

WHAT DOES DIALYZER **THINK** OF ME YOU?

Stavros Aronis
Developer & Trainer @ Erlang Solutions



@Vahnatai
stavros.aronis@erlang-solutions.com

#CodeBEAMSTO

Erlang
SOLUTIONS

About me:

- Student of Kostis Sagonas
- Master thesis on Dialyzer
- PhD in CS (**not** type systems)
- 9yrs Erlang, 1yr “industrially”
- <https://github.com/parapluu/Concuerror>

#CodeBEAMSTO



Goals

Therapy

How to control hatred towards a piece of software?

Dialyzer

How Dialyzer thinks?

Productivity

Why should you add Dialyzer to your build pipeline, today?

Dialyzer?

- Static analysis tool, included in OTP
- Works on single modules or entire applications
- Produces **sound warnings** (no false positives)
 - “Dialyzer is never wrong”
- Finds definite type errors
- Finds unreachable code
- Finds unnecessary tests
- etc.



1.

WHAT DO **YOU**
THINK OF
DIALYZER?

#CodeBEAMSTO

Charisma



Johnny Winn

@johnny_rugger

Follow



I'm sure that dialyzer is always right but it sure can be an ass about it :/

12:03 AM - 9 Apr 2019

1 Retweet 6 Likes



Charisma



Bisse

@SveTob

Follow



Erlang dialyzer is your super smart colleague who looks at your code and comes with bombshell insights 5 minutes later, but he's an antisocial introvert with no communication skills so you have no clue what the fuck he's saying [#elixir](#) [#dialyzer](#)

1:33 PM - 4 Apr 2019

3 Likes



3



Charisma



Chris Keathley

@ChrisKeathley

Follow



Dialyzer is most definitely The Bard in the OTP party

2:45 PM - 21 Feb 2019

7 Retweets 17 Likes



2

7

17



Charisma



Janice @ BangBangCon + Recurse...

@contrepont21

Follow

"either dialyzer misunderstands me or I misunderstand it" -@seancribbs on dialyzex

A quote that's probably generalizable to a whole bunch of software and hardware, tbqh #codebeamsf #myelixirstatus

12:45 AM - 2 Mar 2019

3 Retweets 7 Likes



↻ 3

♡ 7



How does Dialyzer **think?**

#CodeBEAMSTO

How does Dialyzer **think**?

foo(Arg) ->

...

Types: **sets** of values

(e.g. "3.5",

"1 | 2 | 3",

"atom()",

"term()",

"{term(), list()}",

)

How does Dialyzer **think**?

foo(Arg) ->

...

Subtype relationship:

`'foo'` \subseteq `'foo' | 'bar'`

\subseteq `atom()`

\subseteq `atom() | list()`

\subseteq `term()`

How does Dialyzer **think**?

foo (Arg) ->

...

Arg's type: `term()` .

How does Dialyzer **think**?

```
foo(Arg) ->  
  self() ! Arg,  
  ...
```

Arg's type: `term()` .

Any value can be sent as a message.

How does Dialyzer **think**?

```
foo(Arg) ->  
  self() ! Arg,  
  self() ! Arg + 1,  
  ...
```

Arg's type: `number()`.

For "+" to succeed, Arg
needs to be a number.

How does Dialyzer **think**?

```
foo(Arg) ->  
  self() ! Arg,  
  self() ! Arg + 1,  
  Bar = Arg rem 2,  
  ...
```

Arg's type: `integer()`.

For "rem" to succeed, Arg
needs to be an integer.

Bar's type: `integer()`.

How does Dialyzer **think**?

```
foo(Arg) ->
```

```
  self() ! Arg,
```

```
  self() ! Arg + 1,
```

```
  Bar = Arg rem 2,
```

```
  Baz = Arg + Bar,
```

```
  ...
```

Arg's type: `integer()`.

Bar's type: `integer()`.

Baz's type: `integer()`.

And not number!

How does Dialyzer **think**?

```
foo(Arg) ->
```

```
  self() ! Arg,  
  self() ! Arg + 1,  
  Bar = Arg rem 2,  
  Baz = Arg + Bar,  
  Baz ! Bar,  
  ...
```

Arg's type: `integer()`.

Bar's type: `integer()`.

Baz's type: `none()`.

There's no "common" subtype of integer and any acceptable argument for erlang: "!".

How does Dialyzer **think**?

foo.erl:5: Function foo/1 has no local return

foo.erl:10: The call erlang:!(Baz::integer(),Bar::-1 | 0 | 1) will never return since it differs in the 1st argument from the success typing arguments: (atom() | pid() | port() | {atom(),atom()},any())

```
foo(Arg) ->
    self() ! Arg,
    self() ! Arg + 1,
    Bar = Arg rem 2,
    Baz = Arg + Bar,
    Baz ! Bar,
    ...
```

How does Dialyzer **know** types?

#CodeBEAMSTO



Dialyzer trusts **none!**

- ... except for Erlang primitives
- Also collects and uses any specs it finds
- Needs to pre-analyse any code that **you** trust
 - e.g. OTP, dependencies, etc.
 - saves the result in a lookup table (PLT)



How does Dialyzer **think**?

- Types: sets of values (e.g. “1 | 2”, “atom()”, “any()”).
- Dialyzer’s type inference:
 - Assumes that variables can have any value
 - Finds where each variable is used
 - Constraints its values so that its use is ok
 - bottom-up based analysis
 - values must be subtypes of “success typings”
 - Refining:
 - dataflow-based analysis
 - for non-exported function arguments
 - based on local calls



2.

