CSE 105
THEORY OF COMPUTATION

* all HW solutions available on Piazza

Winter 2018 review class

* Review Quiz 10

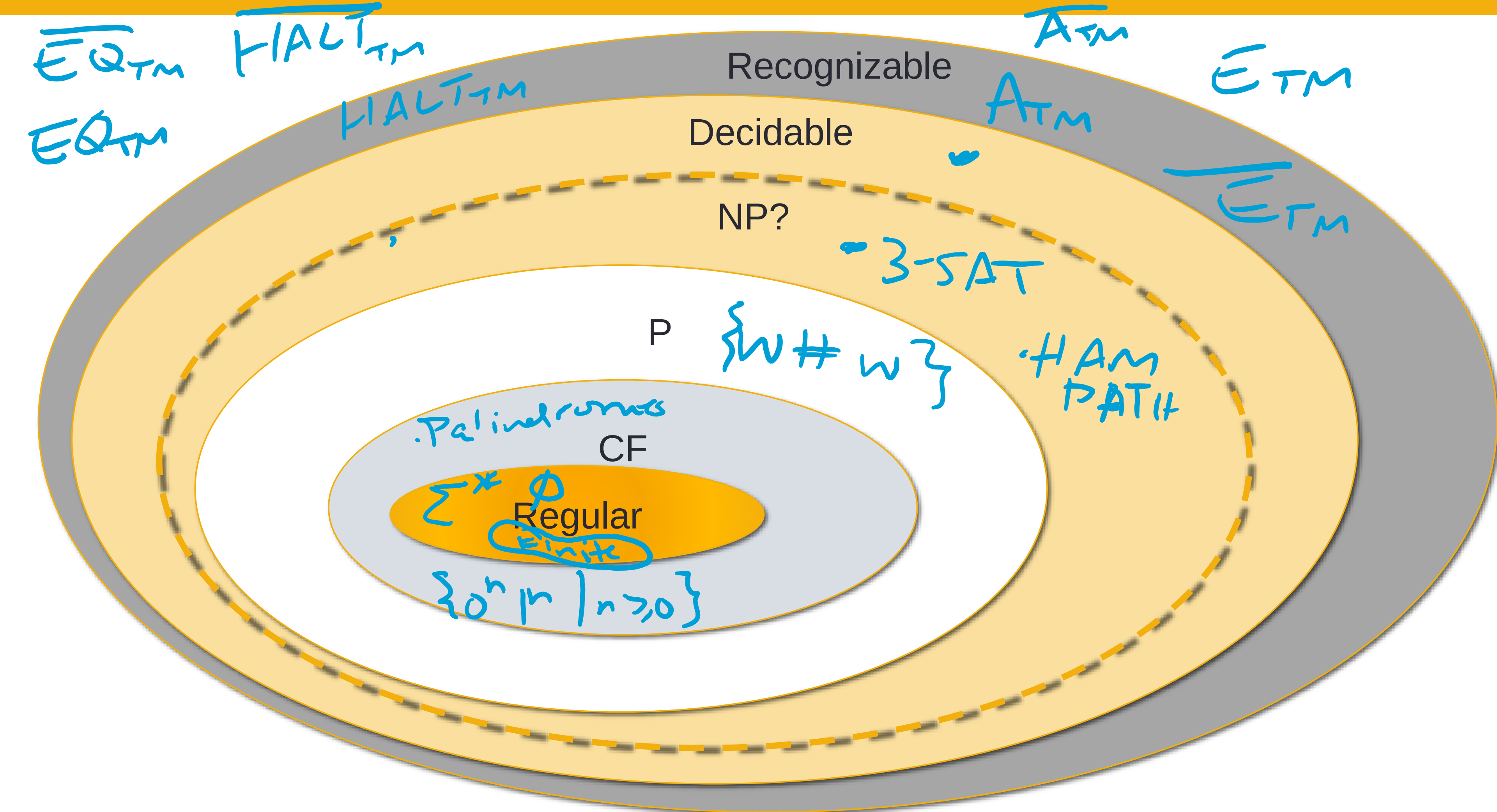
* new individual report today

Today's learning goals

- Summarize key concepts, ideas, themes from CSE 105.
- Approach your final exam studying with confidence.
- Identify areas to focus on while studying for the exam.

Reminders

- CAPE and TA evaluations open
- Final exam **Saturday March 17 11:30am-2:29pm**
- Seat map & study guide on Piazza.
  - Discussion tomorrow will go over some of study guide.



