

Deterministic Finite Automata Finite Automaton (FA) ? QD = All reachable subsets of QE factoring in ?-closures 2. Idea: To avoid enumerating each member of power set, do "lazy creation of states". 31 $q_0 \xrightarrow{0} q_0, q_0 \xrightarrow{1} q_1$ NFA: ? $N = (Q, \Sigma, \delta, q_0, F)$ where $Q = \{q_0, q_1\}$, $\Sigma = \{0, 1\}$, $\delta(q_0, 0) = q_0$, $\delta(q_0, 1) = q_1$, $\delta(q_1, 0) = q_1$, $\delta(q_1, 1) = q_1$ DFA: ? $D = (Q, \Sigma, \delta, q_0, F)$ where $Q = \{q_0, q_1, q_2\}$, $\Sigma = \{0, 1\}$, $\delta(q_0, 0) = q_0$, $\delta(q_0, 1) = q_1$, $\delta(q_1, 0) = q_2$, $\delta(q_1, 1) = q_1$, $\delta(q_2, 0) = q_2$, $\delta(q_2, 1) = q_2$ 1 0 $q_0 \xrightarrow{0} q_0, q_0 \xrightarrow{1} q_1$ 1 $q_1 \xrightarrow{0} q_2, q_1 \xrightarrow{1} q_1$ 0 0 1 Correctness of subset construction Theorem: If D is the DFA constructed from NFA N by subset construction, then $L(D) = L(N)$? Proof: ? Show that $\delta_D(\{q_0\}, w) = \delta_N(q_0, w)$, for all w ? Using induction on w's length: ? Let $w = xa$? $\delta_D(\{q_0\}, xa) = \delta_D(\delta_D(\{q_0\}, x), a) = \delta_N(\delta_N(q_0, x), a) = \delta_N(q_0, xa)$ 32 A bad case where $\#states(DFA) \gg \#states(NFA)$? $L = \{w \mid w \text{ is a binary string such that, the } k\text{th symbol from its end is a } 1\}$? NFA has $k+1$ states ? But an equivalent DFA needs to have at least 2^k states (Pigeon hole principle) ? m holes and $>m$ pigeons $\Rightarrow$ at least one hole has to contain two or more pigeons 33 o An application: Text Search Applications ? Text indexing ? inverted indexing ? For each unique word in the database, store all locations that contain it using an NFA or a DFA ? Find pattern P in text T ? Example: Google querying ? Extensions of this idea: ? PATRICIA tree, suffix tree 35 Advantages & Caveats for NFA ? Great for modeling regular expressions ? String processing – e.g., grep, lexical analyzer ? Could a non-deterministic state machine be implemented in practice? ? Probabilistic models could be viewed as extensions of non deterministic state machines (e.g., toss of a coin, a roll of dice) ? They are not the same though ? A parallel computer could exist in multiple "states" at the same time 36 A few properties of DFAs and NFAs ? A clamping circuit waits for a "1" input, and turns on forever. However, to avoid clamping on spurious noise, we'll design a DFA that waits for two consecutive 1s in a row before clamping on. ? Build a DFA for the following language: $L = \{w \mid w \text{ is a bit string which contains the substring } 11\}$? State Design: ? q_0 : start state (initially off), also means the most recent input was not a 1 ? q_1 : has never seen 11 but the most recent input was a 1 ? q_2 : has seen 11 at least once ? Example #3 ? Build a DFA for the following language: $L = \{w \mid w \text{ is a binary string that has even number of 1s and even number of 0s}\}$ 14 Extension of transitions to paths ? $\delta(q, w)$ = destination state from state q on input string w. ? $\delta(q, wa) = \delta(\delta(q, w), a)$? Work out example #3 using the input sequence $w = 10010$, $a = 1$: ? $\delta(q_0, wa) = ?$ 15 Language of a DFA A DFA A accepts string w if there is a path from q_0 to an accepting (or final) state that is labeled by w. ? i.e., $L(A) = \{w \mid \delta(q_0, w) \in F\}$? i.e., $L(A)$ = all strings that lead to an accepting state from q_0 . 16 o Non-Deterministic Finite Automaton Non-deterministic Finite Automata (NFA) ? A Non-deterministic Finite Automaton (NFA) is called non-deterministic because the machine can exist in more than one state at the same time. ? Transitions could be non-deterministic ? Each transition function therefore maps to a set of states. 18 $q_i \xrightarrow{1} q_j, q_i \xrightarrow{0} q_k, \dots$ Non-deterministic Finite Automata (NFA) ? An NFA consists of: ? Q = A finite set of states ? Σ = A finite set of input symbols (alphabet) ? q_0 = A start state ? F = Set of accepting states ? δ = A transition function, which is a mapping between $Q \times \Sigma \rightarrow \text{subset of } Q$? An NFA is also defined by the 5-tuple: ? $\{Q, \Sigma, q_0, F, \delta\}$ 19 How to use an NFA? ? Input: a word w in Σ^* ? Question: Is w accepted by the NFA? ? Steps: ? Start at the start state q_0 ? For every input symbol in the word w do ? Determine all possible next states from all current states, given the current input symbol in w and the transition function ? If after all symbols in w are consumed and if at least one of the current states is a final state then accept w; ? Otherwise, reject w. 20 NFA for strings containing 01 21 q_0 start $q_1 \xrightarrow{0} q_0, q_1 \xrightarrow{1} q_2$ Final state o $Q = \{q_0, q_1, q_2\}$ o $\Sigma = \{0, 1\}$ o start state = q_0 o $F = \{q_2\}$ o Transition

