Project ARARE

Adaptive Real-Time Analysis & Re-evaluation Engine for Resilient University Timetable Scheduling

A Resilience-First Scheduling Framework

Presented by: Sumit shukla, Om marathe, Tej singh, Utkarsh singh.

3rd Year B.Tech CSE

01. The Problem Statement

Traditional University Timetabling Problems (UTP) are solved statically.

The Gap:

- **Fragility:** Systems fail when faculty are absent or rooms are unavailable.
- **Regeneration Overkill:** Minor changes often require recomputing the entire timetable.
- **Black-Box Logic:** No explainability when constraints are relaxed.
- **Static Nature:** Assumes ideal operating conditions.

02. Project Abstract

ARARE proposes a **resilience-first** scheduling framework.

- **Clash-Free:** Ensures absolute invariants (Faculty/Room conflicts).
- **Localized Re-optimization:** Instead of full regeneration, it performs "targeted repair" during disruptions.
- **Stability > Optimality:** Prioritizes keeping the existing schedule consistent over finding a perfect new one.

03. Motivation & Objectives

Why ARARE?

- To handle hierarchical availability (Global vs. Room-specific).
- To introduce **Constraint Criticality Hierarchy**.

Core Objectives:

1. Model university entities precisely.
2. Implement partial re-optimization algorithms.
3. Provide explainable constraint relaxation (No silent failures).

04. Technical Architecture (Phase 1)

Built for speed, scalability, and modularity.

- **Framework:** Spring Boot (Java)
- **Database:** MySQL
- **Core Entities:** - `College` ➔ `Building` ➔ `Room`
  - `Faculty` ➔ `Subject` ➔ `Timeslot`
  - `Allocation` (The bridge entity)

05. Constraint Modeling

Hard Constraints (Non-Negotiable)

- No Room Clashes.
- No Faculty Clashes.
- Capacity Matching.
- Manual Pre-allocation Locks.

Soft Constraints (Optimization Targets)

- Minimized Faculty Movement.
- Workload Balancing.
- Gap Optimization.

06. The ARARE Scheduling Flow

1. **Initialization:** Loading metadata (Rooms, Faculty, Subjects).
2. **Locking:** Applying manual pre-allocations.
3. **Hard Constraint Mapping:** Establishing the baseline clash-free grid.
4. **Soft Constraint Pass:** Iterative improvement of the schedule.
5. **Validation:** Rule-based checking before finalization.

07. Real-Time Re-evaluation Strategy

What happens when a disruption occurs?

- **Step 1:** Identify "Affected Nodes" only.
- **Step 2:** Freeze "Safe Zones" (unaffected allocations).
- **Step 3:** Perform Localized Re-optimization.
- **Step 4:** If no solution exists, trigger **Graceful Degradation** via the Relaxation Hierarchy.

08. Literature Review & Research Gap

Current research (Genetic Algorithms, Ant Colony Optimization) focuses on **optimality**.

The ARARE Advantage:

- **Constraint Criticality Hierarchy:** Not all constraints are equal.
- **Stability-Oriented:** Minimal disruption to the user experience.
- **Explainable Outputs:** The system tells you *why* a constraint was relaxed.

09. Conclusion & Future Scope

Current State:

A robust university-level engine ensuring operational continuity.

Future Phases:

- Domain-agnostic abstraction (General Resource Allocation).
- Integration with Real-time IoT sensors for room occupancy.

Thank You!

Questions?

Project ARARE: Resilience through Adaptation.