Recognizable
Decidable
NP?
P
CF
Regular
HALT_TM
EQ_TM
A_TM
E_TM
3-SAT
HAM PATH
{w#w}
Palindromes
Σ*
∅
Finite
{0^n 1^n | n≥0}

Model of computation *Formal definition?* *Design?* *Describe language?*	Class of languages *Closure properties?* *Which languages not in class?*
Finite automata -- DFA -- NFA equiv to **Regular expressions**	**Regular languages** To show not in class: Pumping lemma
Push-down automata (state diagram, informal) **CFGs**	**Context-free languages** (stack)
TMs that always halt in polynomial time	**P**
Nondeterministic TMs that halt in polynomial time	**NP**
TMs that always halt aka **Deciders** * (enumerators, nondet TM, 2-tape TM)	**Decidable languages** To show not in class: Diagonalization, reduction
* Turing Machines (in general; may not halt)	**Recognizable languages**

Roadmap of examples

A. Regular language design
B. Undecidability via reduction
C. Closure proofs
D. Determining the language of a PDA /CFG
E. Using Pumping Lemma

Given L, prove it is regular

Construction

Strategy 1: Construct DFA

Strategy 2: Construct NFA

Strategy 3: Construct regular expression

Proof of correctness

WTS 1 if w is in L then w is accepted by ….

WTS 2 if w is not in L then w is rejected by …

Ex: L= { w in {0,1}* | w has odd # of 1s OR starts with 0}

NFA:

Regular expression:

To show a language is **not** regular, we can

A. Show there is a CFG generating A.
B. Use the pumping lemma for regular languages.
C. Show A is undecidable.
D. More than one of the abve.
E. I don't know.

To show a language L is …

Recognizable

- Show there is a TM M with L(M) = L.
- Use closure properties.

Not recognizable

- Prove that L is not decidable and that the complement of L is recognizable.
- *Use closure properties.*

To show a language L is …

Decidable

- Show there is a TM D that always halts and L(D) = L.
- Find a decidable problem L' and show L reduces to L'
- Use closure properties.

Not decidable

- Use diagonalization
- Find an undecidable problem L' and show L' reduces to L.
- *Use closure properties.*

Undecidability via reduction

Theorem: Problem T is undecidable.

Proof Common pattern for many of these proofs.

Assume (towards a contradiction) that T is decidable by TM M_T. Goal: use M_T to build a machine which will decide A_{TM}.

Define M_{ATM} = "On input <M,w>:

1. Using the parameters M and w, construct a different TM X such that if M accepts w, then <X> is in T; if M does not accept w, then <X> is not in T.
2. Run M_T on <X> and accept if M_T accepts, reject if M_T rejects."

Claim: M_{ATM} is decider and $L(M_{ATM}) = A_{TM}$. Then A_{TM} is decidable, contradicting the known fact that A_{TM} is undecidable.

T = { <M> | M is TM and |L(M)| = 1}

Theorem: Problem T is undecidable.

Proof

Assume (towards a contradiction) that T is decidable by TM M_T.
Goal: use M_T to build a machine which will decide A_{TM}.

Define M_{ATM} = "On input <M,w>:

1. Using the parameters M and w, construct a different TM X such that if M accepts w, then <X> is in T; if M does not accept w, then <X> is not in T.
2. Run M_T on <X> and accept if accepts, reject if rejects.

Claim: M_{ATM} is decider and $L(M_{ATM}) = A_{TM}$, Then A_{TM} is decidable, contradicting the known fact that A_{TM} is undecidable.

Goal: reduce $HALT_{TM}$ to T

Given Genie/Decider for T

Want to solve $HALT_{TM}$ *

"On input <M, w>

1. Build X = "On input x

   1. If x = 101, accept.

   2. if x ≠ 101, Run M on w

      if M on w halts & accepts, accept

      if M on w halts & rejects, accept"

2. Ask G about <X>

   if yes, reject; if no, accept"

if <M, w> $\in HALT_{TM}$ then $L(X) = \Sigma^*$

if <M, w> $\notin HALT_{TM}$ then $L(X) = \{101\}$

Undecidability via reduction

Theorem: Problem T is undecidable.

Proof Common pattern for many of these proofs.

Assume (towards a contradiction) that T is decidable by TM M...

Goal: use M_T to b...

Define M_{ATM} = "O...

1. Using the par... such that if M... accept w, the...
2. Run M_T on <X...

Claim: M_{ATM} is de... decidable, contra...

In reduction proofs,

A. We always need to build a new TM X.
B. The auxiliary machine X must be run as part of our algorithm.
C. The auxiliary machine X runs only on w.
D. None of the above.
E. I don't know.

