Theory of Computation
Turing machine closure properties

Arjun Chandrasekhar

Turing Machine Closure Properties

Turing Machine Closure Properties

- We have seen that regular languages are closed under complement, union, intersection, concatenation, star, shuffle, ...

Turing Machine Closure Properties

- We have seen that regular languages are closed under complement, union, intersection, concatenation, star, shuffle, ...
- What operations are decidable languages closed under?

Turing Machine Closure Properties

- We have seen that regular languages are closed under complement, union, intersection, concatenation, star, shuffle, . . .
- What operations are decidable languages closed under?
- What operations are recursively enumerable (RE) langauges closed under?

Turing Machines as Java Programs

Turing Machines as Java Programs

- For these problems, you can always think of Turing Machines as Java programs

Turing Machines as Java Programs

- For these problems, you can always think of Turing Machines as Java programs
  - Or Python if you prefer!

Turing Machines as Java Programs

- For these problems, you can always think of Turing Machines as Java programs
  - Or Python if you prefer!
  - Or Haskell if you're streets ahead :)

Turing Machines as Java Programs

- For these problems, you can always think of Turing Machines as Java programs
  - Or Python if you prefer!
  - Or Haskell if you're streets ahead :)
  - Or (literally) ANY language

Turing Machines as Java Programs

- For these problems, you can always think of Turing Machines as Java programs
  - Or Python if you prefer!
  - Or Haskell if you're streets ahead :)
  - Or (literally) ANY language
- We don't need tape-level descriptions

Turing Machines as Java Programs

- For these problems, you can always think of Turing Machines as Java programs
  - Or Python if you prefer!
  - Or Haskell if you're streets ahead :)
  - Or (literally) ANY language
- We don't need tape-level descriptions
- Java programs are algorithms, and algorithms are Turing machines (Church-Turing thesis)

Closure of Java Programs under Union

Closure of Java Programs under Union

- Let's say I have two java programs called Foo.java, and Bar.java

Closure of Java Programs under Union

- Let's say I have two java programs called Foo.java, and Bar.java
  - Each program a string *w* as input and prints out either ACCEPT or REJECT

Closure of Java Programs under Union

- Let's say I have two java programs called Foo.java, and Bar.java
  - Each program a string w as input and prints out either ACCEPT or REJECT
  - Let's say each program is guaranteed to halt

Closure of Java Programs under Union

- Let's say I have two java programs called Foo.java, and Bar.java
  - Each program a string w as input and prints out either ACCEPT or REJECT
  - Let's say each program is <u>guaranteed</u> to halt
- How would you write a java program called FooBar.java that checks if a string w is accepted by *either* Foo.java or by Bar.java (or both)?

Closure of Java Programs under Union

FooBar.java does the following:

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes w as input

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes *w* as input
2. Run Foo.java and pass *w* as the input

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes w as input
2. Run Foo.java and pass w as the input
3. Run Bar.java and pass w as the input

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes *w* as input
2. Run Foo.java and pass *w* as the input
3. Run Bar.java and pass *w* as the input
4. If either program prints ACCEPT, then FooBar.java prints ACCEPT. Otherwise, it prints REJECT

Closure of Java Programs under Union

Closure of Java Programs under Union

- Let's say I have two java programs called Foo.java, and Bar.java

Closure of Java Programs under Union

- Let's say I have two java programs called Foo.java, and Bar.java
  - Each program a string w as input and prints out either ACCEPT or REJECT

Closure of Java Programs under Union

- Let's say I have two java programs called Foo.java, and Bar.java
  - Each program a string *w* as input and prints out either ACCEPT or REJECT
  - Now let's assume that either program could go into an infinite loop.

Closure of Java Programs under Union

- Let's say I have two java programs called Foo.java, and Bar.java
  - Each program a string w as input and prints out either ACCEPT or REJECT
  - Now let's assume that either program could go into an infinite loop.
- How would you write a java program called FooBar.java that checks if a string w is accepted by either Foo.java or by Bar.java (or both)?

