Coverage in *Theory of Computation* of the ACM recommendations

This lists the appropriate items from the Association for Computing Machinery's *Curriculum Guidelines for Undergraduate Programs in Computer Science* (from 2013), and describes the coverage in *Theory of Computation*. In brief, the text covers all of the points, except that it omits the Chomsky hierarchy in favor of additional Computational Complexity. (The text also covers some material not listed by the ACM.)

AL/Basic Automata Computability and Complexity

Topics:
- Finite-state machines *Full coverage*
- Regular expressions *Full coverage*
- The halting problem *Full coverage*
- Context-free grammars *Partial coverage*
- Introduction to the P and NP classes and the P vs. NP problem *Full coverage*
- Introduction to the NP-complete class and exemplary NP-complete problems (e.g., SAT, Knapsack) *Full coverage*

Learning Outcomes:
1. Discuss the concept of finite state machines. [Familiarity] *Full coverage*
2. Design a deterministic finite state machine to accept a specified language. [Usage] *Full coverage*
3. Generate a regular expression to represent a specified language. [Usage] *Full coverage*
4. Explain why the halting problem has no algorithmic solution. [Familiarity] *Full coverage*
5. Design a context-free grammar to represent a specified language. [Usage] *Partial coverage*
6. Define the classes P and NP. [Familiarity] *Full coverage*
7. Explain the significance of NP-completeness. [Familiarity] *Full coverage*

AL/Advanced Computational Complexity

Topics:
- Review of the classes P and NP; introduce P-space and EXP *Full coverage*
- Polynomial hierarchy *Partial coverage*
- NP-completeness (Cook's theorem) *Full coverage*
- Classic NP-complete problems *Full coverage*
- Reduction Techniques *Full coverage*

Learning Outcomes:
1. Define the classes P and NP. [Familiarity] *Full coverage*
2. Define the P-space class and its relation to the EXP class. [Familiarity] *Partial coverage*
3. Explain the significance of NP-completeness. [Familiarity] *Full coverage*
4. Provide examples of classic NP-complete problems. [Familiarity] *Full coverage*
5. Prove that a problem is NP-complete by reducing a classic known NP-complete problem to it. [Usage] *Full coverage*

AL/Advanced Automata Theory and Computability

Topics:
- Sets and languages
  - Regular languages *Full coverage*
  - Review of deterministic finite automata (DFAs) *Full coverage*
  - Nondeterministic finite automata (NFAs) *Full coverage*
  - Equivalence of DFAs and NFAs *Full coverage*
  - Review of regular expressions; their equivalence to finite automata *Full coverage*
  - Closure properties *Full coverage*
  - Proving languages non-regular, via the pumping lemma or alternative means *Full coverage*
- Context-free languages
  - Push-down automata (PDAs) *Partial coverage*
  - Relationship of PDAs and context-free grammars *Partial coverage*
  - Properties of context-free languages *Partial coverage*

- Turing machines, or an equivalent formal model of universal computation *Full coverage*
- Nondeterministic Turing machines *Full coverage*
- Chomsky hierarchy *Not covered*
- The Church-Turing thesis *Full coverage*
- Computability *Full coverage*
- Rice's Theorem *Full coverage*
- Examples of uncomputable functions *Full coverage*
- Implications of uncomputability *Full coverage*

Learning Outcomes:

1. Determine a language's place in the Chomsky hierarchy (regular, context-free, recursively enumerable). *Not covered*[Assessment]
2. Convert among equivalently powerful notations for a language, including among DFAs, NFAs, and regular expressions, and between PDAs and CFGs. [Usage] *Partial coverage*
3. Explain the Church-Turing thesis and its significance. [Familiarity] *Full coverage*
4. Explain Rice's Theorem and its significance. [Familiarity] *Full coverage*
5. Provide examples of uncomputable functions. [Familiarity] *Full coverage*
6. Prove that a problem is uncomputable by reducing a classic known uncomputable problem to it. [Usage] *Full coverage*