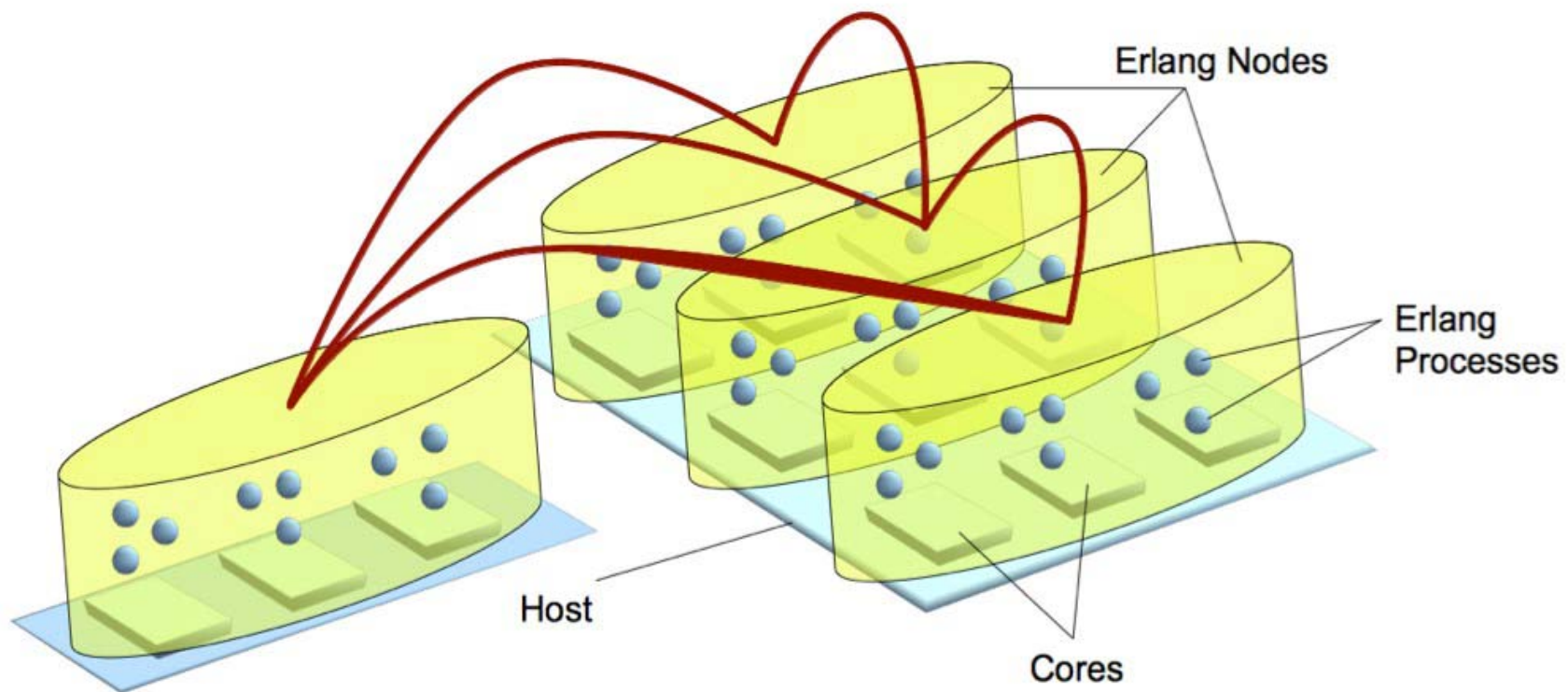# Profiling with Percept2

Huiqing Li and Simon Thompson

University of Kent

Overview

- Erlang multicore …

- Percept

- Percept2

- Other onging work

Erlang multicore

Tools

How to use
these systems
effectively?

Tracing and Profiling

- Understand what your code does and how it makes use of the resources.

- Find bottlenecks.

- Achieve efficient usage of resources and scalability.

Existing tools

- Erlang tracing

- dbg/etop/ttb

- fprof/eprof/cprof

- Percept

- Dtrace

- ...

Percept

Percept

- **P**ercept: **Er**lang **c**oncurr**e**ncy **p**rofiling **t**ool.

- In the Erlang/OTP distribution ...

- ... written by Björn-Egil Dahlberg.

- Monitors process runnability using

  erlang:trace/3, erlang:system_profile/2.

- Offline tool ... 'replay' a computation.

RELEASE

Percept

- How it works
  - Profiling
  - Data analysing
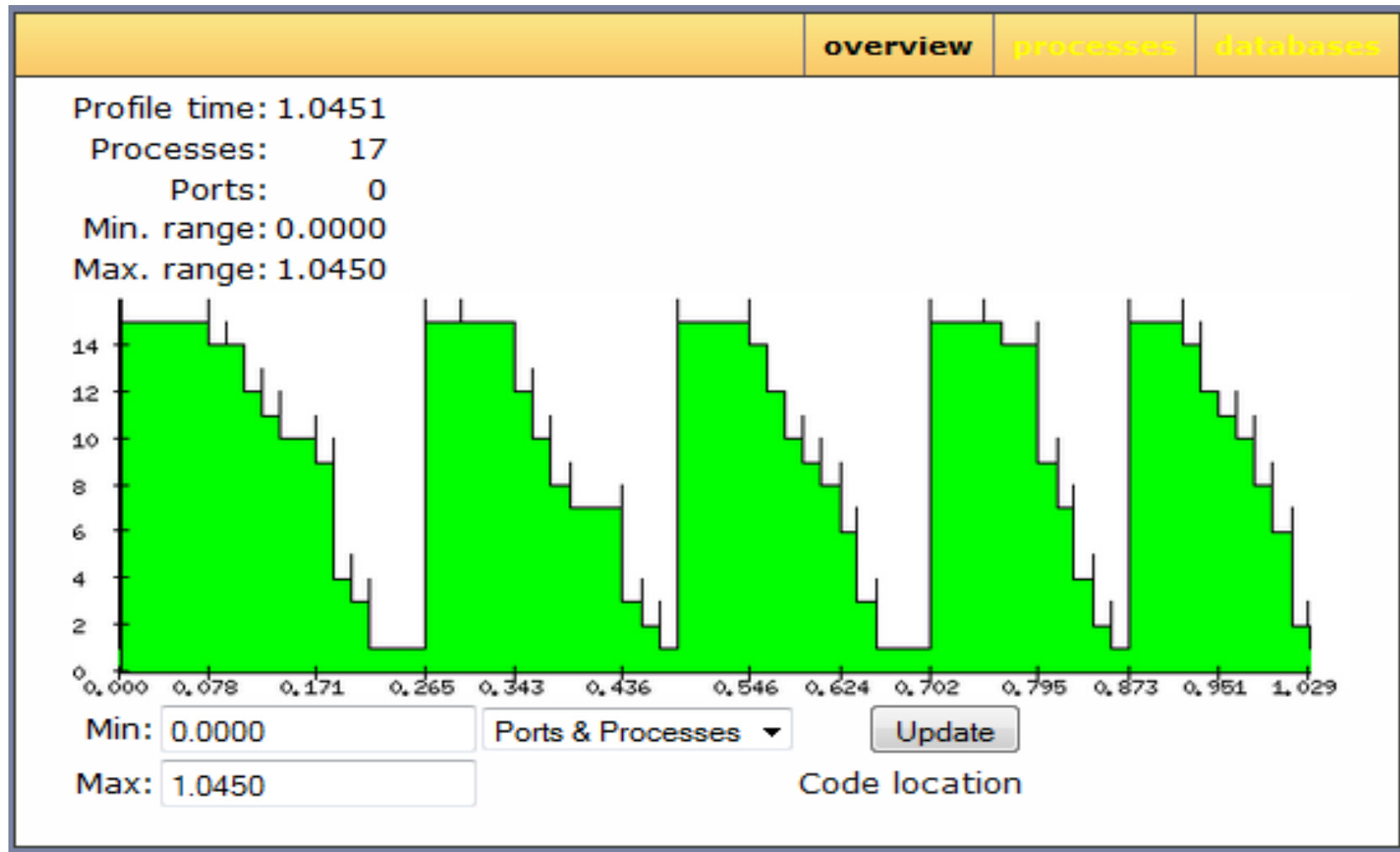 - Web-based data viewing

RELEASE

Percept

- Histogram of active processes vs. time.

- 'Drill down' to process information via Pid.

- Start/end time, parent/child processes, time spent waiting for messages, ...

- View individual process runnability.

Percept

Demo

::::RELEASE

Percept

Percept

					Processes
Select	**Pid**	**Lifetime**	**Entrypoint**	**Name**	**Parent**
☐	<0.37.0>	▮▮▮▮▮	undefined	undefined	undefined
☑	<0.167.0>	▮▮▮▮▮	sorter:main/4	undefined	<0.37.0>
☑	<0.168.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.169.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.170.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.171.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.172.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.173.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.174.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.175.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.176.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.177.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.178.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.179.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.180.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.181.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>
☑	<0.182.0>	▮▮▮▮▮	sorter:loop/0	undefined	<0.167.0>

overview | processes | databases

☑ Select all

[Compare]
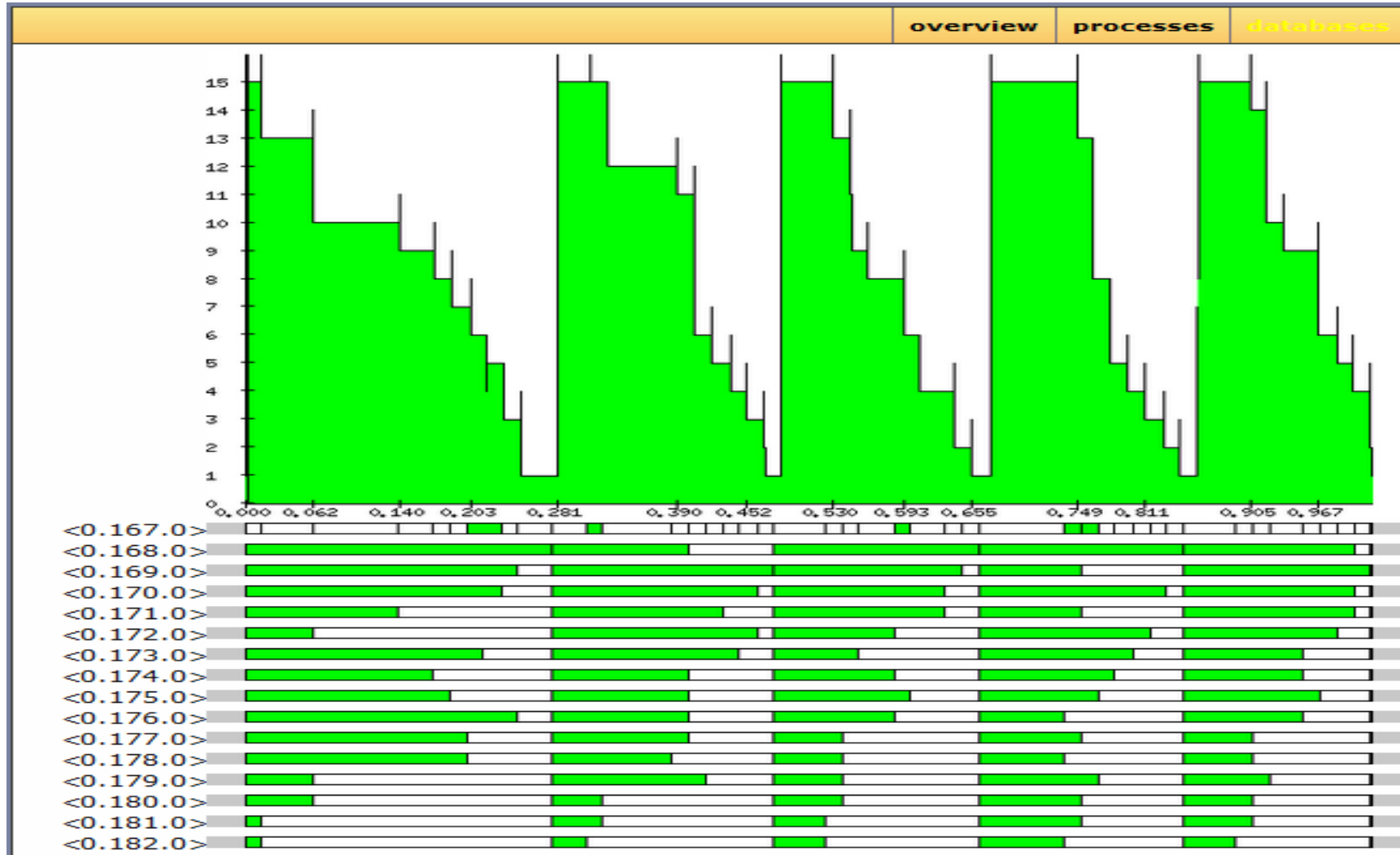
::: RELEASE

Percept

Percept2

Functionality enhancement:

- Scheduler information
- Process communication
- Process run queue migration
- Runnable vs. running
- Dynamic context-aware callgraph
- Process/function accumulated runtime
- Garbage collection time
- Support for distribution

Usability and scalability:

- Scalable process tree
- Cache history webpages
- Selective profiling
- Parallel processing
- . . .