Closure of Java Programs under Union

FooBar.java does the following:

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes w as input

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes w as input
2. Run Foo.java and pass w as the input

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes w as input
2. Run Foo.java and pass w as the input
3. Run Bar.java and pass w as the input

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes *w* as input
2. Run Foo.java and pass *w* as the input
3. Run Bar.java and pass *w* as the input
4. If either program prints ACCEPT, then FooBar.java prints ACCEPT. Otherwise, it prints REJECT

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes *w* as input
2. Run Foo.java and pass *w* as the input
3. Run Bar.java and pass *w* as the input
4. If either program prints ACCEPT, then FooBar.java prints ACCEPT. Otherwise, it prints REJECT

Will this work?

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes w as input
2. ~~Run Foo.java and pass w as the input~~
   This might loop, and we'll never get to run Bar!
3. Run Bar.java and pass w as the input
4. If either program prints ACCEPT, then FooBar.java prints ACCEPT. Otherwise, it prints REJECT

Will this work?

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes w as input

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes w as input
2. Run Foo.java and Bar.java in parallel

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes w as input
2. Run Foo.java and Bar.java in parallel
   - Use some sort of timer to let the machines take turns running

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes w as input
2. Run Foo.java and Bar.java in parallel
   - Use some sort of timer to let the machines take turns running
3. If either program ever prints out ACCEPT, then FooBar.java prints ACCEPT.

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes w as input
2. Run Foo.java and Bar.java in parallel
   - Use some sort of timer to let the machines take turns running
3. If either program ever prints out ACCEPT, then FooBar.java prints ACCEPT.
4. If both print REJECT, Foobar.java prints REJECT.

Closure of Java Programs under Union

FooBar.java does the following:

1. FooBar.java takes w as input
2. Run Foo.java and Bar.java in parallel
   - Use some sort of timer to let the machines take turns running
3. If either program ever prints out ACCEPT, then FooBar.java prints ACCEPT.
4. If both print REJECT, Foobar.java prints REJECT.
5. Otherwise FooBar.java runs forever.

Closure of Decidable Languages under Union

Closure of Decidable Languages under Union

Let's prove that decidable languages are closed under union

Closure of Decidable Languages under Union

Let's prove that decidable languages are closed under union

- Want to show that if A and B are decidable, then $A \cup B$ is decidable

Closure of Decidable Languages under Union

Suppose A and B are decidable

Closure of Decidable Languages under Union

Suppose A and B are decidable

- There are machines M_A, M_B that decide A and B

Closure of Decidable Languages under Union

Suppose A and B are decidable

- There are machines M_A, M_B that decide A and B
- Create a machine M to decide $A \cup B$

Closure of Decidable Languages under Union

Suppose A and B are decidable

- There are machines M_A, M_B that decide A and B
- Create a machine M to decide $A \cup B$
- M does the following on input w:

Closure of Decidable Languages under Union

Suppose A and B are decidable

- There are machines M_A, M_B that decide A and B
- Create a machine M to decide $A \cup B$
- M does the following on input w:
    1. Run M_A on w

Closure of Decidable Languages under Union

Suppose A and B are decidable

- There are machines M_A, M_B that decide A and B
- Create a machine M to decide $A \cup B$
- M does the following on input w:
  1. Run M_A on w
  2. Run M_B on w

Closure of Decidable Languages under Union

Suppose A and B are decidable

- There are machines M_A, M_B that decide A and B
- Create a machine M to decide $A \cup B$
- M does the following on input w:
  1. Run M_A on w
  2. Run M_B on w
  3. If either machine accepts, M accepts. Otherwise, M rejects

Closure of RE Languages under Union

Closure of RE Languages under Union

Let's prove that RE languages are closed under union

Closure of RE Languages under Union

Let's prove that RE languages are closed under union

- Want to show if A and B are RE, then $A \cup B$ is RE

Closure of RE Languages under Union

Suppose A and B are RE

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
    1. Run M_A on w

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
  1. Run M_A on w
  2. Run M_B on w

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
  1. Run M_A on w
  2. Run M_B on w
  3. If either machine accepts, M accepts. Otherwise, M rejects

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
    1. Run M_A on w
    2. Run M_B on w
    3. If either machine accepts, M accepts. Otherwise, M rejects

Will this work?

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
  1. ~~Run M_A on w~~
     This might loop forever!
  2. Run M_B on w
  3. If either machine accepts, M accepts. Otherwise, M rejects

Will this work?

