Foundations of Computer Science
Lecture 28

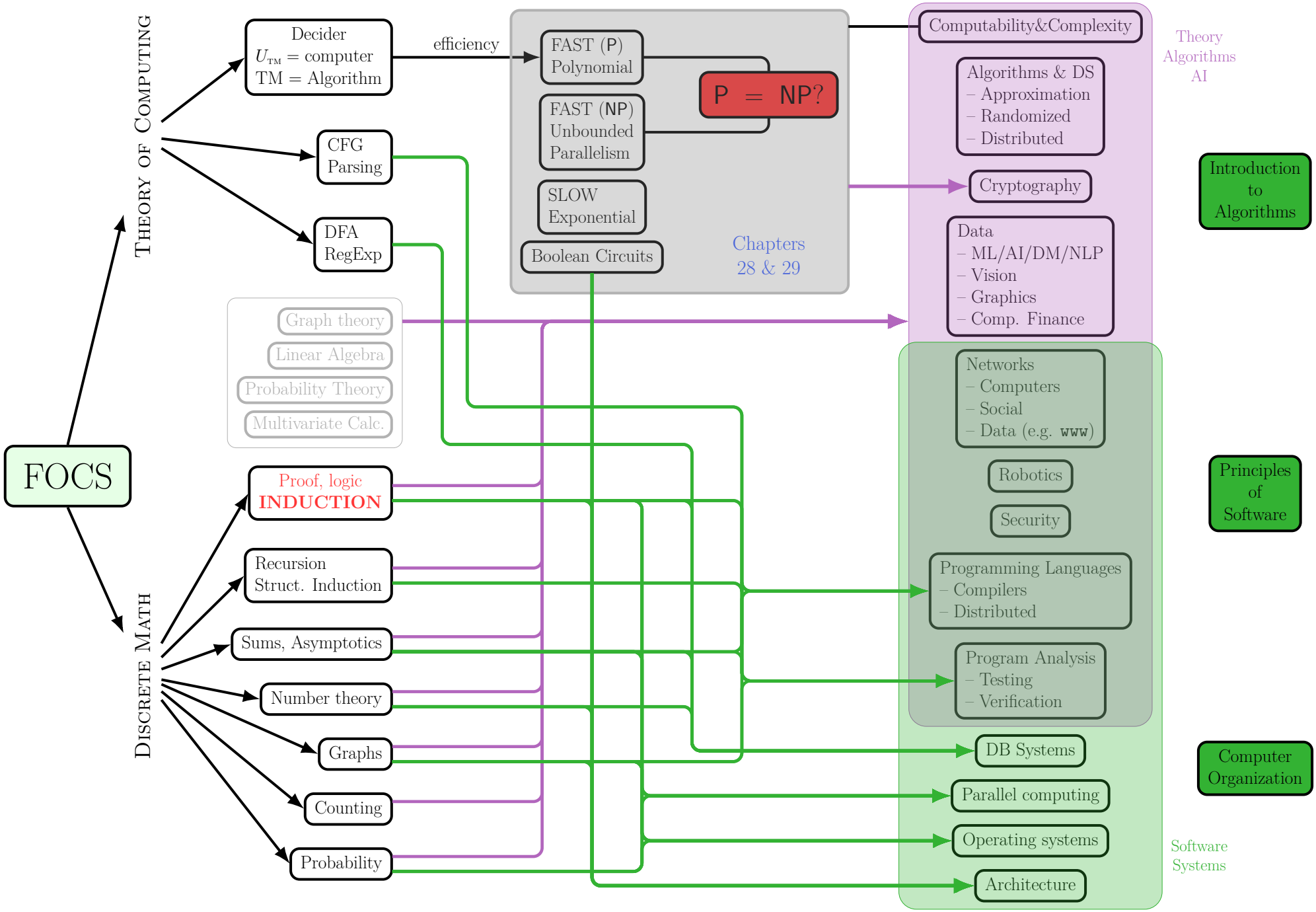
Efficiency: The Class P, NPand NP-Completeness

Running Time
Efficiently Solvable Problems
Boundary Between Efficient and Inefficient

Last Time

1. Computer: Universal Turing Machine U_{TM}
2. Program and Input: $\langle M \rangle \# w$. U_{TM} simulates M on w.
3. No Program Verifier, no Ultimate-Debugger, no PCP-Solver. ☹
4. No means No.

The Path Forward: Focus on Decidable Problems



Today: Efficiency: The Class P, NPand NP-Completeness

1. Time complexity: Asymptotic worst-case analysis.
2. The class P: Efficiently solvable problems.
3. Polynomial on one architecture means polynomial on pretty much any architecture.

Running Time

Time Complexity

Two Tapes vs. One Tape

Decidable But Non -Polynomial

Boundary Between Efficient and Inefficient

. . . the high technology so celebrated today is essentially a mathematical technology.

"To err is human, but to really foul things up you need a computer." – Paul Ehrlich

- **Mariner rocket explodes (1962).** Formula into code bug resulted in no smoothing of deviations.
- **WWWIII (1983)?** Soviet EWS detects 5 US-missiles (bug detected sunlight reflections).
  - **Luckily** *Stanislav "funny feeling in my gut" Petrov* thought: "surely they'd use more missiles?"
- **Therac 25 (1985).** Concurrent programming bug killed patients through massive 100× radiation overdose.
- **AT&T Lines Go Dead (1990).** 75 million calls dropped (one line of buggy code in software upgrade).
- **Patriot missile defense fails (1991). 28 soldiers dead, 100 injured** (rounding error in scud-detection).
- **Pentium floating point long-division bug (1993).** Cost: $475 million – flawed division table.
- **Ariane rocket explosion (1996).** Cost: $500 million – overflow in 64-bit to 16-bit conversion.
- **Y2K (1999).** Cost: $500 billion spent because year was stored as 2 digits to save space.
- **Mars Climate Orbiter Crash (1998).** Cost: $125 million lost due to metric to imperial units bug.
- **Tesla Self-Driving Car (2016). 1 dead**. Auto-pilot didn't "see" tractor-trailer.
- **Financial Disasters:** London Stock Exchange down due to single server bug (**2009**; billions of pounds of trading); Knight Capital computer glitch trigers stock sale (**2012**; 500 million lost and Knight's value drops by 75%).
- **Airline Disasters:**
  - AirFrance 447 2009, **228 dead**: pitot-tube failure feeds inconsistent data to programs which then panic pilot.
  - Spanair 5022, 2008, **154 dead**: malware virus.
  - AdamAir 574, 2007, **102 dead**: navigation system errors (and pilot errors).
  - KoreanAir 801, 1997, **228 dead**: ground proximity warning system bug.
  - AeroPerú 603, 1996, **70 dead**: altimeter failures.
  - Scottish RAF Chinook, 1994, **29 dead**: faulty test program
  - AirFrance 296, 1988, **3 dead**: altimeter bug.
  - IranAir 655, 1988, **290 dead**: shot down by US Aegis combat system (misidentified as attacking military plane).
  - KoreanAir 007, 1983, **269 dead**: autopilot took plane into Soviet airspace where it got shot down.
  - Boeing 737 Max, 2018,2019, **346 dead**: attack sensor + algorithm errors.
- **Software errors cost the U.S. $60 billion annually in rework, lost productivity and actual damages.**

Put effort to make *sure* your program works **fully** correctly **all** the time.