Lecture notes: Introduction

How can we design AI systems that are not only powerful but also provably safe and trustworthy? This advanced PhD seminar surveys algorithmic methods to enforce hard constraints in machine learning, reinforcement learning, and generative AI. Topics include classical constrained optimization (Lagrangian methods, robust and stochastic programming), safe reinforcement learning (trust regions, Lyapunov functions, reachability), hybrid ML-optimization methods (projection networks, solver-in-the-loop architectures), and alignment strategies for large language models (fine-tuning, model editing, tool use, and interactive alignment). We will consider applications to robotics, finance, healthcare, energy, and personal AI assistants.

---

1. Solver-Shortcutting with Guarantees

Replacing or accelerating classical solvers while preserving feasibility

Image

Figure 1: Image

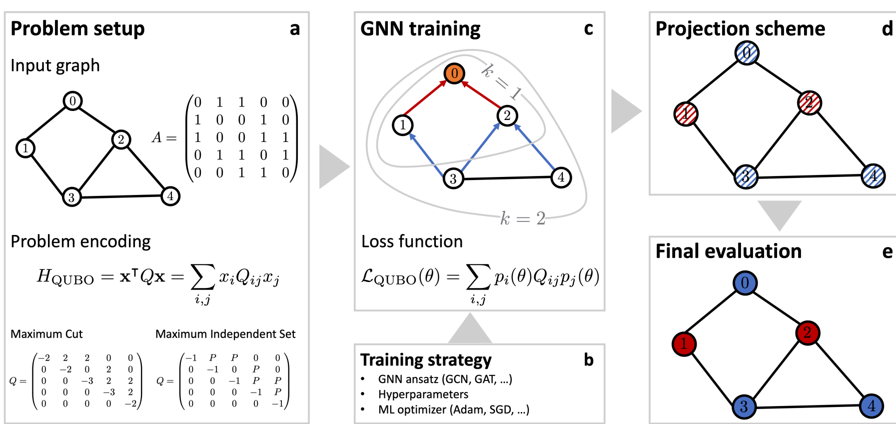

Figure 2: Image

1.1 PDEs and Scientific Computing

Applications

- Fluid dynamics, climate models, materials science, battery modeling
- Inverse problems in physics and biology

