

Transducers

- Transducers are DFAs that produce output and are regarded as a function which takes an input string and produces an output string.
- This may be considered as a more realistic model of computation than simple acceptance or rejection.
- We will observe that adding output to the DFA amounts to extending the model of computation. However, the difference between DFAs and transducers is in fact quite small

Transducers

Hantao Zhang

hzhang@cs.uiowa.edu

The University of Iowa, Department of Computer Science

Based on the slides prepared by Professor Rus

From DFA to DGSM

- Every DFA can be converted into a DGSM by adding a binary output alphabet and the output function ρ as follows: For any $q \in Q$, $a \in \Sigma$,
 1. $\rho(q, a) = 1$ if $\delta(q, a) \in F$
 2. $\rho(q, a) = 0$ if $\delta(q, a) \notin F$
- Then an input string $x \neq \epsilon$ is accepted by the DFA iff the last bit of the DGSM output is 1.

Formal Definition

Definition: A transducer M is a 6-tuple $M = (Q, \Sigma, \Delta, \delta, \rho, q_0)$ where:

1. Q is a finite set of states
2. Σ is the input alphabet
3. Δ is the output alphabet
4. $\delta : Q \times \Sigma \rightarrow Q$ is the next state function
5. $q_0 \in Q$ is the start state
6. $\rho : Q \times \Sigma \rightarrow \Delta^*$ is the output function

Note: Transducers are called by Fleck Deterministic Generalized Sequential Machines (DGSM)

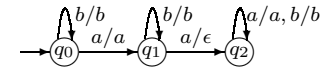
Comparing to DFA, DGSM has Δ and ρ , but not the accept states.

Note

- This machine copies “b” s in state q_0 until first “a” occurs.
- When first “a” occurs, M copies it to the output and then goes to state q_1 . In this state it copies “b”s to output until the second “a” occurs.
- When the second “a” occurs in state q_1 the machine removes it and goes to state q_2 where it copies all input to output.

Example Transducer

Consider the transducer $M = (Q, \Sigma, \Delta, \delta, \rho, q_0)$ where $\Sigma = \Delta = \{a, b\}$ which copies its input to its output while deleting the second occurrence (if any) of the symbol a.



Extending ρ

Here we extend $\rho : Q \times \Sigma \rightarrow \Delta^*$ to $\rho^* : Q \times \Sigma^* \rightarrow \Delta^*$ by:

- $\rho^*(q, \epsilon) = \epsilon$, for each $q \in Q$
- $\rho^*(q, wx) = \rho^*(q, w)\rho(\delta^*(q, w), x)$
for all $w \in \Sigma^*$ and $x \in \Sigma$

Note: this allows us to regard M as defining a function $M : \Sigma^* \rightarrow \Delta^*$, namely, $M(w) = \rho^*(q_0, w)$ and refer to it as DGSM function.

DGSM Computation

- Computation performed by a transducer parallels that of a DFA except that there is a sequence of outputs as well as a sequence of states
- A transducer terminates when all symbols of the output are consumed.
- Formally, given a transducer M is a 6-tuple $M = (Q, \Sigma, \Delta, \delta, \rho, q_0)$ and an input string $x = a_1a_2 \cdots a_k \in \Sigma^*$, where $a_i \in \Sigma$, if

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \rightarrow \cdots \rightarrow a_k q_k$$

by δ , then the output is

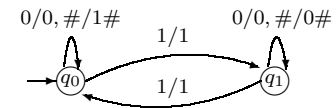
$$\rho(q_0, a_1)\rho(q_1, a_2) \cdots \rho(q_{k-1}, a_k)$$

Interpretation

- M_2 is in state q_0 whenever input has even parity and it is in state q_1 whenever input has an odd parity
- When the output $\#$ (i.e., end of string) is encountered in state q_0 the machine adds 1 to the output thus making it of the odd parity
- When the output $\#$ is encountered in state q_1 the machine adds a 0 to the output thus maintaining the odd parity

Example 2:

In this example $\Sigma = \Delta = \{0, 1, \#\}$ and the DGSM M_2 computes the odd parity, i.e., it transforms $w\#$ where $w \in \{0, 1\}^*$ into $w\pi\#$, $\pi \in \{0, 1\}$ such that the sequence $w\pi$ contains an odd number of 1s.