Countable and uncountable

Countable

- Find bijection with N
- Find a countable superset

Examples

Any language over Σ

Set of all regular languages

Set of rational numbers

Set of integers

Uncountable

- Diagonalization
- Find an uncountable subset

Examples

Set of all subsets of Σ*

Set of infinite binary sequences

Set of real numbers

[0,1]

Closure properties

high level descriptions

	Regular Languages	CFL	Decidable Languages	Recognizable Languages
Union NFA	✔ DFA	✔	✔	✔
Intersection	✔	X	✔	✔
Complement	✔	X	✔	X
Star NFA	✔	✔ CFG	✔	✔
Concatenation NFA	✔	✔	✔	✔

Proving closure

Goal: "The class of ____ languages is closed under ______"

In other words Given a language in specific class, is the result of applying the operation ______ to this language still guaranteed to be in the class?

Proving closure

Given: What does it mean for L to be in class?

e.g. L a regular language, so given a DFA $M_L = (Q_L, \Sigma_L, \delta_L, q_L, F_L)$

with $L(M_L) = L$. Name each of the pieces!

WTS: The result of applying the operation to L is still in this class.

Construction: Build a machine that recognizes the result of applying the operation to L. Start with description in English!

e.g. Let $M = (Q, \Sigma, \delta, q_0, F)$ where Q=… Σ=… δ=… q_0=…F=..

M could be DFA or NFA

Correctness: Prove L(M) = result of applying operation to L

WTS1 if w is in set then w is accepted by M

WTS2 if w is not in the set then w rejected by M.

Claim: The class of recognizable languages is closed under concatenation

Given

WTS

Construction

Correctness

Claim: The class of recognizable languages is closed under concatenation

Given Two recognizable languages A,B and TMs that recognize them: M_A with $L(M_A) = A$ and M_B with $L(M_B) = B$.

WTS The language AB is recognizable.

Construction Define the TM M as "On input w,

1. Nondeterministically split w into w = xy.
2. Simulate running M_A on x. If rejects, reject; if accepts go to 3.
3. Simulate running M_B on y. If rejects, reject; if accepts, accept."

Correctness

Construction Define the TM M as "On input w,

1. Nondeterministically split w into w = xy.
2. Simulate running M_A on x. If rejects, reject; if accepts go to 3.
3. Simulate running M_B on y. If rejects, reject; if accepts, accept.

Correctness Claim that w is in AB iff w is in L(M).

Part 1: Assume w is in AB. Then there are strings x,y such that w = xy and x is in A, y is in B. Running M on w, one of the nondeterministic ways we split w will be into these x,y. In step 2, the computation of M_A on x will halt and accept (because $L(M_A) = A$) so we go to step 3. In that step, the computation of M_B on y will halt and accept (because $L(M_B) = B$ so M accepts w.

Construction Define the TM M as "On input w,

1. Nondeterministically split w into w = xy.
2. Simulate running M_A on x. If rejects, reject; if accepts go to 3.
3. Simulate running M_B on y. If rejects, reject; if accepts, accept.

Correctness Claim that w is in AB iff w is in L(M).

Part 2: Assume w is not in AB. Then there are no strings x,y such that w = xy and x is in A, y is in B. In other words, for each way of splitting w into xy, at least one of the following is true: M_A running on x will reject or loop, M_B running on y will reject or loop. Tracing the computation of M on w, in each one of the nondeterministic computation paths, there is some split w=xy. For each of these splits, in step 2, the computation of M_A on x either loops (in which case M loops on w, so w is not in L(M)) or rejects (in which case M rejects w) or accepts (in which case M goes to step 3). If the computation of M enters step 3, this means that x is in $L(M_A)$ so by our assumption, y is not in $L(M_B)$ so M_B on y must either loop or reject. In either case, M rejects. Thus w is not in L(M).

Proving closure

Given: What does it mean for L to be in class?

e.g. L a regular language, so given a DFA $M_L = (Q_L, \Sigma_L, \delta_L, q_L, F_L)$

with L

WTS: The r

Construction: Build a

the op

e.g. L

M cou

Correctness: Prove L

WTS1

WTS2 if w is not in the set then w rejected by M.

