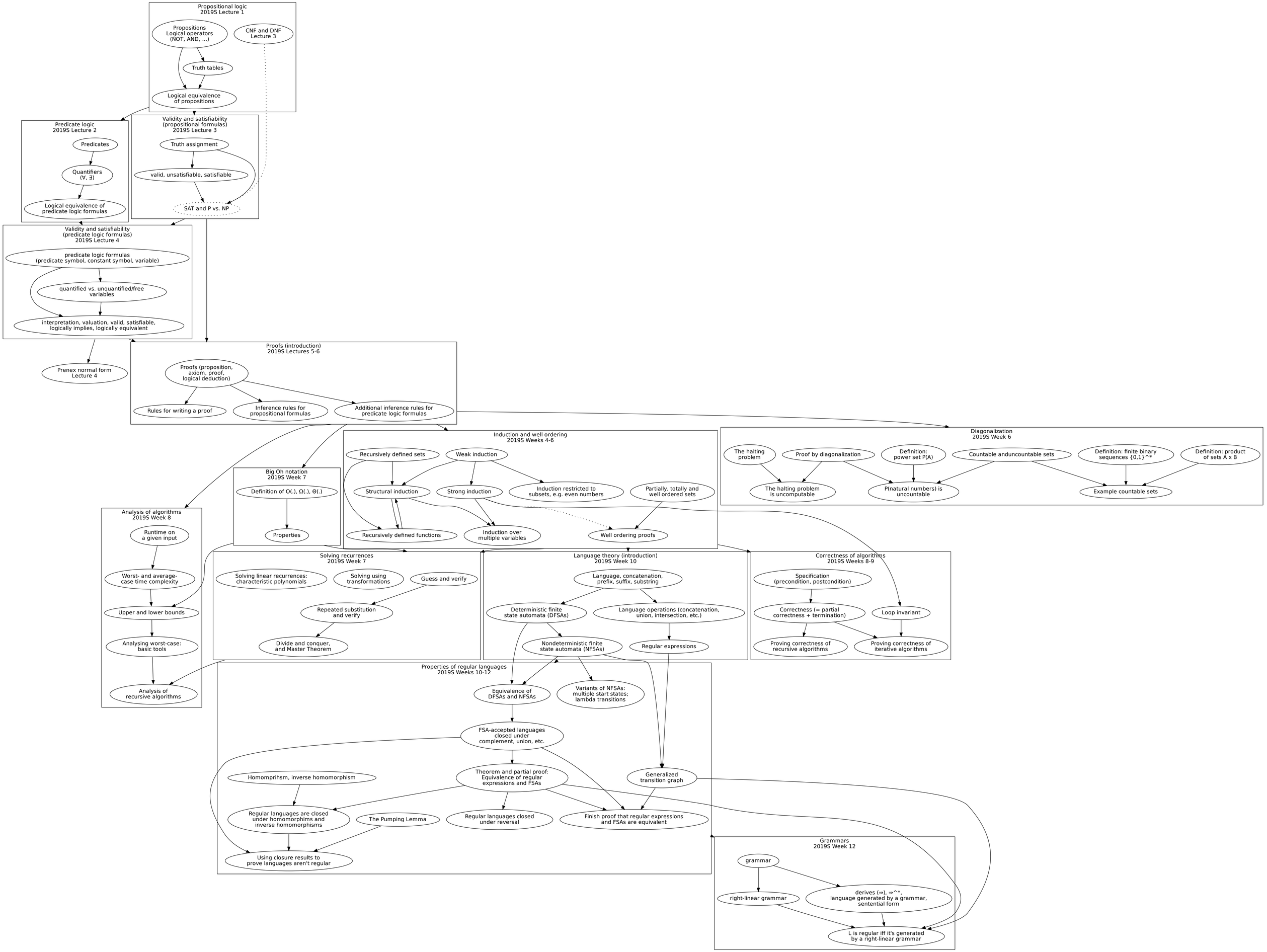
Propositional logic
2019S Lecture 1
Propositions
Logical operators
(NOT, AND, ...)
CNF and DNF
Lecture 3
Truth tables
Logical equivalence
of propositions
Predicate logic
2019S Lecture 2
Predicates
Quantifiers
(∀, ∃)
Logical equivalence of
predicate logic formulas
Validity and satisfiability
(propositional formulas)
2019S Lecture 3
Truth assignment
valid, unsatisfiable, satisfiable
SAT and P vs. NP
Validity and satisfiability
(predicate logic formulas)
2019S Lecture 4
predicate logic formulas
(predicate symbol, constant symbol, variable)
quantified vs. unquantified/free
variables
interpretation, valuation, valid, satisfiable,
logically implies, logically equivalent
Prenex normal form
Lecture 4
Proofs (introduction)
2019S Lectures 5-6
Proofs (proposition,
axiom, proof,
logical deduction)
Rules for writing a proof
Inference rules for
propositional formulas
Additional inference rules for
predicate logic formulas
Diagonalization
2019S Week 6
The halting
problem
Proof by diagonalization
Definition:
power set P(A)
Countable anduncountable sets
Definition: finite binary
sequences {0,1}^*
Definition: product
of sets A x B
The halting problem
is uncomputable
P(natural numbers) is
uncountable
Example countable sets
Induction and well ordering
2019S Weeks 4-6
Recursively defined sets
Weak induction
Structural induction
Strong induction
Induction restricted to
subsets, e.g. even numbers
Partially, totally and
well ordered sets
Recursively defined functions
Induction over
multiple variables
Well ordering proofs
Big Oh notation
2019S Week 7
Definition of O(.), Ω(.), Θ(.)
Properties
Analysis of algorithms
2019S Week 8
Runtime on
a given input
Worst- and average-
case time complexity
Upper and lower bounds
Analysing worst-case:
basic tools
Analysis of
recursive algorithms
Solving recurrences
2019S Week 7
Solving linear recurrences:
characteristic polynomials
Solving using
transformations
Guess and verify
Repeated substitution
and verify
Divide and conquer,
and Master Theorem
Language theory (introduction)
2019S Week 10
Language, concatenation,
prefix, suffix, substring
Deterministic finite
state automata (DFSAs)
Language operations (concatenation,
union, intersection, etc.)
Nondeterministic finite
state automata (NFSAs)
Regular expressions
Correctness of algorithms
2019S Weeks 8-9
Specification
(precondition, postcondition)
Correctness (= partial
correctness + termination)
Loop invariant
Proving correctness of
recursive algorithms
Proving correctness of
iterative algorithms
Properties of regular languages
2019S Weeks 10-12
Equivalence of
DFSAs and NFSAs
Variants of NFSAs:
multiple start states;
lambda transitions
FSA-accepted languages
closed under
complement, union, etc.
Homomprihsm, inverse homomorphism
Theorem and partial proof:
Equivalence of regular
expressions and FSAs
Generalized
transition graph
Regular languages are closed
under homomorphims and
inverse homomorphisms
The Pumping Lemma
Regular languages closed
under reversal
Finish proof that regular expressions
and FSAs are equivalent
Using closure results to
prove languages aren't regular
Grammars
2019S Week 12
grammar
right-linear grammar
derives (⇒), ⇒^*,
language generated by a grammar,
sentential form
L is regular iff it's generated
by a right-linear grammar