Theorem 3.1.3 (Closure Property)

Let DGSM $M = (Q, \Sigma, \Delta, \delta, \rho, q_0)$ and $L \subseteq \Sigma^*$ is regular, then $M(L) \subseteq \Delta^*$ is also regular.

Proof: See Fleck 146, 147

Note

- With DFA our interest is in the language a DFA D recognizes
- With transducers our interest is in the function $f : \Sigma^* \rightarrow \Delta^*$ a transducer M computes

Problem 3.8

Given $L \subseteq \Sigma^*$, define the function $f : \Sigma^* \rightarrow \{0, 1\}^*$ inductively as follows:

1. $f(\epsilon) = \epsilon$
2. For each $w \in \Sigma^*$ and $a \in \Sigma$

$$f(wa) = \begin{cases} f(w)0, & \text{if } wa \notin L \\ f(w)1, & \text{if } wa \in L \end{cases}$$

Show that f is a DGSM function iff L is regular

Assignment (100 points)

Solve the following problems taken from Fleck's book, page 181.

Problem 3.7 Show that the function $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ defined by:

$$f(x) = \begin{cases} 0^n 1^n, & \text{in } x = 0^{2n} \\ x, & \text{otherwise} \end{cases}$$

cannot be realized by a DGSM.

Problem 3.12

Consider the function $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ defined by:

- $f(\epsilon) = \epsilon, f(0) = 0, f(1) = 1$
- For $w_i \in \{0, 1\}$ and $k \geq 2$,
 $f(w_1 w_2 \dots w_k) = w_1 w_2 w_1 w_3 \dots w_1 w_k w_1$

Determine whether or not f is a DGSM function and prove it

Problem 3.10

One of the functions $f_1, f_2 : \{0, 1\}^* \rightarrow \{0, 1\}^*$ can be realized by a DGSM and one is not. Which is which and why?

f_1 : For each $x \in \{0, 1\}^*$, $f_1(x) = w$ where w is obtained from x by deleting all instances of 1

f_2 : $f_2(\epsilon) = \epsilon$ and for each $w \in \{0, 1\}^*$ and $a \in \{0, 1\}$,
 $f_2(wa) = f_2(w)wa$

Note

- k -equivalence are equivalent relations, as one can check directly
 1. Reflexive: $q \stackrel{k}{\equiv} q$ obviously
 2. Symmetric: If $q \stackrel{k}{\equiv} p$ then $p \stackrel{k}{\equiv} q$ because $\rho(q, x) = \rho(p, x)$ implies $\rho(p, x) = \rho(q, x)$
 3. Transitive: result from transitivity of set equality
- **Notation:** Π and Π_k are classes of equivalence defined by:
 1. $[q] \in \Pi$ is defined by $[q] = \{p | q \equiv p\}$
 2. $[q]_k \in \Pi_k$ is defined by $[q]_k = \{p | q \stackrel{k}{\equiv} p\}$

Equivalence of States

- Let M and N be two DGSMs with $\Sigma_M = \Sigma_N = \Sigma$. For each integer $k > 0$, state $q \in Q_m$ is k -equivalent to state $p \in Q_n$ if $\rho_m(q, x) = \rho_n(p, x)$ for all $x \in \Sigma^*$ with $|x| \leq k$ and write it $q \stackrel{k}{\equiv} p$.
- States q and p are equivalent, written $q \equiv p$, if $q \stackrel{k}{\equiv} p$ for all $k \geq 0$
- A DGSM is called distinguished provided it has no two distinct states that are equivalent

Questions

- Given that Σ^* is infinite is there a finite process to determine state equivalence?
- Can k -equivalence be used to develop a series of increasingly better approximations that converge to the equivalence we seek?

