The scale benchmarks and evaluation

This folder contains all scripts and necessary folders to gather evaluation results.

File structure

The following are the components of the artifact:

- `bitcode_llvm14`: Contains all benchmarks mentioned in the paper.
- `yaml`: Contains all configurations files for each tool.
  - `clam.yaml`: the yaml file for running Clam/Crab.
    - Basically, it sets Crab to use Zones and delays widening by 1.
  - `cleam-stats.yaml`: the yaml file to allow Crab show statistics.
  - `mru_no_reduce.yaml`: the yaml file for MRU domain with no reduction.
  - `mru_heuristic_reduce.yaml`: the yaml file for MRU domain with heuristic reduction.
  - `mru_full_reduce.yaml`: the yaml file for MRU domain with full reduction.
- `data`: Stores all csv files collected by script `scripts/get_assert_results.py`.
- `outputs`: Includes the outputs made by each tool.
  - `rgn`: the results from the summarization domain.
  - `obj`: the results from the MRU domain.
- `paper_results`: Shows the results mentioned in the paper.
  - We do not directly generate a LaTeX version. The results mentioned in the paper will be printed as a log file.
- `scripts`: Keeps all scripts for running experiments.
  - `clean_up.sh`: Clean up the results in the previous folders.
  - `get_time_results.py`: Dump timing statistics from the `outputs` folder into `data/*.csv` files.
  - `get_paper_results.py`: Generate the paper table and the scatter plot based on the `data/*.csv` results.
  - `run_experiment.sh`: Runs evaluation for one abstract domain.
  - `run_precision_experiments.sh`: Runs all evaluation for all domains.

The bitcode code

We compiled all the source code into LLVM bitcode (`*.bc`). You can also find the human-readable LLVM IR (`*.ll`).