DA Practical Assignments

Organization, Implementation, and Submission

Martin Grösbacher

Department of Computer Science
University of Salzburg

October 2026

Overview

1 Goals and Programming Model

2 GitHub Workflow

3 Submissions

4 Student Checklist

What are the practical assignments about?

- Implement distributed graph algorithms discussed in the lectures or exercises.

What are the practical assignments about?

- Implement distributed graph algorithms discussed in the lectures or exercises.
- Focus on translating **theoretical pseudocode into working C++**.

What are the practical assignments about?

- Implement distributed graph algorithms discussed in the lectures or exercises.
- Focus on translating **theoretical pseudocode into working C++**.
- Understand what information a node knows, computes, communicates and how the algorithm progresses.

What are the practical assignments about?

- Implement distributed graph algorithms discussed in the lectures or exercises.
- Focus on translating **theoretical pseudocode into working C++**.
- Understand what information a node knows, computes, communicates and how the algorithm progresses.
- The goal is mainly about **understanding the implementation**.

About the framework

- We provide a C++ framework based on the graph-based benchmark suite(**GBBS**).

About the framework

- We provide a C++ framework based on the graph-based benchmark suite(**GBBS**).
- It handles graph storage, parallel execution, message delivery, and round synchronization.

About the framework

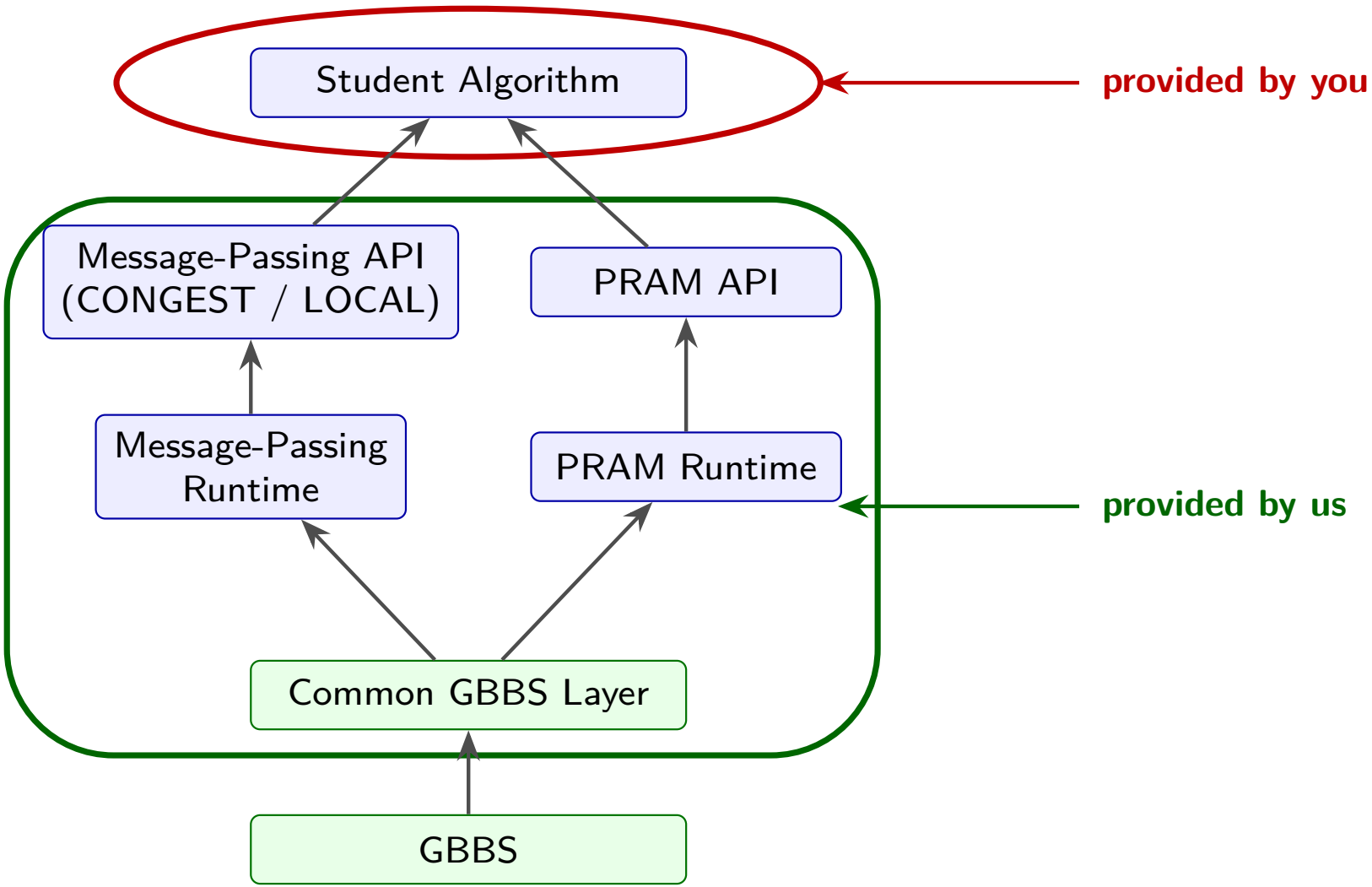
- We provide a C++ framework based on the graph-based benchmark suite(**GBBS**).
- It handles graph storage, parallel execution, message delivery, and round synchronization.
- You implement the algorithm's **local state and per-round behavior**.