DGSM equivalence

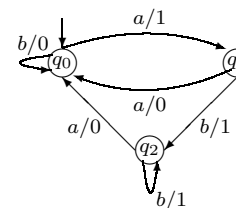
Two DGSM M, N , with input alphabets $\Sigma_m = \Sigma_n$, are equivalent provided that for each $q \in Q_m$ there is $p \in Q_n$ such that $q \equiv p$, and conversely

Note

- q_0 and q_1 are not 1-equivalent because $\rho(q_0, a) = 1$ $\rho(q_1, a) = 0$
- q_1 and q_2 are 1-equivalent because $\rho(q_1, a) = 0$, $\rho(q_2, a) = 0$ and $\rho(q_1, b) = 1$, $\rho(q_2, b) = 1$
- $\Pi_1 = \{[q_0], [q_1, q_2]\}$
- One can verify by observing that:
 1. for any input beginning with "a" both states in $[q_1, q_2]$ initially yield "0" and then continue from $[q_0]$
 2. for any input beginning with "b" both states in $[q_1, q_2]$ yield "1" and then continue with $[q_1, q_2]$
- Hence they are equivalent. I.e. in this case $\Pi = \Pi_1$

Example equivalences

Consider DGSM M below:



Direct Inspection

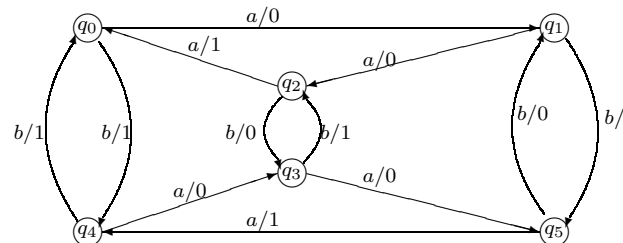
Classes of 1-equivalence:

- $q_2 \stackrel{1}{\equiv} q_5$: $\rho(q_2, a) = \rho(q_5, a) = 1$, $\rho(q_2, b) = \rho(q_5, b) = 0$
Note: q_0 is not equivalent to q_2 or q_5 since $\rho(q_0, b) = 1$
- q_0, q_1, q_3, q_4 are all 1-equivalent as they yield output 0 for input "a" and 1 for input "b"
- That is, $\Pi_1 = [q_0, q_1, q_3, q_4], [q_2, q_5]$

Note: classes of equivalence are more difficult to show. q_0 and q_3 are not 2-equivalent because on "ab", q_0 yield "01" and q_3 yields "00"; however, q_2 and q_5 are 2-equivalent

A more challenging example

Consider DGSM M here:



Note

- The key insight revealed by this result is that we can tell whether or not two k -equivalent states are also $k + 1$ -equivalent by examining k -equivalence of the next states under all letters of Σ
- **Consequence:** from the analysis of k -equivalence and single transitions we can deduce $k + 1$ equivalence

Major result

Let $M = (Q, \Sigma, \Delta, \delta, \rho, q_0)$ be a DGSM and $q, p \in Q$. Then for every $k > 0$:

1. $q \stackrel{k+1}{\equiv} p$ iff $q \stackrel{k}{\equiv} p$ (in Fleck is $q \stackrel{1}{\equiv} p$) and $\delta(q, w) \stackrel{k}{\equiv} \delta(p, w)$ for all $w \in \Sigma$
2. $\Pi_{k+1} = \Pi_k$ implies $\Pi_n = \Pi_k$ for all $n > k$
3. $\Pi_{k+1} = \Pi_k$ implies $q \equiv p$ iff $q \stackrel{k}{\equiv} p$

Proof: see Fleck 151-152

Application

The application of Theorem 3.2.2 is the following algorithm for determining the equivalence of states in a DGSM:

1. Compute Π_1 by inspection of ρ for single letter input
2. Repeat for $k = 1, 2, \dots$ until no class split or $k = n - 1$:
Given Π_k compute Π_{k+1} by:
for each pair $q \stackrel{k}{\equiv} p$:
 - if $\delta(q, w) \stackrel{k}{\equiv} \delta(p, w)$ for all $w \in \Sigma$ then $q \stackrel{k+1}{\equiv} p$
 - otherwise split q and p , that is, set q and p in different classes in Π_{k+1}
3. At termination set $\Pi = \Pi_{k+1}$

Theorem 3.2.2 (Equivalence of States)

Let DGSM $M = (Q, \Sigma, \Delta, \delta, \rho, q_0)$ where $|Q| > n - 2$ states.

