CS 611: Theory of Computation

Hongmin Li

Department of Computer Science
California State University, East Bay

Part I

Closure Properties of Turing Machines

Boolean Operators

Boolean Operators

Proposition

Decidable languages are closed under union, intersection, and complementation.

Boolean Operators

Proposition

Decidable languages are closed under union, intersection, and complementation.

Proof.

Given TMs M_1, M_2 that decide languages L_1, and L_2

- A TM that decides $L_1 \cup L_2$: on input x, run M_1 and M_2 on x, and accept iff either accepts.

Boolean Operators

Proposition

Decidable languages are closed under union, intersection, and complementation.

Proof.

Given TMs M_1, M_2 that decide languages L_1, and L_2

- A TM that decides $L_1 \cup L_2$: on input x, run M_1 and M_2 on x, and accept iff either accepts. (Similarly for intersection.)

Boolean Operators

Proposition

Decidable languages are closed under union, intersection, and complementation.

Proof.

Given TMs M_1, M_2 that decide languages L_1, and L_2

- A TM that decides $L_1 \cup L_2$: on input x, run M_1 and M_2 on x, and accept iff either accepts. (Similarly for intersection.)
- A TM that decides $\overline{L_1}$: On input x, run M_1 on x, and accept if M_1 rejects, and reject if M_1 accepts. $\square$

Regular Operators

Proposition

Decidable languages are closed under concatenation and Kleene Closure.

Proof.

Given TMs M_1 and M_2 that decide languages L_1 and L_2.

- A TM to decide $L_1 L_2$:

Regular Operators

Proposition

Decidable languages are closed under concatenation and Kleene Closure.

Proof.

Given TMs M_1 and M_2 that decide languages L_1 and L_2.

- A TM to decide $L_1 L_2$: On input x, for each of the $|x| + 1$ ways to divide x as yz: run M_1 on y and M_2 on z, and accept if both accept. Else reject.

Regular Operators

Proposition

Decidable languages are closed under concatenation and Kleene Closure.

Proof.

Given TMs M_1 and M_2 that decide languages L_1 and L_2.

- A TM to decide L_1L_2: On input x, for each of the $|x|+1$ ways to divide x as yz: run M_1 on y and M_2 on z, and accept if both accept. Else reject.
- A TM to decide L_1^*:

Regular Operators

Proposition

Decidable languages are closed under concatenation and Kleene Closure.

Proof.

Given TMs M_1 and M_2 that decide languages L_1 and L_2.

- A TM to decide L_1L_2: On input x, for each of the $|x|+1$ ways to divide x as yz: run M_1 on y and M_2 on z, and accept if both accept. Else reject.
- A TM to decide L_1^*: On input x, if $x = \epsilon$ accept. Else, for each of the $2^{|x|-1}$ ways to divide x as $w_1 \dots w_k$ ($w_i \neq \epsilon$): run M_1 on each w_i and accept if M_1 accepts all. Else reject. $\square$

Inverse Homomorphisms

Proposition

Decidable languages are closed under inverse homomorphisms.

Inverse Homomorphisms

Proposition

Decidable languages are closed under inverse homomorphisms.

Proof.

Given TM M_1 that decides L_1, a TM to decide $h^{-1}(L_1)$ is:

Inverse Homomorphisms

Proposition

Decidable languages are closed under inverse homomorphisms.

Proof.

Given TM M_1 that decides L_1, a TM to decide $h^{-1}(L_1)$ is: On input x, compute $h(x)$ and run M_1 on $h(x)$; accept iff M_1 accepts. $\square$

Homomorphisms

Proposition

Decidable languages are not closed under homomorphism

Homomorphisms

Proposition

Decidable languages are not closed under homomorphism

Proof.

We will show a decidable language L and a homomorphism h such that $h(L)$ is undecidable

Homomorphisms

Proposition

Decidable languages are not closed under homomorphism

Proof.

We will show a decidable language L and a homomorphism h such that $h(L)$ is undecidable

- Let $L = \{xy \mid x \in \{0,1\}^*, y \in \{a,b\}^*, x = \langle M, w \rangle$, and y encodes an integer n such that the TM M on input w will halt in n steps $\}$

Homomorphisms

Proposition

Decidable languages are not closed under homomorphism

Proof.

We will show a decidable language L and a homomorphism h such that $h(L)$ is undecidable

- Let $L = \{xy \mid x \in \{0,1\}^*, y \in \{a,b\}^*, x = \langle M, w \rangle$, and y encodes an integer n such that the TM M on input w will halt in n steps $\}$
- L is decidable: can simply simulate M on input w for n steps

Homomorphisms

Proposition

Decidable languages are not closed under homomorphism

Proof.

We will show a decidable language L and a homomorphism h such that $h(L)$ is undecidable

- Let $L = \{xy \mid x \in \{0,1\}^*, y \in \{a,b\}^*, x = \langle M, w \rangle$, and y encodes an integer n such that the TM M on input w will halt in n steps $\}$
- L is decidable: can simply simulate M on input w for n steps
- Consider homomorphism h: $h(0) = 0$, $h(1) = 1$, $h(a) = h(b) = \epsilon$.

