The R Project for Statistical Computing

Google Summer of Code Proposal

CI Optimization for R Package Performance Testing

Sagnik Mandal

March 31, 2025 v3.0

Contents

1. Project Info 3
2. Bio 3
3. Contact Information 4
4. Affiliation 4
5. Schedule Conflicts 4
6. Mentors 4
7. Coding Plan and Methods 5
    - 7.1. Project Scope 5
    - 7.2. Broad Tasks 5
    - 7.3. Detailed Coding Plan 6
        - 7.3.1. Package Minification 6
        - 7.3.2. Updating Atime to Use Cached Packages 6
        - 7.3.3. Artifact Caching & Retrieval Workflow 6
        - 7.3.4. Support for PRs from Forks 6
        - 7.3.5. Testing and Documentation 7
8. Timeline 7
    - 8.1. Pre-GSoC Period ($CURRENT_DATE - May 8, 2025) 7
    - 8.2. Community Bonding Period (May 8 - June 1, 2025) 7
    - 8.3. Week 1: Core Minification System (June 2-8) 7
    - 8.4. Week 2: Caching Integration (June 9-15) 7
    - 8.5. Week 3: Updating Atime and Workflow Integration (June 16 - June 22) 7
    - 8.6. Week 4: Support for Forks and Advanced Metrics (June 23 - June 29) 7
    - 8.7. Week 5: Testing and Documentation (June 30 - July 6) 8
    - 8.8. Final Week (July 7 - July 14) 8
9. Management of Coding Project 8
    - 9.1. Communication Strategy 8
    - 9.2. Testing 8
10. Management of Coding Project 8
    - 10.1. Communication Strategy 8
    - 10.2. Testing 8
11. Test Submissions 8

1. Project Info

Project title: Optimizing a performance testing workflow by reusing minified R package versions between CI runs

Short Title: CI Optimization for R Package Performance Testing

Idea Page: Idea Description on the R GSoC Wiki

2. Bio

I am Sagnik Mandal, a sophomore pursuing an Integrated Dual Degree in Materials Science and Technology at IIT (BHU), Varanasi. I have been a self-taught programmer since over 6 years now, and have also contributed to various open source projects, and used to maintain a couple of packages on the Arch Linux User Repository (AUR).

I first learnt about R, when I was following a online course on "Dealing with materials data : collection, analysis and interpretation", this course discussed data analysis using R for materials science. I have also been working under a professor at my institute on a project that involves data analysis and machine learning for material science applications.

I have used Github Actions in some of my personal projects, last year I applied to Joplin for GSoC to work on a similar project where I had planned to use Github Actions to automatically fetch build Joplin Plugins, along with preventing malicious code from being executed. While the project was shortlisted by Joplin, I couldn't make it to the final list of selected students, but this experience has helped me understand the working of Github Actions.

Other information about me can be found on my resume.

3. Contact Information

Name: Sagnik Mandal

Postal Address: 503, Satish Dhawan Hostel, IIT (BHU), Varanasi, Uttar Pradesh, India, 221005 (Timezone: UTC+5.5°)

Telephone(s): +91-7470989815°

Email(s): sagnik.mandal.mst23@iitbhu.ac.in°, acriticalcynic@outlook.com°

Other communications channels: Google Meet, Zoom°, Discord°, WhatsApp°

4. Affiliation

Institution: Indian Institute of Technology (Banaras Hindu University), Varanasi, India

Program: B.Tech. & M.Tech. (Integrated Dual Degree) in Materials Science and Technology

Stage of Completion: Sophomore, expected graduation in 2028

Contact to Verify: Dr. Chandan Upadhyay° (*email: cupadhyay.mst@iitbhu.ac.in°*)

5. Schedule Conflicts

I have my Summer Break from May 10 - July 10, 2025, so for majority of the project duration, I will be able to dedicate 35 hours a week to the project. I have already setup my build environment and through the tests I have also tackled parts of the project.

The first and the last week of the project will be a bit tight for me, as I will be travelling back home and then back to college, but I will try to make up for the lost time by working extra hours during the rest of the project duration.

6. Mentors

Evaluating Mentor: Anirban Chetia (anirban166) (*email: ac4743@nau.edu°*)

Co-Mentor(s): Toby Dylan Hocking (tdhock) (*email: toby.hocking@r-project.org°*)

Contact with Mentors: I have been in touch with both Anirban and Toby over Email and Github.