$\forall q, p \in Q [q \equiv p] \text{ iff } q \stackrel{n-1}{\equiv} p$

Proof: See Fleck 152,152

Applying the algorithm

1. Step 1: as we already know, $\Pi_1 = \{[q_0, q_1, q_3, q_4], [q_2, q_5]\}$

2. Step 2: $k = 1$:

$\delta(q_0, a) = q_1$, $\delta(q_1, a) = q_2$; since q_1 and q_2 are not in the same class q_0, q_1 are not 2-equivalent; hence q_0 and q_1 are placed in different classes in Π_2

$\delta(q_0, a) = q_1$, $\delta(q_4, a) = q_3$, q_1, q_3 are 1-equivalent; $\delta(q_0, b) = q_4$, $\delta(q_4, b) = q_0$, q_0, q_4 are 1-equivalent. Hence, q_0, q_4 are 2-equivalent. Similarly, $[q_2, q_5]$ does not split. Consequently, $\Pi_2 = \{[q_0, q_4], [q_1, q_3], [q_2, q_5]\}$

Step 2; $k = 2$: Similar analysis shows $\Pi_3 = \Pi_2$

3. Step 3: termination occurs with

$\Pi_2 = \Pi_3 = \Pi = \{[q_0, q_4], [q_1, q_3], [q_2, q_5]\}$

Observations

• $\Pi_k = \{C_1^k, C_2^k, \dots, C_m^k\}$

• $p \stackrel{k}{\equiv} q$ and $\delta(q, w) \stackrel{k}{\equiv} \delta(p, w)$ means that $\exists C_i^k, C_j^k \in \Pi_k$, such that $p, q \in C_i^k$ and $\delta(p, w), \delta(q, w) \in C_j^k$

• Condition $\delta(q, w)$ not k -equivalent to $\delta(p, w)$ means that $\delta(p, w) \in C_j^k$ and $\delta(q, w) \in C_r^k$ for $j \neq r$

Induction basis: $k = 1$

- When $n > 1$, if 1-equivalence were to have a single equivalence class then for any two states, p and q , and any letter $w \in \Sigma$, $\delta(p, w) = \delta(q, w)$
- Then by Lemma 3.2.1, $q \stackrel{2}{\equiv} p$; i.e., $\Pi_2 = \Pi_1$ and again, by the same lemma $\Pi_2 = \Pi_1 = \Pi$
- This implies that all states are equivalent, which contradicts the assumption that M is distinguished and $n > 1$. Hence, 1-equivalence has at least 2 classes

Theorem 3.2.3 (Fleck)

In a distinguished DGSM M with n states, for each k , $1 \leq k < n$, a k -equivalence has at least $k + 1$ equivalence classes, and no k -equivalence class has more than $n - k$ states

Proof: by induction on k .

Note: For $n = 1$ this is vacuously true because there is no k with $1 \leq k < 1$

Conclusion

Since each equivalence class contains at least one state, if there are $k+1$ classes, k of them contain at least k elements, leaving at most $n - k$ for the other class.

Induction step: $k := k + 1$

Assume that for $1 \leq k \leq n - 2$, k -equivalence has at least $k + 1$ classes and consider $(k + 1)$ -equivalence.