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
  1. Run M_A and M_B in parallel

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
  1. Run M_A and M_B in parallel
     1.1 Run M_A for one step

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
  1. Run M_A and M_B in parallel
     1.1 Run M_A for one step
     1.2 Run M_B for one step

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
  1. Run M_A and M_B in parallel
     1.1 Run M_A for one step
     1.2 Run M_B for one step
     1.3 Run M_A for one step

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
  1. Run M_A and M_B in parallel
     1.1 Run M_A for one step
     1.2 Run M_B for one step
     1.3 Run M_A for one step
     1.4 Run M_B for one step

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
  1. Run M_A and M_B in parallel
     1.1 Run M_A for one step
     1.2 Run M_B for one step
     1.3 Run M_A for one step
     1.4 Run M_B for one step
     1.5 ...

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
  1. Run M_A and M_B in parallel
     1.1 Run M_A for one step
     1.2 Run M_B for one step
     1.3 Run M_A for one step
     1.4 Run M_B for one step
     1.5 ...
  2. If either M_A or M_B (ever) accepts, then M accepts

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a machine M to recognize $A \cup B$
- M does the following on input w:
  1. Run M_A and M_B in parallel
     1.1 Run M_A for one step
     1.2 Run M_B for one step
     1.3 Run M_A for one step
     1.4 Run M_B for one step
     1.5 ...
  2. If either M_A or M_B (ever) accepts, then M accepts
  3. If neither machine (ever) accepts, then M will never accept - which is sufficient

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a nondeterministic machine M to recognize $A \cup B$

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a nondeterministic machine M to recognize $A \cup B$
- M does the following on input w:

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a nondeterministic machine M to recognize $A \cup B$
- M does the following on input w:
  1. On input w, nondeterministically guess whether to $w \in A$ or $w \in B$

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a nondeterministic machine M to recognize $A \cup B$
- M does the following on input w:
    1. On input w, nondeterministically guess whether to $w \in A$ or $w \in B$
    2. Either run M_A or M_B, depending on which language you guessed

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a nondeterministic machine M to recognize $A \cup B$
- M does the following on input w:
    1. On input w, nondeterministically guess whether to $w \in A$ or $w \in B$
    2. Either run M_A or M_B, depending on which language you guessed
    3. If the guessed machine accepts M will accept

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a nondeterministic machine M to recognize $A \cup B$
- M does the following on input w:
    1. On input w, nondeterministically guess whether to $w \in A$ or $w \in B$
    2. Either run M_A or M_B, depending on which language you guessed
    3. If the guessed machine accepts M will accept
    4. If neither machine accepts, M will not accept no matter how it guesses (which is sufficient)

Closure of RE Languages under Union

Suppose A and B are RE

- There are machines M_A, M_B that recognize A and B
- Create a nondeterministic machine M to recognize $A \cup B$
- M does the following on input w:
  1. On input w, nondeterministically guess whether to $w \in A$ or $w \in B$
  2. Either run M_A or M_B, depending on which language you guessed
  3. If the guessed machine accepts M will accept
  4. If neither machine accepts, M will not accept no matter how it guesses (which is sufficient)
- Every nondeterministic TM can be converted to a deterministic TM

Closure Properties of Turing Machines

- Prove that decidable languages are closed under intersection
- Prove that RE languages are closed under intersection

Closure of Decidable Languages under Intersection

- Suppose A and B are decidable

Closure of Decidable Languages under Intersection

- Suppose A and B are decidable
- Let M_A and M_B decide A and B, respectively

Closure of Decidable Languages under Intersection

- Suppose A and B are decidable
- Let M_A and M_B decide A and B, respectively
- We construct a machine M to decide $A \cap B$

Closure of Decidable Languages under Intersection

- Suppose A and B are decidable
- Let M_A and M_B decide A and B, respectively
- We construct a machine M to decide $A \cap B$
- M does the following:

Closure of Decidable Languages under Intersection

- Suppose A and B are decidable
- Let M_A and M_B decide A and B, respectively
- We construct a machine M to decide $A \cap B$
- M does the following:
    1. M takes w as input