7. Coding Plan and Methods

7.1. Project Scope

The project will involve these three repositories:

1. data.table
   - `.ci/atime/tests.R` contains benchmark tests for various data.table functions, including performance regression tests across different versions. Currently, about 15 test cases install multiple versions of data.table, resulting in **approximately 28 different builds per run**.
   - `.github/workflows/performance-tests.yml` triggers the `Autocomment-atime-results` action to run performance tests on pull requests.
2. Autocomment-atime-results
   - `action.yml` defines the GitHub Action that sets up the R environment, installs required packages, and runs tests via the atime package. After testing, it logs execution time and comments on the PR with performance results.
3. atime
   - `R/versions.R` contains functions to build and install different versions of the package. Key functions include `atime_versions_install`, which installs various git versions with modified names; `pkg.edit.default`, which modifies package files to support multiple installations; and `atime_versions_exprs`, which creates benchmark expressions with correct package references.

7.2. Broad Tasks

1. **Package Minification**:
   - Develop a script that extracts an R package tarball, removes unnecessary files (e.g., vignettes, documentation, tests), and installs the minimal version using `R CMD INSTALL`.
2. **Updating Atime to Use Cached Packages**:
   - Update the `atime_versions_install` function to check for locally cached packages before building from source.
   - Use the `rappdirs` package for cross-platform support to locate the cache directory.
3. **Artifact Caching & Retrieval Workflow**:
   - Modify the `Autocomment-atime-results` workflow to check for and retrieve cached minified packages.
   - If an artifact is not available or fails integrity checks (using SHA-256), rebuild and upload it for future use.
4. **Support for PRs from Forks**:
   - Adapt the workflow to securely handle PRs from forks by using `pull_request_target` events or conditional artifact uploads to avoid exposing secrets.
5. **Testing and Documentation**:
   - Test the caching mechanism and minification process on various versions of data.table.
   - Document the process to assist future contributors.

7.3. Detailed Coding Plan

7.3.1. Package Minification

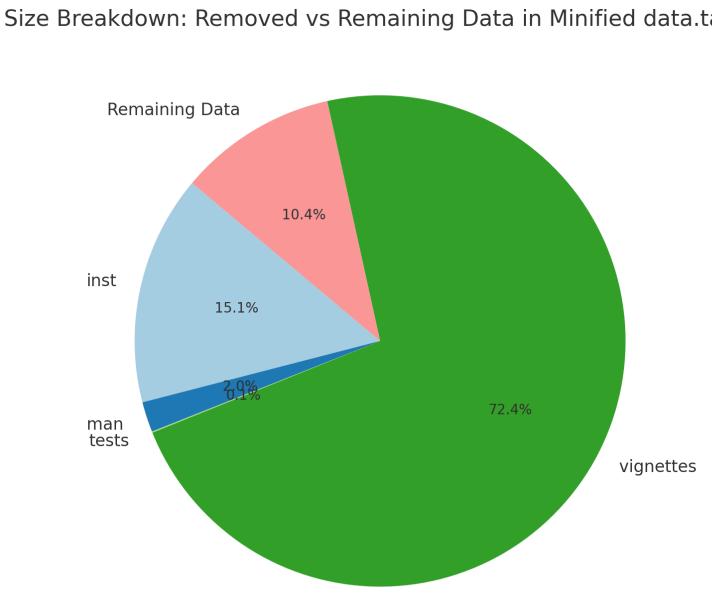

Figure 1: Package Minification

Removing unnecessary files and directories (such as `vignettes/` and `inst/`) can reduce the package size by up to 90%. This reduction directly translates into faster installations and reduced CI runtime.

7.3.2. Updating Atime to Use Cached Packages

Enhance the `atime_versions_install` function to:

- Check for the presence of a cached minified package based on a unique artifact name (version number and SHA-256 checksum).
- If found, verify its integrity using SHA-256 and install it directly.
- If not found, build the minified package from source and then cache it for future use.

This update will allow the package to be used in both CI and local environments seamlessly.

7.3.3. Artifact Caching & Retrieval Workflow

Integrate artifact caching into the GitHub Action by:

- Adding a step at the beginning of the workflow to download cached artifacts if they exist.
- Incorporating conditional logic to rebuild the package if artifact download fails or if the integrity check does not pass.
- Uploading newly built minified packages as artifacts at the end of the workflow for reuse in future runs.

