Alloy repair hint generation based on historical data

Ana Barros
U.Porto, Faculty of Engineering
INESC TEC

Henrique Neto
University of Minho
INESC TEC

Alcino Cunha
University of Minho
INESC TEC

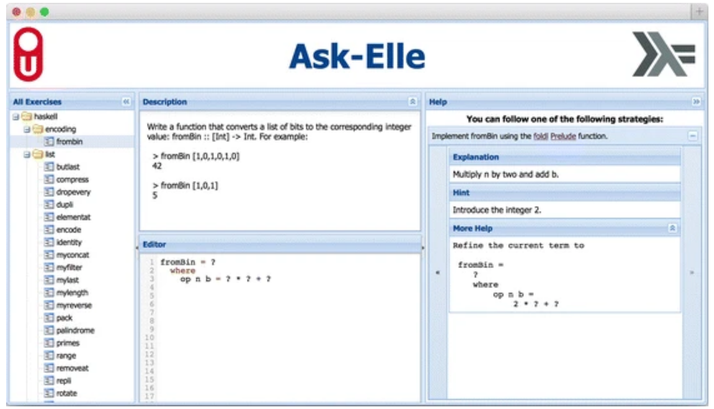
Nuno Macedo
U.Porto, Faculty of Engineering
INESC TEC

Ana C. R. Paiva
U.Porto, Faculty of Engineering
INESC TEC

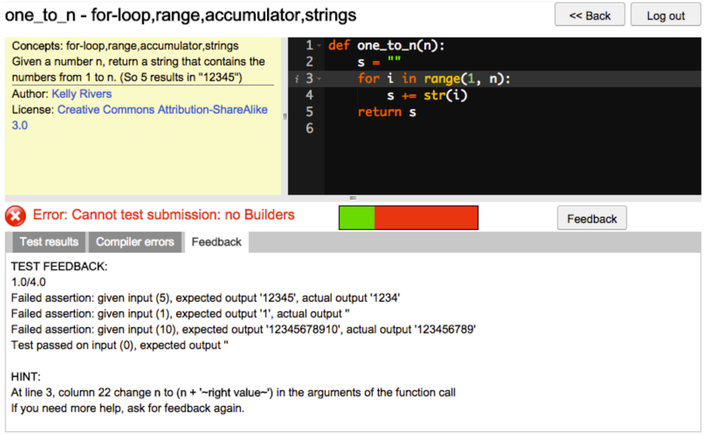
FM'24
September 2024, Milan, Italy

Learning Programming

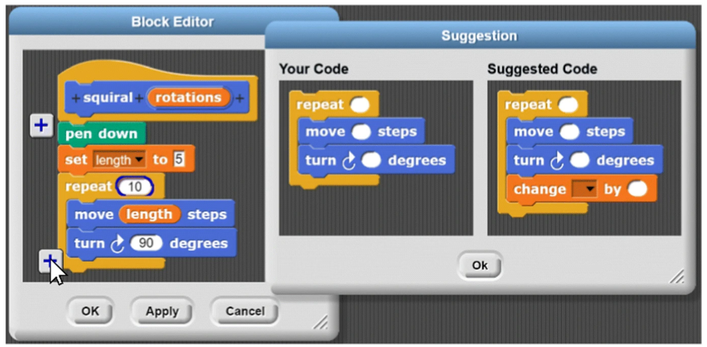
- Programming tutoring systems have long employed **automated hints**
- Many user studies on learning impact, conclusions depend on **language** or **type of hints**



Ask-Elle, Haskell



ITAP@CodeWebs, Python



iSnap, Snap!