Percept

RELEASE

Percept2

- Download

    https://github.com/RefactoringTools/percept2

- Install

    $ configure; $make; $(sudo) make install

- Documentation

    http://refactoringtools.github.com/percept2

- Papers

    - Multicore Profiling for Erlang Programs Using Percept2 (EW2013)
    - Multi-level Visualization of Concurrent and Distributed Computation in Erlang (VLC2013)

RELEASE

Percept2

- How to use

  - Profile a complete application run.

```
percept2:profile(FileSpec, Entry, Options).

-type profile_option()::procs_basic | procs    |
                        ports_basic | ports    |
                        schedulers  | running|
                        message     | migration |
                        garbage_collection | s_group |
                        all |
                        {callgraph,[module_name()]}.

-type filespec()::file:filename()
              |{file:filename(),wrap,Suffix::string(),
               WrapSize::pos_integer()}.
```

Percept2

- How to use

  - Profile while application is running.

    ```
    percept2:profile(FileSpec, Options).

    percept2:stop_profile().
    ```

Percept2

- ## How to use

  - Profile a particular part of your application by instrumenting your code with `percept2:profile/2` and `percept2:stop_profile/0` commands.

Percept2

- How to analyze trace data

  - analyze a single trace file.

    ```
    percept2:analyze([FileName]).
    ```

  - analyze a colection of trace files in parallel.

    ```
    percept2:analyze(FileBaseName, Suffix,
                     StartIndex, EndIndex).
    ```

- How to visualize profiling data.

  ```
  percept2:start_webserver(8888) or

  percept2:start_webserver()
  ```

  then go to `localhost:8888` `(or the port number allocated automatically)` in your web browser.

Percept2

Demo

RELEASE

Percept2: function profiling

- Existing function profiling tools for Erlang:

Tool	Results	Size of result	Effects on program execution time	Records number of calls	Records Execution time	Records called by	Records garbage collection
fprof	per process to screen/file	large	significant slowdown	yes	total and own	yes	yes
eprof	per process/function to screen/file	medium	small slowdown	yes	only total	no	no
cover	per module to screen/file	small	moderate slowdown	yes, per line	no	no	no
cprof	per module to caller	small	small slowdown	yes	no	no	no

Percept2 →

Percept2: function profiling

Tool	Results	Size of result	Effects on program execution time	Records number of calls	Records Execution time	Records called by	Records garbage collection
fprof	per process to screen/file	large	significant slowdown	yes	total and own	yes	yes
Percept2	per process to file *but can be selective*	*depends on*	yes	only total	yes	yes	
eprof	per process/function to screen/file	medium	small slowdown	yes	only total	no	no
cover	per module to screen/file	small	moderate slowdown	yes, per line	no	no	no
cprof	per module to caller	small	small slowdown	yes	no	no	no

... Percept2 only profiles functions defined in the modules specified by user.

RELEASE

Percept2: distribution

- Inter-node communication.

- s_group structure.

- Built-on Erlang trace builder(ttb).

RELEASE

Percept2

Demo

:::RELEASE

Percept2

- Interpreting profiling data.
  - Abnormal ratio between the number of runnable processes and the number of schedulers available.
  - Very heavy messaging passing
  - Very large messages
  - Long garbage collection time.
  - Long waiting time
  - Very shorted-lived processes
  - Heavily loaded processes
  - Workload imbalance
  - Etc.

Percept2

Percept2

Select [+/-]		Pid	Lifetime	Name	Parent	#RQ_chgs	#msgs _recv	avg_size msg_recv	#msgs _sent ▲	avg_size msg_sent	Entrypoint	Callgraph
☐	-	<15711.109.0>	▉▉▉	undefined	undefined	0	217424	9	217417	15	undefined	no callgraph/time
☐	-	<15723.107.0>	▉▉▉	undefined	undefined	0	217422	9	217416	14	undefined	no callgraph/time
☐	-	<15723.105.0>	▉▉▉	undefined	undefined	0	17392	32	17549	96	undefined	no callgraph/time
☐	-	<15711.107.0>	▉▉▉	undefined	undefined	0	11456	40	11612	136	undefined	no callgraph/time
☐	-	<15727.108.0>	▉▉▉	undefined	undefined	0	80249	18	1924	19	undefined	no callgraph/time
☐	-	<15728.132.0>	▉▉▉	undefined	undefined	0	1868	19	1649	15	undefined	no callgraph/time
☐	-	<15726.111.0>	▉▉▉	undefined	undefined	0	4153	19	1644	16	undefined	no callgraph/time
☐	-	<15726.134.0>	▉▉▉	undefined	undefined	0	1980	18	1634	16	undefined	no callgraph/time
☐	-	<15728.112.0>	▉▉▉	undefined	undefined	0	2030	19	1625	15	undefined	no callgraph/time
☐	-	<15726.116.0>	▉▉▉	undefined	undefined	0	2275	18	1619	15	undefined	no callgraph/time
☐	-	<15725.95.0>	▉▉▉	undefined	undefined	0	2028	18	1615	16	undefined	no callgraph/time
☐	-	<15725.115.0>	▉▉▉	undefined	undefined	0	1980	19	1614	15	undefined	no callgraph/time
☐	-	<15726.102.0>	▉▉▉	undefined	undefined	0	2186	18	1608	15	undefined	no callgraph/time
☐	-	<15726.94.0>	▉▉▉	undefined	undefined	0	2317	17	1606	15	undefined	no callgraph/time
☐	-	<15726.114.0>	▉▉▉	undefined	undefined	0	2102	18	1606	15	undefined	no callgraph/time

overview   processes   ports   function act!

RELEASE

Percept2

Pid	<0.41.0>
Node	'test_percept@127.0.0.1'
Name	server
Entrypoint	proc_lib:init_p/5
Arguments	

Timetable		**Timestamp**	**Profile Time**
	Start	{1386,69002,86003}	0.0000
	Stop	{1386,70016,106103}	1014.0201

Parent	<0.37.0>
Children	
RQ_history	[3]
{#msg_received, avg_msg_size}	{3005,60}
{#msg_sent, avg_msg_size}	{1005,59}
garbage collection time (is secs)	0.0001
accumulated runtime (in secs)	0.0103
Callgraph/time	show call graph/time

percentage of total waiting time	total	mean	stddev	#recv	module:function/arity
100%	1013.9975	1.0140	0.0005	1000	timer:sleep/1
0%	0.0042	0.0000	0.0000	1022	gen_server:loop/6
0%	0.0001	0.0001	0	1	code_server:call/2
0%	0.0000	0.0000	0	1	init:request/1
0%	0.0000	0.0000	0	1	erlang:put/2

RELEASE

Percept2: case study
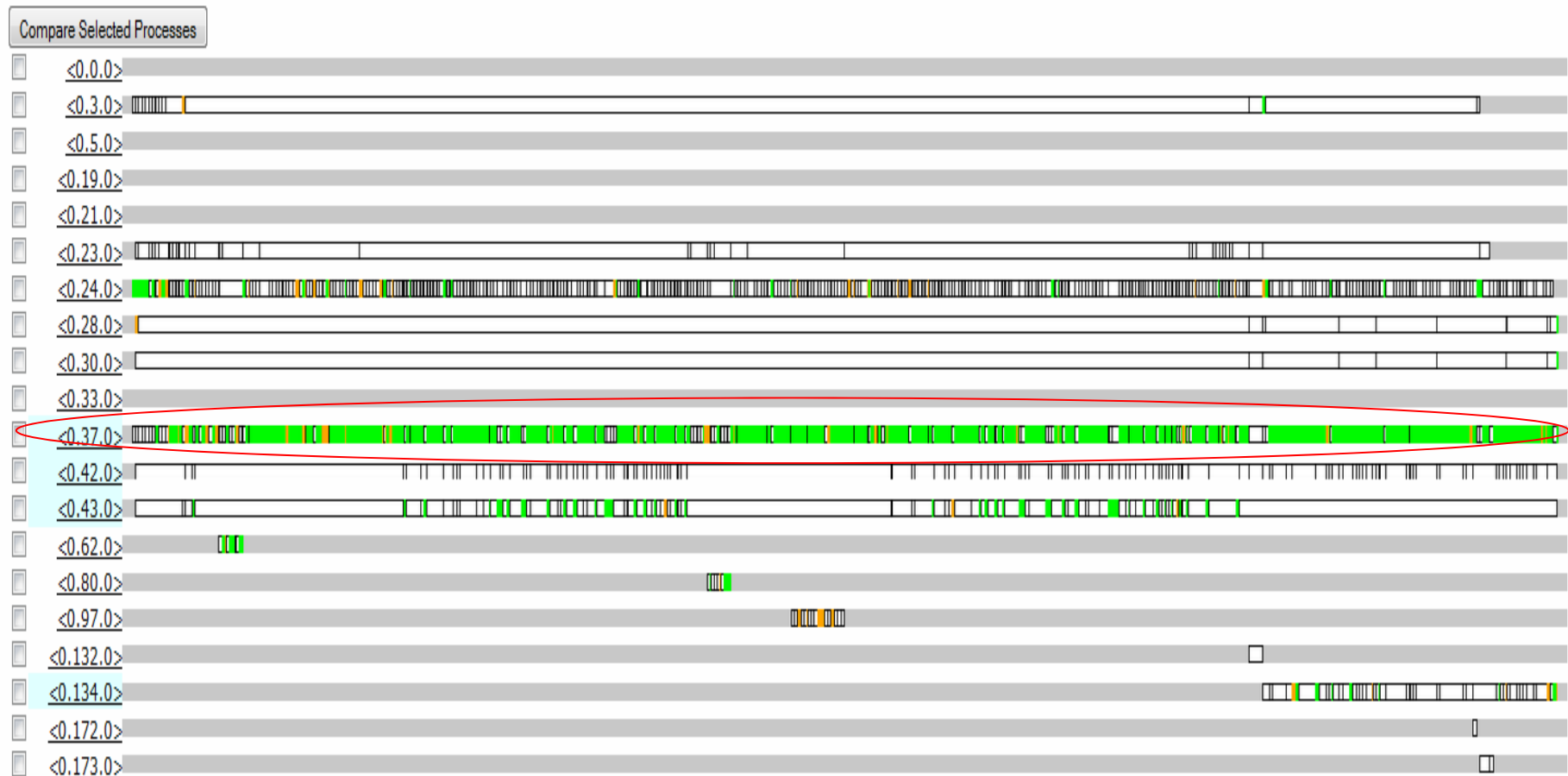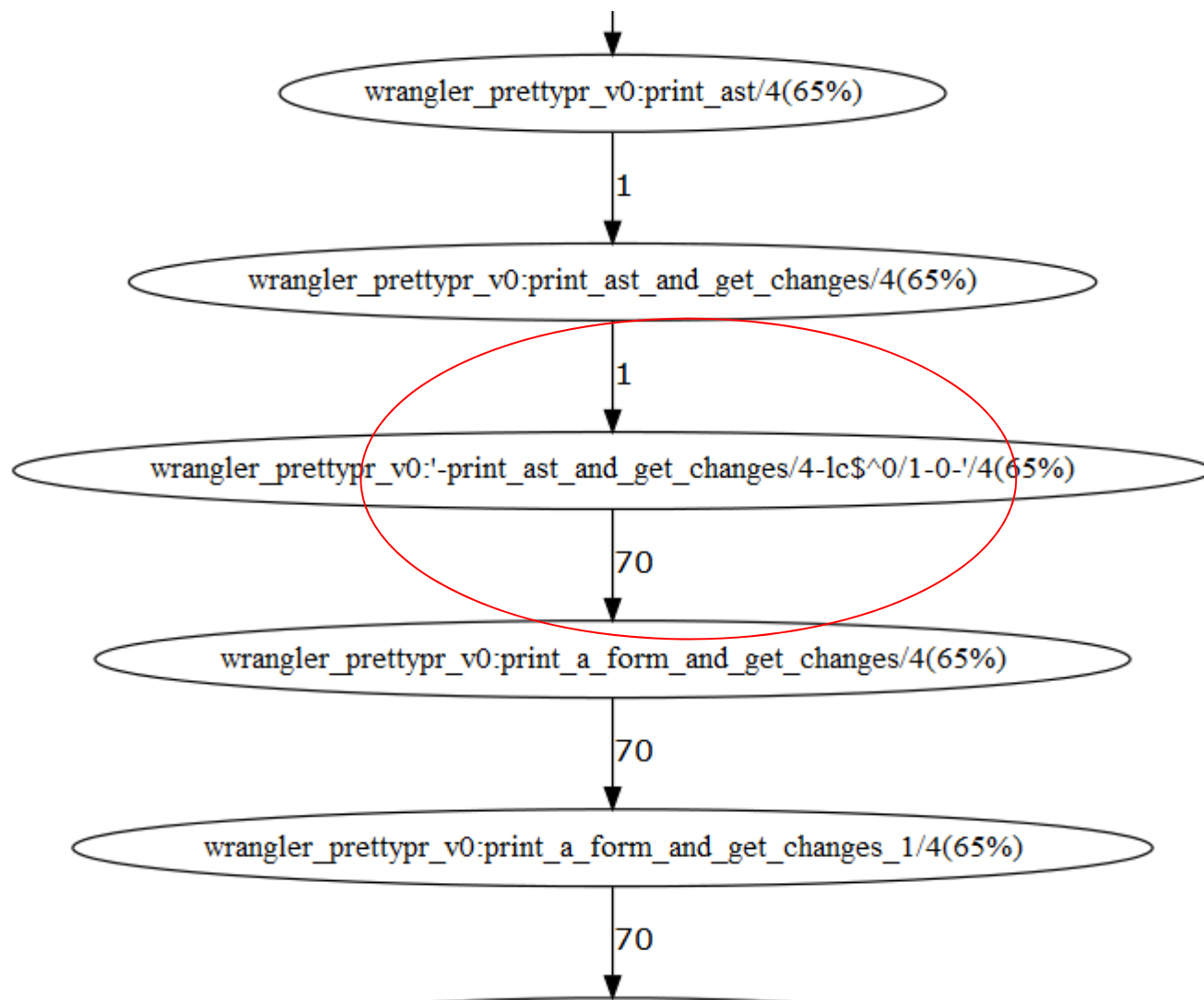
Percept2