To prove the class of recognizable languages is closed under ______ the constructions may involve building a

A. Two-tape Turing machine.
B. Nondeterministic decider.
C. Enumerator.
D. All of the above.
E. I don't know.

Claim: The class of decidable languages is closed under reversal

Given

WTS

Construction

Correctness

Claim: The class of decidable languages is closed under reversal

Given A decidable language L, with a decider TM D: L(D)=L

WTS There is a decider that decides L^R = {w | w^R is in L}

Construction Define the TM M as "On input w:

1. Make a copy of w in reverse.
2. Simulate running D on this copy.
3. If D accepts, accept. If D rejects, reject.

Correctness If w is in L^R then in step 1, M builds w^R and in step 2, the computation of D on w^R will accept (because L(D) = L), so in step 3, M accepts w. If w is not in L^R then in step 1, M builds w^R and in step 2, the computation of D on w^R will rejept (because L(D) = L), so in step 3, M rejects w.

Claim: The class of decidable languages is closed under reversal

Given A decidable language L, with a decider TM D: L(D)=L

WTS There is a decider that decides $L^R = \{w \mid w^R \text{ is in } L\}$

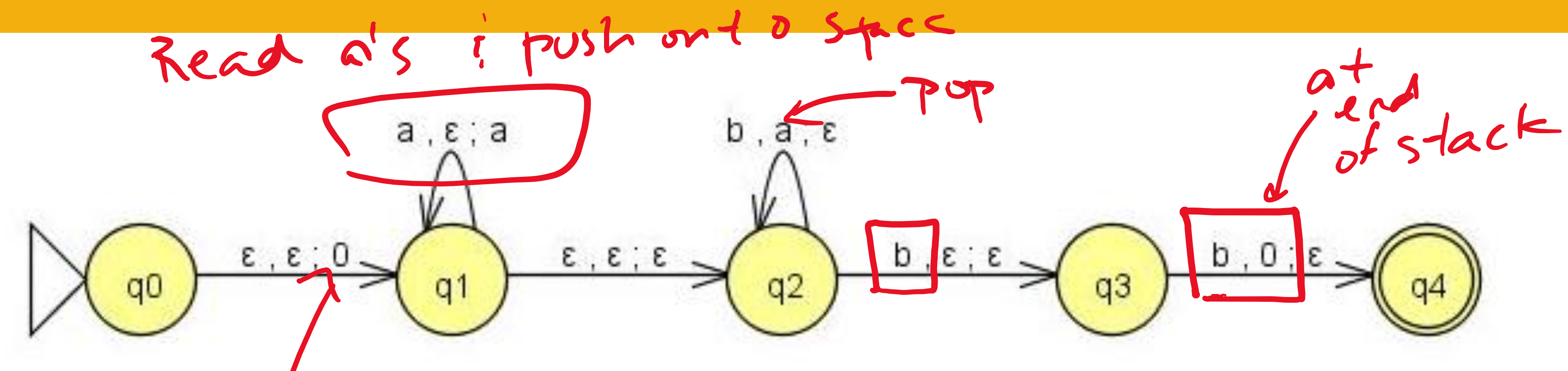
Construction Def

1. Make a copy o
2. Simulate runni
3. If D accepts, ac

Correctness If w i ,
the computation of
step 3, M accepts
and in step 2, the c)
= L), so in step 3, I

Is this how we proved that the class of regular languages is closed under reversal?

A. Yes.
B. No – but we could modify our earlier proof to make a copy of w^R and then run the DFA on it.
C. No – and this strategy won't work for automata.
D. I don't know.



end of stack symbol

What is the language of this PDA?

A. {w | # of b's in w ≥ # of a's in w}
B. {w | w = $a^n b^{n+1}$ for some n≥0}
C. {w | w = $a^n b^{n+2}$ for some n≥0}
D. {w | w = $a^n b^{2n}$ for some n≥0}
E. {w | w = $0a^n b^{2n}0$ for some n≥0}

What is the language of CFG

with rules

$$S \rightarrow aSb \mid bY \mid Ya$$

$$Y \rightarrow bY \mid Ya \mid \varepsilon$$

(Using) Pumping Lemma

Theorem: L = {w w^R | w is in {0,1}* } is not regular.

Proof (by contradiction): Assume, towards a contradiction, that L is regular. Then by the Pumping Lemma, there is a pumping length, p, for L. Choose s to be the string __________. The Pumping Lemma guarantees that s can be divided into parts s=xyz such that |xy| ≤p, |y|>0, and for any i≥0, xy^iz is in L. But, if we let i=_____, we get the string _______ which is not in L, a contradiction. Thus L is not regular.

(Using) Pu

Theorem: L = {w

Proof (by contra

that L is regular.

pumping length,

_________. The Pumping Lemma guarantees that s can be divided into parts s=xyz such that |xy| ≤p, |y|>0, and for any i≥0, xy^iz is in L. But, if we let i=_____, we get the string _______ which is not in L, a contradiction. Thus L is not regular.

