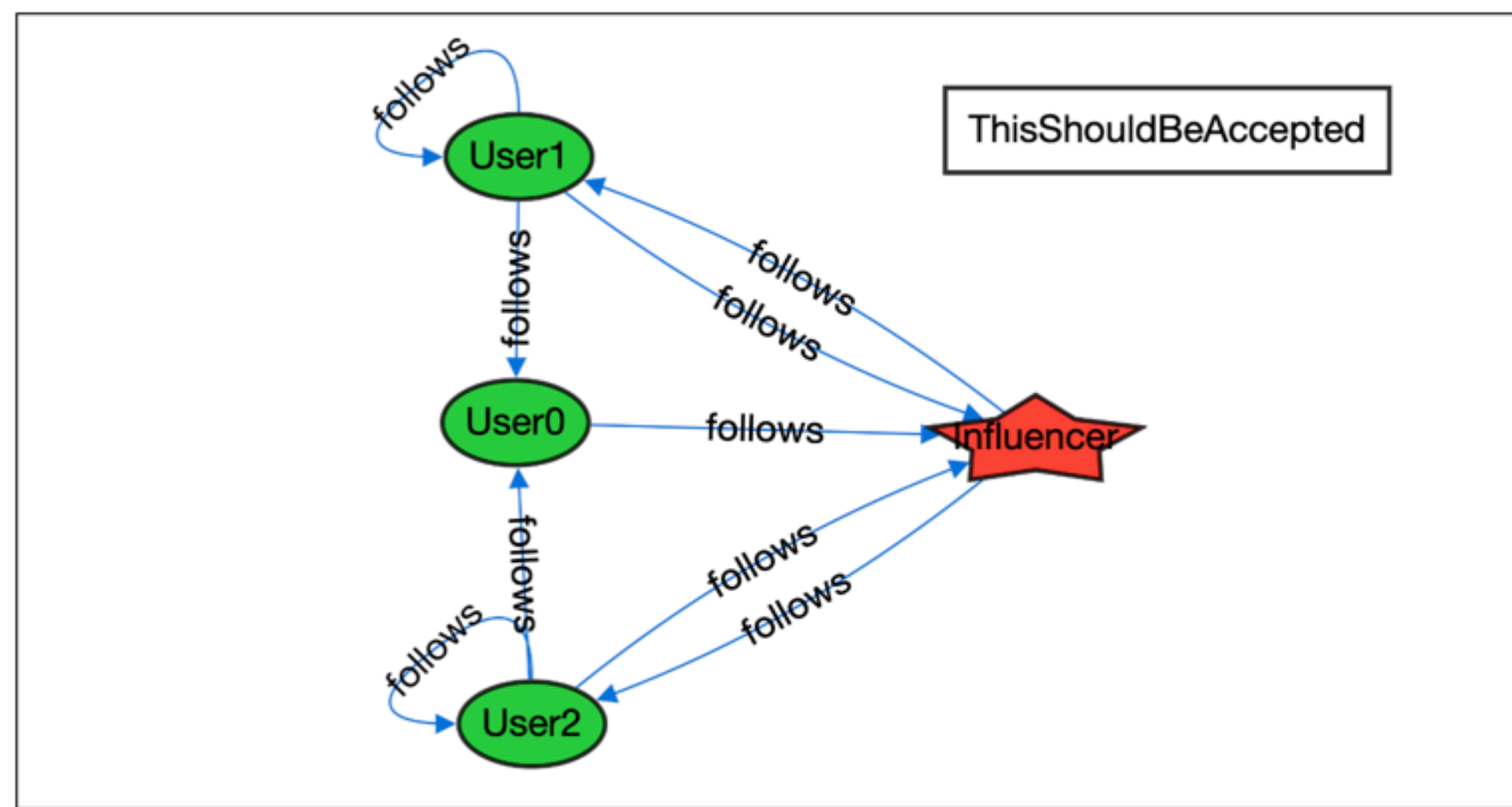


Navigation controls: Previous instance (left arrow), Next instance (right arrow), and Share instance (link icon).

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).

Privacy Policy: Alloy4Fun logs all user interactions, including code and options, which may be analyzed and made public for research purposes. Do not share confidential information. However, Alloy4Fun does not collect any personally identifiable information.

Counter-example found. check Spec is invalid.



Previous instance

Next instance

Share instance

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).

Privacy Policy: Alloy4Fun logs all user interactions, including code and options, which may be analyzed and made public for research purposes. Do not share confidential information. However, Alloy4Fun does not collect any personally identifiable information.

```
sig User { follows : set User }  
sig Influencer extends User {}
```

```
pred Spec { // Influencers are followed by everyone else  
  
}
```

```
//SECRET
```

```
pred Oracle { all u : Influencer | follows.u = User - u }
```

```
//SECRET
```

```
abstract one sig RejectedBy {}
```

```
//SECRET
```

```
sig ThisShouldBeRejected, ThisShouldBeAccepted extends RejectedBy {}
```

```
//SECRET
```

```
check Spec {  
  (some ThisShouldBeRejected iff (Spec and not Oracle))  
  implies  
  (Spec iff Oracle)  
} for 4
```

Springer Nature | IEEE Xplore | MIT Libraries | ieeexplore.ieee.org

Aluminum: Principled scenario exploration through minimality

Conferences > 2013 35th International Conference on Information Visualization (InfoVis)

Publisher: IEEE

[Cite This](#) [PDF](#)

[Tim Nelson](#); [Salman Saghafi](#); [Daniel J. Dougherty](#); [Kathi Fisler](#); [Shriram Krishnamurthi](#) All Authors

53 Cites in Papers | **148** Full Text Views

Abstract: Scenario-finding tools such as Alloy are widely used to understand the consequences of specifications, with applications to software modeling, security analysis, and verification. This paper focuses on the exploration of scenarios: which scenarios are presented first, and how to traverse them in

More Like This

The Potential for Synergy between Information Visualization and Software Engineering Visualization
2008 12th International Conference Information Visualisation
Published: 2008

[Feedback](#)



Not Secure — alloy4fun.inesctec.pt



Counter-example found. check Spec is invalid.



Previous instance



Next instance



Share instance

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).