Percept2
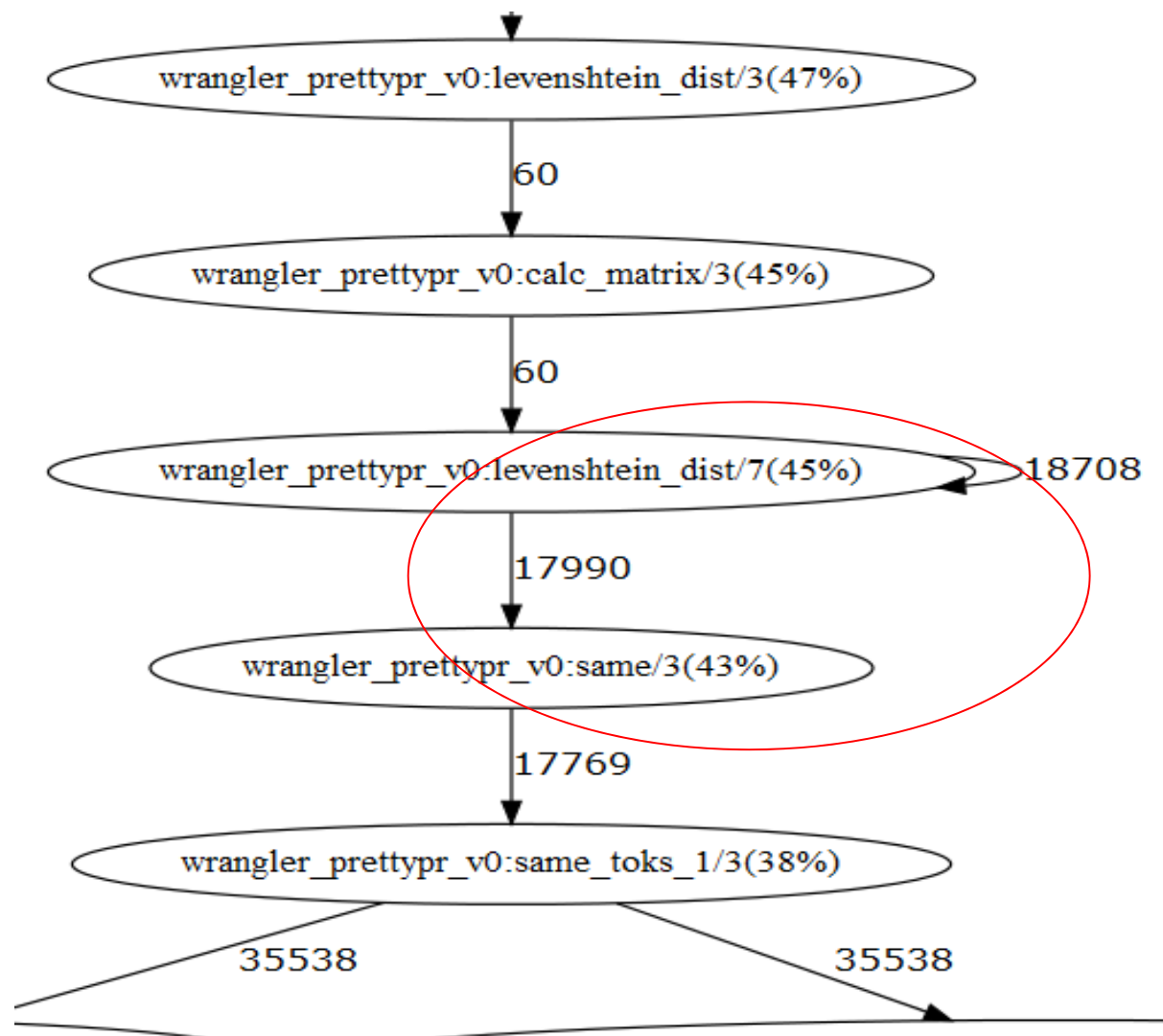
Percept2: process runnability

The number of runnable processes at any time.
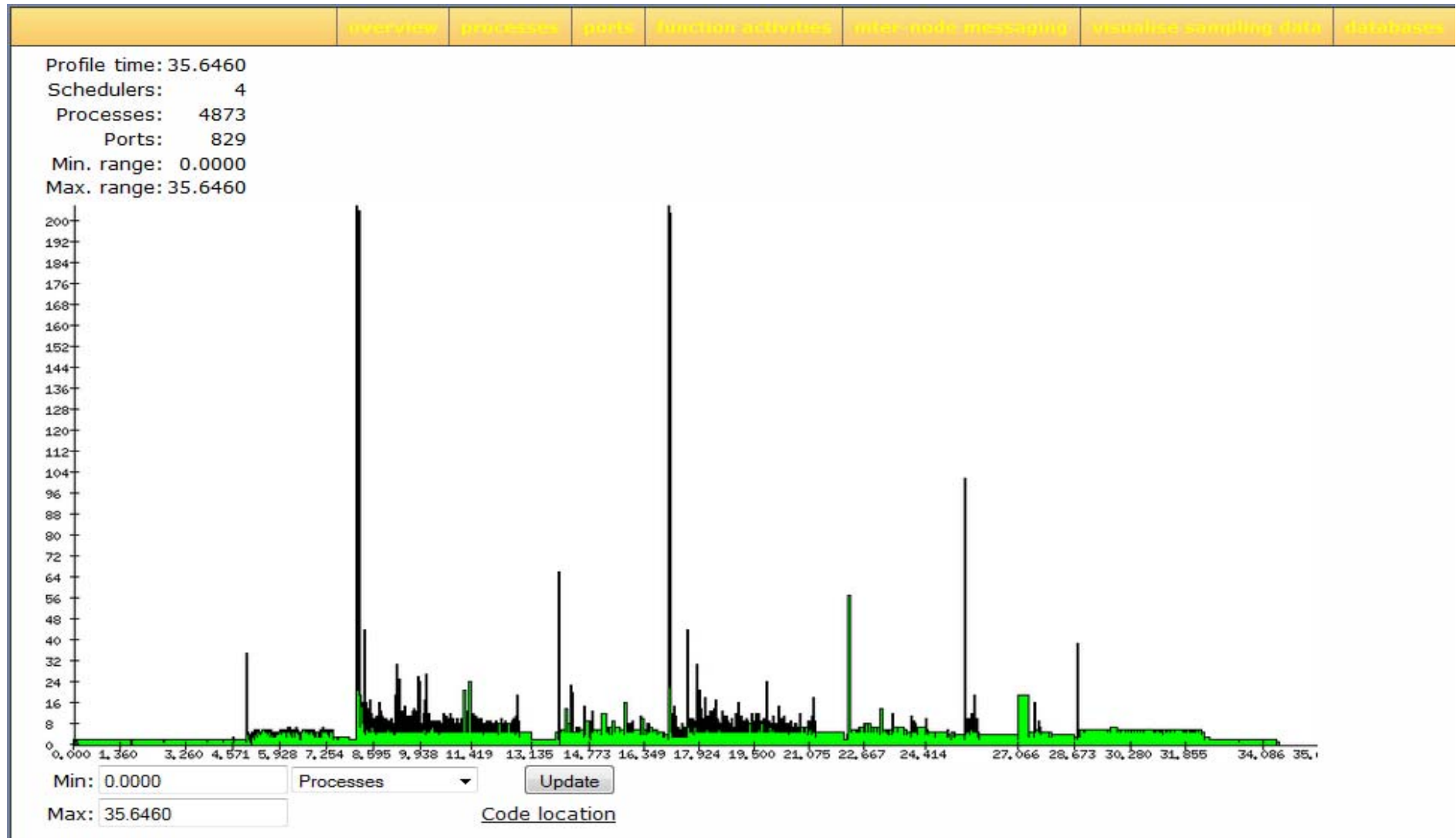
Percept2: scheduler info

The number of schedulers active at any time.

Percept2: process info tree

Process info presented in expandable tree format.

Percept2: process info tree

Process info presented in expandable tree format.

Percept2: process info

Information about a particular process

Pid	<0.4981.0>
Name	undefined
Entrypoint	dialyzer_worker:loop/2
Arguments	

Timetable		Timestamp	Profile Time
	Start	{1370,944591,720014}	4.4930
	Stop	{1370,944591,766136}	4.5391

Parent	<0.4978.0>
Children	<0.5045.0>
RQ_history	[1,2,1,4]
{#msg_received, avg_msg_size}	{15,6344}
{#msg_sent, avg_msg_size}	{3,1645}
accumulated runtime (in secs)	0.0309
Callgraph/time	no callgraph/time

percentage of total waiting time	total	mean	stddev	#recv	module:function/arity
98%	0.0144	0.0144	0	1	compile:do_compile/2
1%	0.0002	0.0000	0.0000	8	prim_file:drv_get_response/1
0%	0.0001	0.0000	0.0000	4	erlang:port_command/2
0%	0.0000	0.0000	0	1	dialyzer_coordinator:get_next_label/2
0%	0.0000	0.0000	0	1	dialyzer_worker:loop/2

Percept2: process communication

Process info presented in expandable tree format.

Select	[+/-]	Pid	Lifetime	Name	Parent	#RQ_chgs	#msgs_received	#msgs_sent
☐	-	<0.22.0>	▬	undefined	undefined	0	{3,11}	{3,8}
☐	-	<0.20.0>	▬	undefined	undefined	0	{3,8}	{0,0}
☐	-	<0.25.0>	▬	undefined	undefined	0	{1,71}	{1,51}
☐	-	<0.24.0>	▬	undefined	undefined	0	{731,2187}	{364,4287}
☐	[-]	<0.38.0>	▬	undefined	undefined	1	{1785,25590}	{300,136}

Select	[+/-]	Pid	Lifetime	Name	Parent	#RQ_chgs	#msgs_received	#msgs_sent	
☐	[-]	<0.4977.0>	▬	undefined	<0.38.0>	0	{71,467524}	{70,471185}	dialyzer_analysis_callg

Select	[+/-]	Pid	Lifetime	Name	Parent	#RQ_chgs	#msgs_received	#msgs_sent	Entrypoint
☐	[-]	<0.4978.0>	▬	undefined	<0.4977.0>	1175	{4856,167}	{209,158804}	dialyzer_analysis_callgraph:analysis_start/2 ca