7.3.4. Support for PRs from Forks

To securely handle PRs from forks:

- Use the `pull_request_target` event to ensure that repository secrets are only used when safe.
- Disable or modify artifact upload steps in forked PRs to prevent unauthorized access.
- Clearly document the workflow adaptations for contributors from forked repositories.

7.3.5. Testing and Documentation

- Implement unit tests for the minification script and the caching logic.
- Benchmark the CI runtime with and without the caching mechanism.
- Document the workflow configuration and the rationale behind each step to facilitate community contributions.

8. Timeline

Total Hours: 175 (35 hours/week × 5 weeks) **Project Duration:** June 2 - July 14, 2025

8.1. Pre-GSoC Period ($CURRENT_DATE - May 8, 2025)

- Set up repositories with mirrored historical branches for:
  - `data.table`
  - Fork of `Autocomment-atime-results`
  - Local `atime` development environment
- Develop an initial prototype for the package minification script.
- Audit current CI runtimes and resource usage.

8.2. Community Bonding Period (May 8 - June 1, 2025)

- Initial discussions with mentors and community to finalize project goals.
- Begin preliminary work on the minification script, despite limited availability in the first week due to travel.

8.3. Week 1: Core Minification System (June 2-8)

Objective: Implement and stabilize the package minification script.

- Finalize the file exclusion list based on empirical size analysis of 10 historical versions.
- Develop a versioned artifact naming convention.

8.4. Week 2: Caching Integration (June 9-15)

Objective: Integrate caching into the `atime` package.

- Modify `atime_versions_install` to check for cached packages.
- Implement SHA-256 verification for artifact integrity.
- Develop a fallback mechanism to rebuild packages if needed.

8.5. Week 3: Updating Atime and Workflow Integration (June 16 - June 22)

Objective: Complete updates in the atime package and integrate caching in the workflow.

- Finalize changes in the atime package for local and CI caching.
- Begin integrating artifact retrieval and upload steps in the GitHub Action.

8.6. Week 4: Support for Forks and Advanced Metrics (June 23 - June 29)

Objective: Adapt the workflow for PRs from forks and implement performance metrics.

- Update workflow to securely handle forked PRs.
- Add logging for download/install times and disk space usage.
- Simulate CI runs and document performance gains.

8.7. Week 5: Testing and Documentation (June 30 - July 6)

Objective: Final testing and comprehensive documentation.
- Execute unit and integration tests across various data.table versions.
- Document the process, performance improvements, and usage instructions.

8.8. Final Week (July 7 - July 14)

- Finalize documentation and code clean-up.
- Prepare project reports and submit final evaluations.

9. Management of Coding Project

9.1. Communication Strategy

- Hold bi-weekly sync calls with mentors via Google Meet/Zoom.
- Maintain a weekly blog to document progress and challenges.
- Use GitHub issues and pull requests to track incremental improvements and receive community feedback.

9.2. Testing

- Develop unit tests for the minification script and caching logic.
- Use multiple data.table versions to test the caching mechanism.
- Benchmark CI runtimes before and after the optimization to quantify time and resource savings.

10. Management of Coding Project

10.1. Communication Strategy

- Bi-weekly sync calls with mentors (Google Meet/Zoom etc.) to track progress and discuss blockers.
- Regular updates will be added on a weekly blog which I will maintain. This will also act as the report for the project.

10.2. Testing

- Add unit tests for the minification script and the caching workflow.
- Test the workflow with different versions of data.table to ensure that the cached packages are being used correctly.

11. Test Submissions

EASY: Script to Minify R Package°
- Wrote a script that is agnostic to the package name and version, and can be used to minify any R package tarball. It also installs the minified package using `R CMD INSTALL`, to ensure that the package is installable.

MEDIUM: Github Action to Minify R Packages and Upload as Artifact°
- Created a GitHub Action that reads the package name and version from the issue description, checks if the package is already minified, and if not, minifies the package and uploads it as an artifact. The action also installs the minified package using `R CMD INSTALL`.

- It then comments on the issue with package size details, and time taken to minify the package.

HARD: Supporting PRs from Forks in Autocomment-atime-results°

- Modified the `Autocomment-atime-results` workflow to support PRs from forks. The workflow is divided into two parts: one that runs the tests and uploads the results as artifacts, and another that downloads the results and comments on the PR with the results. The workflow has been tested to work with PRs from forks and non-forks.
- Then cloned the data.table repository, with all its historical branches required for the atime results, and tested the workflow to work with PRs from forks and non-forks.