Closure of Decidable Languages under Intersection

- Suppose A and B are decidable
- Let M_A and M_B decide A and B, respectively
- We construct a machine M to decide $A \cap B$
- M does the following:
  1. M takes w as input
  2. Run M_A on w

Closure of Decidable Languages under Intersection

- Suppose A and B are decidable
- Let M_A and M_B decide A and B, respectively
- We construct a machine M to decide $A \cap B$
- M does the following:
  1. M takes w as input
  2. Run M_A on w
  3. Run M_B on w

Closure of Decidable Languages under Intersection

- Suppose A and B are decidable
- Let M_A and M_B decide A and B, respectively
- We construct a machine M to decide $A \cap B$
- M does the following:
    1. M takes w as input
    2. Run M_A on w
    3. Run M_B on w
    4. If M_A and M_B both accept w, then M accepts w

Closure of Decidable Languages under Intersection

- Suppose A and B are decidable
- Let M_A and M_B decide A and B, respectively
- We construct a machine M to decide $A \cap B$
- M does the following:
  1. M takes w as input
  2. Run M_A on w
  3. Run M_B on w
  4. If M_A and M_B both accept w, then M accepts w
  5. If either machine rejects, then M rejects

Closure of RE Languages under Intersection

- Suppose A and B are RE

Closure of RE Languages under Intersection

- Suppose A and B are RE
- Let M_A and M_B recognize A and B, respectively

Closure of RE Languages under Intersection

- Suppose A and B are RE
- Let M_A and M_B recognize A and B, respectively
- We construct a machine M to recognize $A \cap B$

Closure of RE Languages under Intersection

- Suppose A and B are RE
- Let M_A and M_B recognize A and B, respectively
- We construct a machine M to recognize $A \cap B$
- M does the following:

Closure of RE Languages under Intersection

- Suppose A and B are RE
- Let M_A and M_B recognize A and B, respectively
- We construct a machine M to recognize $A \cap B$
- M does the following:
  1. M takes w as input

Closure of RE Languages under Intersection

- Suppose A and B are RE
- Let M_A and M_B recognize A and B, respectively
- We construct a machine M to recognize $A \cap B$
- M does the following:
  1. M takes w as input
  2. Run M_A and M_B in parallel on w

Closure of RE Languages under Intersection

- Suppose A and B are RE
- Let M_A and M_B recognize A and B, respectively
- We construct a machine M to recognize $A \cap B$
- M does the following:
  1. M takes w as input
  2. Run M_A and M_B in parallel on w
  3. If M_A and M_B both accept, M accepts w

Closure of RE Languages under Intersection

- Suppose A and B are RE
- Let M_A and M_B recognize A and B, respectively
- We construct a machine M to recognize $A \cap B$
- M does the following:
  1. M takes w as input
  2. Run M_A and M_B in parallel on w
  3. If M_A and M_B both accept, M accepts w
  4. If $w \notin A \cap B$ then M might loop forever but that's ok

Closure Properties of Turing Machines

For any language A, let

$$\#(A) = \{w = w_1\#w_2\#\ldots\#w_n | w_i \in A\}$$

i.e. several strings in A each separated by a $\#$ sign

Closure Properties of Turing Machines

For any language A, let

$$\#(A) = \{w = w_1\#w_2\#\ldots\#w_n | w_i \in A\}$$

i.e. several strings in A each separated by a $\#$ sign

- Prove that decidable languages are closed under $\#$
- Prove that RE languages are closed under $\#$

Closure of Decidable Languages under

Closure of Decidable Languages under

- Suppose A is decidable

Closure of Decidable Languages under

- Suppose A is decidable
- Let M_A decide A

Closure of Decidable Languages under

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide $\#(A)$.