About the framework

- We provide a C++ framework based on the graph-based benchmark suite(**GBBS**).
- It handles graph storage, parallel execution, message delivery, and round synchronization.
- You implement the algorithm's **local state and per-round behavior**.
- You should not need to write any code regarding threads, locks, memory barriers, or networking.

The exact programming interface will be provided with the assignments.

About the framework



Provided Repository

- We publish a **public course repository** containing the framework and assignment skeletons.

Provided Repository

- We publish a **public course repository** containing the framework and assignment skeletons.
- Each student creates a **private fork** on GitHub.

Provided Repository

- We publish a **public course repository** containing the framework and assignment skeletons.
- Each student creates a **private fork** on GitHub.
- Work in your own fork: implement, commit, and push changes.

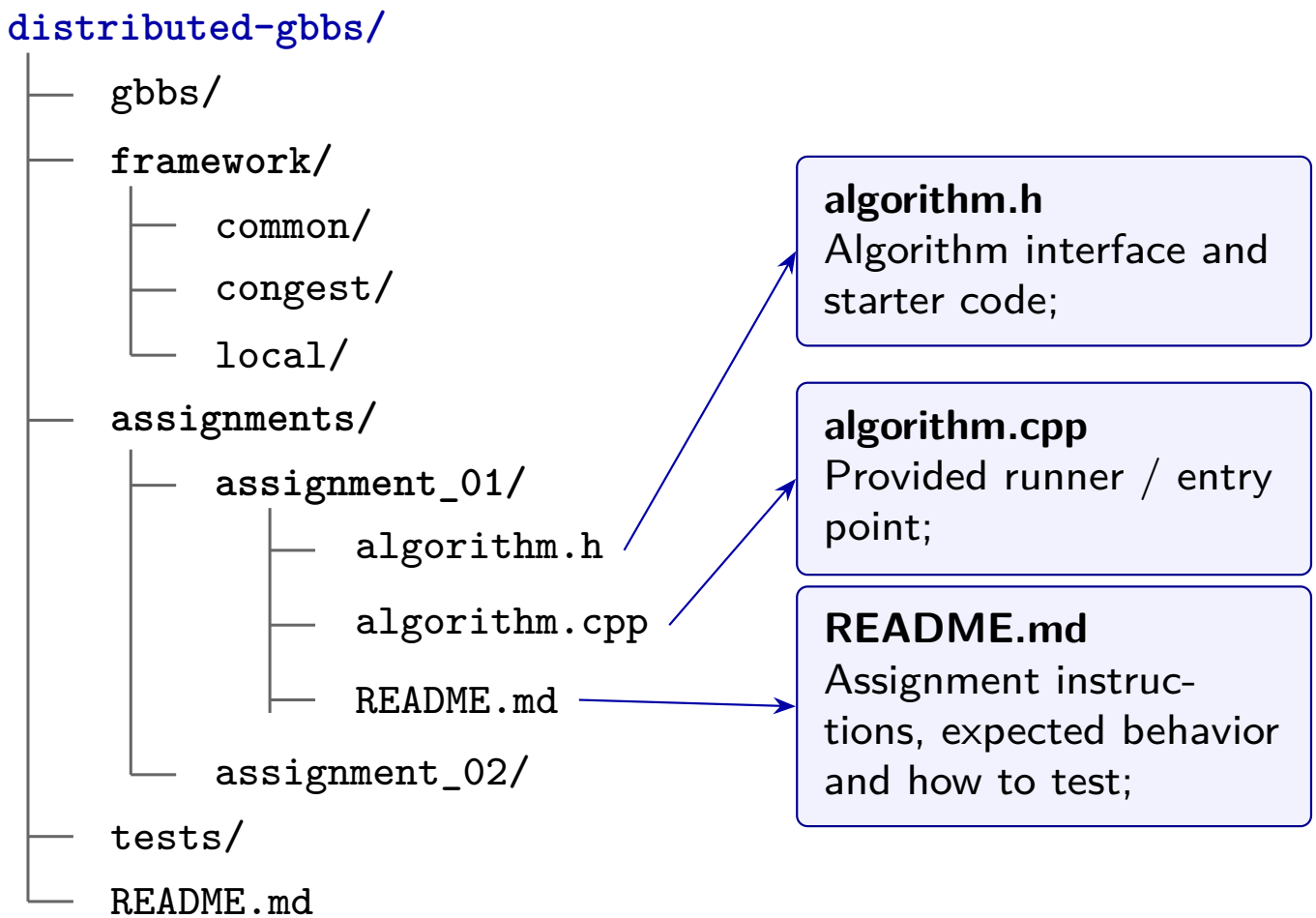
Provided Repository

- We publish a **public course repository** containing the framework and assignment skeletons.
- Each student creates a **private fork** on GitHub.
- Work in your own fork: implement, commit, and push changes.
- New assignments are released in the original course repository.