- Since Π_{k+1} is a refinement of Π_k , either (a) Π_{k+1} has more classes than Π_k or (b) $\Pi_{k+1} = \Pi_k$
- In case (a) Π_{k+1} has at least $k + 2$ classes (because Π_k has $k + 1$ classes by induction step)
- In case (b), by Lemma 3.2.1, $\Pi_{k+1} = \Pi_n$ and therefore has $n > k + 1$ classes

Isomorphic DGSMs

Let $M = (Q_1, \Sigma, \Delta, \delta_1, \rho_1, q_0)$ and $N = (Q_2, \Sigma, \Delta, \delta_2, \rho_2, p_0)$ be DGSMs.

Definition: M is state isomorphic to N if there is a function $\alpha : Q_1 \rightarrow Q_2$ which is one-to-one and onto (i.e., a bijection) so that for all $q \in Q_1$ and $w \in \Sigma$

- (i) $\rho_1(q, w) = \rho_2(\alpha(q), w)$, and
- (ii) $\alpha(\delta_1(q, w)) = \delta_2(\alpha(q), w)$

Lemma 3.2.4

Let $M = (Q_1, \Sigma, \Delta, \delta_1, \rho_1, q_0)$ and $N = (Q_2, \Sigma, \Delta, \delta_2, \rho_2, p_0)$ be DGSMs and $q \in Q_1, p \in Q_2$: if $q \equiv p$ then $\delta_1^*(q, x) \equiv \delta_2^*(p, x)$ for all $x \in \Sigma^*$

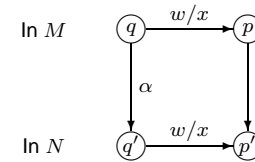
Proof by contradiction: assume that $q \equiv p$ but $\delta_1^*(q, x) \not\equiv \delta_2^*(p, x)$ for some $x \in \Sigma^*$. Then there is $y \in \Sigma^*$ so that $\rho_1(\delta_1(q, x), y) \neq \rho_2(\delta_2(p, x), y)$.

Then, $\rho_1(q, xy) = \rho_1(q, x)\rho_1(\delta_1(q, x), y) \neq \rho_1(q, x)\rho_2(\delta_2(p, x), y) = \rho_2(p, x)\rho_2(\delta_2(p, x), y) = \rho_2(p, xy)$, which contradicts the equivalence of q and p and hence, $\delta_1(q, x) = \delta_2(p, x)$ for all $x \in \Sigma^*$

Note

- If two DGSMs do not have the same number of states, they cannot be isomorphic
- For two DGSMs with n states, there are $n!$ different 1-1 and onto mappings to consider as potential isomorphisms

Illustration of Isomorphism



Note: The critical structural relationship is: *for each transition in M,*
 $\delta_1(q, w) = p$, *if* $\alpha(q) = q'$ *and* $\alpha(p) = p'$ *then* $\delta_2(q', w) = p'$ *in N*

Note

- Both functions δ' and ρ' are well defined
- It is immaterial which of the states are chosen from the class
- Thus, for $q, p \in [q]$, $q \equiv p$ and so $\rho(q, w) = \rho(p, w)$
- By Lemma 3.2.1, $\delta(q, w) \equiv \delta(p, w)$, i.e.,
 $[\delta(q, w)] = [\delta(p, w)]$

Fundamental theorem

Among all DGSMs equivalent to a given DGSM M , there is a unique one (up to isomorphism) which has the fewest states and is distinguished. This is called the reduced machine of M

Proof: By construction. We will show that if $M = (Q, \Sigma, \Delta, \delta, \rho, q_0)$ then its reduced machine is $N = (\Pi, \Sigma, \Delta, \delta', \rho', [q_0])$ where for all $[q] \in \Pi$ and $w \in \Sigma$:

$$\delta'([q], w) = [\delta(q, w)] \text{ and } \rho'([q], w) = \rho(q, w)$$

Claim 2

N is distinguished

Proof: let $[q]$ and $[p]$ be two distinct equivalence classes. Then $q \not\equiv p$ and $\exists x \in \Sigma^* [\rho'([q], x) = \rho(q, x) \neq \rho(p, x) = \rho'([p], x)]$

