Families of Languages

Robert M. Keller
Harvey Mudd College
March 2010

Families of Languages

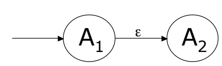
- Let F be a **family** (set) of languages.
- Examples:
  - The family of finite languages.
  - The family of regular languages.
  - The family of co-finite languages (complement within Σ^* is finite).
  - The family of all languages.

Closure Properties

- A family F is **closed under** an operator if the application of that operator to a language or languages in F results in a language which is also in F.
- Example: The family of regular languages is closed under $\cup$, concatenation, and *.
- Exercise: Determine whether the other families on the preceding page are closed under these same operators. (Create a matrix.)

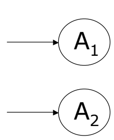
Proof of Some Closure Properties using the Subset Construction

- The family of languages accepted by DFA (or NFA; we know they are the same) is closed under **concatenation**.
- **Proof**: Suppose L_1 and L_2 are accepted by DFA's. To show that $L_1 L_2$ is also, connect the corresponding acceptors A_1 and A_2 by adding ε transitions from each accepting state of A_1 to each start state of A_2. Then modify the start and accepting states appropriately and convert this NFA to a DFA. ***Figuratively***,



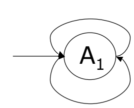
Proof of Some Closure Properties using the Subset Construction

- Show that the family of DFA languages is closed under union. Hint:



Proof of Some Closure Properties using the Subset Construction

- Show that the family of DFA languages is closed under *. Hint:



Summary

- The family of languages accepted by DFA's is closed under concatenation, union, and *.

Proof that every regular language is accepted by a DFA.

- The family of languages accepted by DFA's is closed under concatenation, union, and *. These correspond exactly to the regular operators.
- Furthermore, the languages of a single 1–letter string are accepted by DFA's, as are ∅ and ε.
- Therefore every regular language is accepted by a DFA.

Kleene's Theorem

- A language is regular iff it is accepted by some DFA.
- Proof:
  - We showed DFA ⇒ regular by solving a system of equations.
  - We showed regular ⇒ DFA by the subset construction and closure properties.

The family of DFA languages is closed under complement.

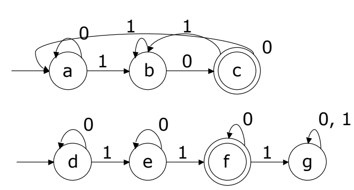
- For complement, we only need reverse the roles of accepting and non-accepting states in the accepting automaton.
- Thus the family of regular languages is closed under complement, even though complement is not a regular operator. (It is sometimes seen in an **extended** version of regular expressions.)

The family of DFA languages is closed under intersection.

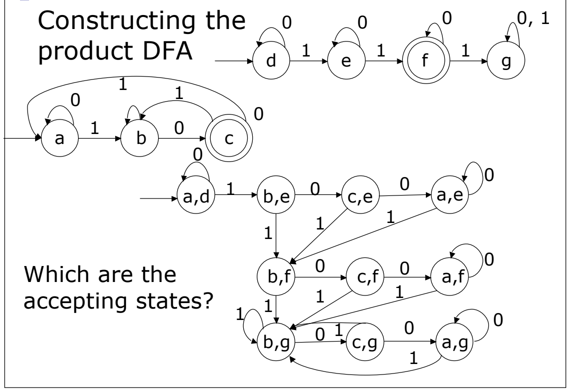
- Proof: Adjoin the two automata and use the subset construction with the set of two start states as the start state. This effectively constructs an automaton, called the **product automaton**, that simulates the behavior of both of the original automata in parallel.
- Choose accepting states appropriately. The new automaton will automatically be deterministic, as the originals were.
- Thus the family of regular languages is closed under intersection, although ∩ is not a regular operator.
- The product automaton can be used for any binary set operator (∩, ∪, -, ⊕, ≡, etc.). Only the accepting states are different.

Example: A DFA for the **intersection** of languages given by regular expressions (0∪1)*10 and 0*10*10*

The individual DFA are:



Constructing the product DFA



Which are the accepting states?

Notes on Product Construction

- Depending on the intended implementation, it may be better **not** to construct the composite machine explicitly, but rather leave it decomposed.
- The rationale is that there are generally fewer states in the sum of the two machines than in the composite.
- We sometimes try to go the other way: **decompose** a complex machine into a product of simpler machines.