2025

CS 4402

Comparative Programming Languages

LEARNING JOURNAL 8

SANA UR REHMAN

INSTRUCTOR: JIM CASALE

This week's focus was on an entirely new programming paradigm: logic programming. My process involved a thorough review of the Unit 8 learning materials, covering the foundations of logic programming, unification, resolution, and the syntax of Prolog. To apply these concepts, I researched and composed my discussion forum post, where I investigated how Prolog is uniquely suited for solving complex constraint-satisfaction problems. Following this, I completed the self-quiz to test my immediate comprehension and took the review quiz to begin consolidating knowledge for the final exam.

My initial reaction to logic programming was that it felt like a significant mental shift. Coming from imperative languages, where I explicitly define *how* to solve a problem step-by-step, the declarative nature of Prolog—where you define *what* is true—was both fascinating and challenging. It required me to stop thinking about loops and variables and start thinking about facts, rules, and relationships.

CHALLENGES AND DISCOVERIES

The most challenging aspect was internalizing the concepts of unification and resolution. Trusting Prolog's built-in backtracking engine to find solutions, rather than trying to manage the search process myself, was a difficult habit to break. The provided readings laid the groundwork, but the real "aha" moment came while working on the discussion post.

I chose to explore university timetabling, which I learned is a classic combinatorial problem. Researching this application revealed how Prolog, especially with Constraint Logic Programming (CLP) extensions, can elegantly solve problems that would be incredibly cumbersome to code imperatively. Instead of writing complex algorithms to check for conflicts, one can simply declare the constraints: a student cannot be in two places at once, or a room cannot exceed its capacity. As noted in survey literature, this fusion of logic programming and constraint solving provides practical power for a wide array of complex problems (Jaffar & Maher, 1994). This application demonstrated that the "magic" of Prolog isn't magic at all, but a powerful combination of pattern matching (unification) and a depth-first search (backtracking).

GAINED SKILLS AND PERSONAL REALIZATIONS

This week, the primary skill I gained is *declarative thinking*. I am learning to model a problem's domain by defining its facts and rules, which is a powerful abstraction. This approach is highly applicable in fields like database design (SQL is largely declarative) and artificial intelligence.

As a learner, this unit highlighted my own cognitive biases. I am strongly conditioned to think imperatively. This challenge forced me to approach problem-solving from a completely different angle, reinforcing the core purpose of this "Comparative Programming Languages" course. The self-quiz and review quiz acted as my primary feedback mechanisms, confirming whether I had correctly understood the abstract mechanics of how Prolog finds an answer. For example, seeing a query trace helped solidify how recursion and backtracking work in tandem.

The ideas for this week are directly applicable. While I may not write a timetabling system, understanding logic programming provides a new mental tool for breaking down complex problems based on rules and relationships, which is valuable in any software design context (Rudová et al., 2003).

The most important thing I am still thinking about is this separation of *logic* from *control*. Being able to define the "logic" of a problem and trust the engine to handle the "control" (the search) is an incredibly powerful concept that I am just beginning to appreciate.

References

Jaffar, J., & Maher, M. J. (1994). Constraint logic programming: A survey. *Journal of Logic Programming, 19-20*, 503-581. https://courses.grainger.illinois.edu/cs522/sp2016/ConstraintLogicProgrammingASurvey.pdf

Rudová, H., Murray, K., & Hnìtynka, P. (2003). *University course timetabling with soft constraints*. Masaryk University. https://www.unitime.org/papers/patat03.pdf

Wordcount: 551