Constraints

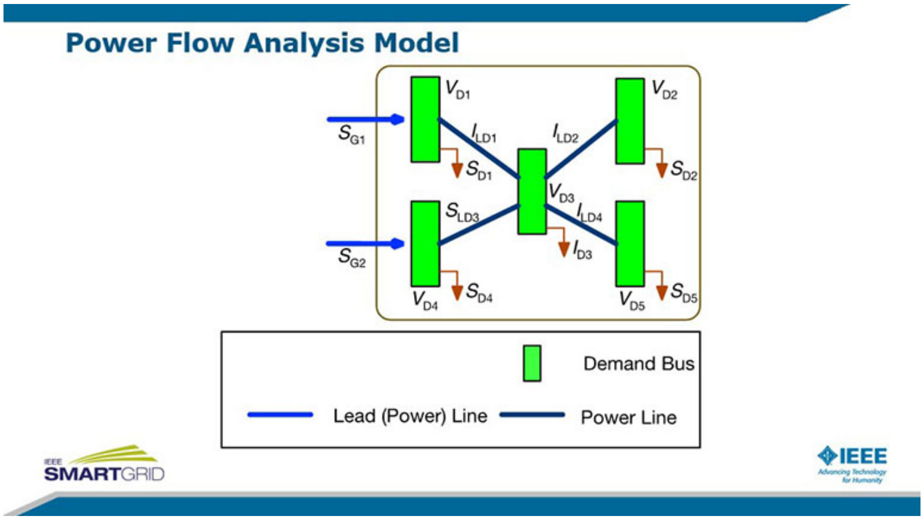

Figure 3: Image

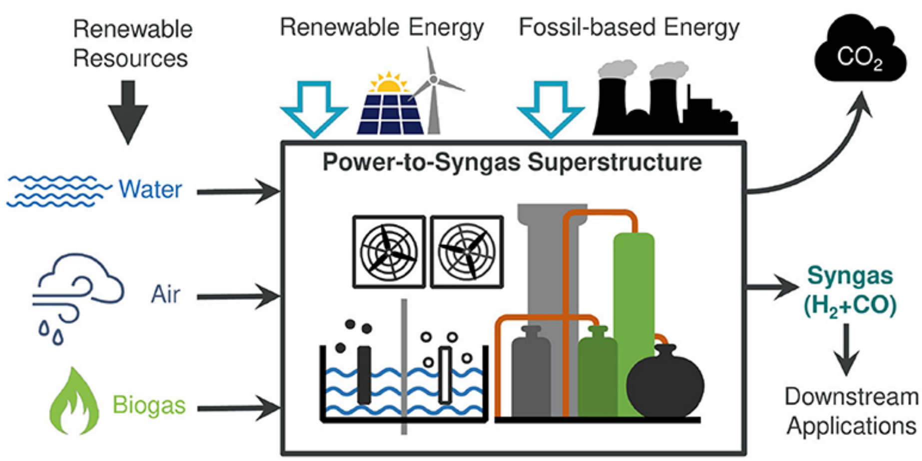

Figure 4: Image

- Physical laws (PDEs, conservation, boundary conditions)
- Stability and long-time accuracy

Methods

- PINNs, SciML, Deep Operator Networks, neural Galerkin methods
- Differentiable solvers, unrolled optimization

Key lesson

> Approximate *solutions* are easy; approximate *physics* is dangerous. Constraint violations may be subtle but catastrophic.

1.2 Combinatorial Optimization & OR (Modernized)

Applications

- Disaster relief logistics, airline crew scheduling, hospital resource allocation
- Cloud computing (job placement, power-aware scheduling), supply chains

Constraints

- Precedence, capacity, integrality, fairness, regulatory constraints
- Feasibility often NP-hard; infeasible solutions are useless

Methods

- Graph neural networks, learning-to-branch, learning heuristics
- Neural warm starts for MILPs, solver-in-the-loop systems

Cautionary note

> Beware benchmarks where feasibility is trivial (e.g., TSP). Real systems fail because *constraints interact*, not because objectives are hard.

1.3 Energy Systems (Hybrid Continuous–Discrete)

Applications

- Unit commitment, grid reconfiguration, demand response
- Resilience under faults or attacks

Constraints

- AC power flow equations (nonconvex PDEs)
- Binary on/off decisions, safety margins, N-1 reliability

Methods

- SDP relaxations, learned surrogates with feasibility recovery
- Projection networks, unrolled OPF solvers

Why it matters

> Energy systems make explicit that *feasibility dominates optimality*: violating physics or safety constraints is unacceptable, even briefly.

