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 q0 q1 0 0,1 q2 1 NFA: ?N 0 1 q0 {q0 ,q1 } {q0 } q1 O {q2 } *q2 O O DFA: ?D 0 1 [q0] [q0 ,q1] [q0] [q0 ,q1] [q0 ,q1] [q0 ,q2] *[q0 ,q2] [q0 ,q1] [q0] [q0] 1 0 [q0 ,q1] 1 [q0 ,q2] 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 ?D ({q0 }, w) ? ?N (q0 , w} , for all w ? Using induction on w's length: ? Let w = xa ? ?D ({q0 }, xa) ? ?D (?N (q0 , x}, a) ? ?N (q0 , w} 32 A bad case where #states(DFA) #states(NFA) ? L = {w I w is a binary string such that, the k 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 => 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 I w is a bit string which contains the substring 11} ? State Design: ? q0 : start state (initially off), also means the most recent input was not a 1 ? q1 : has never seen 11 but the most recent input was a 1 ? q2 : has seen 11 at least once ? Example #3 ? Build a DFA for the following language: L = { w I w is a binary string that has even number of 1s and even number of 0s} 14 Extension of transitions to paths ? ? (q, w) = destination state from state q on input string w. ? ? (q, wa) = ? (?(q, w), a) ? Work out example #3 using the input sequence w = 10010, a = 1: ? ? (q0 , wa) = ? 15 Language of a DFA A DFA A accepts string w if there is a path from q0 to an accepting (or final) state that is labeled by w. ? i.e., L(A) = { w I ?(q0 ,w) ? F } ? i.e., L(A) = all strings that lead to an accepting state from q0. 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 qi 1 1 qj qk … Non-deterministic Finite Automata (NFA) ? An NFA consists of: ? Q = A finite set of states ? ? = A finite set of input symbols (alphabet) ? q0 = A start state ? F = Set of accepting states ? ? = A transition function, which is a mapping between Q x ? -> subset of Q ? An NFA is also defined by the 5-tuple: ? {Q, ? , q0 , F, ? } 19 How to use an NFA? ? Input: a word w in ?* ? Question: Is w accepted by the NFA? ? Steps: ? Start at the start state q0 ? 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 q0 start q1 0 0,1 0,1 1 q2 Final state o Q = {q0 ,q1 ,q2 } o ? = {0,1} o start state = q0 o F = {q2 } 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 I w 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: ? (q, ?) = {q} ? Induction: ? Let ? (q0 , w) = {p1 , p2…, pk } ? ? (pi , a) = Si for I =1, 2…, k ? Then, ? (q0 , wa) = S1 U S2 U … U Sk 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 I ?(q0 , w) ? F != ? } 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 = {QE , ?,?E , q0 , FE } be an ?-NFA Goal: To build DFA D = {QD , ?, ?D , {qD }, FD } such that L(D)=L(E) Construction: 1.Equivalence of DFA & NFA ????= {0, 1} ???2.?????4.