r/rust • 5 mo. ago
germandiago

What do you Rustaceans think of Dafny language?

I stepped by this language and since Rust makes a lot emphasis on memory safety and lack of undefined behavior, I would like to know your opinions on the features of Dafny language, which is a language that goes a step beyond in correctness via proofs: https://dafny.org/

What do you think of it?

19 23 Share

Add a comment

Sort by: Best Search Comments

PikachuKiiro • 5mo ago

Should probably be asking Coq or Lean users instead of rust.

44 Reply

regnskogen • 5mo ago

The way I see it Dafny and Rust have different and complementary goals. Personally, I don't like programming in Dafny or in any other theorem prover, automatic or otherwise. I think they (including Dafny) are neat for prototyping algorithms to figure out if your ideas work outside of your head. As the old saying goes: a constraint solver is a powerful tool to figure out which constraints you forgot to put in!

Writing large, practical and reusable systems in Dafny is somewhat like writing them in Brainfuck, except it's at least potentially useful.

memory-related incorrectness. You can still write logic bugs to your heart's content. For most systems this is a far better (in the sense of more fun and efficient with programmer time) trade off and will probably always be than going for provable correctness. I think this is a branding problem with Rust, where people go around calling it safe when "less unsafe" would be a lot closer to the truth.

28 Reply

germandiago OP • 5mo ago

I can imagine a subset of algorithms like STL or sinilar (sort, partition, stable sort) verified thorugh tools like this I guess.

7 Reply

5 more replies

matthieum • 5mo ago

I see Dafny as the opposite end of the spectrum:

- Rust is pragmatic first, with an emphasis on correctness.
- Dafny is correct first.

And I am afraid that for large programs, the lack of pragmatism makes Dafny untractable.

Instead, so far, I am more enthusiastic about opt-in correctness guarantees as provided by say, Prusti or Creusot.

See for example one of Creusot's demos:

```
extern crate creusot_contracts;

use creusot_contracts::{
    logic::{Int, OrdLogic, Seq},
    *,
};

#[predicate]
fn sorted_range<T: OrdLogic>(s: Seq<T>, l: Int, u: Int) -> bool {
    pearlite! {
        forall<i : Int, j : Int> l <= i && i < j && j < u ==> s[i] <= s[j]
    }
}

#[predicate]
fn sorted<T: OrdLogic>(s: Seq<T>) -> bool {
    sorted_range(s, 0, s.len())
```

```
#[ensures((^v)@.permutation_of(v@))]
pub fn gnome_sort<T: Ord + DeepModel>(v: &mut Vec<T>)
where
    T::DeepModelTy: OrdLogic,
{
    let old_v = snapshot! { v };
    let mut i = 0;
    #[invariant(sorted_range(v.deep_model(), 0, i@))]
    #[invariant(v@.permutation_of(old_v@))]
    while i < v.len() {
        if i == 0 || v[i - 1].le(&v[i]) {
            i += 1;
        } else {
            v.swap(i - 1, i);
            i -= 1;
        }
    }
}
```

It's a relatively lightweight set of annotations over otherwise regular code, which allows Creusot to prove invariants, and thereby the correctness (adherence to the specification).

The opt-in nature also means that when the specification becomes intractable -- expressing the invariants of an entire system is... challenging -- then one can just NOT opt-in, saving engineering time & compute-time (if the prover is even up to the task...).

This gives solid bricks, upon which to build a flexible system, and for now I'd argue it's a very sweet point of the spectrum.

21 Reply

WormRabbit • 5mo ago

Every theorem prover allows allows you to introduce unchecked assumptions. Basically like `unsafe` blocks in Rust: something that the compiler can't verify and trusts you to get right. In that sense the intractable cases are handled mostly the same. The difference is in the defaults: do we assume that most of our code is mechanically verifiable, such that we only need to introduce a few unchecked primitives, or do we assume that most of it is a vague mess with rare islands of formal correctness? Rust has bet on the former with memory safety, and succeeded. It's still up to debate whether this bet would pay off for more complex properties.

