COT 6410 Spring2020 **Sample Final Exam**

1. Let set **A** be **infinite recursive**, **B** be **re non-recursive** and **C** be **non-re**. Using the terminology **(REC) recursive**, **(RE) re, non-recursive**, **(NR) non-re (possibly co-re)**, categorize each set by dealing with the cases I present, saying whether or not the set can be of the given category and briefly, but convincingly, justifying each answer (BE COMPLETE). You may assume sets like $\aleph$ are infinite REC; **K** and $\mathbf{K_0}$ are RE; and **TOTAL** is non-re. You may also assume, for any set **S**, the existence of comparably hard sets
$S_E = \{2x | x \in S\}$ and $S_D = \{2x+1 | x \in S\}$.

   a.) $\mathbf{A + B = \{ x \mid x = y + z, \text{ for some } y \in A \text{ and some } z \in B \}}$

   REC: **$A = \aleph$, $B = K_0$, $A+B = \{ x \mid x \geq \min y \in K_0 \}$.**

   This is the complement of a finite set and is hence decidable as the finite set is.

   b.) $\mathbf{A \cap C = \{ x \mid x \in A \text{ and } x \in C\}}$

   RE: **$A = E = \{2x \mid x \in \aleph\}$, $C = TOTAL_D \cup K_E$.**

   $A \cap C = K_E$ which is RE.

2. Choosing from among **(REC) recursive**, **(RE) re non-recursive, (coRE) co-re non-recursive**, **(NRNC) non-re/non-co-re**, categorize each of the sets in a) through d). Justify your answer by showing some minimal quantification of some known recursive predicate.

   a) $\mathbf{A = \{ <f,g> \mid \exists x\ \varphi_f(x)\downarrow \text{ and } \varphi_g(x)\downarrow \text{ and } \varphi_g(x) = \varphi_f(x) \}}$.

   $\exists$<x,t > [STP(f,x,t) & STP(g,x,t) & Value(f,x,t) = Value(g,x,t)] ______ *RE*

   b.) $\mathbf{B = \{ f \mid \text{range}(\varphi_f) \text{ is empty} \}}$

   $\forall$<x,t > [~STP(f,x,t)] ______ *co-RE*

   c.) $\mathbf{C = \{ <f,x> \mid \varphi_f(x)\downarrow}$ **but takes at least 10 steps to do so }**

   $\exists$t [STP(f,x,t) & ~STP(f,x,9)] ______ *RE*

   d.) $\mathbf{D = \{ f \mid \varphi_f}$ **diverges for some value of x }**

   $\exists$x $\forall$t [~STP(f,x,t)] ______ *NRNC*

3. Looking back at Question 1, which of these are candidates for using Rice's Theorem to show their unsolvability? Check all for which Rice Theorem might apply.

   a) ____√ **b)** ____√ **c)** ____ **d)** ____√

4. Let **S** be an arbitrary semi-decidable set. By definition, **S** is the domain of some partial recursive function $\mathbf{g_S}$. Using $\mathbf{g_S}$, constructively show that **S** is the range of some partial recursive function, $\mathbf{f_S}$. No proof is required; just the construction is needed here.

   $f_S(x) = x * \exists t\ [\ STP(x, g_s, t)\ ]$ or

   $f_S(x) = x * (g_s(x) - g_s(x) + 1)$

5. Using the definition that **S** is recursively enumerable iff **S** is the range of some effective procedure $\mathbf{f_S}$ (partial recursive function), prove that if both **S** and its complement **~S** are recursively enumerable (using enumerating effective procedures $\mathbf{f_S}$ and $\mathbf{f_{\sim S}}$) then **S** is decidable. To get full credit, you must show the characteristic function for **S**, χ_S, in all cases. Also, be sure to discuss why your χ_S works.
Look at end of this sample exam for alternative, simpler problems (this one is too hard).

Define steps = $\mu\langle y,t\rangle[(STP(f_S,y,t)\ \&\ (VALUE(f_S,y,t)=x))$ ***or***
$(STP(f_{\sim S},y,t)\ \&\ (VALUE(f_{\sim S},y,t)=x))\ ])$

Define $\chi_S(x) = STP(f_S,\langle steps\rangle_1,\langle steps\rangle_2)\ \&\ (VALUE(f_S,\langle steps\rangle_1,\langle steps\rangle_2) = x$