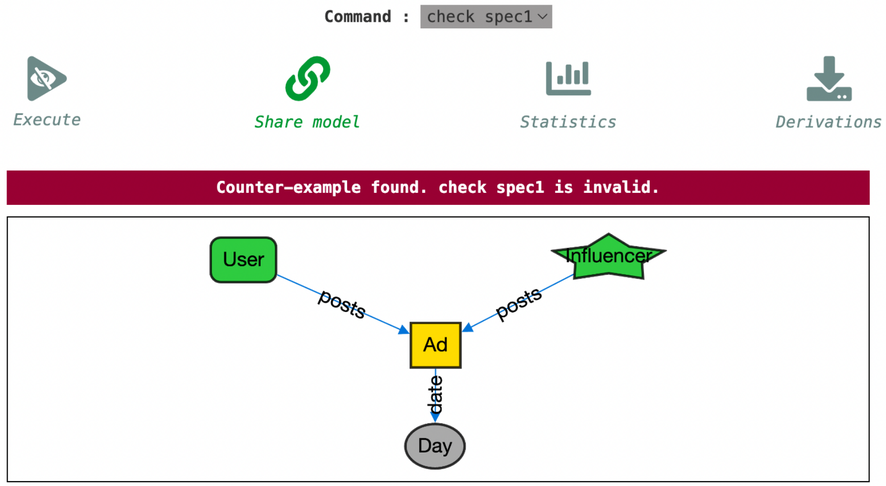
Learning Specification

- Platforms to support learning formal specification still incipient
- Being backed by solvers, counter-examples are the natural kind of feedback
- Study [ICSE-SEET'24] showed that for novices:
  - Counter-examples are not helpful
  - Next-step hints improve performance without reducing learning retention
- Can next-step hints from programming be adapted to the specification context?

```
sig User {
   follows : set User,
   posts   : set Photo
}
sig Influencer extends User {}
sig Photo {
   date    : one Day
}
sig Ad extends Photo {}
sig Day {}

/* Every photo is posted by one user. */
pred spec1 {
  all p:Photo | some u:User | u->p in posts
}
/* Influencers are followed by everyone else. */
pred spec2 {

}
```



Alloy4Fun, Alloy

Next-step Hints

- Given incorrect code, guide the student towards a correct version

Search-based

- Syntactic search for a correct solution
- Can be based on automated repair
- Effective when close to a solution
- Difficult to assess hint quality

Data-driven

- Follow previous successful attempts
- Distance from solution irrelevant, as long as previously submitted
- Hints mimic typical student behaviour
- Require extensive data set of submissions