A. s = 000000111111, i=6

B. s=0^p0^p, i=2

C. s=0^p110^p, i=2

D. More than one of the above.

E. I don't know.

P and NP

P: Languages decidable in polynomial time on deterministic Turing machines.

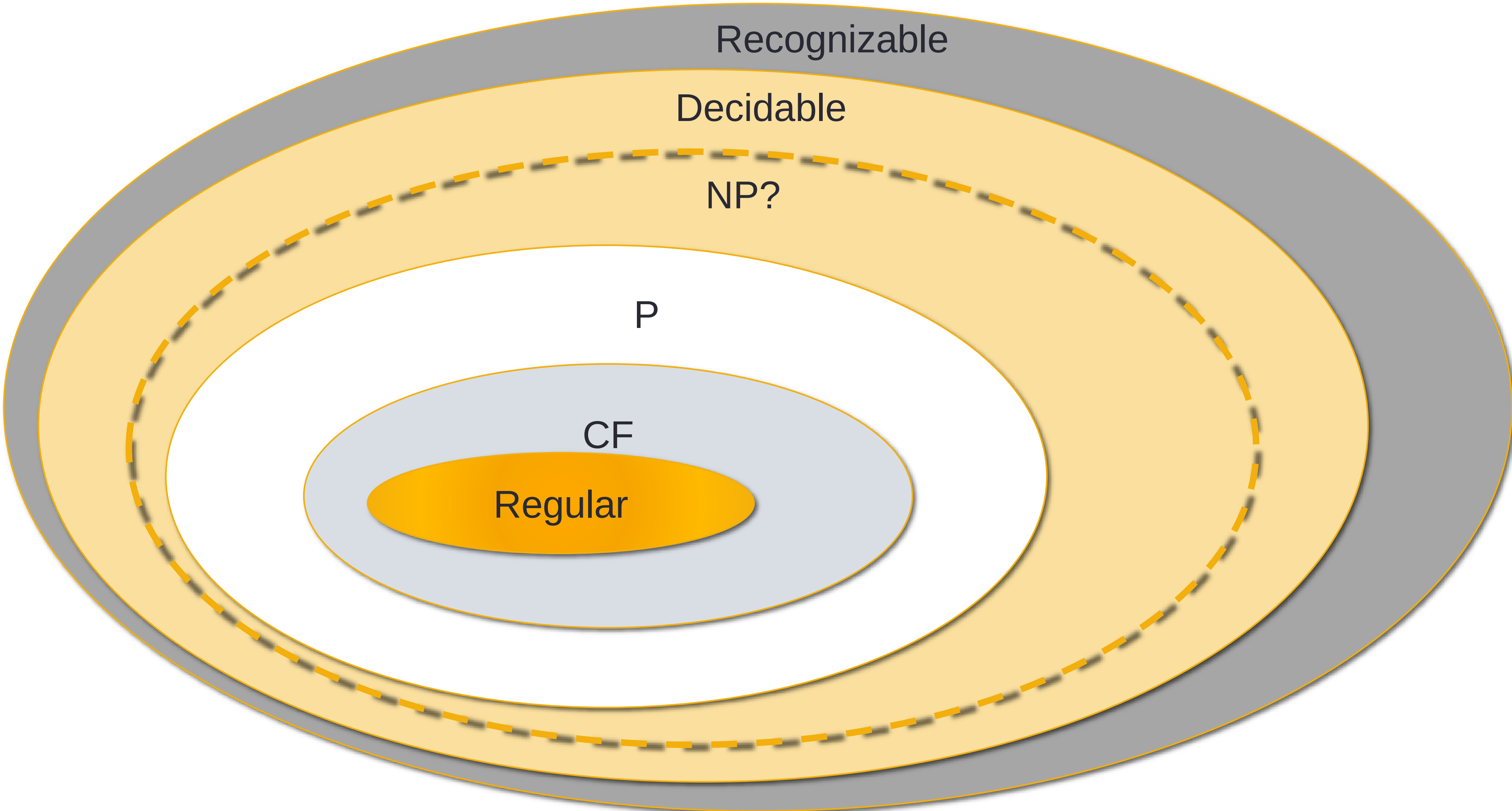
e.g. PATH, Simple arithmetic, CFL's, etc.

NP: Languages decidable in polynomial time on nondeterministic Turing machines.

e.g. TSP, SAT, CLIQUE, etc.

Know P$\subseteq$NP and

if an NP-complete problem is in P, then P=NP.


Recognizable
Decidable
NP?
P
CF
Regular


Im POSSIBLe