Homomorphisms

Proposition

Decidable languages are not closed under homomorphism

Proof.

We will show a decidable language L and a homomorphism h such that $h(L)$ is undecidable

- Let $L = \{xy \mid x \in \{0,1\}^*, y \in \{a,b\}^*, x = \langle M, w\rangle$, and y encodes an integer n such that the TM M on input w will halt in n steps $\}$
- L is decidable: can simply simulate M on input w for n steps
- Consider homomorphism h: $h(0) = 0$, $h(1) = 1$, $h(a) = h(b) = \epsilon$.
- $h(L) =$

Homomorphisms

Proposition

Decidable languages are not closed under homomorphism

Proof.

We will show a decidable language L and a homomorphism h such that $h(L)$ is undecidable

- Let $L = \{xy \mid x \in \{0,1\}^*, y \in \{a,b\}^*, x = \langle M, w \rangle$, and y encodes an integer n such that the TM M on input w will halt in n steps $\}$
- L is decidable: can simply simulate M on input w for n steps
- Consider homomorphism h: $h(0) = 0$, $h(1) = 1$, $h(a) = h(b) = \epsilon$.
- $h(L) = \text{HALT}$ which is undecidable. □

Boolean Operators

Proposition

R.E. languages are closed under union, and intersection.

Boolean Operators

Proposition

R.E. languages are closed under union, and intersection.

Proof.

Given TMs M_1, M_2 that recognize languages L_1, L_2

Boolean Operators

Proposition

R.E. languages are closed under union, and intersection.

Proof.

Given TMs M_1, M_2 that recognize languages L_1, L_2

- A TM that recognizes $L_1 \cup L_2$: on input x, run M_1 and M_2 on x in parallel, and accept iff either accepts.

Boolean Operators

Proposition

R.E. languages are closed under union, and intersection.

Proof.

Given TMs M_1, M_2 that recognize languages L_1, L_2

- A TM that recognizes $L_1 \cup L_2$: on input x, run M_1 and M_2 on x in parallel, and accept iff either accepts. (Similarly for intersection; but no need for parallel simulation) □

Complementation

Proposition

R.E. languages are not closed under complementation.

Proof.

A_{TM} is r.e. but $\overline{A_{\mathrm{TM}}}$ is not. □

Deciding vs. Recognizing

Proposition

If L and $\overline{L}$ are recognizable, then L is decidable

Proof.

Program P for deciding L, given programs P_L and $P_{\overline{L}}$ for recognizing L and $\overline{L}$:

Deciding vs. Recognizing

Proposition

If L and $\overline{L}$ are recognizable, then L is decidable

Proof.

Program P for deciding L, given programs P_L and $P_{\overline{L}}$ for recognizing L and $\overline{L}$:

- On input x, simulate P_L and $P_{\overline{L}}$ on input x.

Deciding vs. Recognizing

Proposition

If L and $\overline{L}$ are recognizable, then L is decidable

Proof.

Program P for deciding L, given programs P_L and $P_{\overline{L}}$ for recognizing L and $\overline{L}$:

- On input x, simulate P_L and $P_{\overline{L}}$ on input x. Whether $x \in L$ or $x \notin L$, one of P_L and $P_{\overline{L}}$ will halt in finite number of steps.
- Which one to simulate first?

Deciding vs. Recognizing

Proposition

If L and $\overline{L}$ are recognizable, then L is decidable

Proof.

Program P for deciding L, given programs P_L and $P_{\overline{L}}$ for recognizing L and $\overline{L}$:

- On input x, simulate P_L and $P_{\overline{L}}$ on input x. Whether $x \in L$ or $x \notin L$, one of P_L and $P_{\overline{L}}$ will halt in finite number of steps.
- Which one to simulate first? Either could go on forever.

Deciding vs. Recognizing

Proposition

If L and $\overline{L}$ are recognizable, then L is decidable

Proof.

Program P for deciding L, given programs P_L and $P_{\overline{L}}$ for recognizing L and $\overline{L}$:

- On input x, simulate P_L and $P_{\overline{L}}$ on input x. Whether $x \in L$ or $x \notin L$, one of P_L and $P_{\overline{L}}$ will halt in finite number of steps.
- Which one to simulate first? Either could go on forever.
- On input x, simulate in parallel P_L and $P_{\overline{L}}$ on input x until either P_L or $P_{\overline{L}}$ accepts

Deciding vs. Recognizing

Proposition

If L and $\overline{L}$ are recognizable, then L is decidable

Proof.

Program P for deciding L, given programs P_L and $P_{\overline{L}}$ for recognizing L and $\overline{L}$:

- On input x, simulate P_L and $P_{\overline{L}}$ on input x. Whether $x \in L$ or $x \notin L$, one of P_L and $P_{\overline{L}}$ will halt in finite number of steps.
- Which one to simulate first? Either could go on forever.
- On input x, simulate in parallel P_L and $P_{\overline{L}}$ on input x until either P_L or $P_{\overline{L}}$ accepts
- If P_L accepts, accept x and halt. If $P_{\overline{L}}$ accepts, reject x and halt.