The big downside of formal systems based on SMT solvers, like Creusot, is that proof search is an entirely black box process. You throw a bunch of assumptions in, and hope that the computer would prove or disprove your claim. If it can do neither, it is entirely unclear why, and what assumptions may be missing. This could be better for general-purpose complex programs, where the programmer has only a vague idea why something could be correct, but it's a downside for more well-scoped

can guess your reasoning from generic assumptions.

12 Reply

3 more replies

1 more reply

whatever73538 • 5mo ago

MaxHaydenChiz • 5mo ago

Dafny is more well known because it is used by schools to teach, but in terms of industry usage Why3 is probably the relevant framework since a lot of embedded verification tools are built on turning Ada and C code into something Why3 can process.

The embedded Rust folks seem to be pretty determined to make something that can replace C and Ada, and I'm very much rooting for them.

In terms of the usage in certain industries, it is important that Rust, especially unsafe Rust, eventually gets similar capabilities. And I think that people would be better programmers if they understood, at least conceptually, what you need to assert and where in order to know thugs are correct. Most of the overhead in using verification tools in industry is bad tooling, poor library support, and the like.

The actual overhead is probably about 50%, but it turns into 3-5x because of all the tedious stuff and crappy tools. I'm hopeful that Rust can improve this situation since there's a big focus on good tools in general.

As for Dafny itself, what I don't like about Dafny is that it relies on a complicated collection of multiple external black box SMT solvers. Things could be fine for one version and then break on an update even with no changes to your code.

I'd *like* for the Rust version to do something similar to "Hammer" in Coq and Isabelle (which were the tools used to make the certified C compiler and the seL4 kernel respectively). Essentially, they record the "answer" from the SMT solver in a way that can be quickly checked by a smaller / simpler theorem prover.

So you have a smaller base of trusted code (few thousand lines) and you get reproducability.

It would legit be great if someone added Rust extraction to Coq and Why3. A lot of the work for verifying different pieces of Rust is already done in Coq. And being able to extract verified FP code to a Rust library directly without needing to make a C wrapper would be nice. (Same with stiff like lock free data structures and other things that you do need to formally verify in some way.)

But I think we are some years away from this because unsafe doesn't have a rigorously defined memory model yet and there are debates about the merits of stacked borrows vs tree borrows and various other things.

ridicalis • 5mo ago

Dat logo tho

9 Reply

tjhance • 5mo ago

Dafny was one of the major inspirations behind the Rust verification tool Verus. The big lesson from Verus's development is that Dafny's style of VC generation works even better when used in conjunction with Rust-style ownership types. Specifically, one of the most challenging aspects of Dafny's approach is how it deals with its mutable objects ("dynamic frames"). However, if you use ownership types instead, this aspect of the encoding can be entirely excised. Thus, with Rust, you can get all the benefits of SMT-based modular function verification, without the downsides of dynamic frames.

5 Reply

ctesibius • 5mo ago

It's going to depend on context. If you need to choose a language for a commercial project, it can be helpful to look at a big database for a similar project and see what issues arose in practice, because they often don't correspond to the issues language designers want to address.

So to give an example: I used to "own" a Windows device managment system. This was intended for corporate use, and allowed you to specify what AV, VPN versions were installed, check that certain software was running before a VPN was brought up, and so on. It also managed 3G and public WiFi connections. This was a large and quite complex bit of software, and was maintained over several years after launch, so that gave me a good database of the problems which were found during development and operation.

No memory handling problems were found. That doesn't mean that none existed, of course, but generally we were not doing anything very complicated such as you might have if you did some thing like write a compiler.

and other assertions, then passed that over as part of the spec for the system (which was not written in Java). This could then be incorporated in to the unit tests. That's as close as I have ever needed to come to a formal spec.