Closure of Decidable Languages under

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide $\#(A)$.
- M does the following:

Closure of Decidable Languages under

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide $\#(A)$.
- M does the following:
    1. M takes w as input

Closure of Decidable Languages under

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide $\#(A)$.
- M does the following:
  1. M takes w as input
  2. Check that $w = w_1 \# w_2 \ldots \# w_n$ (i.e. correct format)

Closure of Decidable Languages under

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide $\#(A)$.
- M does the following:
  1. M takes w as input
  2. Check that $w = w_1\#w_2\ldots\#w_n$ (i.e. correct format)
  3. Run M_A on each w_i

Closure of Decidable Languages under $\#$

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide $\#(A)$.
- M does the following:
  1. M takes w as input
  2. Check that $w = w_1\#w_2\ldots\#w_n$ (i.e. correct format)
  3. Run M_A on each w_i
  4. If M_A accepts each w_i accept. Otherwise, reject

Closure of RE Languages under

Closure of RE Languages under $\#$

- Suppose A is RE

Closure of RE Languages under

- Suppose A is RE
- Let M_A recognize A

Closure of RE Languages under

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize $\#(A)$

Closure of RE Languages under $\#$

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize $\#(A)$
- M does the following:

Closure of RE Languages under

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize $\#(A)$
- M does the following:
  1. M takes w as input

Closure of RE Languages under

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize $\#(A)$
- M does the following:
  1. M takes w as input
  2. Check that $w = w_1 \# w_2 \ldots \# w_n$ (i.e. correct format)

Closure of RE Languages under

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize $\#(A)$
- M does the following:
  1. M takes w as input
  2. Check that $w = w_1 \# w_2 \ldots \# w_n$ (i.e. correct format)
  3. Run M_A in parallel on each w_i

Closure of RE Languages under

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize $\#(A)$
- M does the following:
  1. M takes w as input
  2. Check that $w = w_1\#w_2\ldots\#w_n$ (i.e. correct format)
  3. Run M_A in parallel on each w_i
  4. If M_A accepts each w_i, then M accepts w.

Closure of RE Languages under

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize $\#(A)$
- M does the following:
  1. M takes w as input
  2. Check that $w = w_1\#w_2\ldots\#w_n$ (i.e. correct format)
  3. Run M_A in parallel on each w_i
  4. If M_A accepts each w_i, then M accepts w.
  5. If any $w_i \notin A$ then M may loop forever, and that's ok

Closure Properties of Turing Machines

Recall that for any language A, let

$$A^* = \{w = w_1 w_2 \dots w_n | w_i \in A\}$$

Closure Properties of Turing Machines

Recall that for any language A, let

$$A^* = \{w = w_1 w_2 \ldots w_n | w_i \in A\}$$

- Prove that decidable languages are closed under Kleene star
- Prove that RE languages are closed under Kleene star

Closure of Decidable Languages under Kleene Star

Closure of Decidable Languages under Kleene Star

- Suppose A is decidable

Closure of Decidable Languages under Kleene Star

- Suppose A is decidable
- Let M_A decide A

Closure of Decidable Languages under Kleene Star

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^*

Closure of Decidable Languages under Kleene Star

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^*
- M does the following:

Closure of Decidable Languages under Kleene Star

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^*
- M does the following:
  1. M takes w as input

Closure of Decidable Languages under Kleene Star

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^*
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \ldots w_n$

Try all possible ways of splitting up w

Split 1: w | w | w | w | w | w | w ...

Split 2: w w w w | w w | w ...

Split 3: w | w w | w w | w w ...

Split 4: w | w w w w w w ...

Closure of Decidable Languages under Kleene Star

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^*
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \dots w_n$
     2.1 For each way of splitting it up, run M_A on each w_i

Closure of Decidable Languages under Kleene Star

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^*
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \dots w_n$
     - 2.1 For each way of splitting it up, run M_A on each w_i
     - 2.2 If M_A accepts each w_i, then M accepts

Closure of Decidable Languages under Kleene Star

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^*
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \ldots w_n$
     - 2.1 For each way of splitting it up, run M_A on each w_i
     - 2.2 If M_A accepts each w_i, then M accepts
     - 2.3 Otherwise move on to the next way of splitting up w

Closure of Decidable Languages under Kleene Star

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^*
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \dots w_n$
     - 2.1 For each way of splitting it up, run M_A on each w_i
     - 2.2 If M_A accepts each w_i, then M accepts
     - 2.3 Otherwise move on to the next way of splitting up w
  3. If all splits are rejected, then M rejects

Closure of RE Languages under Kleene Star

Closure of RE Languages under Kleene Star

- Suppose A is RE

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \ldots w_n$

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \ldots w_n$
     2.1 For each way of splitting it up, run M_A on each w_i

