Artifact documentation

This is the artifact documentation for the article *A Separation Logic for Heap Space under Garbage Collection*. The virtual machine archive file `artifact.ova` (4.1 GB) should be provided with this documentation.

The files we refer to in this documentation appear in the virtual machine ready to navigate, but can also be browsed online at:

https://gitlab.inria.fr/fpottier/diamonds/-/tree/master/project

List of claims

This is a list of the claims that are made in the article, and for each claim, where to find the corresponding proof.

Figure 3

Page 8, Figure 3, lists the following laws:

- Join|-> is an instance of an Iris rule (so, not really a claim – see https://iris-project.org/)
- Join<-| corresponds to `Lemma mapsfrom_join_equiv` in file `ph.v`
- Covariance<-| corresponds to `Lemma mapsfrom_weaken` in file `ph.v`
- Confront|-><-| corresponds to `Lemma exploit_mapsto_mapsfrom_multiplicity` in file `ph.v`
- Zero<> corresponds to `Lemma diamond_0` in file `wp_diamonds.v`
- Join<> corresponds to `Lemma diamond_split` in file `wp_diamonds.v`
- Zero†† is true by definition: `Notation "†† ls"` (in file `ph.v`) is an iteration over the set `ls`, which is empty in this case, so it is equal to `True`
- Join†† corresponds to `Lemma ddag_join` in file `ph.v`
- CloudEmpty corresponds to `Lemma ocloud_empty` in file `wp_free.v`. Note the inversion of order of arguments: "D cloudˆn P" in the paper corresponds to `ocloud P n D` in the development ("P" is generally named `antecedents`) ("D" is generally named `locs` or `ls`)
- CloudCons corresponds to `Lemma ocloud_cons` in file `wp_free.v`.
- Free corresponds to `Lemma supd_free_group` in file `wp_free.v`.
- Cleanup corresponds to `Lemma ph_cleanup_group` in file `ph.v`.

Page 10, Law FreeSingleton corresponds to `Lemma supd_free_singleton` in file `wp_free.v`

Page 12, Figure 4. Each rule corresponds to the corresponding `Lemma wp_*` in the corresponding `wp_*.v` file, for example `Lemma wp_skip` in file `wp_skip.v` for the rule named Skip.

Note that the rules are sometimes presented in weakest precondition form, that is to say `P |- wp c Q` instead of `{ P } c { Q }`. Sometimes the Hoare triple form is presented, but the latter is actually a notation for a weakest precondition formula. (See also the section "The Program Logic SL<>" of file `README.md`.)

Page 14, Theorem 4.1 corresponds to `Theorem adequacy` in file `adequacy.v`.

Page 14, Theorem 4.2 is about the validity of rules in Figures 3 and 4, which are detailed above.

Page 17-18, the reference-counting style versions of reasoning rules correspond to lemmas in file `rc.v` and have the same name as their more general counteparts, for example RC-Move corresponds to `Lemma wp_move` in file `rc.v`. Law RC-Free-Singleton corresponds to `Lemma supd_free_cleanup_singleton` in file `rc.v`.

Page 19, Figure 8, the specification for list copy corresponds to `Lemma list_copy_spec_combined` in file `examples_lists.v`.

Page 21, Figure 9, the specifications for stack functions create, push, and pop, correspond to lemmas `Lemma new_stack_spec`, `Lemma stack_push_spec`, `stack_pop_spec`, in file `stack_example.v`, and the law for deallocating a stack corresponds to `Lemma free_stack` of the same file.

Download, installation, sanity-testing

Option 1: virtual machine

The 4.1 GB file `artifact.ova` provided with this documentation can be imported into VirtualBox 6.1 through File > Import Appliance. It expands into a virtual machine that takes up some additional 11 GB. The VM's login and password are both "test", although no login should be required.

Inside the VM, one can open file e.g. `adequacy.v` by opening a terminal (Ctrl-Alt-T) and typing `coqide project/src/adequacy.v`. Replace `coqide` with `emacs` to run Proof General instead.

Option 2: installation for OPAM users

Reviewers familiar with Coq and OPAM may find it simpler to clone the following git repository and follow the instructions in file `project/INSTALL.md`.

`https://gitlab.inria.fr/fpottier/diamonds`

Option 3: from-scratch installation

We built a script to set up all softwares dependencies in a newly-created VM, and reviewers experiencing difficulties following `INSTALL.md` and wishing to avoid downloading the provided VM might try follow the script we used to build the virtual machine, as it is intended to install everything from scratch, even though it will modify one's opam existing configuration if any (as opposed to `INSTALL.md` which creates a local opam directory). In theory one should only need to run:

```
wget -O- https://gitlab.inria.fr/fpottier/diamonds/-/raw/master/project/setup-all.sh | bash
```

Which should download the project, its dependencies and compile everything.

This script is intended to be run in a new virtual machine, it has been tested with a fresh minimal installation of Ubuntu 20.04 running on a VirtualBox with 2048 MB of RAM and 15 GB disk, and took about 35 minutes.

The script may fail at the very end when the opam environment is out of sync, in which case one should open a new terminal and run `cd project; make`.

Evaluation instructions

Reviewers can check that all files compile and that no `Admitted` or `Axiom` remains; running `make` and `make axioms` completes that. The latter runs in particular `grep 'Admitted' src/*.v`.

One should also open some select `.v` files inside CoqIDE or Proof General and evaluate the whole file to check that no errors occurs and to verify that the objects and statements mentioned in the claims are what they are supposed to be.

Additional artifact description

We believe the repository is self-contained: the structure of the project, as well as a link to installation instructions, appear in the `README.md` file of the `project` directory of the repository: `https://gitlab.inria.fr/fpottier/diamonds/`