Privacy Policy: Alloy4Fun logs all user interactions, including code and options, which may be analyzed and made public for research purposes. Do not share confidential information. However, Alloy4Fun does not collect any personally identifiable information.

claude.ai

Influencer follower specification in Alloy Free plan · Upgrade Share

Consider the following Alloy model.

```
sig User { follows : set User }
sig Influencer extends User {}
```

Can you give me an english sentence that explains what is being formalised by the following constraint.

```
all u : Influencer | follows.u = User
```

10:58 < 2/2 >

The constraint says: **every influencer is followed by all users.**

Breaking it down — `follows.u` is the set of users who follow `u` (the relational transpose/join), and setting that equal to `User` means that set must be the entire `User` population, for every `u` that is an `Influencer`.

Write a message...

+ Sonnet 4.6

Claude is AI and can make mistakes. Please double-check responses.

```
sig User { follows : set User }  
sig Influencer extends User { }
```



Try changing this expression!

```
pred Spec { // Influencers are followed by everyone else  
  all u : Influencer | follows.u = User  
}
```

link.springer.com

SPRINGER NATURE Link

Log in

Find a journal Publish with us Track your research Search

Home > Empirical Software Engineering > Article

An empirical evaluation of pre-trained large language models for repairing declarative formal specifications

Open access | Published: 25 July 2025

Volume 30, article number 149, (2025) [Cite this article](#)

You have full access to this [open access](#) article

[Download PDF](#) [Save article](#)

[Download](#) [Save](#)

[Empirical Software Engineering](#)

[Aims and scope](#) [Submit manuscript](#)

[Sections](#) [Figures](#) [References](#)

[Abstract](#)

[Introduction](#)

[Background and Motivation](#)

[Methodology](#)

[Allow Analyzer Feedback Levels](#)

Mohannad Alhanahnah, Md Rashedul Hasan, Lisong Xu & Hamid Bagheri

2671 Accesses 10 Citations [Explore all metrics](#)

Abstract

Automatic Program Repair (APR) has garnered significant attention as a



dl.acm.org

ACM DIGITAL LIBRARY | Association for Computing Machinery | Universidade do Minho | Browse | About | Sign in | Register






Journals | Magazines | Proceedings | Books | SIGs | Conferences | People | Search ACM Digital Library | Advanced Search

Conference | Proceedings | Upcoming Events | Authors | Affiliations | Award Winners


Home > Conferences > ICSE > Proceedings > ICSE-SEET '24 > Assessing the impact of hints in learning formal specification






RESEARCH-ARTICLE | OPEN ACCESS |  

Assessing the impact of hints in learning formal specification

Authors:  Alcino Cunha,  Nuno Macedo,  José Creissac Campos,  Iara Margolis,  Emanuel Sousa | [Authors Info & Claims](#)


ICSE-SEET '24: Proceedings of the 46th International Conference on Software Engineering: Software Engineering Education and Training
Pages 151 - 161 • <https://doi.org/10.1145/3639474.3640050>



Published: 24 May 2024 [Publication History](#) 





3 330     

Abstract

Background. Many programming environments include automated feedback in the form of hints to help novices learn autonomously. Some experimental studies investigated the impact of automated



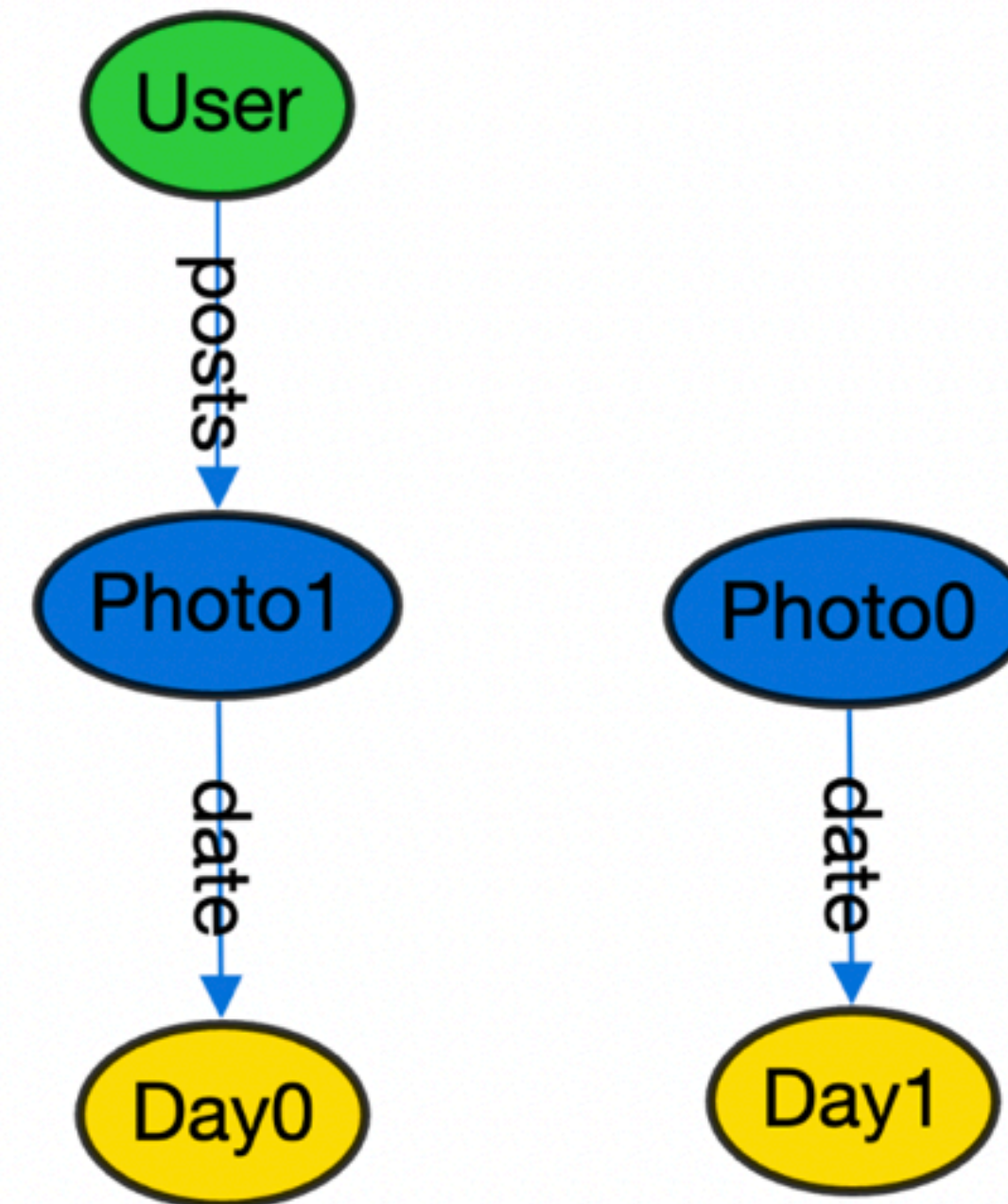
   

Fix the following incorrect specification:

`all p : Photo | lone posts.p`

The example below is allowed by the following incorrect specification but should be forbidden:

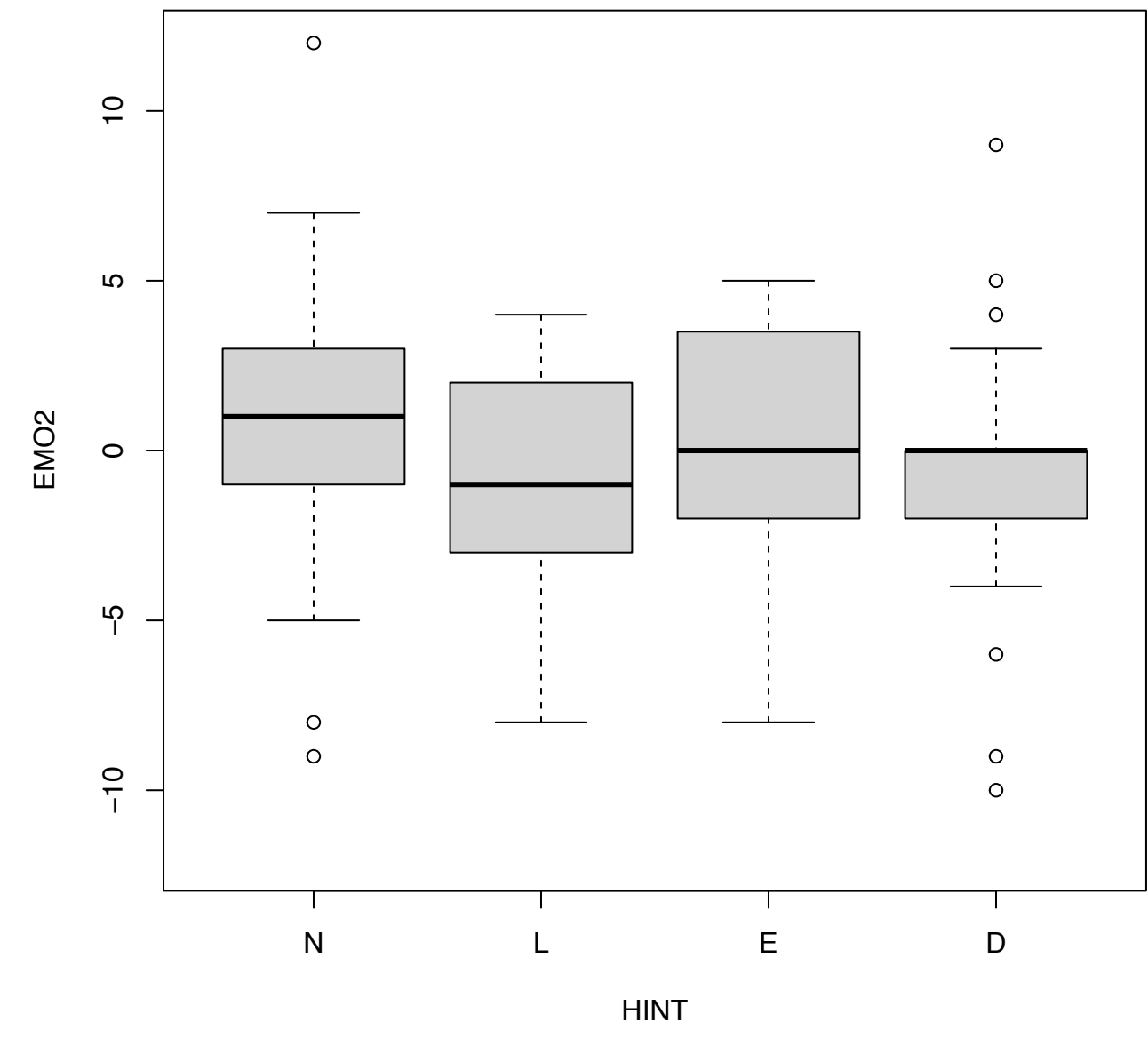
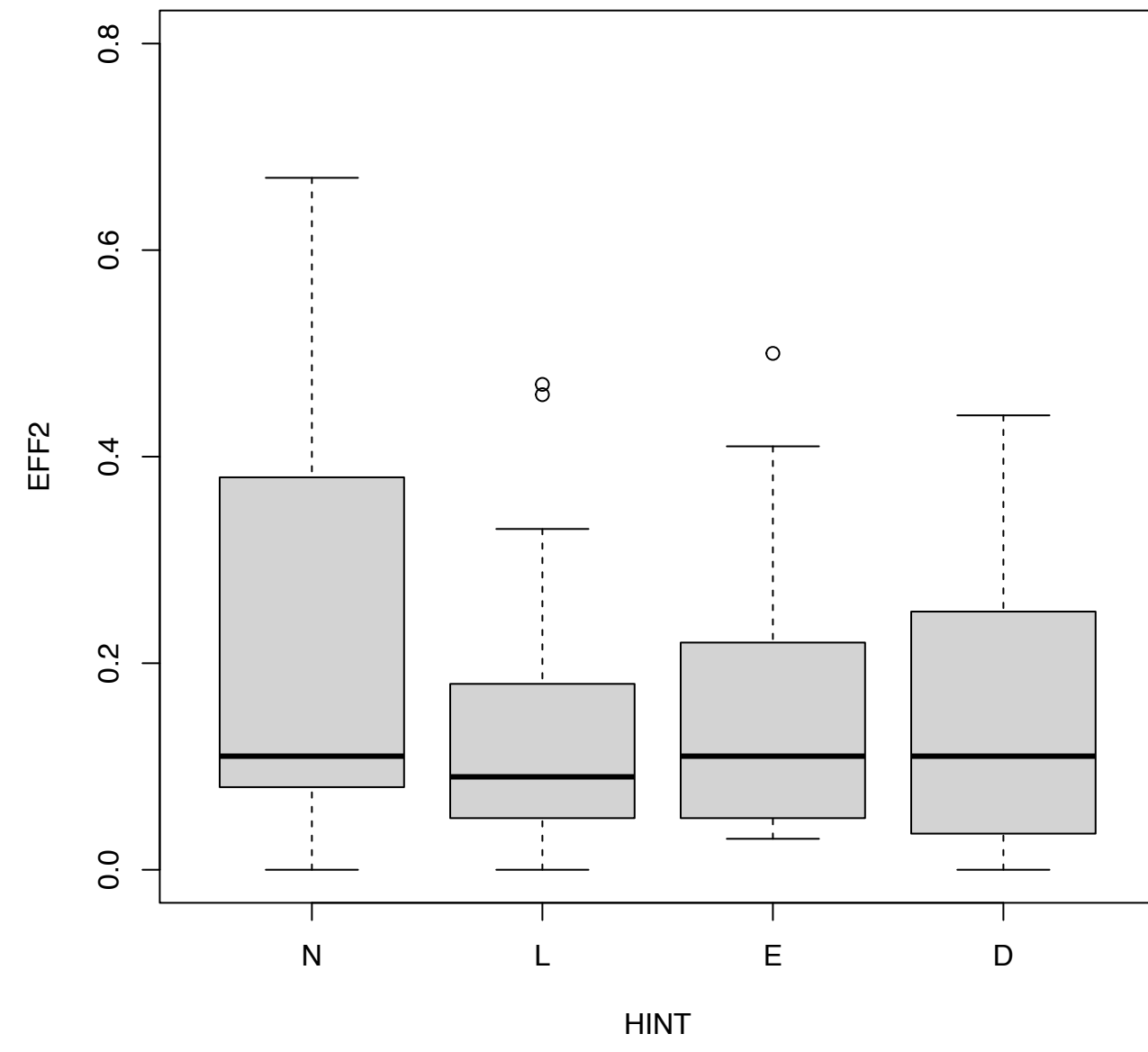
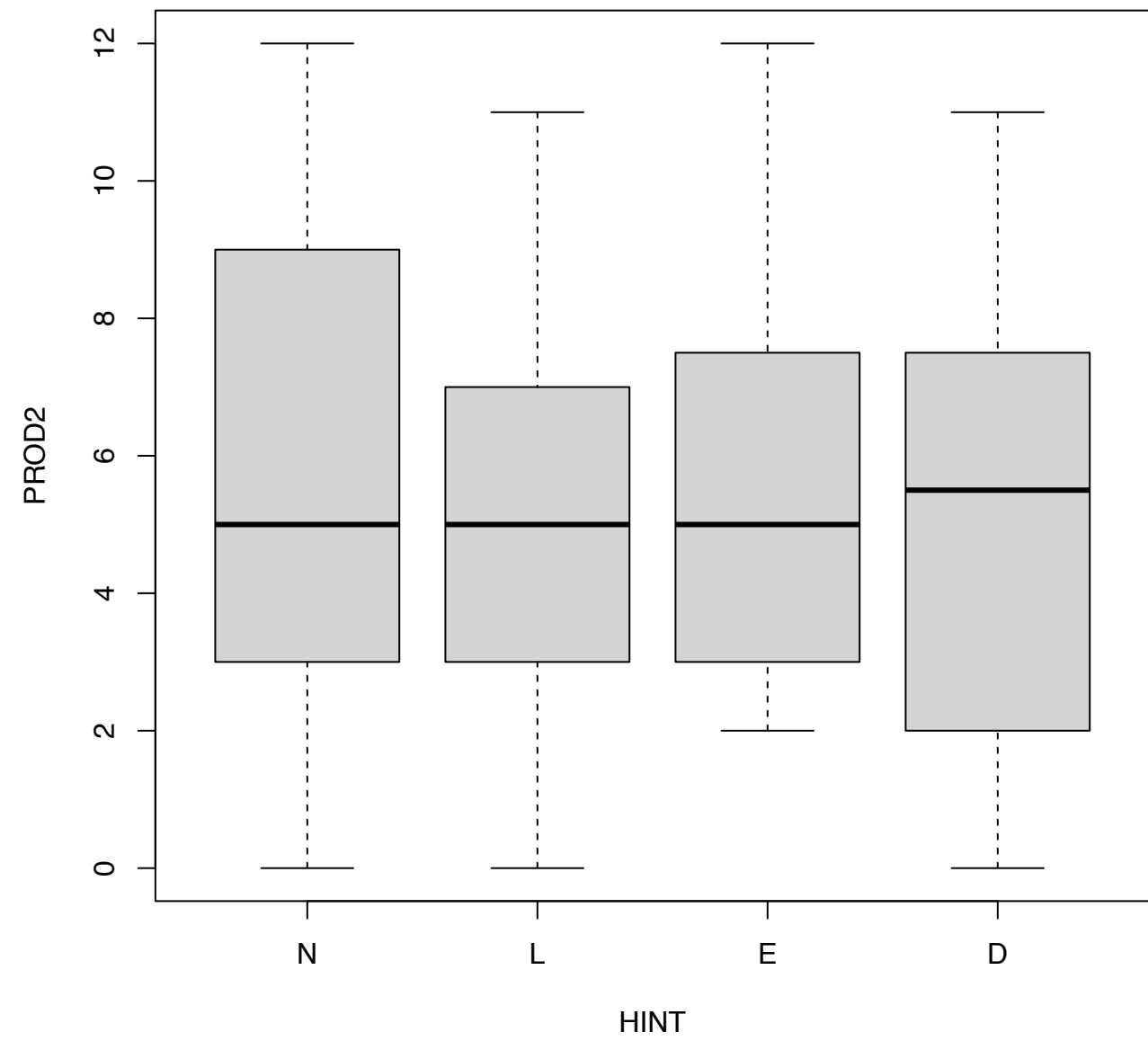
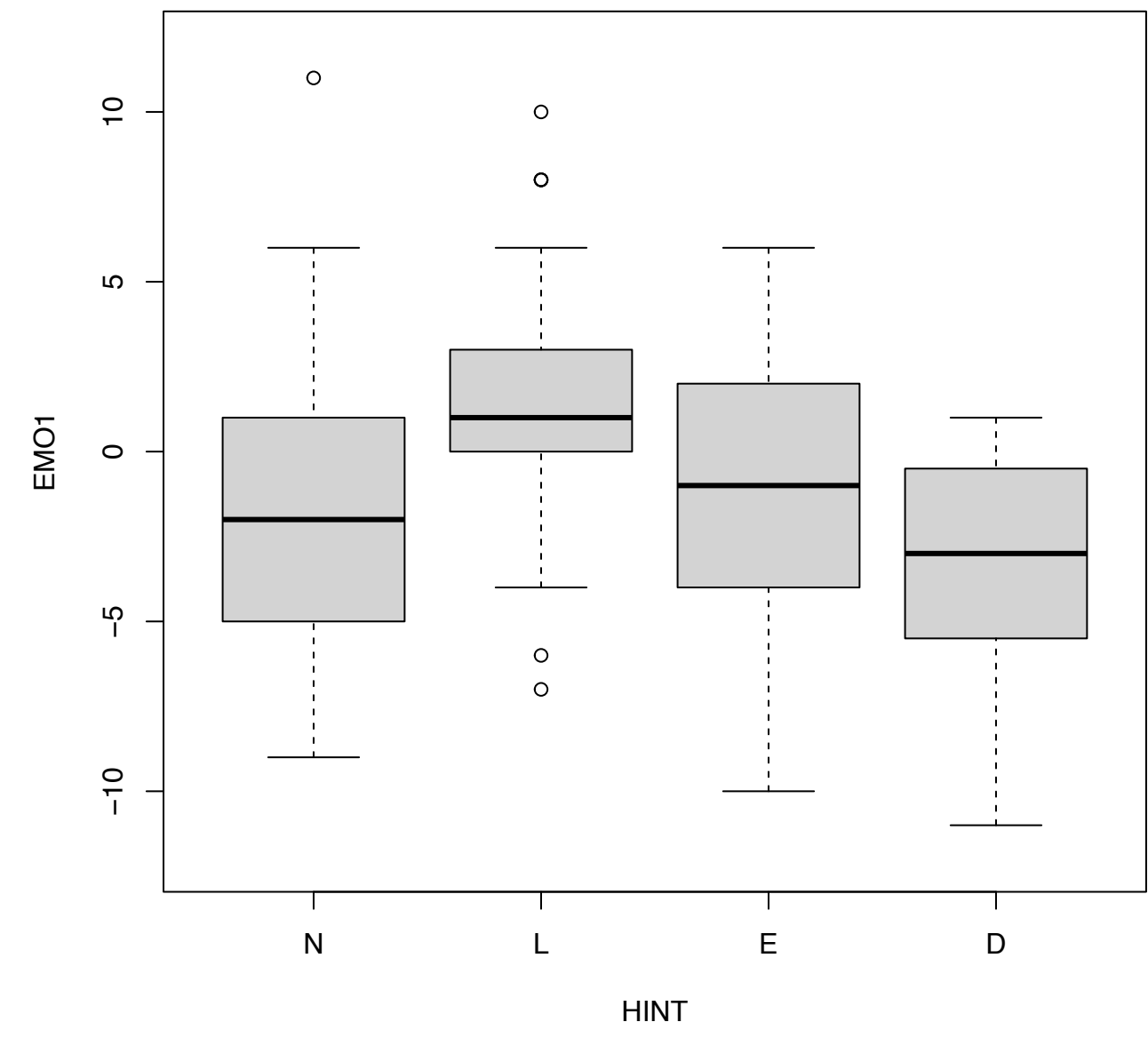