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \dots w_n$
     2.1 For each way of splitting it up, run M_A on each w_i
     2.2 If M_A accepts each w_i, then M accepts

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \ldots w_n$
     2.1 For each way of splitting it up, run M_A on each w_i
     2.2 If M_A accepts each w_i, then M accepts
     2.3 Otherwise move on to the next way of splitting up w

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \ldots w_n$
     2.1 For each way of splitting it up, run M_A on each w_i
     2.2 If M_A accepts each w_i, then M accepts
     2.3 Otherwise move on to the next way of splitting up w
  3. If all splits are rejected, then M rejects

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \ldots w_n$
     2.1 For each way of splitting it up, run M_A on each w_i
     2.2 If M_A accepts each w_i, then M accepts
     2.3 Otherwise move on to the next way of splitting up w
  3. If all splits are rejected, then M rejects

Will this work?

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \dots w_n$
     2.1 For each way of splitting it up, run M_A on each w_i
     2.2 If M_A accepts each w_i, then M accepts
     2.3 ~~Otherwise move on to the next way of splitting up w~~
     M_A may loop on some w_i, and we don't get to try other splits
  3. If all splits are rejected, then M rejects

Will this work?

Closure of RE Languages under Kleene Star

- Suppose A is RE

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \ldots w_n$ in parallel

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \ldots w_n$ in parallel
     2.1 For each way of splitting it up, run M_A on each w_i in parallel.

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \ldots w_n$ <u>in parallel</u>
     2.1 For each way of splitting it up, run M_A on each w_i <u>in parallel.</u>
  3. If M_A accepts each w_i <u>for any split</u>, then M accepts

Closure of RE Languages under Kleene Star

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^*.
- M does the following:
  1. M takes w as input
  2. Try all possible ways of splitting up $w = w_1 w_2 \ldots w_n$ in parallel
     - 2.1 For each way of splitting it up, run M_A on each w_i in parallel.
  3. If M_A accepts each w_i for any split, then M accepts
  4. If every way of splitting up w fails, M may loop, but that's ok

Closure Properties of Turing Machines

- Suppose A is RE

Closure Properties of Turing Machines

- Suppose A is RE
- Let M_A recognize A

Closure Properties of Turing Machines

- Suppose A is RE
- Let M_A recognize A
- Create a nondeterministic machine M to recognize A^* as follows:

Closure Properties of Turing Machines

- Suppose A is RE
- Let M_A recognize A
- Create a <u>nondeterministic</u> machine M to recognize A^* as follows:
  1. M takes w as input

Closure Properties of Turing Machines

- Suppose A is RE
- Let M_A recognize A
- Create a nondeterministic machine M to recognize A^* as follows:
  1. M takes w as input
  2. Nondeterministically guess how to split up $w = w_1 w_2 \dots w_n$

Closure Properties of Turing Machines

- Suppose A is RE
- Let M_A recognize A
- Create a nondeterministic machine M to recognize A^* as follows:
  1. M takes w as input
  2. Nondeterministically guess how to split up $w = w_1 w_2 \ldots w_n$
  3. Run M_A on each w_i in parallel

Closure Properties of Turing Machines

- Suppose A is RE
- Let M_A recognize A
- Create a nondeterministic machine M to recognize A^* as follows:
  1. M takes w as input
  2. Nondeterministically guess how to split up $w = w_1 w_2 \dots w_n$
  3. Run M_A on each w_i in parallel
  4. If M_A accepts each w_i (i.e. we guessed correctly) then M accepts

Closure Properties of Turing Machines

- Suppose A is RE
- Let M_A recognize A
- Create a nondeterministic machine M to recognize A^* as follows:
  1. M takes w as input
  2. Nondeterministically guess how to split up $w = w_1 w_2 \dots w_n$
  3. Run M_A on each w_i in parallel
  4. If M_A accepts each w_i (i.e. we guessed correctly) then M accepts
- Every nondeterministic TM can be converted to a deterministic TM

Closure Properties of Turing Machines

- Prove that decidable languages are closed under complement
- Prove that RE languages are closed under complement