Next-step Hints

- Given incorrect code, guide the student towards a correct version

Search-based

- Syntactic search for a correct solution
- Can be based on automated repair
- Effective when close to a solution
- Difficult to assess hint quality

TAR

SEFM'22

Data-driven

- Follow previous successful attempts
- Distance from solution irrelevant, as long as previously submitted
- Hints mimic typical student behaviour
- Require extensive data set of submissions

Teaching Alloy with Alloy4Fun

- Specification challenges:
  - *given an Alloy model …*
  - *… write a specification for the following requirement*
- Correctness checked by solving:
  - shows witness of inconsistency

```
sig User {
  follows : set User,
  posts : set Photo }
sig Influencer extends User {}
sig Photo {
  date : one Day }
sig Ad extends Photo {}
sig Day {}

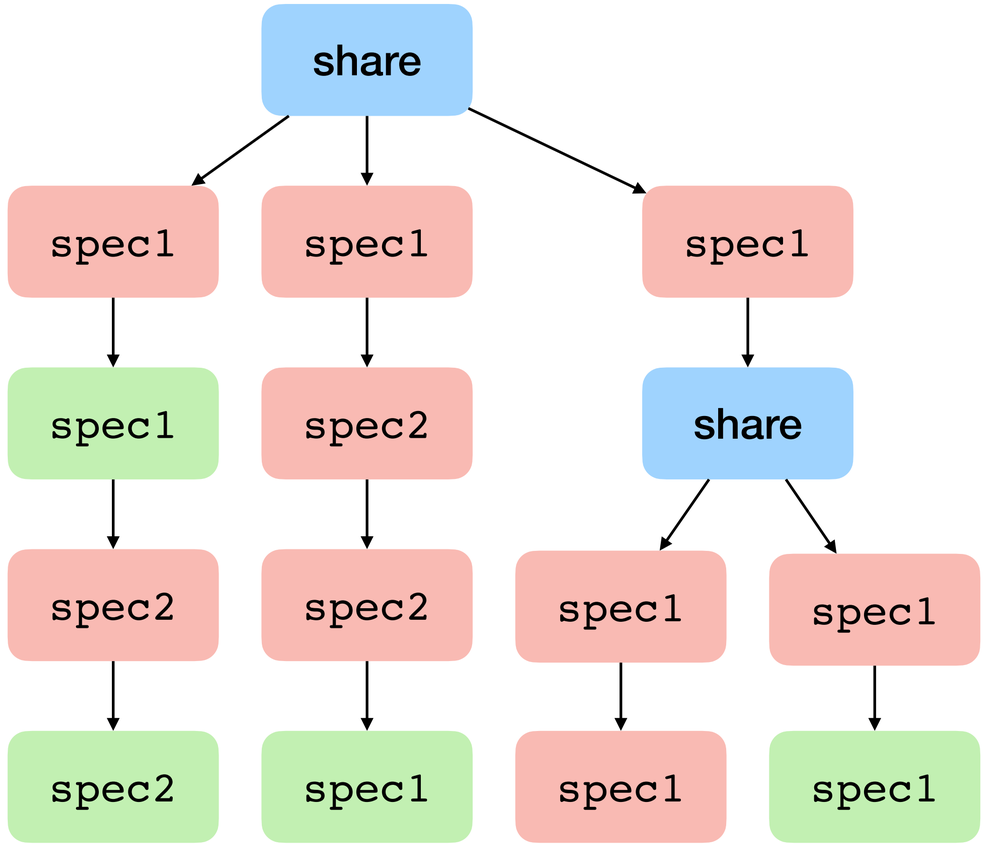
/* Every photo is posted by one user. */
pred spec1 { }
//SECRET
pred oracle1 { all p: Photo | one posts.p }
//SECRET
check spec1 { oracle1 iff spec1 } for 3

/* Influencers are followed by everyone else. */
pred spec2 { }
//SECRET
pred oracle2 { all u:User | Influencer-u in u.follows }
//SECRET
check spec2 { oracle2 iff spec2 } for 3
```