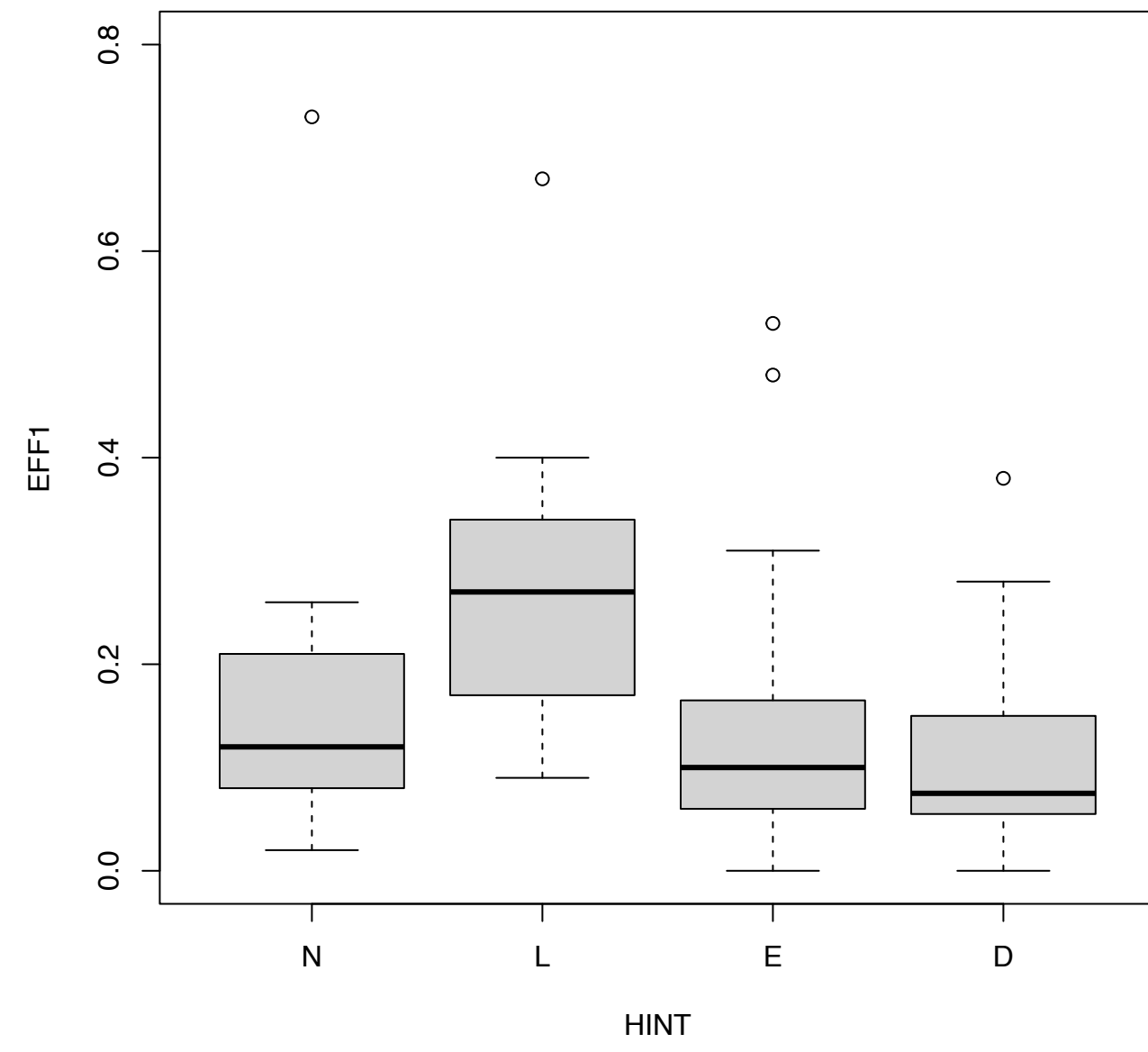
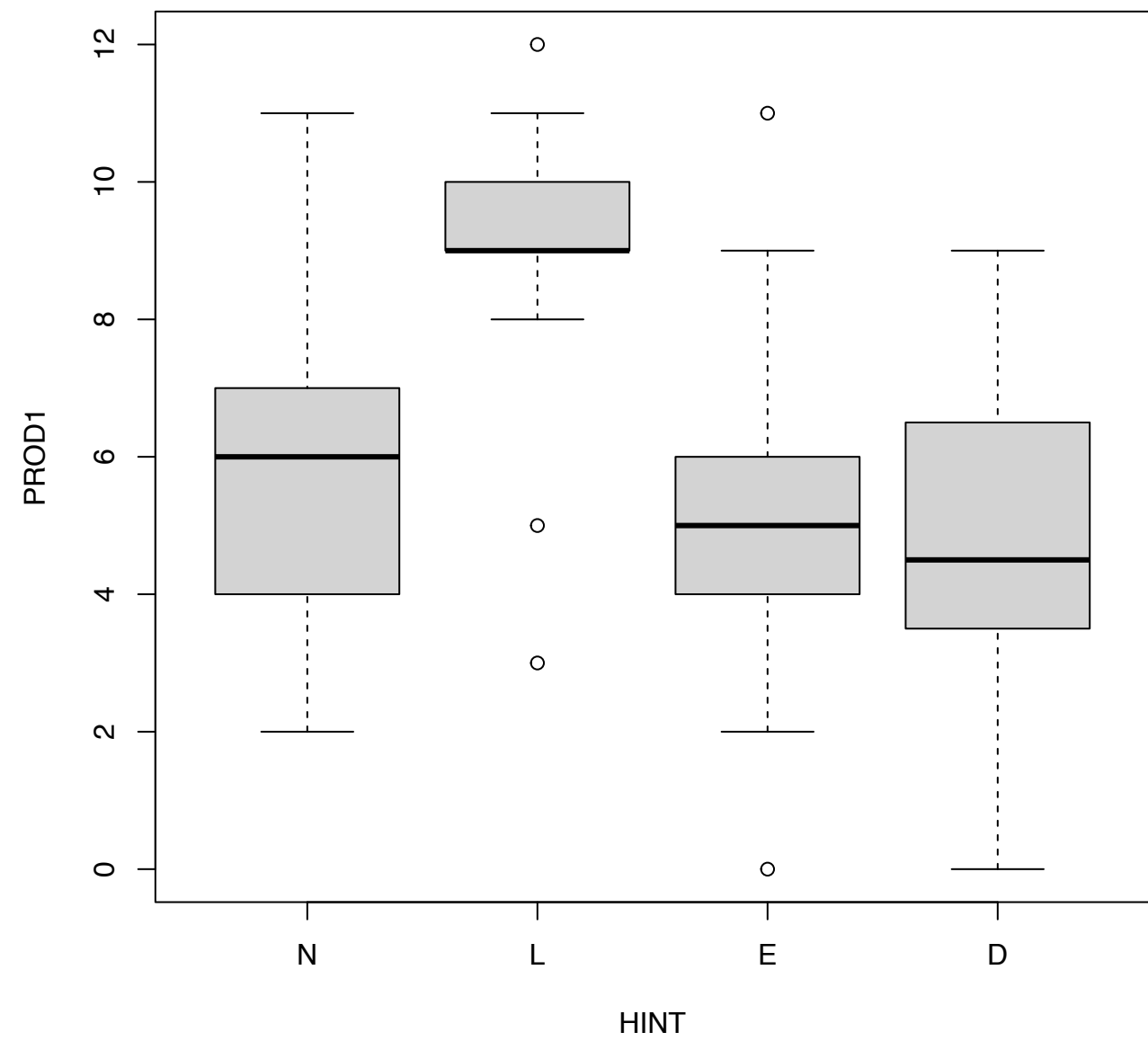
`all p : Photo | lone posts.p`