Closure of Decidable Languages under Complement

- Suppose A is decidable

Closure of Decidable Languages under Complement

- Suppose A is decidable
- Let M_A decide A

Closure of Decidable Languages under Complement

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^c

Closure of Decidable Languages under Complement

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^c
- M does the following:

Closure of Decidable Languages under Complement

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^c
- M does the following:
    1. M takes w as input

Closure of Decidable Languages under Complement

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^c
- M does the following:
  1. M takes w as input
  2. Run M_A on w

Closure of Decidable Languages under Complement

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^c
- M does the following:
  1. M takes w as input
  2. Run M_A on w
  3. If M_A accepts, M rejects

Closure of Decidable Languages under Complement

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^c
- M does the following:
  1. M takes w as input
  2. Run M_A on w
  3. If M_A accepts, M rejects
  4. If M_A rejects, M accepts

Closure of Decidable Languages under Complement

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^c
- M does the following:
  1. M takes w as input
  2. Run M_A on w
  3. If M_A accepts, M rejects
  4. If M_A rejects, M accepts
- M_A always halts, so M always halts

Closure of Decidable Languages under Complement

- Suppose A is decidable
- Let M_A decide A
- Create a machine M to decide A^c
- M does the following:
  1. M takes w as input
  2. Run M_A on w
  3. If M_A accepts, M rejects
  4. If M_A rejects, M accepts
- M_A always halts, so M always halts
- M accepts $w \Leftrightarrow M_A$ rejects $w \Leftrightarrow w \notin A$

Closure of RE Languages under Complement

- Suppose A is RE

Closure of RE Languages under Complement

- Suppose A is RE
- Let M_A recognize A

Closure of RE Languages under Complement

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^c.

Closure of RE Languages under Complement

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^c.
- M does the following:

Closure of RE Languages under Complement

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^c.
- M does the following:
    1. M takes w as input

Closure of RE Languages under Complement

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^c.
- M does the following:
  1. M takes w as input
  2. Run M_A on w

Closure of RE Languages under Complement

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^c.
- M does the following:
  1. M takes w as input
  2. Run M_A on w
  3. If M_A accepts, M rejects

Closure of RE Languages under Complement

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^c.
- M does the following:
  1. M takes w as input
  2. Run M_A on w
  3. If M_A accepts, M rejects
  4. If M_A rejects, M accepts

Closure of RE Languages under Complement

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^c.
- M does the following:
  1. M takes w as input
  2. Run M_A on w
  3. If M_A accepts, M rejects
  4. If M_A rejects, M accepts

Will this work?

Closure of RE Languages under Complement

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^c.
- M does the following:
  1. M takes w as input
  2. Run M_A on w
  3. If M_A accepts, M rejects
  4. If M_A rejects, M accepts
  5. If M_A loops, then M loops

Will this work?

Closure of RE Languages under Complement

- Suppose A is RE
- Let M_A recognize A
- Create a machine M to recognize A^c.
- M does the following:
  1. M takes w as input
  2. Run M_A on w
  3. If M_A accepts, M rejects
  4. If M_A rejects, M accepts
  5. If M_A loops, then M loops
- M may not accept strings that are part of A^c

Will this work?

Closu
Comp

-
-
-
-

- A^c

Will

Closure of RE Languages under Complement

- As it turns out, RE languages are NOT closed under complement.

Closure of RE Languages under Complement

- As it turns out, RE languages are NOT closed under complement.
- We will study techniques to prove such statements next week.

Closure Properties of Turing Machines

Recap

Closure Properties of Turing Machines

Recap

- Decidable languages are closed under union, intersection, complement, Kleene star

Closure Properties of Turing Machines

Recap

- Decidable languages are closed under union, intersection, complement, Kleene star
- RE languages are closed under union, intersection, Kleene star

Closure Properties of Turing Machines

Recap

- Decidable languages are closed under union, intersection, complement, Kleene star
- RE languages are closed under union, intersection, Kleene star
  - We need to be careful and run machines/computation paths in parallel

Closure Properties of Turing Machines

Recap

- Decidable languages are closed under union, intersection, complement, Kleene star
- RE languages are closed under union, intersection, Kleene star
  - We need to be careful and run machines/computation paths in parallel
- RE languages are not closed under complement