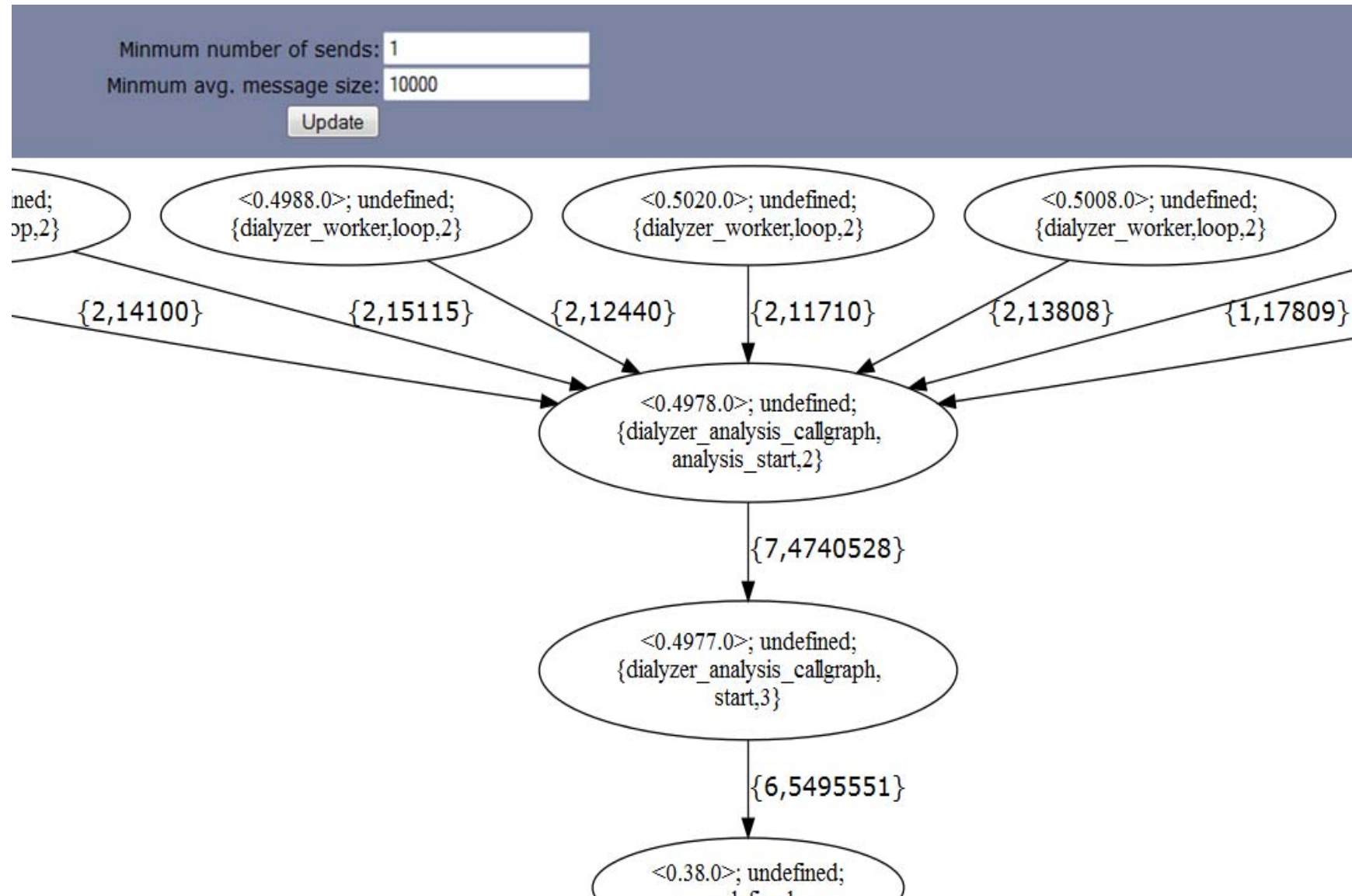| Select	[+/-]	Pid	Lifetime	Name	Parent	#RQ_chgs	#msgs_received	#msgs_sent	Entrypoint	Callgraph
☐	-	<0.4979.0>	▬	undefined	<0.4978.0>	0	{1,17}	{0,0}	dialyzer_callgraph:race_data_server_loop/1	no callgraph/time
☐	-	<0.*1*.0>	▬	'4790 procs omitted'	<0.4978.0>	10679	{11647,1378}	{20379,63}	dialyzer_worker:loop/2	no callgraph/time
☐	-	<0.*0*.0>	▬	'7 procs omitted'	<0.4978.0>	2044	{9461,21}	{4727,12}	dialyzer_coordinator:regulator_loop/2	no callgraph/time
☐	-	<0.7863.0>	▬	undefined	<0.4978.0>	708	{3479,21}	{1739,12}	dialyzer_coordinator:regulator_loop/2	no callgraph/time
☐	-	<0.9786.0>	▬	undefined	<0.4978.0>	1	{1,12}	{2,352}	dialyzer_worker:loop/2	no callgraph/time

☐ Select all   ☐ Include children procs   ☐ Include omitted procs
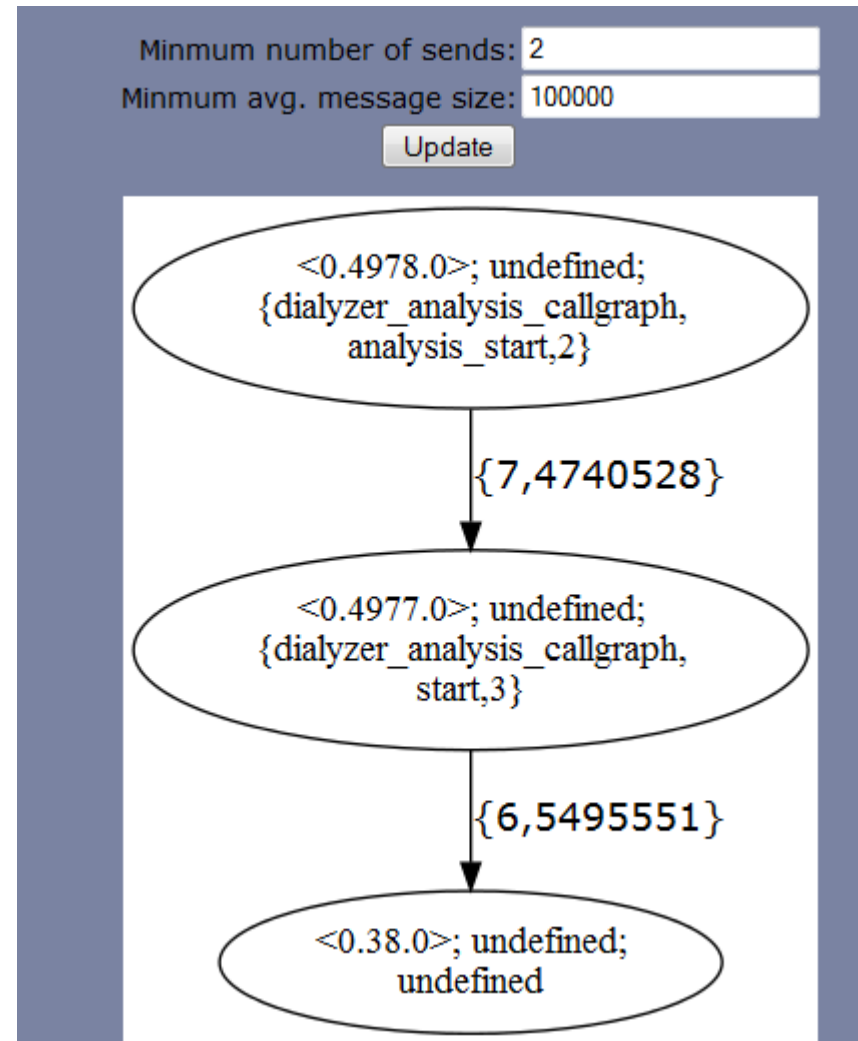
[Compare]   **Process Tree Graph**   **Process Communication**

Percept2: process communication

Process tree in graph representation

Percept2: process communication

Process info presented in expandable tree format.

Select	[+/-]	Pid	Lifetime	Name	Parent	#RQ_chgs	#msgs_received	#msgs_sent
☐	-	<0.22.0>		undefined	undefined	0	{3,11}	{3,8}
☐	-	<0.20.0>		undefined	undefined	0	{3,8}	{0,0}
☐	-	<0.25.0>		undefined	undefined	0	{1,71}	{1,51}
☐	-	<0.24.0>		undefined	undefined	0	{731,2187}	{364,4287}
☐	[-]	<0.38.0>		undefined	undefined	1	{1785,25590}	{300,136}

Select	[+/-]	Pid	Lifetime	Name	Parent	#RQ_chgs	#msgs_received	#msgs_sent	
☐	[-]	<0.4977.0>		undefined	<0.38.0>	0	{71,467524}	{70,471185}	dialyzer_analysis_callg

Select	[+/-]	Pid	Lifetime	Name	Parent	#RQ_chgs	#msgs_received	#msgs_sent	Entrypoint
☐	[-]	<0.4978.0>		undefined	<0.4977.0>	1175	{4856,167}	{209,158804}	dialyzer_analysis_callgraph:analysis_start/2 ca

Select	[+/-]	Pid	Lifetime	Name	Parent	#RQ_chgs	#msgs_received	#msgs_sent	Entrypoint	Callgraph
☐	-	<0.4979.0>		undefined	<0.4978.0>	0	{1,17}	{0,0}	dialyzer_callgraph:race_data_server_loop/1	no callgraph/time
☐	-	<0.*1*.0>		'4790 procs omitted'	<0.4978.0>	10679	{11647,1378}	{20379,63}	dialyzer_worker:loop/2	no callgraph/time
☐	-	<0.*0*.0>		'7 procs omitted'	<0.4978.0>	2044	{9461,21}	{4727,12}	dialyzer_coordinator:regulator_loop/2	no callgraph/time
☐	-	<0.7863.0>		undefined	<0.4978.0>	708	{3479,21}	{1739,12}	dialyzer_coordinator:regulator_loop/2	no callgraph/time
☐	-	<0.9786.0>		undefined	<0.4978.0>	1	{1,12}	{2,352}	dialyzer_worker:loop/2	no callgraph/time