Change the highlighted operator in the following incorrect specification:

```
all p : Photo | lone posts.p
```

The following specification incorrectly states that every photo is posted by at most one user:
all p : Photo | lone posts.p



Collecting and mining data

Alloy4Fun

```
1 sig User {
2   follows : set User
3 }
4 sig Influencer extends User {}
5
6 pred Spec {
7   // Influencers are followed by everyone else
8 }
9
10 //SECRET
11 pred Oracle {
12   all u : Influencer | follows.u = User - u
13 }
14 //SECRET
15 check Spec {
16   Spec iff Oracle
17 } for 4
```



Execute



Share model



Statistics



Derivations



To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).

Privacy Policy: Alloy4Fun logs all user interactions, including code and options, which may be analyzed and

```

{
  "_id": "vqBvekqeWwturoCu9",
  "code": "sig User {\n    follows : set User \n}\n\nsig Influencer extends User {}\n\npred Spec {\n    // Influencers are followed by everyone else\n\n}\n\n//SECRET\n\npred Oracle {\n    all u : Influencer | follows.u = User - u \n}\n\n//SECRET\n\ncheck Spec { \n    Spec iff Oracle \n} for 4",
  "derivationOf": null,
  "children": [
    {
      "_id": "MeN6ByQPq7rRyBNft",
      "code": "sig User {\n    follows : set User \n}\n\nsig Influencer extends User {}\n\npred Spec {\n    // Influencers are followed by everyone else\n\n\tall u : Influencer | follows.u = User\n}\n",
      "derivationOf": "vqBvekqeWwturoCu9",
      "children": [
        {
          "_id": "MoQM7CvHa3TkRFNAR",
          "code": "sig User {\n    follows : set User \n}\n\nsig Influencer extends User {}\n\npred Spec {\n    // Influencers are followed by everyone else\n\n\tall u : Influencer | follows.u = User - u\n}\n",
          "derivationOf": "MeN6ByQPq7rRyBNft",
          "children": []
        }
      ]
    }
  ]
}

```

Alloy4Fun

```
1 sig User {
2   follows : set User
3 }
4 sig Influencer extends User {}
5
6 pred Spec {
7   // Influencers are followed by everyone else
8 }
9
10 //SECRET
11 pred Oracle {
12   all u : Influencer | follows.u = User - u
13 }
14 //SECRET
15 check Spec {
16   Spec iff Oracle
17 } for 4
```



Execute



Share model



Statistics



Derivations

To get started with Alloy4Fun check out this [introduction](#), along with some example challenges. More information about Alloy is available at [AlloyTools](#).

