Formal Methods in Software Engineering

§1 Annotation

In this course, you will master the logical foundations and decision procedures that underpin modern software verification. Starting from propositional and first-order logic, we advance through SAT and SMT solving, study the theory of computation to understand the limits of automated reasoning, and culminate in deductive program verification using Hoare logic and Dafny. Every theoretical concept is reinforced through hands-on work with industrial-strength tools — CaDiCaL, Z3, and Dafny — so you can translate formal guarantees into practice.

§2 Learning Outcomes

By the end of the semester, students should be able to:

- Apply propositional and first-order logic for formal specification: syntax, semantics, normal forms, proof systems.
- Encode combinatorial and verification problems as SAT instances in CNF and solve them using modern SAT solvers (CaDiCaL).
- Formulate problems in first-order theories and solve them using SMT solvers (Z3) via the SMT-LIB language.
- Classify decision problems by decidability and complexity (P, NP, co-NP, PSPACE), and explain why program verification is undecidable in general.
- Specify and verify programs using Design by Contract, Hoare logic, weakest preconditions, and Dafny.
- Critically assess industrial and academic applications of formal methods.

§3 Prerequisites

Students are expected to have prior exposure to:

- Discrete mathematics (propositional logic, set theory)
- Basic proof techniques (natural deduction)
- Automata theory and formal languages
- At least one programming language

Experience with software engineering or systems design is helpful but not required.

§4 Course Format

- 12 weeks, 1 lecture + 1 practice session per week (all groups together).
- **Lectures**: Present theoretical foundations and methods.

- **Practices**: Hands-on exercises, tool demos, and problem solving.
- **Assignments**: Four homework sets reinforcing core concepts via tool-based labs.
- **Exam**: Assess understanding of theoretical and applied aspects of formal methods.

§5 Course Structure

5.1. Propositional Logic, Normal Forms, and SAT (Weeks 1–2)

- PL recap: syntax, semantics, validity, satisfiability, entailment.
- Normal forms: NNF, CNF, DNF. Tseitin transformation and equisatisfiability.
- SAT problem. Cook–Levin theorem. SAT encodings. DIMACS format.
- Algorithms: DPLL, CDCL. Modern SAT solvers.

5.2. First-Order Logic & Theories (Weeks 3–4)

- FOL syntax: signatures, terms, formulas, free vs bound variables.
- FOL semantics: structures, variable assignments, evaluation.
- First-order theories: T-satisfiability, T-validity. Decidability landscape.
- Gödel's completeness and incompleteness theorems (stated).
- Decision problems as languages. Decidable vs recognizable.

5.3. Computation Theory & SMT (Weeks 5–6)

- Complexity classes: P, NP, co-NP, PSPACE. NP-completeness.
- Halting problem, Rice's theorem, reductions.
- Satisfiability Modulo Theories: QF_UF, QF_LIA, QF_LRA, QF_IDL, QF_AX, QF_BV.
- DPLL(T) architecture. Theory solvers. Nelson–Oppen combination.
- SMT-LIB v2 language. Z3 solver.

5.4. Program Verification & Dafny (Weeks 7–10)

- Program correctness, Design by Contract.
- Floyd–Hoare logic: Hoare triples, weakest preconditions.
- Loop invariants and termination.
- Dafny: `requires`, `ensures`, `invariant`, `decreases`, recursive functions, lemmas, algebraic data types, arrays, sequences, sorting.

5.5. Advanced Topics & Applications (Weeks 11–12)

- Bounded model checking, verification pipelines (VCGen → SMT).

- Model checking overview: transition systems, temporal logic.
- Formal methods in industry: seL4, CompCert, AWS (TLA+), Intel.
- Student presentations.

§6 Assignments

#	Topic	Tools	Weight	Due
1	SAT Encodings	DIMACS, CaDiCaL	25%	Week 3
2	SMT Solving	SMT-LIB, Z3	25%	Week 6
3	Program Verification	Dafny	25%	Week 10
4	Advanced Dafny	Dafny	25%	Week 12

Students also prepare a literature review presentation on an industrial or academic application of formal methods.

§7 Grading and Evaluation

Homework (40%)	Review (20%)	Exam (30%)	Participation (10%)

7.1. Homework Assignments (40%)

Four assignments covering SAT encodings, SMT solving, and program verification with Dafny.

7.2. Literature Review & Presentation (20%)

Students analyze a real case study of formal methods in industry or research.

7.3. Final Exam (30%)

Tests theoretical understanding (logic, complexity, Hoare logic) and tool-based reasoning (SAT/SMT encoding, Dafny specifications).

7.4. Participation (10%)

Attendance, engagement in practice sessions, and collaboration.

§8 Course Policies

- Standard university policies on academic integrity, attendance, and accommodations apply.
- Students are encouraged to regularly collaborate and discuss concepts, but all submitted work must be their own unless explicitly stated otherwise.
- Late submissions will be penalized unless prior arrangements are made with the instructor.

§9 Resources

Lecture notes, slides, and additional readings will be uploaded in the course GitHub repo: https://github.com/Lipen/formal-methods-course.