☐ Select all   ☐ Include children procs   ☐ Include omitted procs

Compare   **Process Tree Graph**   **Process Communication**

Percept2: process communication

Percept2: process communication

Percept2: compare process runnability

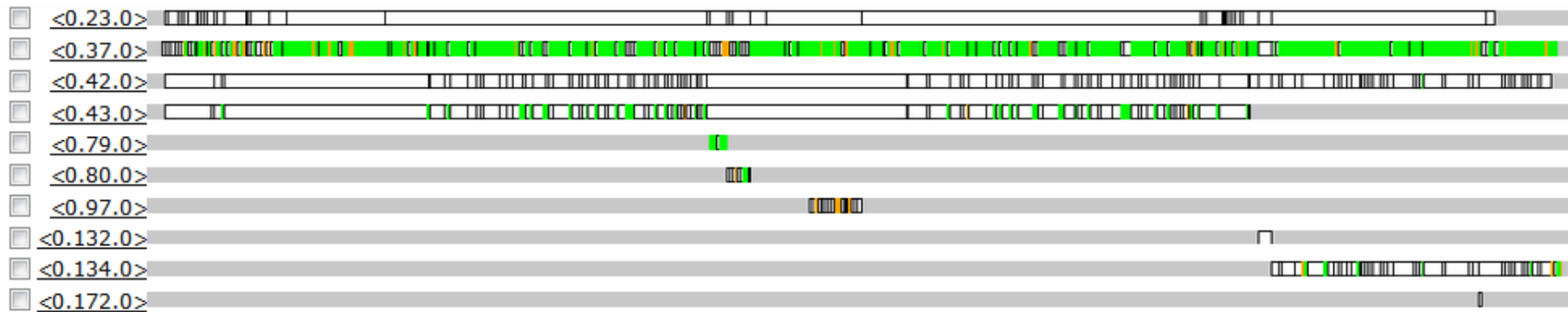# Percept2: compare process runnability
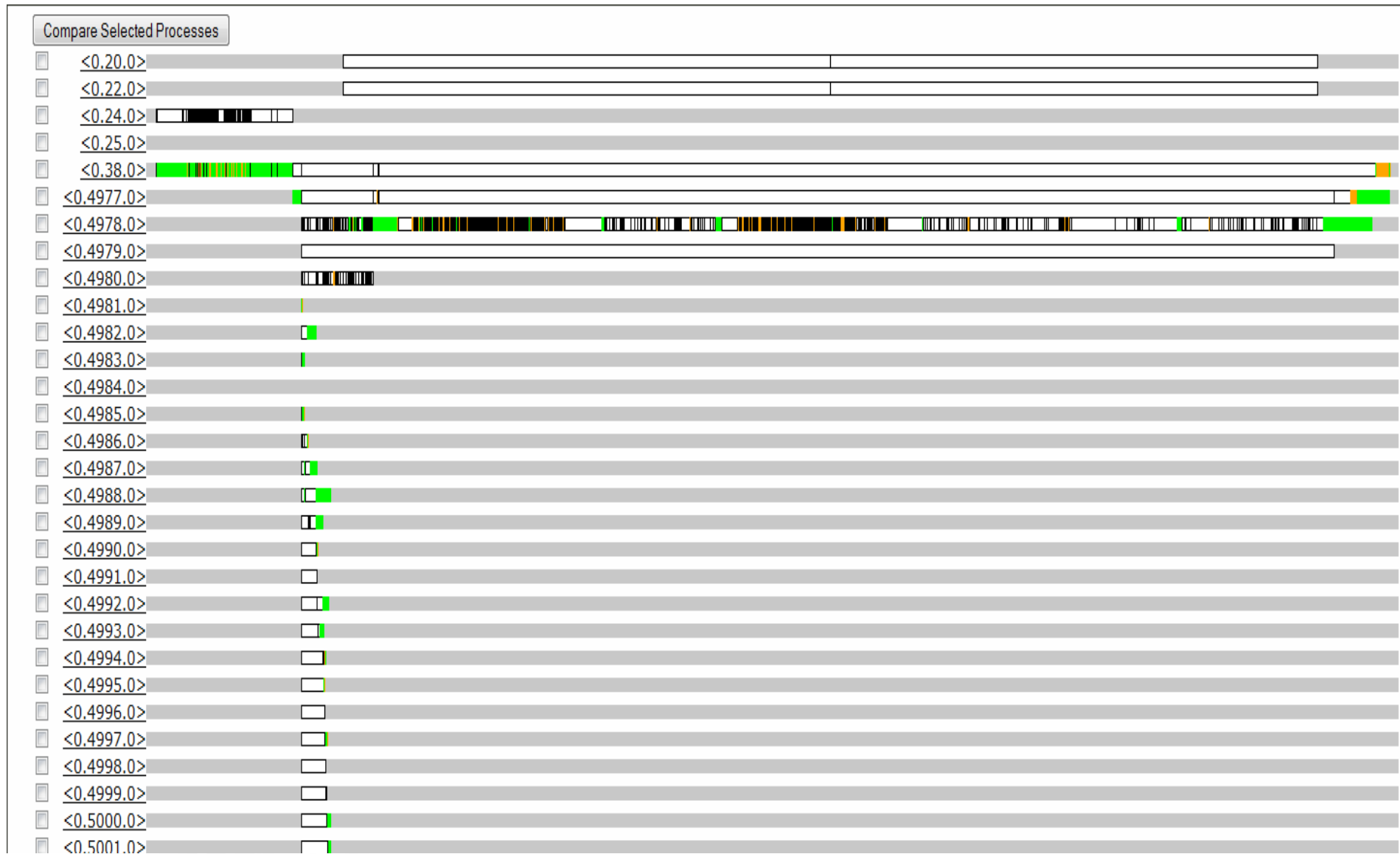
Percept2: compare process runnability
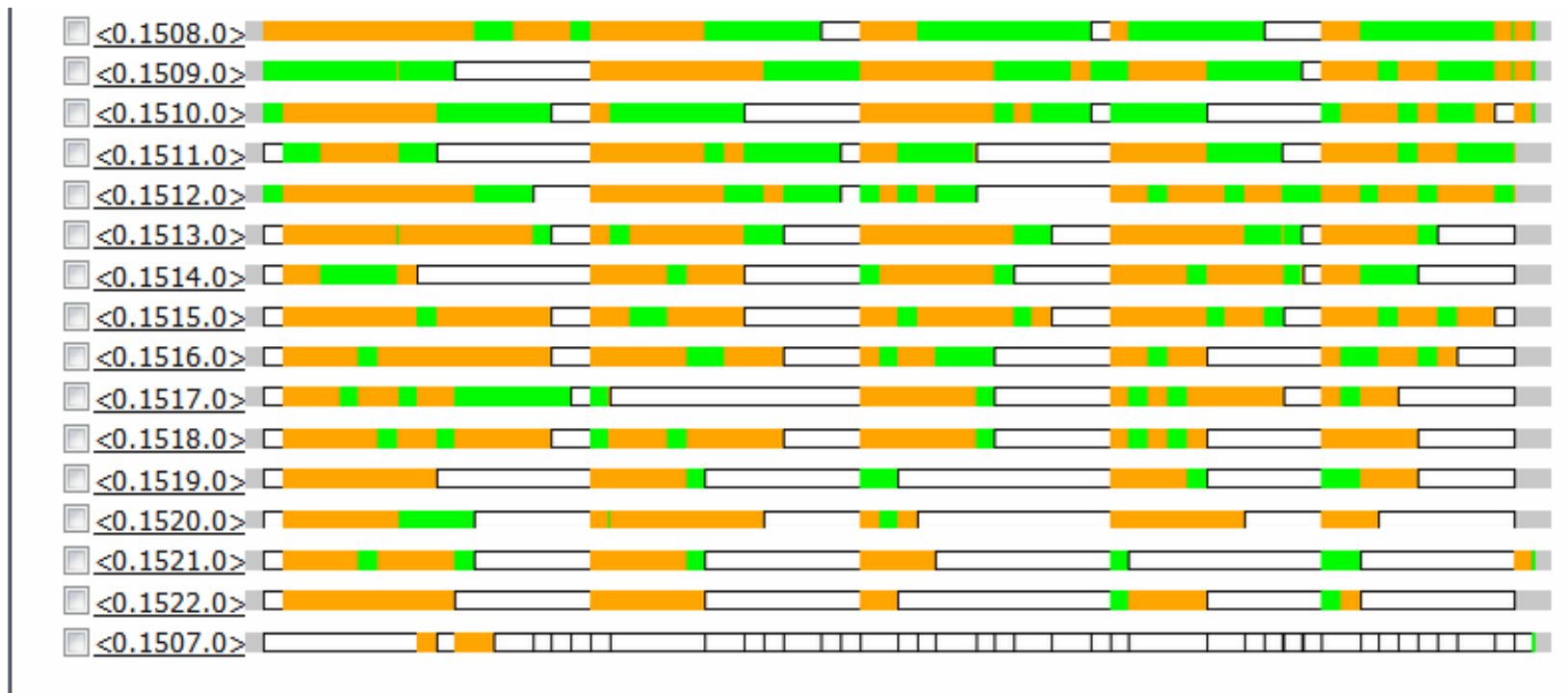
Green: running; Orange: runnable but not running; White: blocked.

RELEASE
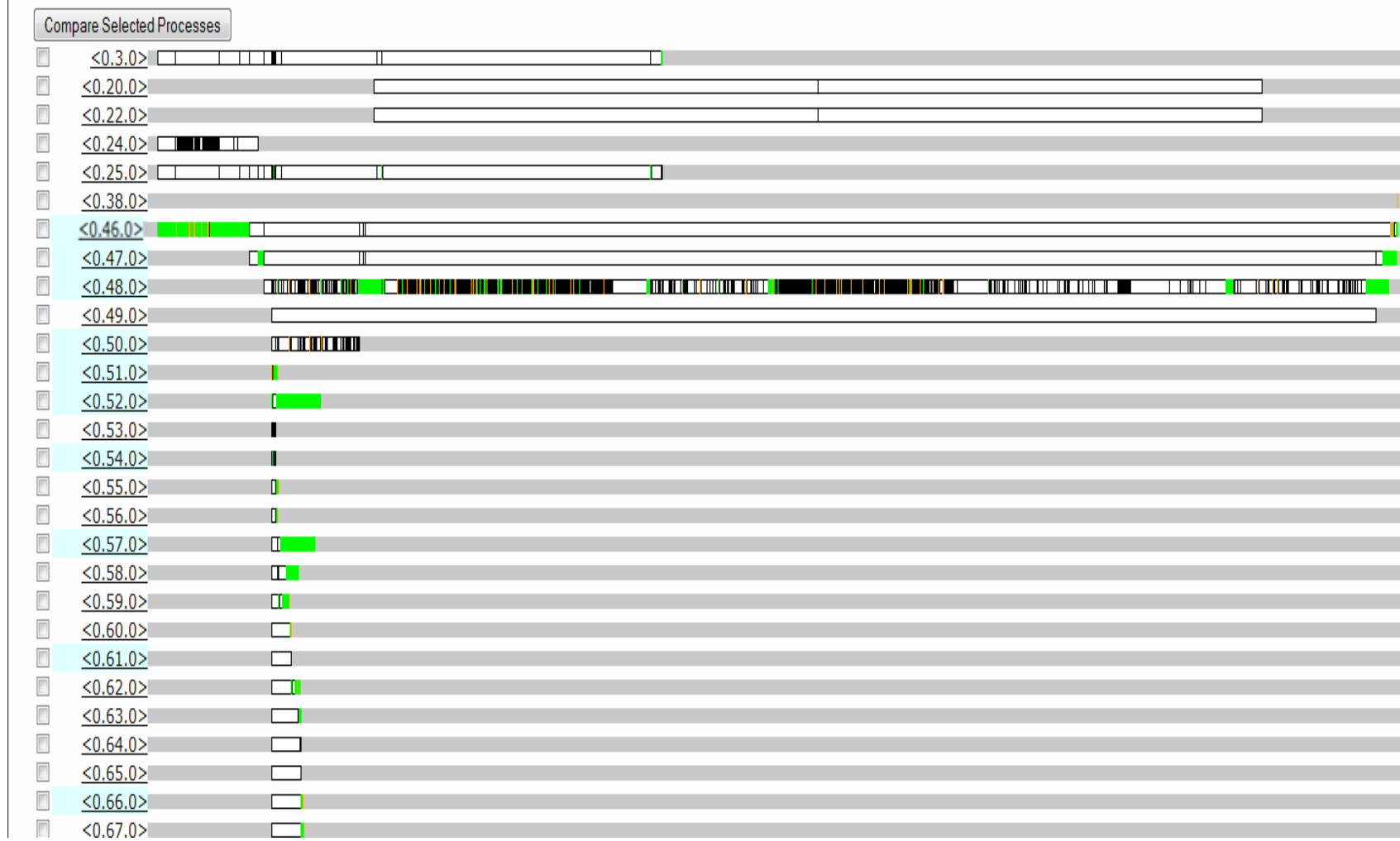
Percept2: compare process runnability

Heavily loaded processes/load imbalance

Very short-lived processes

RELEASE

Percept2: function profiling

A process' pid is hightlighted if it has a callgraph generated.

Percept2: function profiling

Pid	<0.46.0>
Name	undefined
Entrypoint	p2d:'-run/0-fun-0-'/0
Arguments	

		Timestamp	Profile Time
Timetable	Start{1370,980952,698000}		0.2030
	Stop{1370,980994,600004}		42.1050

Parent	<0.38.0>
Children	<0.47.0>
RQ_history	[]
{#msg_received, avg_msg_size}	{0,0}
{#msg_sent, avg_msg_size}	{0,0}
accumulated runtime (in secs)	2.7635
Callgraph/time	show call graph/time

percentage of total waiting time	total	mean	stddev	#recv	module:function/arity
100%	38.6250	6.4375	12.6777	6	dialyzer_cl:cl_loop/2
0%	0.0522	0.0001	0.0012	463	gen:do_call/4
0%	0.0241	0.0027	0.0051	9	code_server:call/2
0%	0.0023	0.0000	0.0000	1970	prim_file:drv_get_response/1

RELEASE

Percept2: function profiling
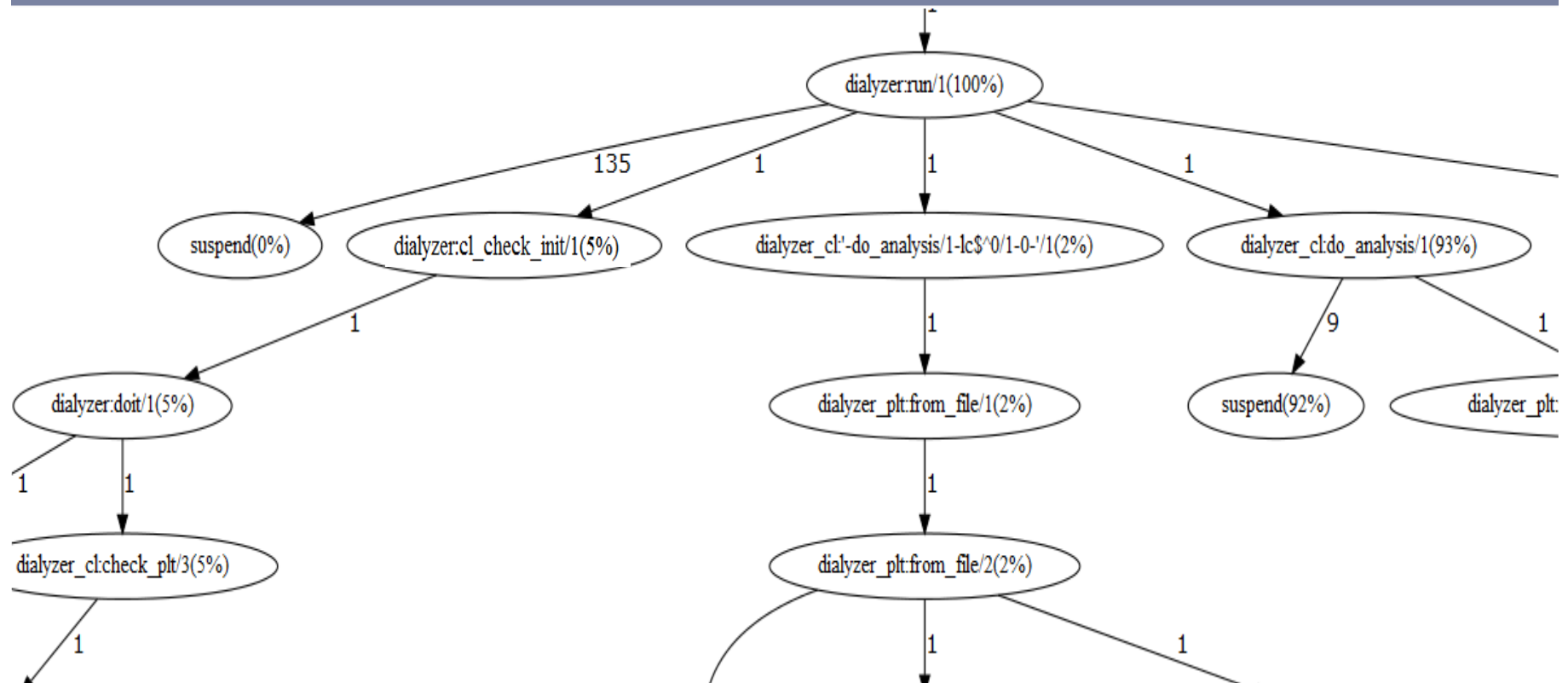
Percept2: function profiling

Accumulated time during which a function within this process is in a running state.