Teaching Alloy with Alloy4Fun

- Alloy4Fun stores every interaction
- Students can backtrack to previous attempts, a session is a tree
- Successful paths are branches that arrive at correct solutions
- We've been collecting data in our classes for several years

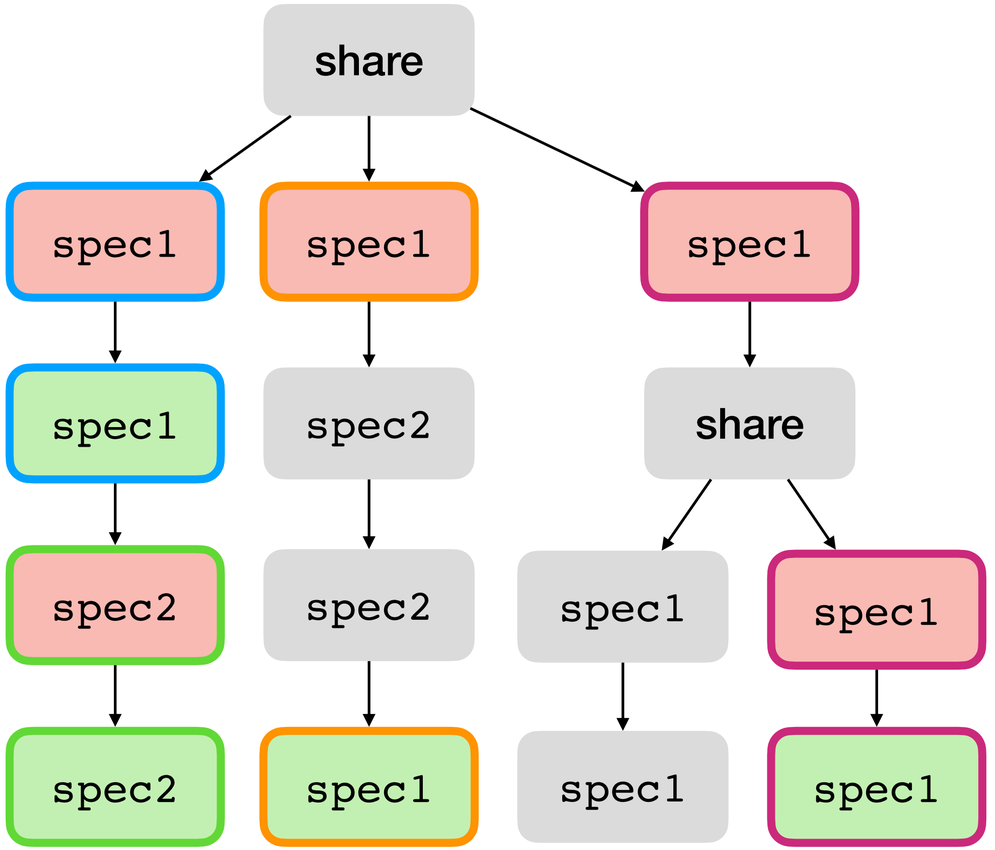
https://doi.org/10.5281/zenodo.8123547



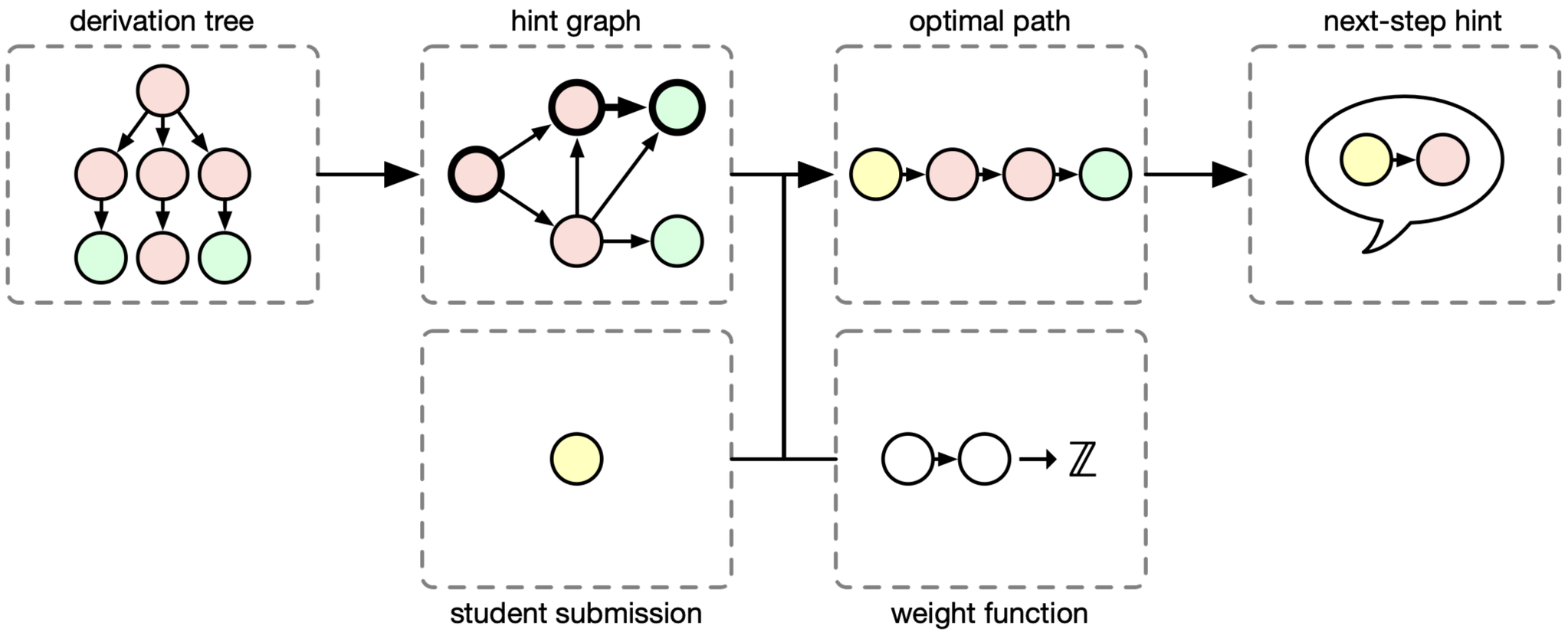
Teaching Alloy with Alloy4Fun

- Alloy4Fun stores every interaction
- Students can backtrack to previous attempts, a session is a tree
- Successful paths are branches that arrive at correct solutions
- We've been collecting data in our classes for several years