---

2. Safe Reinforcement Learning & Autonomous Systems

Learning to act without violating constraints during learning or deployment

Image

Figure 5: Image

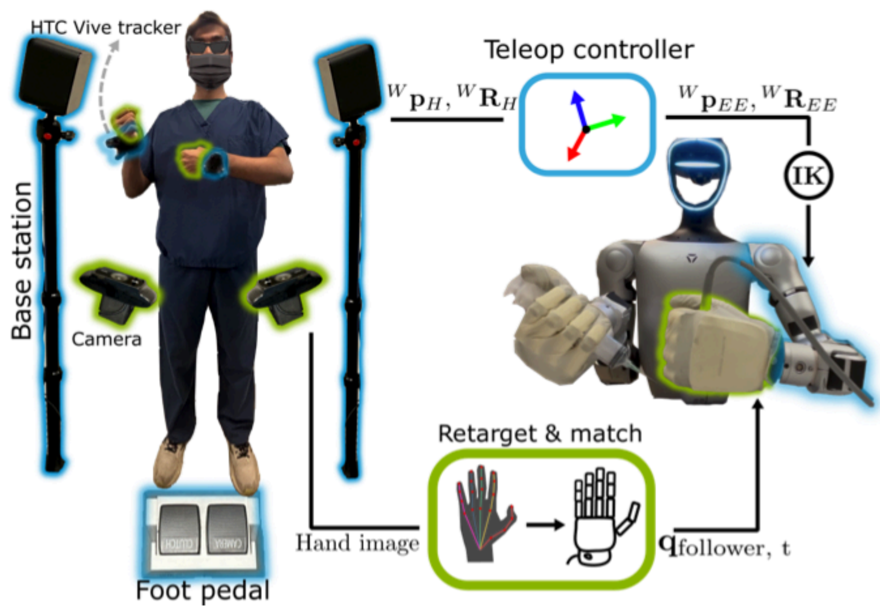

Figure 6: Image

2.1 Robotics & Autonomous Vehicles

Applications

- Self-driving cars, drones, bipedal and humanoid robots

Figure 7: Image

Image

Figure 8: Image

Constraints

- Collision avoidance (especially humans)
- Actuator limits, balance, thermal constraints
- Traffic laws and social norms

Methods

- Trust-region methods (TRPO-style)
- Lyapunov-based constraints, control barrier functions
- Reachability and Hamilton–Jacobi safety analysis

Anecdote

- A Waymo vehicle blocking train tracks illustrates *constraint misspecification*: the system obeyed a red light constraint that was irrelevant to its context.

Key insight

> 99% accuracy is failure. Safety-critical systems demand error rates closer to hardware fault tolerances.