Pid	<0.46.0>
Entrypoint	p2d:'-run/0-fun-0-'/0
M:F/A	dialyzer:cl_check_init/1
Call count	1
Accumulated time (in secs)	2.0614
Callers	module:function/arity call count
	dialyzer:run/1 1
Called	module:function/arity call count
	dialyzer:doit/1 1

Percept2: scalability

- Parallelise Percept2 itself.

- Multiple log files ...

- ... that can be processed in parallel ...

- ... and integrated into a single result.

Percept2: scalability

percept - DEFOTP - DEFARGS
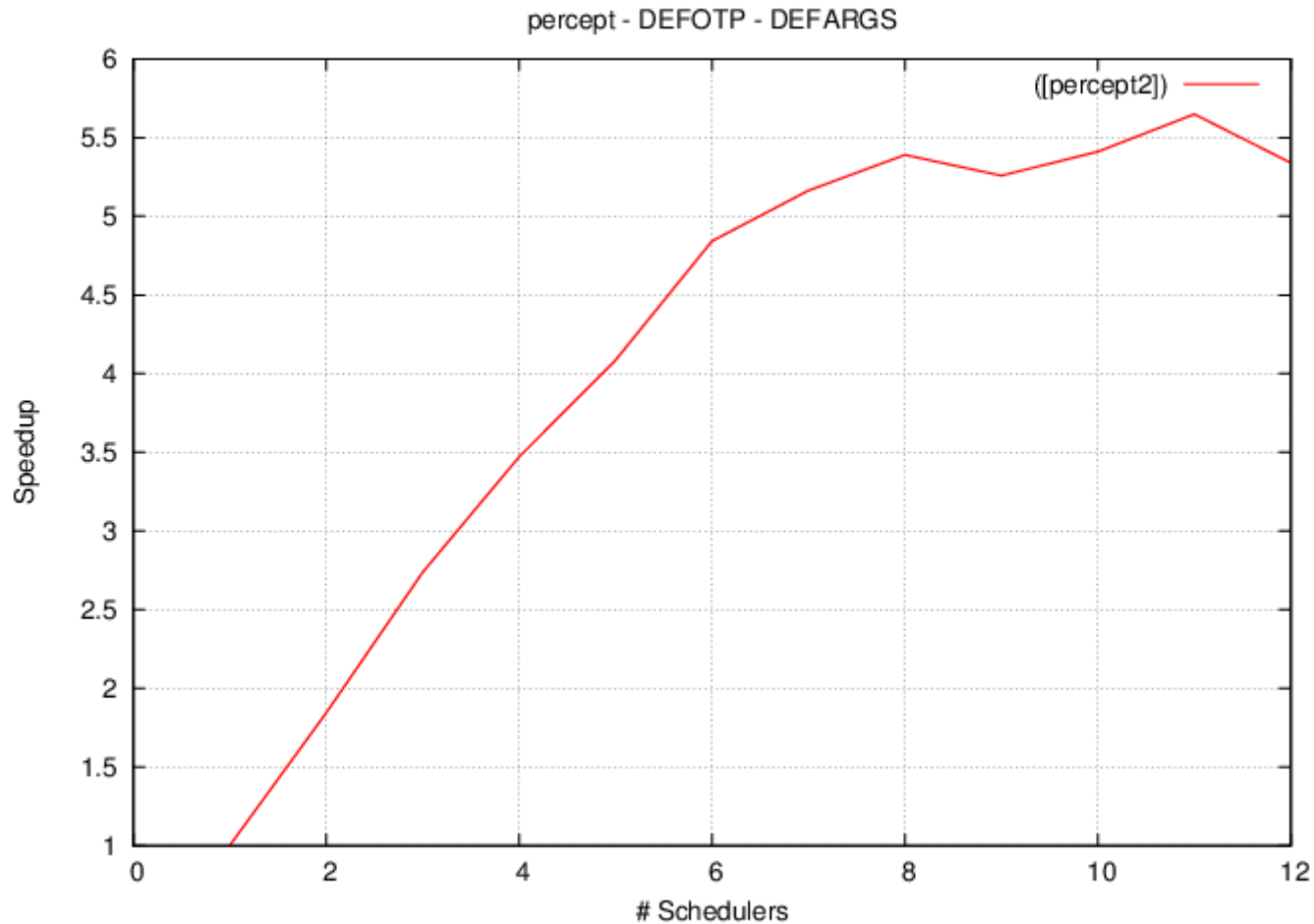
Scalability of Percept2 when analysing 5 trace files (total size: 1.36G, 11,008,609 traces messages) in parallel.

RELEASE

Percept2

More features are being added ...

RELEASE

Percept2: case study

Clone detection in Wrangler

| parse files | → | flatten trees | → | identify candidates | → | check candidates |

Percept2: case study
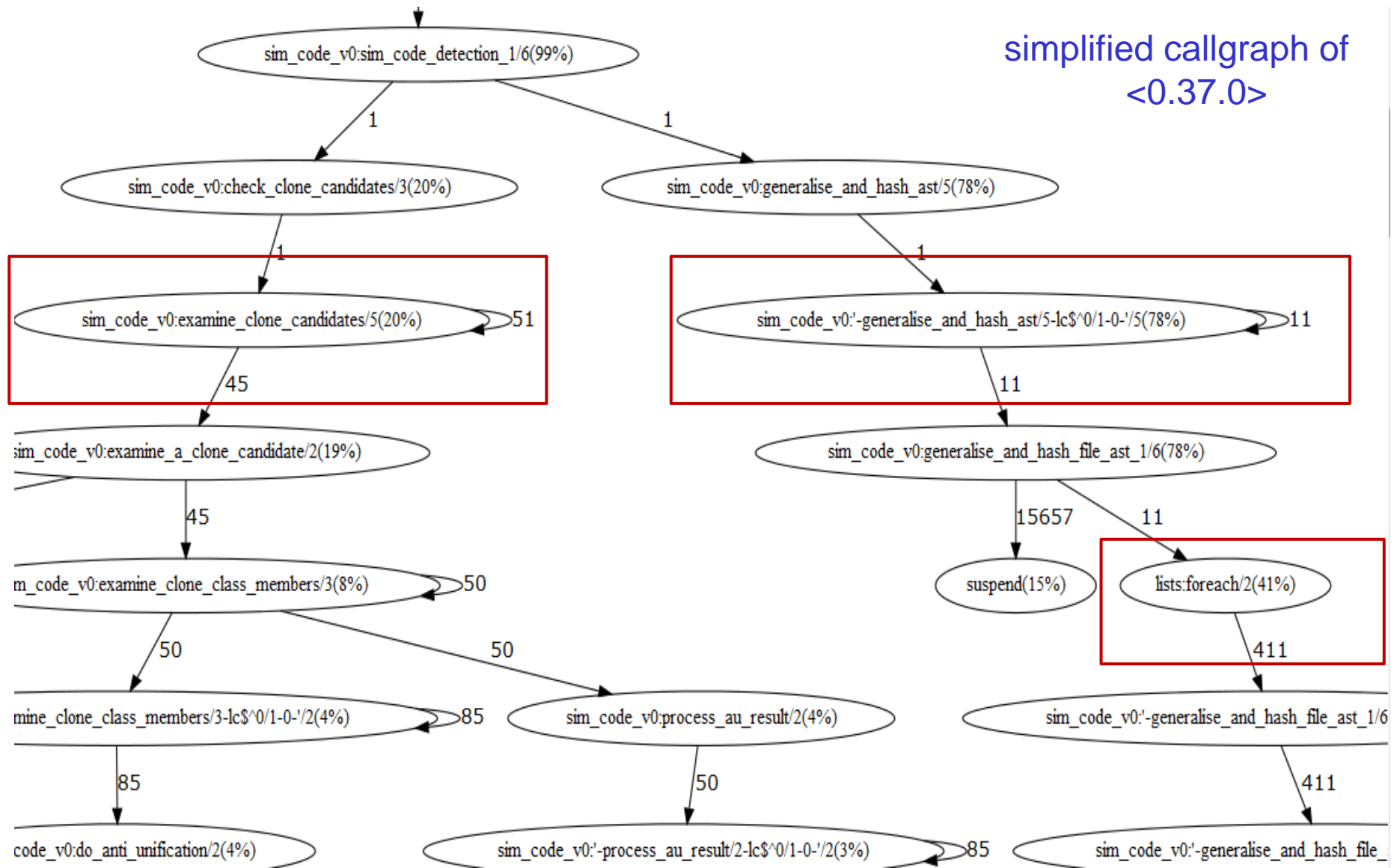
Percept2: case study