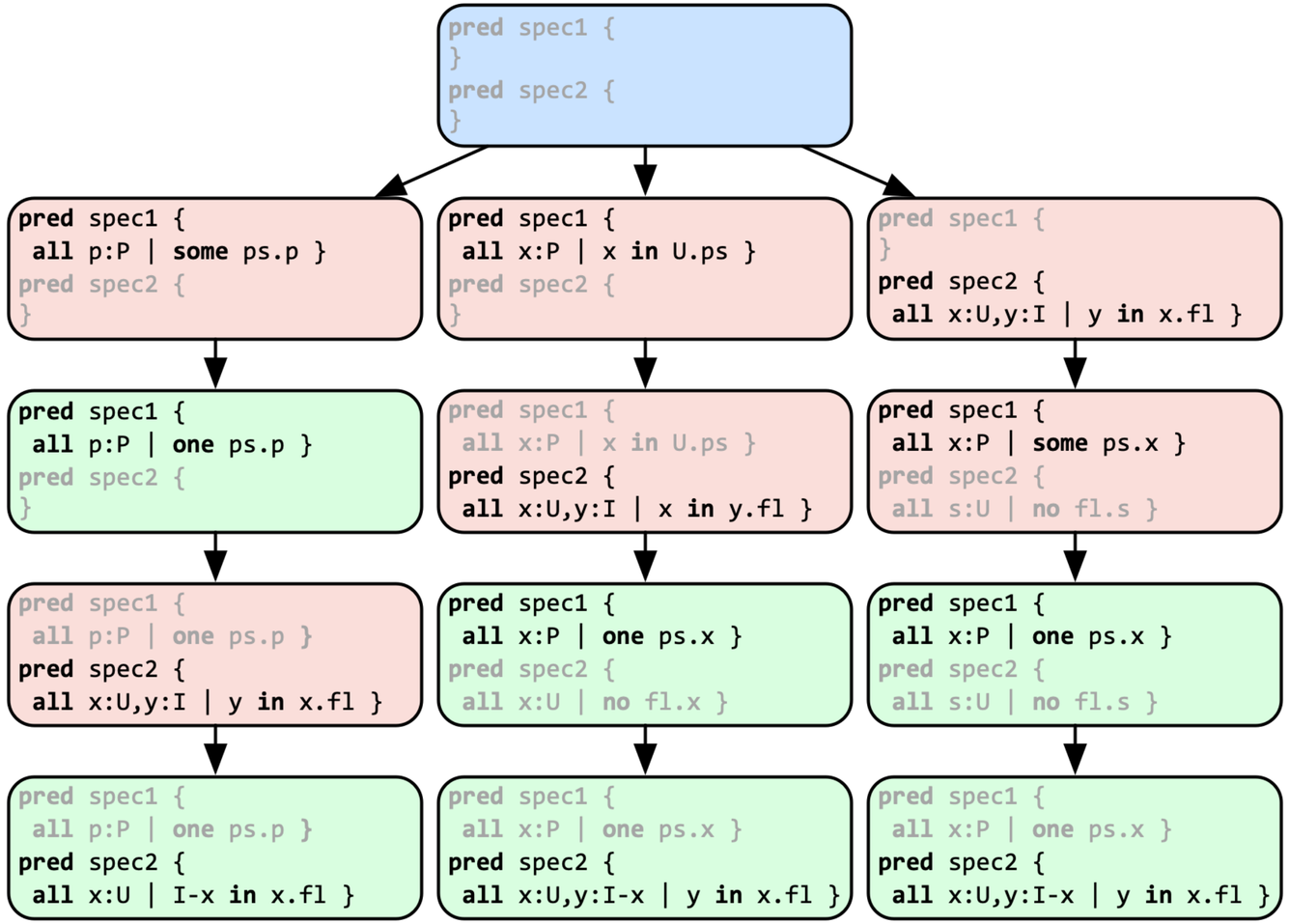
https://doi.org/10.5281/zenodo.8123547



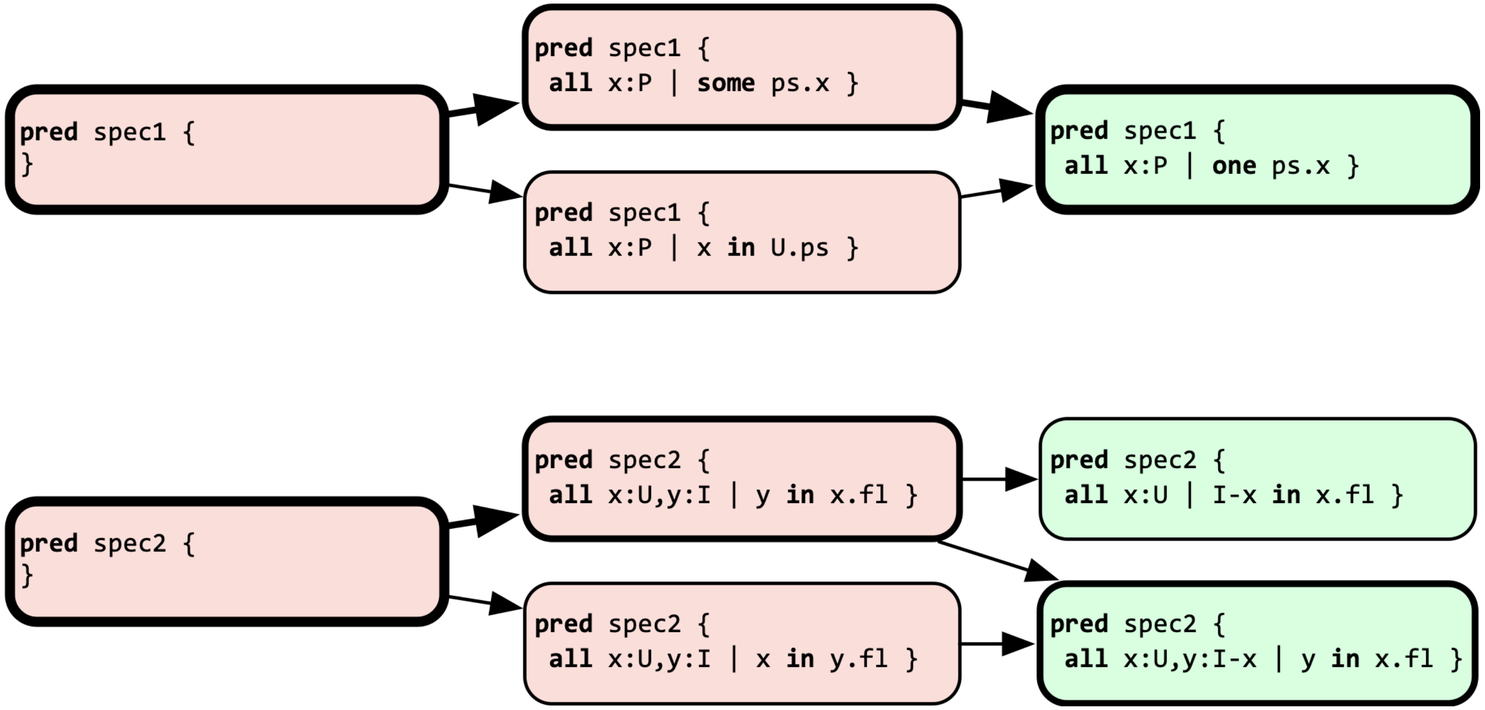
Overview


derivation tree
hint graph
optimal path
next-step hint
student submission
weight function
ℤ