MORE

THOUGHTS?

#CodeBEAMSTO



Commitments?



Rob Howard (🇬🇧)

@damncabbage

Follow

Replying to @damncabbage @Storakatten

And making use bolt-on type systems where I can, eg. Flow / TypeScript.

(An exception: Erlang/Elixir's Dialyzer; it's almost deliberately written to give you a false sense of confidence, eg. it won't complain if there's at least *one* working path through a broken function.)

4:55 PM - 11 May 2019

2 Likes



1



2



Testing & Verification Tools

Program analyses tools	Sound	Complete
Under-approximate	Report only real bugs (may miss some)	Find all bugs (may report false positives)
Over-approximate	Produce only real safety proofs (may reject safe programs)	Prove safety for all safe programs (may accept unsafe programs)

https://twitter.com/johannes_kinder/status/1105138480303218688

The bright side of life



Igor Clark

@igorclark

Follow



umm, **#erlang**'s **#dialyzer** is the absolute shiznit. quick pass over a project found a couple of really non-obvious bugs in no time, in library/dependency code as well as mine. if ever motivation for adding more spec/type annotations was needed then this definitely provides it!

5:55 PM - 20 Feb 2019

2 Retweets 9 Likes



1



2



9



#CodeBEAMSTO



3.

**EVEN MORE
THOUGHTS?**

#CodeBEAMSTO



Speed?

 **Louis** @louispilfold · Apr 26
I should do more talks

3 replies 11 likes

 **Evadne W.** @evadne · Apr 26
Do a detailed one on what is wrong with Dialyzer

1 reply 3 likes

 **Louis** @louispilfold [Follow](#)

Replying to @evadne

I'll just run it on a project and we can collectively wait for it to finish

10:18 AM - 26 Apr 2019

1 Like 

1 reply 1 like



Speed?



Tony Collen @tcollen · Apr 18

Leaned heavily on Behaviours and Typespecs in #elixirlang today. It worked out most excellent 🤘👍



3



1



Tony Collen

@tcollen

Follow

I also wish Dialyzer were about 100x faster

1:39 AM - 18 Apr 2019

1 Like



1



dialyzer --statistics -n --apps <all of OTP>

```
compile (+0.07s): 18.57s (1410 modules)
clean   (+0.01s): 0.79s
remote  (+0.14s): 9.43s
order   (+0.57s): 4.95s
typesig (+0.00s): 59.18s ( 88558 SCCs)
order   (+0.00s): 4.02s
refine  (+0.00s): 30.97s (1410 modules)
order   (+0.00s): 5.77s
typesig (+0.00s): 61.03s ( 70444 SCCs)
order   (+0.00s): 2.31s
refine  (+0.00s): 30.42s (1340 modules)
order   (+0.00s): 0.16s
typesig (+0.00s): 8.87s ( 3107 SCCs)
order   (+0.00s): 0.06s
refine  (+0.00s): 8.13s ( 247 modules)
order   (+0.00s): 0.00s
typesig (+0.00s): 1.24s ( 11 SCCs)
order   (+0.00s): 0.00s
refine  (+0.00s): 0.40s ( 2 modules)
warning (+1.58s): 34.35s (1410 modules)
(+ 0.45s)
```

done in 4m43.61s

On my 2014 Quad-core
Macbook Pro



dialyzer +S 1:1 --statistics -n --apps <all of OTP>

```
compile (+0.09s): 40.06s (1410 modules)
clean (+0.01s): 0.81s
remote (+0.13s): 9.65s
order (+0.60s): 5.00s
typesig (+0.00s): 129.66s ( 88558 SCCs)
order (+0.00s): 4.05s
refine (+0.00s): 65.26s (1410 modules)
order (+0.00s): 5.86s
typesig (+0.00s): 137.26s ( 70444 SCCs)
order (+0.00s): 2.38s
refine (+0.00s): 67.58s (1340 modules)
order (+0.00s): 0.17s
typesig (+0.00s): 18.38s ( 3107 SCCs)
order (+0.00s): 0.06s
refine (+0.00s): 17.36s ( 247 modules)
order (+0.00s): 0.00s
typesig (+0.00s): 1.24s ( 11 SCCs)
order (+0.00s): 0.00s
refine (+0.00s): 0.36s ( 2 modules)
warning (+1.42s): 77.77s (1410 modules)
(+ 0.50s)
```

done in 9m46.04s

If restricted to one core



4.

Conclusions

#CodeBEAMSTO



Why & how I use Dialyzer?

- To check my types & specs
- To simplify my code
- As the first job on CI test runs
- Properly focused (only my code, not dependencies)
 - On sane dependencies, possibly patched
- With a cached PLT
- With HiPE enabled

More reads

- Decoding Dialyzer

<http://devonestes.herokuapp.com/decoding-dialyzer>

- Spot The Discrepancies with Dialyzer for Erlang

<http://tech.adroll.com/blog/dev/2019/02/19/erlang-dialyzer.html>

- Chemanalysis: Dialyzing Elixir

<https://codesync.global/media/chemanalysis-dialyzing-elixir-sean-cribbs>

THANK YOU!
YOUR
THOUGHTS?

Stavros Aronis
Developer & Trainer @ Erlang Solutions



@Vahnatai
stavros.aronis@erlang-solutions.com

#CodeBEAMSTO

Erlang
SOLUTIONS