Privacy Policy: Alloy4Fun logs all user interactions, including code and options, which may be analyzed and



Not Secure — alloy4fun.inesctec.pt



Alloy4Fun

Statistics consider *challenges* all *check* commands in the *root* model. Unless explicitly stated, executions statistics include *all* commands, including those that are introduced by the user in a session. Recall that for challenges *correctness* is unsatisfiability. Statistics are updated once per day.

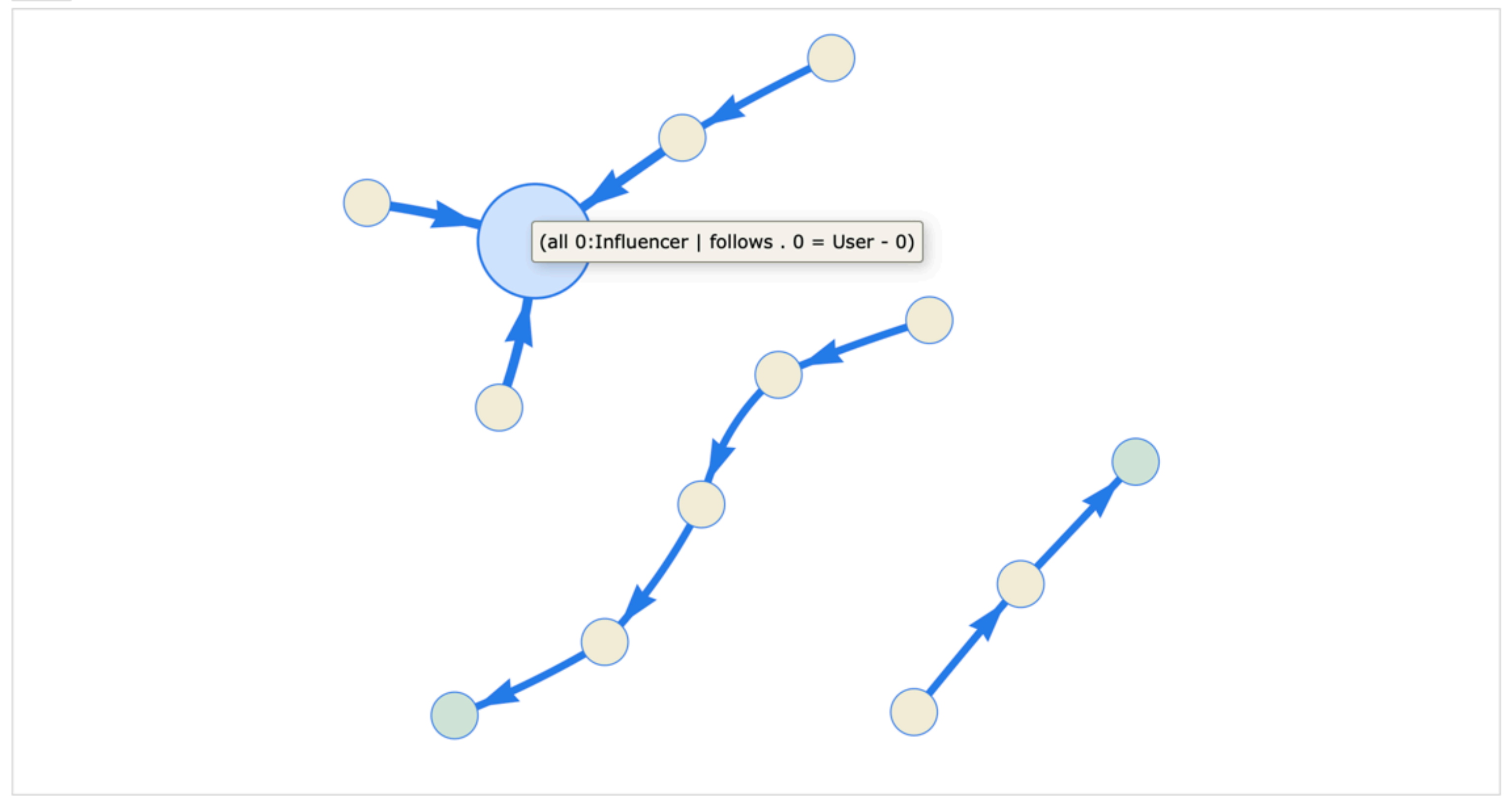
Challenge name	unknown	Extraction time	2026-05-07T12:18:16.550785
# sub-challenges	1	Metric catalog	Simple online metrics
Total sessions ⓘ	5	Sessions by # solved challenges ⓘ	5
7	Average session ⓘ	3.4	Longest session ⓘ
0.37	Average length all solved ⓘ	3.4	Average % unsatisfiable ⓘ
17	Challenge executions ⓘ	5	Sessions all solved ⓘ
10	Unsat executions ⓘ	15	Total executions ⓘ
2	Compile-time errors ⓘ	5	Sat executions ⓘ
1	Total shares ⓘ	2	Error executions ⓘ
1	Number of iterations ⓘ	1	Total warnings ⓘ
0		0	Shared models ⓘ
			Average iterations ⓘ

Sessions by length ⓘ



Submission graph ⓘ

Spec






Full derivation tree

zenodo

Published October 2025 | Version MFES 24/25 Dataset Open

Alloy4Fun Dataset for 2024/25

Nuno Macedo¹ ; Alcino Cunha² ; Ana C. R. Paiva¹ 

[Show affiliations](#)

This dataset contains models submitted by students in the Alloy4Fun platform to solve the challenge models from various editions of formal methods courses in the University of Minho (UM) and the University of Porto (UP) between the fall of 2019 and the spring of 2025, totalling around 185.000 entries. Participants include those enrolled in the optional MSc course "Specification and Modelling" (EM) and the mandatory MSc course "Formal Methods in Software Engineering" (MFES) in UM, and the optional MSc course "Formal Methods for Critical Systems" (MFS) in UP. Note that since the challenges' permalinks are publicly available, the dataset may contain submissions from other participants outside the classroom context.

The analysis of the 2021 dataset is reported in the Science of Computer Programming paper "Experiences on Teaching Alloy with an Automated Assessment Platform" (extending the ABZ'20 conference version analysing the 2020 dataset).