Provided Repository

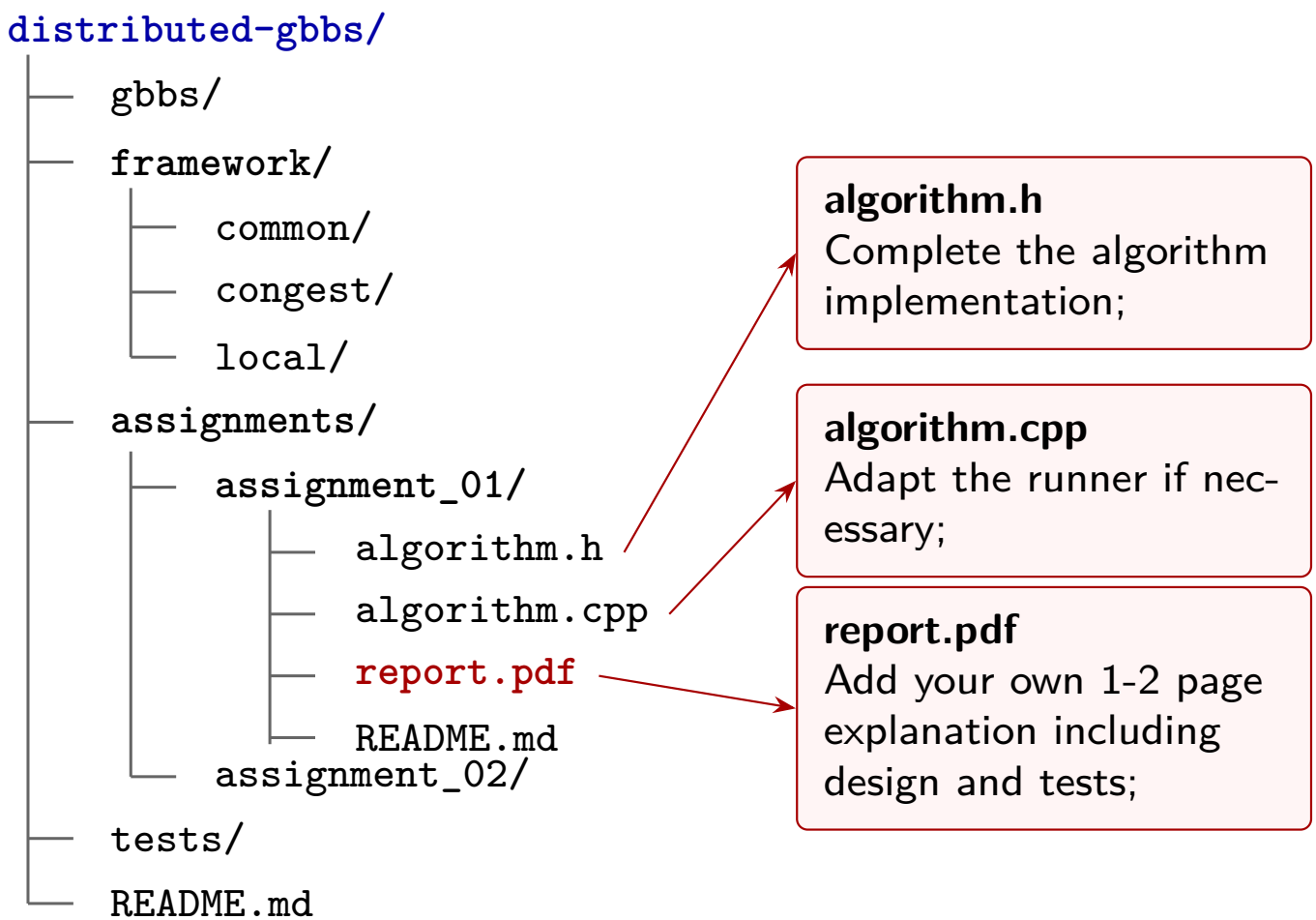
- We publish a **public course repository** containing the framework and assignment skeletons.
- Each student creates a **private fork** on GitHub.
- Work in your own fork: implement, commit, and push changes.
- New assignments are released in the original course repository.
- You fetch and merge these updates into your fork.

Provided Repository: At Assignment Release



We provide the structure and skeleton code.

Provided Repository: At Submission



You provide the implementation and the report.

Step by step Setup

1. Fork the course repository and make the fork private.
2. Send me a collaboration invitation. My Github username is "MartinGroesbacher".
3. Clone your fork and register the original as `upstream`:

```
git clone https://github.com/YOU/COURSE-REPO.git
cd COURSE-REPO
git remote add upstream https://github.com/COURSE/COURSE-REPO.git
git remote -v
```

New assignments

During an assignment:

```
git add assignments/assignment_01/
git commit -m "Implement assignment 1"
git push origin main
```

New assignments

During an assignment:

```
git add assignments/assignment_01/
git commit -m "Implement assignment 1"
git push origin main
```

When a new assignment is released:

```
git fetch upstream
git merge upstream/main
git push origin main
```

New assignments

During an assignment:

```
git add assignments/assignment_01/
git commit -m "Implement assignment 1"
git push origin main
```

When a new assignment is released:

```
git fetch upstream
git merge upstream/main
git push origin main
```

- Do not modify framework files unless explicitly instructed.