Hence, $[q] \not\equiv [p]$, thus proving Claim 2

Claim 1

For each $q \in Q$, $q \equiv [q]$ and hence $M \equiv N$

Proof: By induction on the length of the input

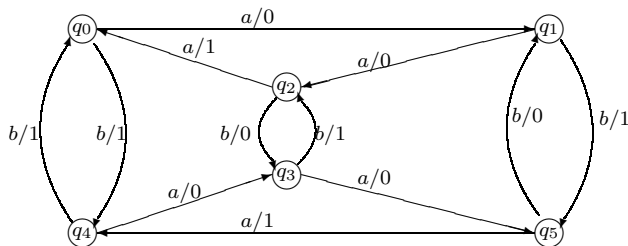
- **Induction basis:** $|w| = 0$: $\rho'([q], \epsilon) = \rho(q, \epsilon) = \epsilon$, for all $q \in Q$
- **Induction step:** assume claim 1 for all $w \in \Sigma^*$, $|w| = k$, and consider the input wx , $x \in \Sigma$. Then we have:
 - $\rho'([q], wx) = \rho'([q], w)\rho'(\delta'([q], w), x)$ (by DGSM property)
 - $= \rho(q, w)\rho'(\delta'([q], w), x)$ (by induction hypothesis)
 - $= \rho(q, w)\rho'(\delta(q, w), x)$ (by δ' definition)
 - $= \rho(q, w)\rho(\delta(q, w), x)$ (by ρ^{prime} definition)
 - $= rho(q, wx)$ (by DGSM property)

Hence, since q and $[q]$ yield the same output for all inputs;

Claim 1 is thus proven

Example

Consider again DGSM M :



We have shown that $\Pi = \{[q_0, q_4], [q_1, q_3], [q_2, q_5]\}$

Claim 3

Any distinguished DGSM M_1 equivalent to M is isomorphic to N

Proof: by construction. Let $M_1 = (Q_1, \Sigma, \Delta, \delta_1, \rho_1, q_1^0)$.

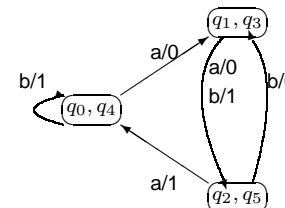
- Define $\alpha : Q_1 \rightarrow \Pi$ as follows: since $M_1 \equiv M$, for each $q_1 \in Q_1$ there is $q \in Q$ so that $q_1 \equiv q$. Let $\alpha(q_1) = [q]$. Since $M_1 \equiv M \equiv N$, α is total and onto. Since M_1 and N are both distinguished, α is one-to-one
- One can check directly that α satisfies the properties of an isomorphism

Application to DFA

- We can view a DFA as a DGSM that has a binary output alphabet:
 1. entering a final state corresponds to "1" output and
 2. entering a non-final state corresponds to "0" output
- Then the DGSM reduction process can be applied to DFA by the following procedure:

Reduced DGSM of M

The reduced machine is obtained by taking the classes $[q_0, q_4], [q_1, q_3], [q_2, q_5]$ as states and by associating the input/output behavior of the states in a class to the entire class:



Final Observations

- The unique reduced machine property is valid only with DFA. For NFA this property is lost because ϵ input cannot distinguish states

Reduction Algorithm for a DFA

1. Discard all states that are unreachable from the start state of the DFA. They affect the transducer's capability but not the recognizer
2. Add outputs from $\{0, 1\}$ to the DFA defining ρ by:

$$\rho(q, w) = \begin{cases} 0, & \text{if } \delta(q, w) \text{ is non-final} \\ 1, & \text{if } \delta(q, w) \text{ is final} \end{cases}$$

3. Perform the reduction process on DGSM produced by step (2), but do not place final and non-final states in the same Π_1 equivalence classes; note that ϵ input does not distinguish states in a DGSM but it does distinguish final and non-final states in a DFA.
4. Ignore the output associated with the reduced DGSM obtained in step (3); this is the minimal DFA