Name	Permalink	Courses (Students)	Entries
Trash FOL	sDLK7uBCbgZon3znd	EM 19/20 (~20) & 20/21 (~20), MFS 21/22 (~10) & 22/23 (~10)	4333
Classroom FOL	YH3ANm7Y5Qe5dSYem	EM 19/20 (~20) & 20/21 (~20), MFS 21/22 (~10) & 22/23 (~10)	5106
Trash RL	PQAJE67kz8w5NWJuM	EM 19/20 (~20) & 20/21 (~20)	4929
Classroom RL	zRAn69AocpkmxXZnW	EM 19/20 (~20) & 20/21 (~20)	6813

Reading Assistant

- Outline
- Highlights
- Abstract
- Keywords
- 1. Introduction
- 2. Alloy4Fun overview
- 3. Formalizing Alloy4Fun with Alloy
- 4. Writing and mining challenges
- 5. Experiences on teaching with Alloy
- 6. Concluding remarks and future work
- CRedit authorship contribution statement
- Declaration of Competing Interest
- Acknowledgements
- References
- Show full outline

Home > Software Engineering and Formal Methods > Conference paper

Right or Wrong – Understanding How Users Write Software Models in Alloy

Conference paper | First Online: 26 November 2024

pp 309–327 | [Cite this conference paper](#)

Access provided by Massachusetts Institute of Technology

Download book PDF

Download book EPUB

Save conference paper

Ana Jovanovic & Allison Sullivan

Part of the book series: [Lecture Notes in Computer Science](#) ((LNCS, volume 15280))

Included in the following conference series: [International Conference on Software Engineering and Formal Methods](#)



Software Engineering and Formal Methods

(SEFM 2024)

- Sections
- Figures
- References

- Abstract
- Introduction
- Background
- Experiment SetUp
- Empirical Evaluation

Cornell University

arXiv > cs > arXiv:2510.2

Computer Science > Software

[Submitted on 27 Oct 2025 (v1), last revis

Validating Formal S

Alcino Cunha, Nuno Macedo

Validation is a central activity wh
to define upfront test cases or sc
cases is burdensome and error p
an empirical evaluation of using
natural language requirements. I
formalized in the Alloy specificat
from other closed- and open-so
effective at generating positive (a
requirement, and that can detect

Subjects: **Software Engineering** (cs.
ACM classes: D.2.1; D.2.4; D.2.5
Cite as: arXiv:2510.23350 [cs.SE]
(or arXiv:2510.23350v2 [cs.
https://doi.org/10.48550/

Submission history

From: Nuno Macedo [view email]
[v1] Mon, 27 Oct 2025 14:02:20 UTC
[v2] Wed, 18 Feb 2026 16:04:20 UTC

Bibliographic Tools Code, Da

Bibliographic and Citation

IEEE.org IEEE Xplore IEEE SA IEEE Spectrum More Sites Donate Cart Create Account Personal Sign In

IEEE Xplore® Browse My Settings Help

MIT Libraries Access provided by: MIT Sign Out

IEEE

All [Search Icon]

ADVANCED SEARCH

Conferences > 2025 40th IEEE/ACM Internatio... ?

Automated Combinatorial Test Generation for Alloy

Publisher: IEEE

Cite This PDF

Agustín Borda ; Germán Regis ; Nazareno Aguirre ; Marcelo Frias ; Pablo Ponzio All Authors

26 Full Text Views

Abstract

Document Sections

Introduction

Abstract: Specifications are an essential component of software development, and getting specifications right, especially formal specifications, can be very challenging. While the use of tools such as model finders and model checkers can be very effective for specification analysis through property checking, researchers have also realized that by the explicit provision of

HIGH PERFORMANCE COMPUTING Technologies, Solutions to Exascale Systems, and Beyond eLearning Course Program

LEARN MORE IEEE

More Like This

Software Development Management Planning



IEEE Transactions on Software Engineering Published: 1984

Cellular automata-based Feedback

zenodo

Published October 20, 2025 | Version MFES 24/25 Dataset Open

Alloy4Fun Semantic Entries

Macedo, Nuno¹ ; Cunha, Alcino² 

[Show affiliations](#)

This dataset groups together entries from the [Alloy4Fun dataset](#) according to their semantic and syntactic equivalence. Each entry is a student attempt to formalize a requirement (i.e., a challenge). Each challenge is a command that tests the equivalence of a student predicate against a hidden oracle. This grouping reduces ~51.000 entries into ~7.000 semantically distinct specifications (and ~17.000 syntactically distinct).

Semantic equivalence is tested with the Alloy Analyzer (with a default scope of 3 for top-level signatures).

Syntactic equivalence is tested after a simple normalization (variable anonymization and operator commutativity).

Each file `original_cmd.json` contains the information regarding a challenge represented by command labeled `cmd` of an Alloy4Fun exercise whose model has id `original`, namely:

- **original**: the ID of the original model in the Alloy4Fun database
- **cmd**: the challenge command label
- **pred**: the challenge predicate label
- **entry_count**: the total number of entries for the challenge
- **semantic_count**: the number of semantically distinct entries
- **syntactic_count**: the number of syntactically distinct entries
- **scope**: the scope used to test equivalence
- **groups**: the actual semantic groups, containing:
 - **entry_count**: the total number of entries in the semantic group
 - **syntactic_count**: the number of syntactically distinct entries
 - **correct**: whether a correct solution to the challenge
 - **elems**: the actual syntactic groups, containing:
 - **entry_count**: the total number of entries in the syntactic group
 - **code**: a representative code submission

The following Alloy4Fun exercises were considered

Conclusion

Some lessons

- Students like the gamification aspects of challenges

Some lessons

- Students like the gamification aspects of challenges
- Feedback that is good for experts is not necessarily good for novices

Some lessons

- Students like the gamification aspects of challenges
- Feedback that is good for experts is not necessarily good for novices
- Too many challenges with automatic feedback may be prejudicial

Some lessons

- Students like the gamification aspects of challenges
- Feedback that is good for experts is not necessarily good for novices
- Too many challenges with automatic feedback may be prejudicial
- Open data is awesome for research

The future

- Move from “specification challenges” to “validation challenges”

The future

- Move from “specification challenges” to “validation challenges”
- Rethink all the formal methods teaching approach in the AI era

Thanks!

This work is funded by national funds through FCT – Fundação para a Ciência e a Tecnologia, I.P., under the support UID/50014/2025 (<https://doi.org/10.54499/UID/50014/2025>).