table {q2 {q} 2 *q} 2 {q2 q ?} 1 {q0 {q} 0 ,q1 q} 0 0 1 states symbols What is an "error state"? A DFA for recognizing the key word "price" ? An NFA for the same purpose: ? Transitions into a dead state are implicit 22 q0 p q1 r q2 i q3 c q4 e q5 qe Any other input symbol q0 p q1 r q2 i q3 c q4 e q5 Any symbol Example #3 ? Build an NFA for the following language: $L = \{ w \mid w \text{ ends in } 01 \}$? ? ? Other examples ?

Keyword recognizer (e.g., if, then, else, while, for, include, etc.) ? Strings where the first symbol is present somewhere later on at least once 23 Extension of ? to NFA Paths ? Basis: $\delta(q, ?) = \{q\}$? Induction: ? Let $\delta(q_0, w) = \{p_1, p_2, \dots, p_k\}$? $\delta(p_i, a) = S_i$ for $i = 1, 2, \dots, k$? Then, $\delta(q_0, wa) = S_1 \cup S_2 \cup \dots \cup S_k$ 24 Language of an NFA ? An NFA accepts w if there exists at least one path from the start state to an accepting (or final) state that is labeled by w ? $L(N) = \{ w \mid \delta(q_0, w) \cap F \neq \emptyset \}$ 25 Differences between NFA and DFA ? DFA 1. All transitions are deterministic ? Each transition leads to exactly one state 2. For each state, transition on all possible symbols (alphabet) should be defined 3. Accepts input if the last state visited is in F 4. Harder to construct because of the number of states 5. Practical implementation is feasible ? NFA 1. Some transitions could be non deterministic ? A transition may lead to a more than one state 2. Not all symbol transitions need to be defined explicitly (if undefined will go to an error state – this is just a design convenience, not to be confused with "non determinism") 3. Locate regular languages in the Chomsky Hierarchy 10 The Chomsky Hierarchy 11 Regular (DFA) Context free (PDA) Context sensitive (LBA) Recursively– enumerable (TM) o A containment hierarchy of classes of formal languages Example #1 ? Informally, it is a state diagram that comprehensively captures all possible states and transitions that a machine can take while responding to a stream or sequence of input symbols. Practical implementations limited but emerging (e.g., Micron automata processor) 26 Note: NFAs and DFAs are equivalent in power to recognize languages. (All accept Regular Languages) 43 Eliminating ?-transitions Let $E = \{Q_E, ?, ?E, q_0, F_E\}$ be an ?-NFA Goal: To build DFA $D = \{Q_D, ?, ?D, \{q_D\}, F_D\}$ such that $L(D) = L(E)$ Construction: 1. Equivalence of DFA & NFA $?? ?? = \{0, 1\}$.???2.?????4