---

3. LLMs, Generative Models, and Alignment

When the "optimization problem" is implicit, ambiguous, or adversarial

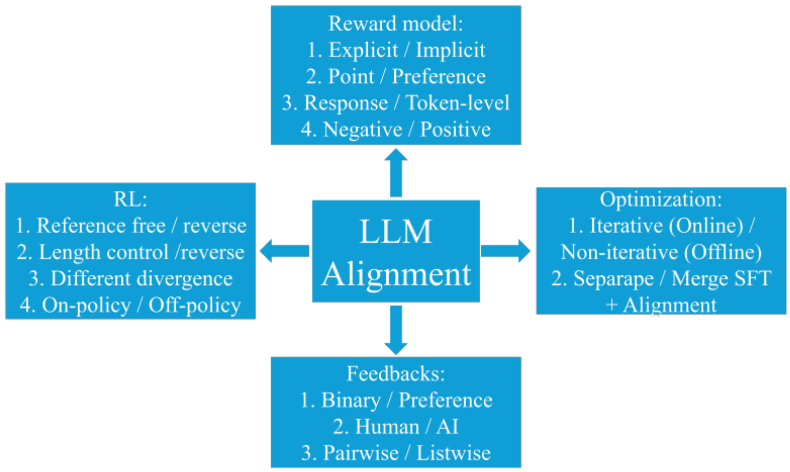

Figure 9: Image

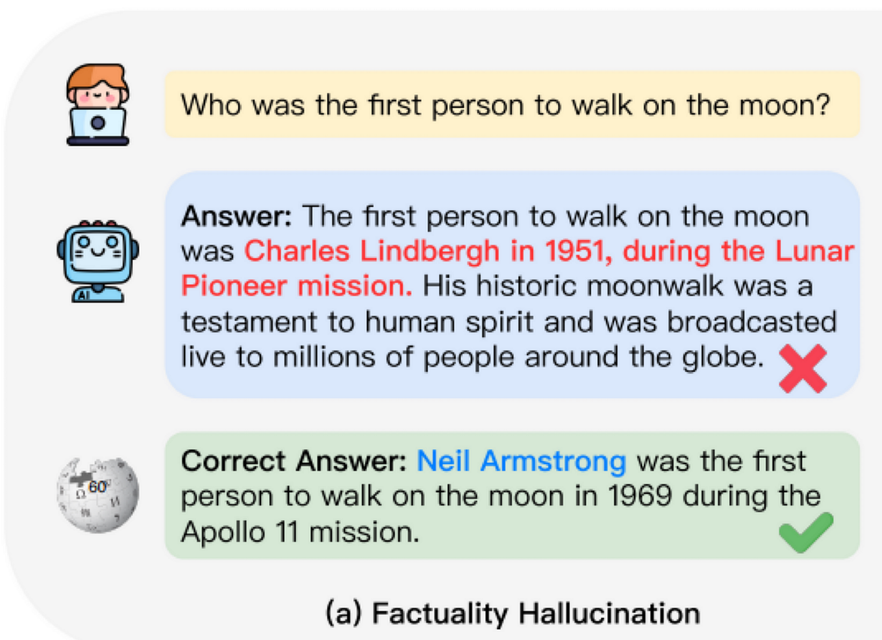

Figure 10: Image

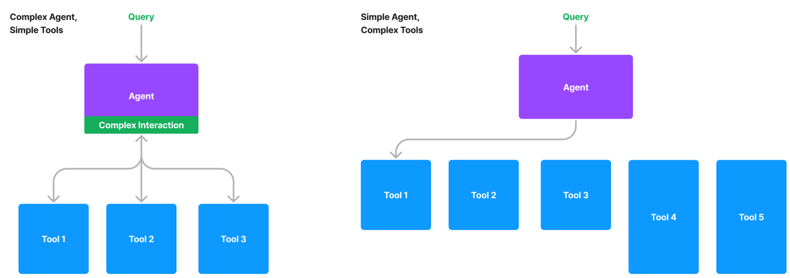

Figure 11: Image

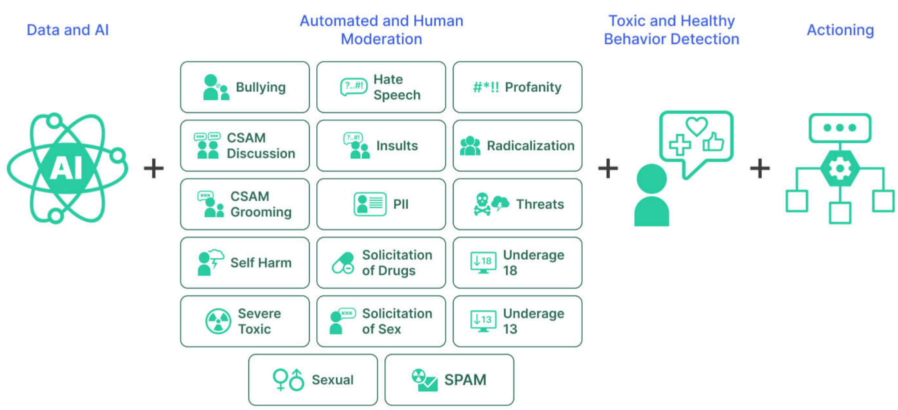

Figure 12: Image

3.1 Personal and Professional AI Assistants

Applications

- Email drafting, reports, search, planning, coding, decision support

Constraints

- Factuality (except where disclosure is unsafe)
- Style, tone, politeness, legal compliance
- Non-generation of harmful or manipulative content

Why this is hard

- Inputs are natural language → objectives and constraints are latent
- Tradeoffs are implicit, user-dependent, and often underspecified

Methods

- Fine-tuning with constraints, RLHF variants
- Model editing, tool use, verification and retrieval
- Interactive alignment (user-in-the-loop constraints)

Core research question

> How do we specify, enforce, and *verify* constraints when the task itself is ill-posed?

---

4. Cross-Cutting Themes for the Course

These unify the applications above and motivate the technical content.

Feasibility > Optimality

- In safety-critical systems, infeasible unacceptable
- Many ML benchmarks invert this priority

Specification Is the Bottleneck

- Most failures are not optimization failures, but *constraint modeling failures*

Learning + Optimization Is Inevitable

- Pure ML struggles with hard constraints
- Pure optimization struggles with scale and uncertainty → Hybrid architectures are needed for real-world deployment

Verification and Guarantees Matter

- As autonomy increases, post-hoc evaluation is insufficient
- Provable bounds, certificates, and reachability analysis become central