Percept2: case study

overloaded process
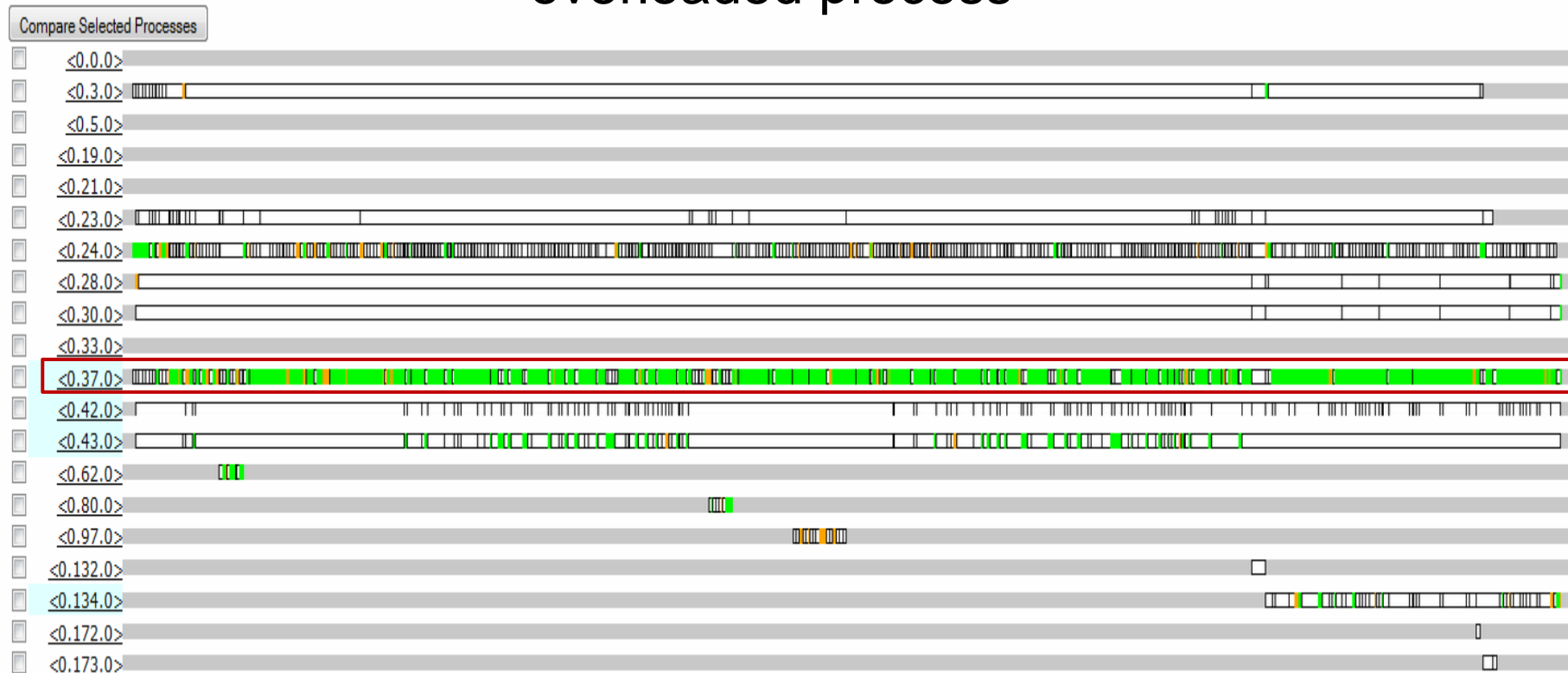
Percept2: case study

simplified callgraph of <0.37.0>

Percept2: case study
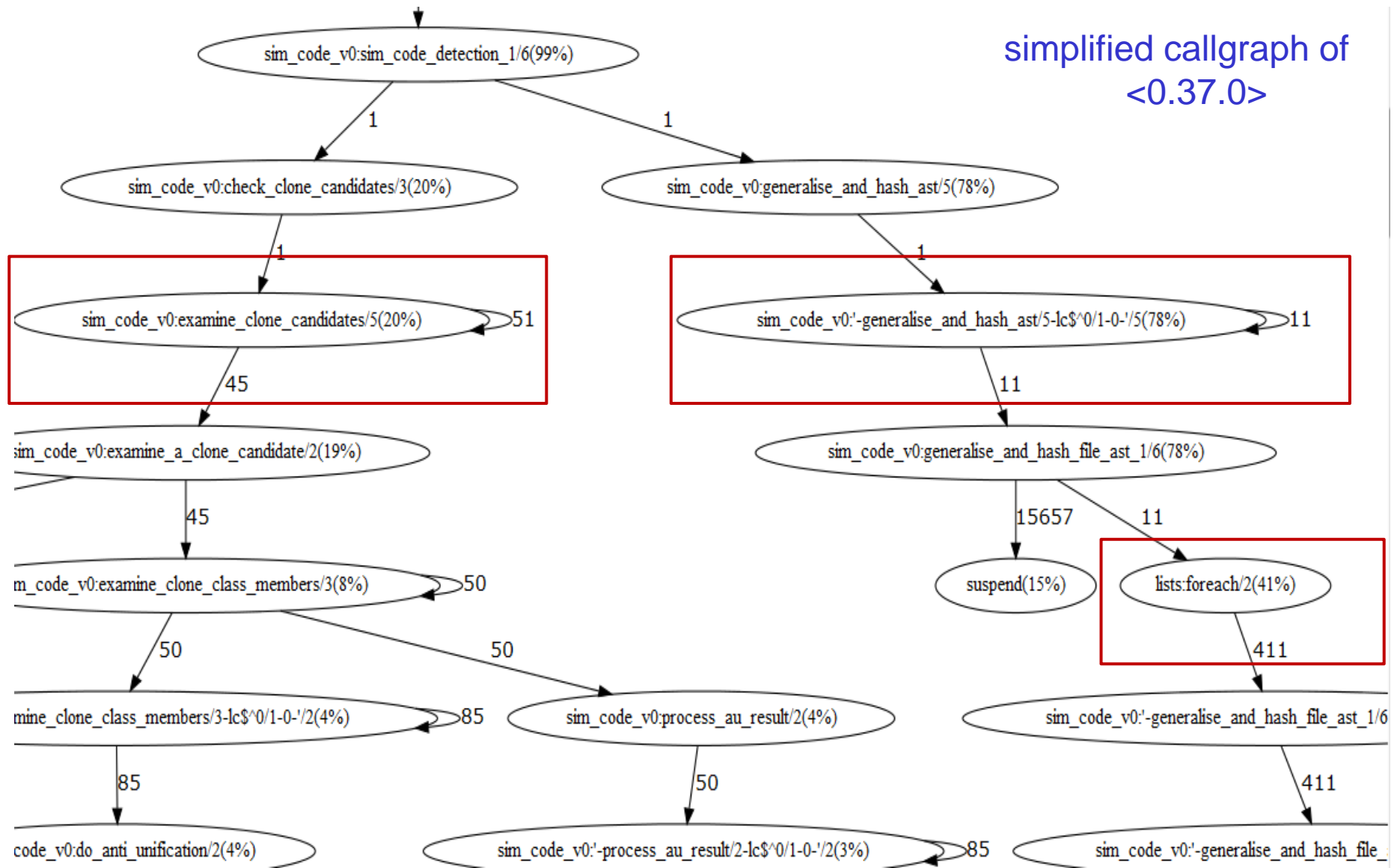
Refactoring #1: List comprehension to parallel map

```
[generalise_and_hash_file_ast_1(
    File, Threshold,ASTPid,true,SearchPaths,TabWidth)
    ||File <- Files]
```

```
para_lib:pmap(
    fun(File) -> generalise_and_hash_file_ast_1(
        File, Threshold, ASTPid,true,SearchPaths,TabWidth)
    end, Files)
```

RELEASE

Percept2: case study

Percept2: case study

Refac #2: sequential 'foreach' to parallel 'foreach'
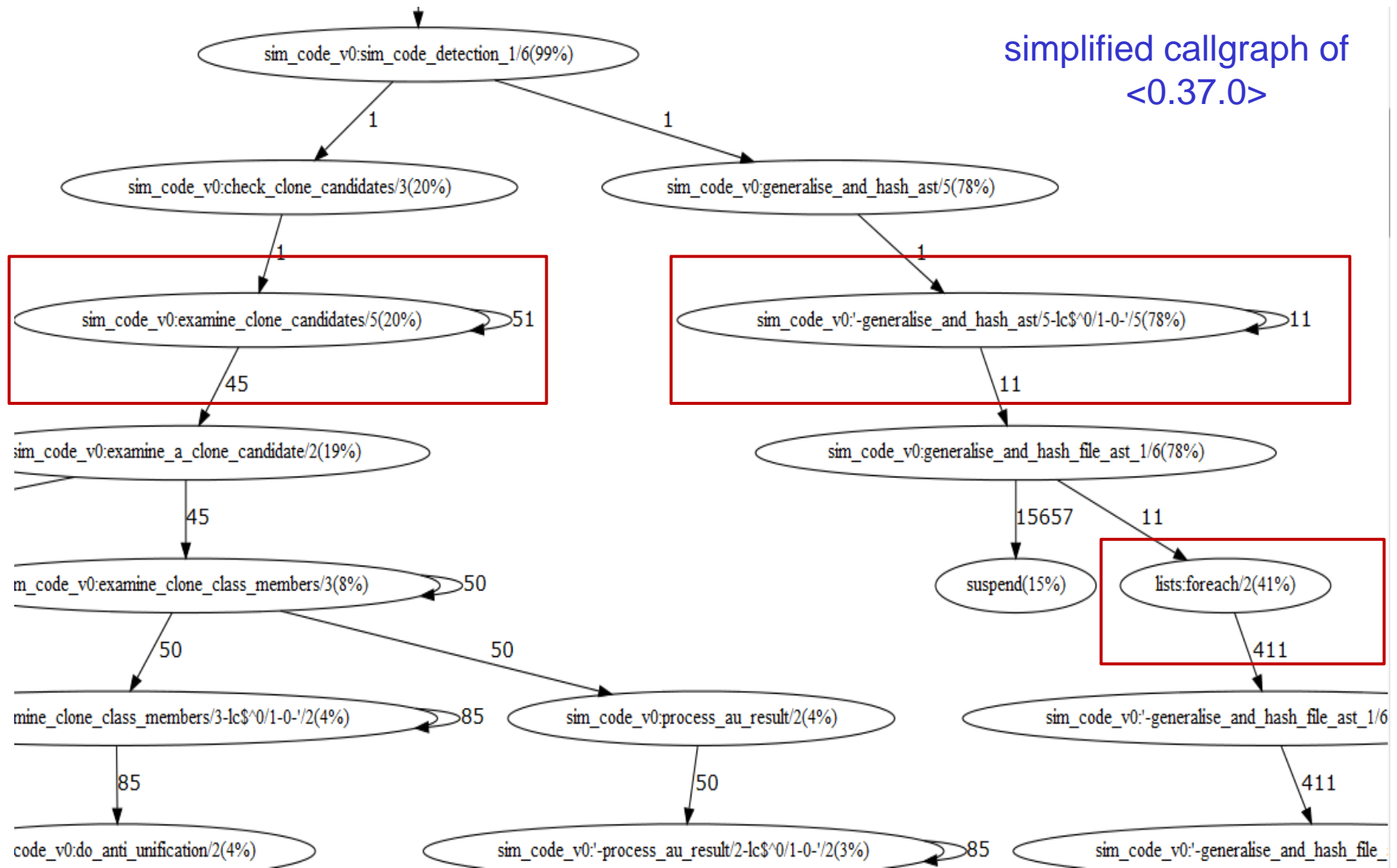
```
lists:foreach(fun (Form)-> F(Form) end, Forms)
```