Graph Construction



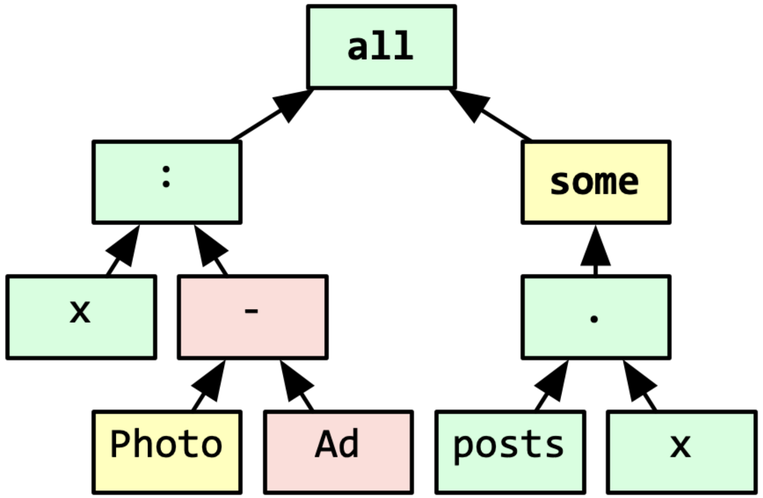
Graph Construction

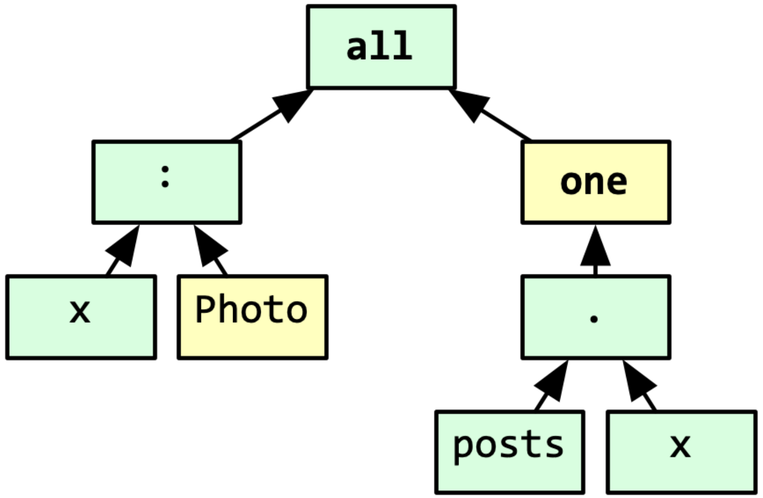


Optimal Next State

- Calculate the *optimal* path towards a correct state
- Policy based on a *metric* between two consecutive states
  - Syntactic attributes (complexity, …)
  - Popularity attributes (times submitted, …)
- The target state is the *next state* in the optimal path
  - Not necessarily a correct state

Hint Message Generation

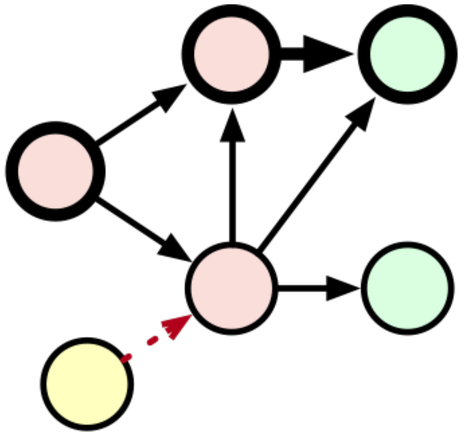




Keep going! It seems like you have unnecessary information in your expression. Try simplifying your expression by deleting the difference operator (-).

Handling Missing Hits

- Purely data-driven approaches fail for submissions not previously seen
- We implement a syntactic procedure to search for close existing nodes
- Based on mutations previous defined for the Alloy language



Deployment

```
1  sig User {
2     follows : set User,
3     posts   : set Photo
4  }
5  sig Influencer extends User {}
6  sig Photo {
7     date    : one Day
8  }
9  sig Ad extends Photo {}
10 sig Day {}
11
12 /* Every photo is posted by one user. */
13 pred spec1 {
14   all p:Photo-Ad | some posts.p
15 }
16 /* Influencers are followed by everyone else. */
17 pred spec2 {
18
19 }
```

Command : check spec1


Execute


Hint


Share model


Statistics


Derivations

Counter-example found. check spec1 is invalid.

Keep going! It seems like you have unnecessary information in your expression. Try simplifying your expression by deleting the difference operator (-).