The big which actually appeared were things like visual layout, modem compatibility, internationalisation - loads of boring stuff which is difficult to address at the language level. But that's the point. Language designers put in features to address the types of bugs that they *can* address, and that's not necessarily the ones which come up in practice. Now that's not to suggest that something like Dafny cannot add value, but it is important to find application domains where its strengths correspond to real problems showing up in bug databases.

1 Reply

SadPie9474 • 5mo ago

Top 1% Commenter

Most of the Rust community has come to recognize that Dafny is effectively the world's first programming language, and that's not an exaggeration. It has precisely the same goal as Rust: for the language to prove its own correctness for you. Rust takes this to a certain extent, but falls far short of anything practical. Dafny allows you to do everything that Rust does but to a more thorough extent. Languages that have existed before Dafny are merely languages that can help *you do* programming; Dafny is the first language that is itself a programming language.

-20 Reply

r/ProgrammingLanguages • 2 yr. ago

The Dafny Programming and Verification Language

39 upvotes · 13 comments

r/rust • 8 yr. ago

15 upvotes · 19 comments

r/ProgrammingLanguages • 3 yr. ago

How does the Dafny Programming Language compile-time check its constraints?

24 upvotes · 13 comments

r/programming • 8 yr. ago

Dafny: A Language and Program Verifier for Functional Correctness

27 upvotes · 21 comments

r/programming • 9 yr. ago

Dafny: a verification-aware programming language

29 upvotes · 27 comments

r/PointlessStories • 3 mo. ago

How I didn't recognise my own language

247 upvotes · 12 comments

r/Transcription • 1 mo. ago

Trying to identify the language

341 upvotes · 39 comments

r/rust • 1 yr. ago

Nine Rules to Formally Validate Rust Algorithms with Dafny

5 upvotes · 7 comments

r/singularity • 5 mo. ago

Daphne Koller says although biology is 5-7 years behind language models, the explosion in biological data means that AI will soon be able to make causal inferences about disease...

621 upvotes · 60 comments

r/programming • 10 mo. ago

Dafny is a verification-aware programming language

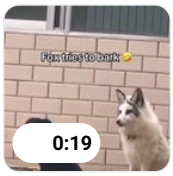
r/asklinguistics • 2 mo. ago

Why are some languages better for certain styles of rhyming?

42 upvotes · 19 comments

r/PetsareAmazing • 2 mo. ago

Learning other languages



1.5K upvotes · 44 comments

r/CuratedTumblr • 7 mo. ago

Language learning

1.7K upvotes · 33 comments

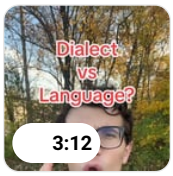
r/functionalprogramming • 4 mo. ago

Popularity of different functional languages

53 upvotes · 43 comments

r/LanguageTips2Mastery • 1 mo. ago

What's the difference between a Dialect and a Language?



109 upvotes · 31 comments

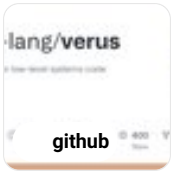
r/no • 3 mo. ago

Can you say no in another language, dialect or slang here?

182 upvotes · 1K comments

r/rust • 10 mo. ago

GitHub - verus-lang/verus: Verified Rust for low-level systems code



81 upvotes · 10 comments

r/TvShows • 7 mo. ago

Subtitled shows: What do you think about the translation of cultural terms?

5 upvotes · 1 comment

What do you think of writing using Markdown?

1 upvote · 5 comments

r/bisexualadults • 6 mo. ago

What do you think of my manifesto?

r/no • 3 mo. ago

Are you allowed to say the y word in other languages?

40 upvotes · 122 comments

r/AskEurope • 27 days ago

What languages are you fluent in?

234 upvotes · 1.1K comments

r/asklinguistics • 20 days ago

Are some languages inherently harder to learn?

34 upvotes · 56 comments

r/webdev • 6 mo. ago

What do you think about RxJS ?

18 upvotes · 71 comments

r/maryland • 5 mo. ago

What do you guys think of this dialect analysis of our region? Is it accurate to you?

67 upvotes · 80 comments