Deciding vs. Recognizing

Proof (contd).

In more detail, P works as follows:

On input x
for $i = 1, 2, 3, \ldots$
    simulate P_L on input x for i steps
    simulate $P_{\overline{L}}$ on input x for i steps
    if either simulation accepts, break
if P_L accepted, accept x (and halt)
if $P_{\overline{L}}$ accepted, reject x (and halt)

Deciding vs. Recognizing

Proof (contd).

In more detail, P works as follows:

```
On input x
for i = 1, 2, 3, ...
    simulate P_L on input x for i steps
    simulate P_L̄ on input x for i steps
    if either simulation accepts, break
if P_L accepted, accept x (and halt)
if P_L̄ accepted, reject x (and halt)
```

(Alternately, maintain configurations of P_L and $P_{\overline{L}}$, and in each iteration of the loop advance both their simulations by one step.) □

Regular Operations

Proposition

R.E languages are closed under concatenation and Kleene closure.

Proof.

Given TMs M_1 and M_2 recognizing L_1 and L_2

- A TM to recognize $L_1 L_2$:

Regular Operations

Proposition

R.E languages are closed under concatenation and Kleene closure.

Proof.

Given TMs M_1 and M_2 recognizing L_1 and L_2

- A TM to recognize L_1L_2: On input x, do in parallel, for each of the $|x| + 1$ ways to divide x as yz: run M_1 on y and M_2 on z, and accept if both accept. Else reject.

Regular Operations

Proposition

R.E languages are closed under concatenation and Kleene closure.

Proof.

Given TMs M_1 and M_2 recognizing L_1 and L_2

- A TM to recognize L_1L_2: On input x, do in parallel, for each of the $|x|+1$ ways to divide x as yz: run M_1 on y and M_2 on z, and accept if both accept. Else reject.
- A TM to recognize L_1^*:

Regular Operations

Proposition

R.E languages are closed under concatenation and Kleene closure.

Proof.

Given TMs M_1 and M_2 recognizing L_1 and L_2

- A TM to recognize $L_1 L_2$: On input x, do in parallel, for each of the $|x| + 1$ ways to divide x as yz: run M_1 on y and M_2 on z, and accept if both accept. Else reject.
- A TM to recognize L_1^*: On input x, if $x = \epsilon$ accept. Else, do in parallel, for each of the $2^{|x|-1}$ ways to divide x as $w_1 \ldots w_k$ ($w_i \neq \epsilon$): run M_1 on each w_i and accept if M_1 accepts all. Else reject. □

Homomorphisms

Proposition

R.E. languages are closed under both inverse homomorphisms and homomorphisms.

Proof.

Let TM M_1 recognize L_1.

- A TM to recognize $h^{-1}(L_1)$:

Homomorphisms

Proposition

R.E. languages are closed under both inverse homomorphisms and homomorphisms.

Proof.

Let TM M_1 recognize L_1.

- A TM to recognize $h^{-1}(L_1)$:On input x, compute $h(x)$ and run M_1 on $h(x)$; accept iff M_1 accepts.

Homomorphisms

Proposition

R.E. languages are closed under both inverse homomorphisms and homomorphisms.

Proof.

Let TM M_1 recognize L_1.

- A TM to recognize $h^{-1}(L_1)$:On input x, compute $h(x)$ and run M_1 on $h(x)$; accept iff M_1 accepts.
- A TM to recognize $h(L_1)$:

Homomorphisms

Proposition

R.E. languages are closed under both inverse homomorphisms and homomorphisms.

Proof.

Let TM M_1 recognize L_1.

- A TM to recognize $h^{-1}(L_1)$:On input x, compute $h(x)$ and run M_1 on $h(x)$; accept iff M_1 accepts.
- A TM to recognize $h(L_1)$: On input x, start going through all strings w, and if $h(w) = x$, start executing M_1 on w

Homomorphisms

Proposition

R.E. languages are closed under both inverse homomorphisms and homomorphisms.

Proof.

Let TM M_1 recognize L_1.

- A TM to recognize $h^{-1}(L_1)$:On input x, compute $h(x)$ and run M_1 on $h(x)$; accept iff M_1 accepts.
- A TM to recognize $h(L_1)$: On input x, start going through all strings w, and if $h(w) = x$, start executing M_1 on w, using dovetailing to interleave with other executions of M_1.

Homomorphisms

Proposition

R.E. languages are closed under both inverse homomorphisms and homomorphisms.

Proof.

Let TM M_1 recognize L_1.

- A TM to recognize $h^{-1}(L_1)$:On input x, compute $h(x)$ and run M_1 on $h(x)$; accept iff M_1 accepts.
- A TM to recognize $h(L_1)$: On input x, start going through all strings w, and if $h(w) = x$, start executing M_1 on w, using dovetailing to interleave with other executions of M_1. Accept if any of the executions accepts. □