Quantitative Evaluation

- Selected 12 exercises (125 challenges) from the Alloy4Fun dataset
- 66k (syntactically correct) submissions
- Split for graph construction (70%) and hint requests (30%)

Exercise	Chls.	Submission	Parsing
Social Network	8	22690	14943
Courses	15	22516	14911
Train Station	10	8158	6388
Production Line	10	8078	6058
Trash LTL	19	5279	4352
Classroom FOL	14	5893	4376
Classroom RL	14	6341	4248
Trash RL	9	4361	3059
Trash FOL	9	4092	2719
Graphs	7	3211	2481
Labelled TS	6	3382	2076
Curriculum Vitae	4	1199	854

Quantitative Evaluation

- Suitable hit rate, even without mutation-based enhancement
- Time to create the graphs suitable for frequent updates
- Time to return a hint negligible
- Better rate and performance than existing search-based approach

Graph Construction	Nodes		134
	Graph generation		7 s
	Policy application		1 s
Hint Request	**Data-driven**	**39%**	**0.01 s**
	Data-driven + Mutations	57%	0.58 s
	Search-based (TAR)	30%	30.22 s

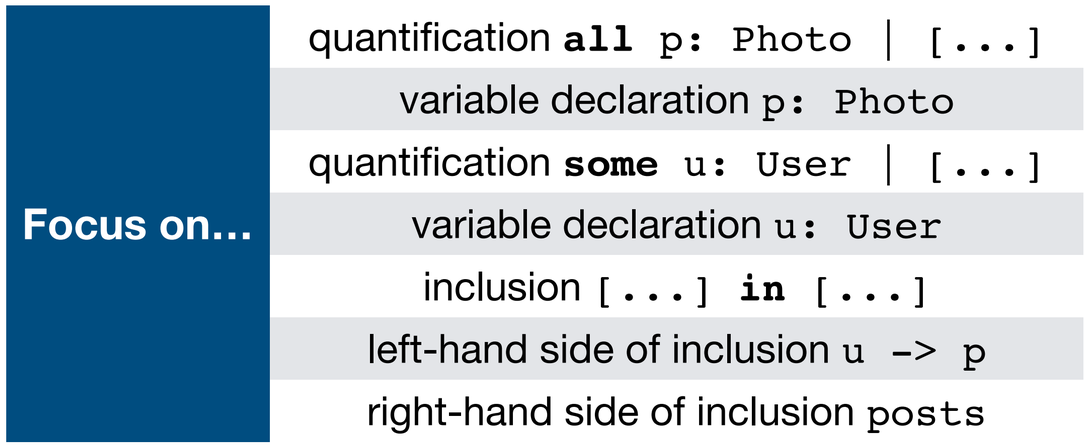
average per challenge

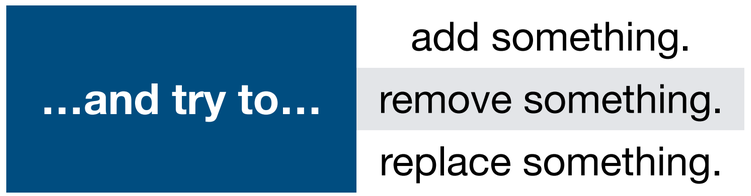
Qualitative Evaluation

- Asked Alloy instructors to write hints for 6 incorrect specifications
  - *Suggest to a student an **operation** on a **location***
- Received 8 replies

Incorrect spec for *“Every photo is posted by one user.”*

```
all p: Photo | some u: User | u -> p in posts
```





Qualitative Evaluation

- Compared with replies with those generated with different policies
- Best fitting policies used both syntactic and historical data
  - Generated hint was selected by at least an expert in all specifications

Policy	Any expt. Location	Most expt. Location	Any expt. Loc. + Op.	Most expt. Loc. + Op.
Edge complexity	5	3	4	2
Edge popularity	4	3	4	3
Edge complexity x Edge popularity	**6**	**5**	**6**	**3**

Conclusions

- First data-driven approach to hint generation for learning formal specification
- Implemented in Alloy4Fun and complemented with mutation-based component for missing hits
- With the right policy, can emulate hints provided by experts
- Experts also suggested other kinds of hints that may be worth exploring

Alloy repair hint generation based on historical data

Ana Barros
U.Porto, Faculty of Engineering
INESC TEC

Henrique Neto
University of Minho
INESC TEC

Alcino Cunha
University of Minho
INESC TEC

Nuno Macedo
U.Porto, Faculty of Engineering
INESC TEC

Ana C. R. Paiva
U.Porto, Faculty of Engineering
INESC TEC







This work is financed by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia within project EXPL/CCI-COM/1637/2021.

SpecRep, https://haslab.github.io/SpecRep/