⬇

```
para_lib:foreach(
        fun (Form)-> F(Form) end, Forms, ?Parallel)
```

Percept2: case study

simplified callgraph of
<0.37.0>

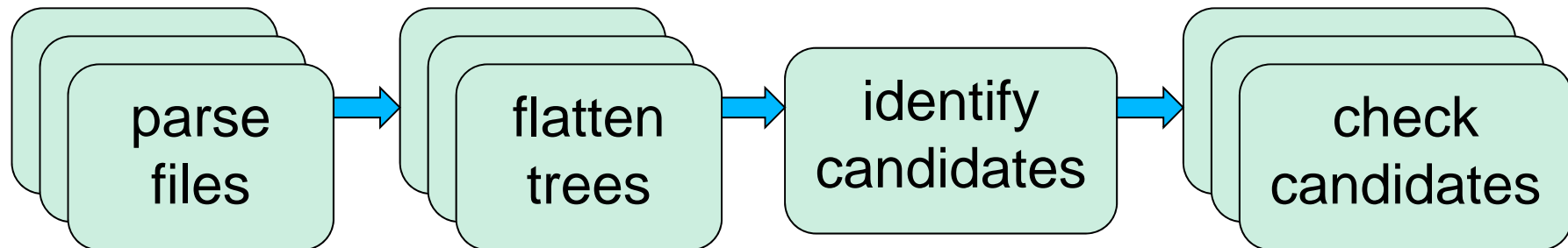# Refac #3: recursive function to parallel 'foreach'

```erlang
examine_clone_candidates([],_Thresholds,CloneCheckerPid,_Num) ->
  get_final_clone_classes(CloneCheckerPid);
examine_clone_candidates([C|Cs],Thresholds,CloneCheckerPid,Num) ->
  output_progress_msg(Num),
  NewClones = examine_a_clone_candidate(C, Thresholds),
  add_new_clones(CloneCheckerPid, {C, NewClones}),
  examine_clone_candidates(Cs,Thresholds,CloneCheckerPid,Num+1).
```
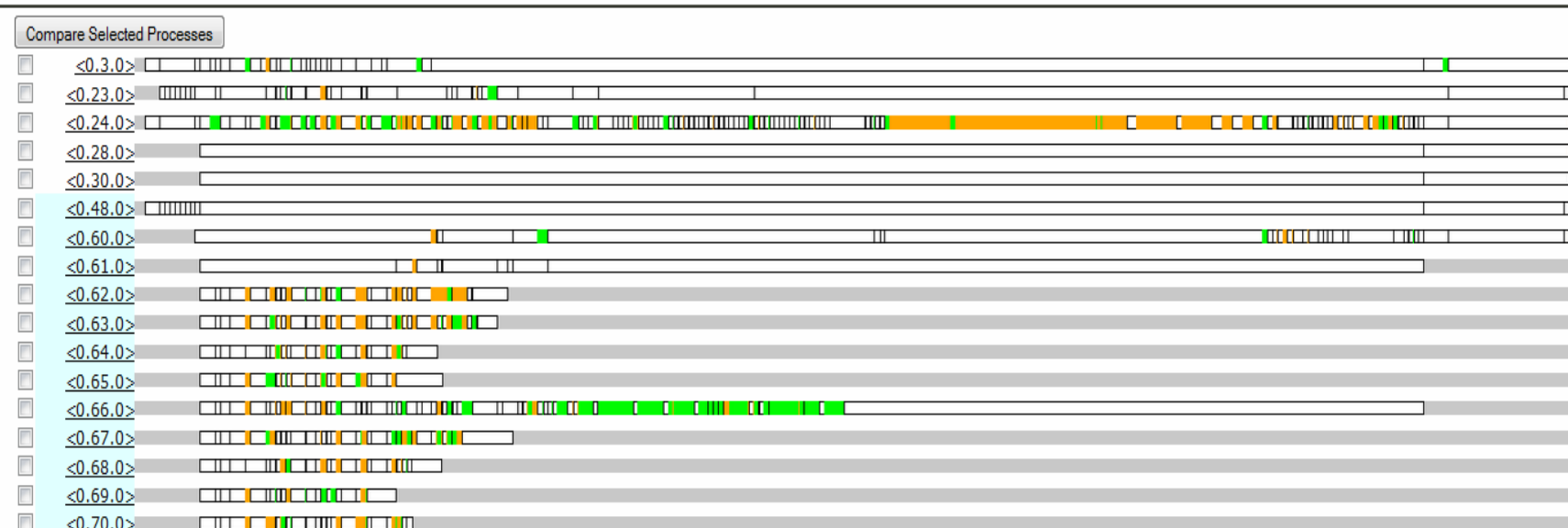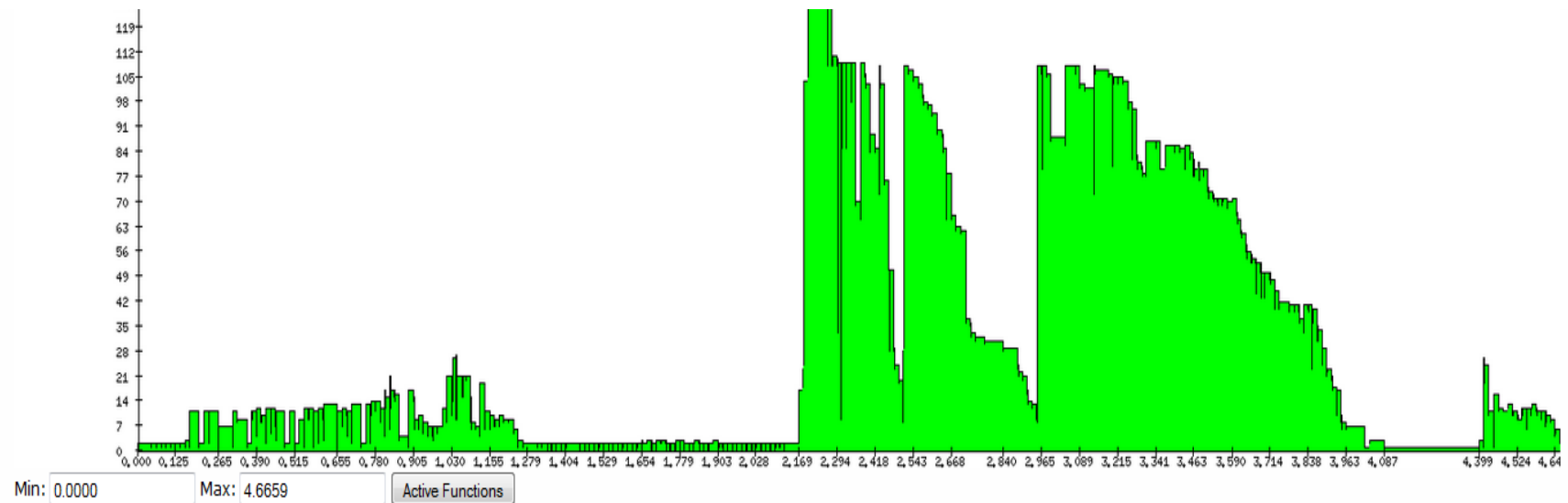
⬇

```erlang
examine_clone_candidates(Cs, Thresholds, CloneCheckerPid) ->
  NumberedCs = lists:zip(Cs, lists:seq(1, length(Cs))),
  para_lib:pforeach(fun({C, Nth}) ->
                      examine_a_clone_candidate(
                        {C,Nth},Thresholds,CloneCheckerPid)
                    end, NumberedCs),
  get_final_clone_classes(CloneCheckerPid).
examine_a_clone_candidate({C,Nth},Thresholds,CloneCheckerPid) ->
  output_progress_msg(Nth),
  NewClones = examine_a_clone_candidate(C, Thresholds),
  add_new_clones(CloneCheckerPid, {C, NewClones}).
```
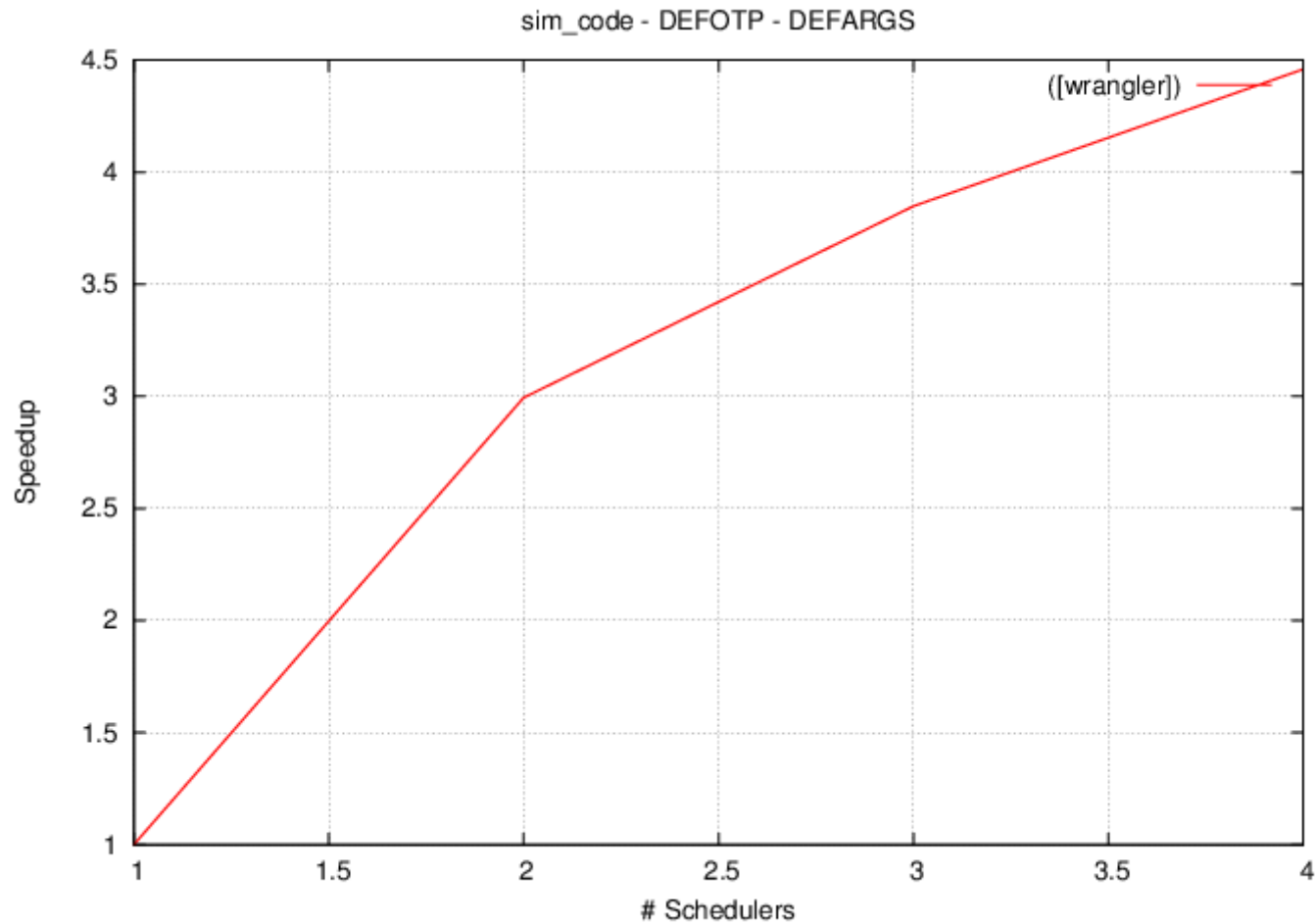
Percept2: case study

Clone detection in Wrangler

parse files → flatten trees → identify candidates → check candidates

RELEASE

Percept2: case study

Percept2: case study
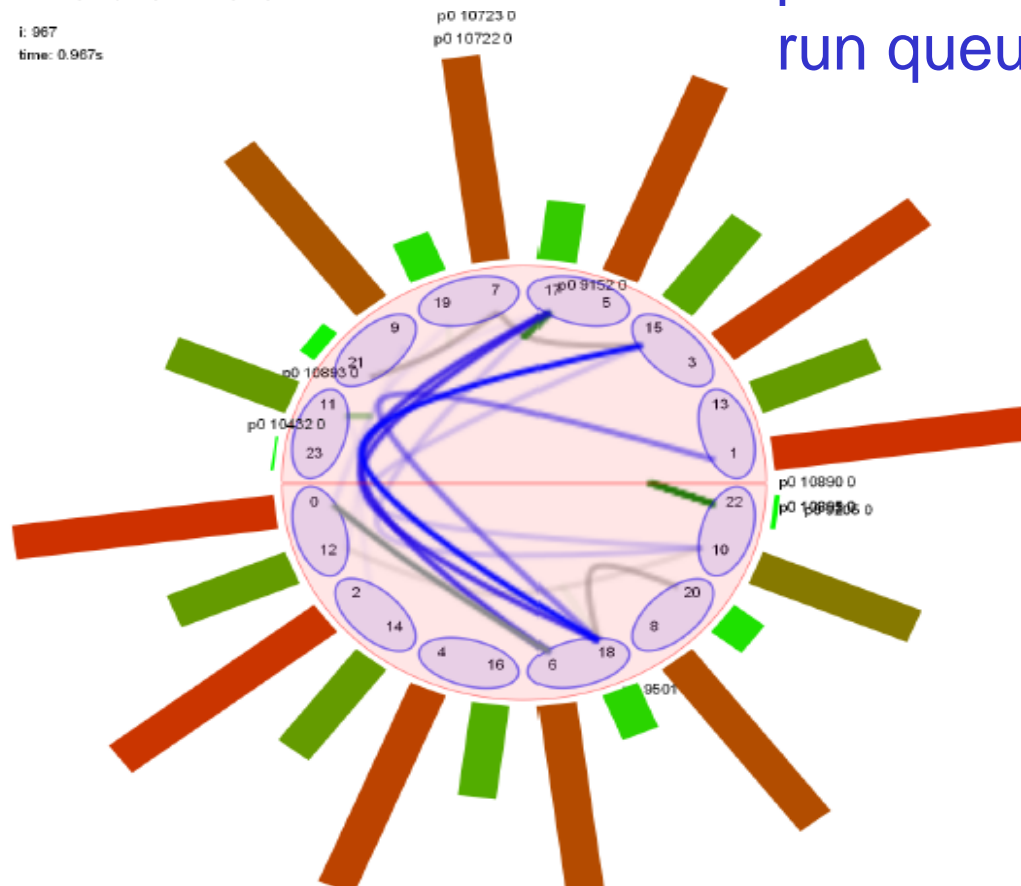

sim_code - DEFOTP - DEFARGS

([wrangler])

Other ongoing work

- Online profiling tool

    -- Devo: a web socket based online profiling and visualisation tool.

Other ongoing work

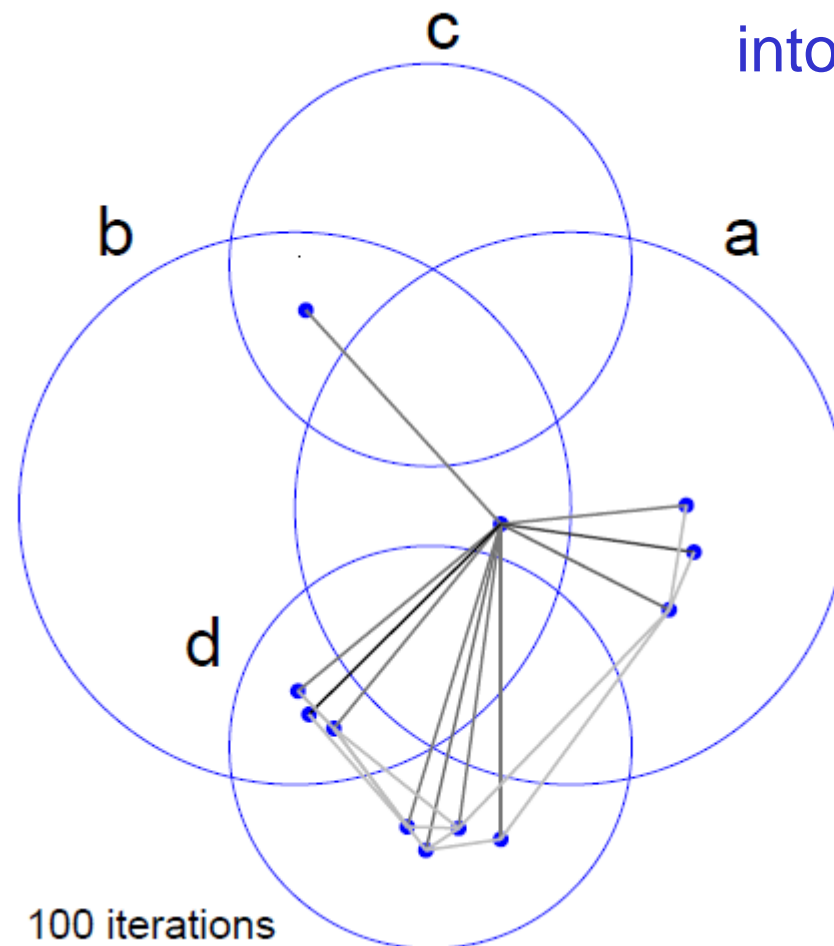- Online visualisation. process migration and run queue size

http://profsjt.blogspot.co.uk/2012/11/animating-multicore-erlang.html

Other ongoing work

- Online visualisation.

interaction between Erlang nodes grouped into s_groups.

100 iterations

Percept2

Get involved!

https://github.com/RefactoringTools/percept2

https://github.com/RefactoringTools/devo