How do you submit?

1. Complete the algorithm implementation in your fork.

How do you submit?

1. Complete the algorithm implementation in your fork.
2. Write a short report about how the implementation works in the designated assignment directory.

How do you submit?

1. Complete the algorithm implementation in your fork.
2. Write a short report about how the implementation works in the designated assignment directory.
3. **Commit and push both code and report before the deadline.**

How do you submit?

1. Complete the algorithm implementation in your fork.
2. Write a short report about how the implementation works in the designated assignment directory.
3. **Commit and push both code and report before the deadline.**
4. At the deadline, we collect a snapshot of each registered public fork.

How do you submit?

1. Complete the algorithm implementation in your fork.
2. Write a short report about how the implementation works in the designated assignment directory.
3. **Commit and push both code and report before the deadline.**
4. At the deadline, we collect a snapshot of each registered public fork.

Important

Local, unpushed changes are **not** part of your submission. Any changes to your repository after the deadline will be ignored. Do not push executables.

Report

- Explain how your code implements the algorithm from the lecture.

Report

- Explain how your code implements the algorithm from the lecture.
- Describe the local state, messages, and round-by-round updates.

Report

- Explain how your code implements the algorithm from the lecture.
- Describe the local state, messages, and round-by-round updates.
- Explain why your implementation respects restrictions of the underlying distributed model.

Report

- Explain how your code implements the algorithm from the lecture.
- Describe the local state, messages, and round-by-round updates.
- Explain why your implementation respects restrictions of the underlying distributed model.
- Describe how you tested it, including problems you encountered.

Aim

A short, clear explanation of **your actual code**, not a generic restatement of the lecture.

Report format is pdf. The report should at most be 1-2 pages.

AI usage

- **AI assistance is permitted**, including help with programming.

AI usage

- **AI assistance is permitted**, including help with programming.
- You remain responsible for the correctness of your submission.

AI usage

- **AI assistance is permitted**, including help with programming.
- You remain responsible for the correctness of your submission.
- You must be able to explain and reason about **all submitted code**.

AI usage

- **AI assistance is permitted**, including help with programming.
- You remain responsible for the correctness of your submission.
- You must be able to explain and reason about **all submitted code**.
- Permitted AI assistance does **not** mean copying of other students' solutions. Plagiarism will mean 0 points for every submission involved.

Short oral checks

- Students may randomly be invited to a **short oral discussion** of their submission.

Short oral checks

- Students may randomly be invited to a **short oral discussion** of their submission.
- The purpose is to verify **understanding**, not whether a particular tool was used.

Checklist for submissions

1. Your GitHub fork is private.

Checklist for submissions

1. Your GitHub fork is private.
2. You invited me as a collaborator to your fork.

Checklist for submissions

1. Your GitHub fork is private.
2. You invited me as a collaborator to your fork.
3. Your implementation builds and runs using the provided instructions.

Checklist for submissions

1. Your GitHub fork is private.
2. You invited me as a collaborator to your fork.
3. Your implementation builds and runs using the provided instructions.
4. The required report is present in the assignment directory.

Checklist for submissions

1. Your GitHub fork is private.
2. You invited me as a collaborator to your fork.
3. Your implementation builds and runs using the provided instructions.
4. The required report is present in the assignment directory.
5. All relevant changes are **committed and pushed**.

Checklist for submissions

1. Your GitHub fork is private.
2. You invited me as a collaborator to your fork.
3. Your implementation builds and runs using the provided instructions.
4. The required report is present in the assignment directory.
5. All relevant changes are **committed and pushed**.
6. You can explain how the code follows the requested algorithm.

Questions?

Questions?

Repository URL, assignment deadlines, and detailed instructions will be provided in due time on the course website.