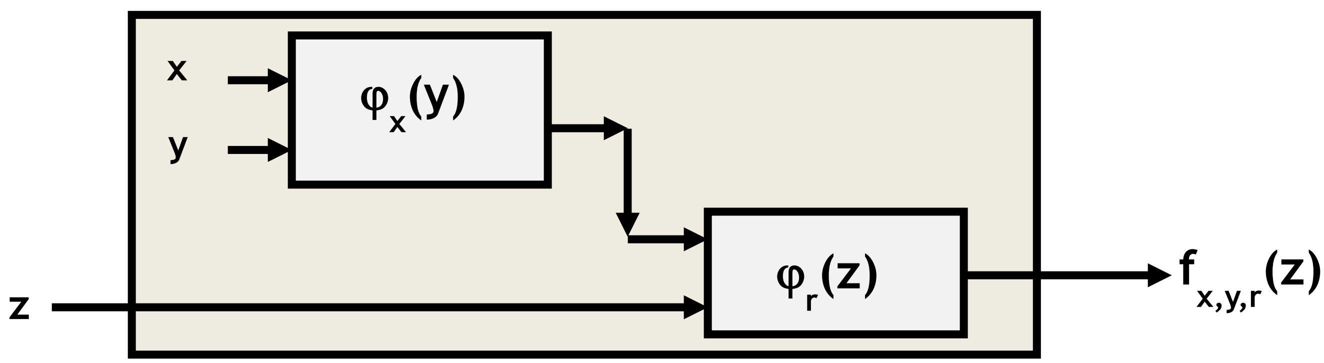
If $x \in S$ ***then*** $\exists y\ f_S(y) = x$ ***and so*** $\exists\langle y,t\rangle[(STP(f_S,y,t)\&(VALUE(f_S,y,t)=x))]$ ***and so*** $\chi_S(x) = 1$ ***(true)***

If $x \notin S$ ***then*** $\exists y\ f_{\sim S}(y)=x$ ***and so*** $\exists\langle y,t\rangle[\ (STP(f_{\sim S},y,t)\ \&\ (VALUE(f_{\sim S},y,t)=x))\ \&$
$(\sim STP(f_S,y,t)$ ***or*** $(VALUE(f_S,y,t)\neq x))\ ]]$ ***and so*** $\chi_S(x)=0$ ***(false)***

Because of this $\chi_S(x)$ ***always converges and produces 1 (true) iff*** $x \in S$

Thus, $\chi_S(x)$ ***meets our requirements.***

6. Rice's Theorem deals with attributes of certain types of problems **P** about partial recursive functions and their corresponding sets of indices $\mathbf{S_P}$. The following image describing a function $\mathbf{f_{x,y,r}}$ is central to understanding Rice's Theorem.



Explain the meaning of this by indicating:

a.) What assumption do we make about what kind of functions are not in P?
We assume no function with empty domain has property P.

b.) What is **r**, how is it chosen and how can we guarantee its existence?
r is the index of some function with property P. One must exist since P is non-trivial.

c.) Using recursive function notations, write down precisely what $\mathbf{f_{x,y,r}}$ computes for the Strong Form of Rice's Theorem.
$f_{x,y,r}(z) = \varphi_x(y) - \varphi_x(y) + \varphi_r(z)$

How does this function $\mathbf{f_{x,y,r}}$ behave with respect to **x**,**y** and **r**, and how does that relate to the original problem, **P**, and set, $\mathbf{S_P}$?

If $\varphi_x(y)\downarrow$ **then** $f_{x,y,r}(z) = \varphi_r(z)\ \forall z$ **and** $f_{x,y,r} \in S_P$.

If $\varphi_x(y)\uparrow$ **then** $f_{x,y,r}(z)\uparrow\ \forall z$ **and** $f_{x,y,r} \notin S_P$.
Thus, we could decide the halting problem if we could decide membership in $\mathbf{S_P}$**, so P is an undecidable problem.**

7. Define **NAT = { f | range(f) = ℵ }**. That is, **f ∈NAT** iff **f**'s range includes every natural number.

a.) Show some minimal quantification of some known recursive predicate that provides an upper bound for the complexity of **NAT**.

∀k ∃<x, t> [STP(f,x,t) && (Value(f,x,t) == k)]

b.) Use Rice's Theorem to prove that **NAT** is undecidable.

First, NAT is non-trivial as the identity, I(x)=x, is in NAT and the Constant Zero, Z(x)=0, is not.

Second, let f and g be arbitrary indices of arbitrary effective procedures, such that range(φ_f) = range(φ_g).

f is in NAT iff range(φ_f) = ℵ iff range(φ_g) = ℵ if g is in NAT

This means NAT satisfies both properties of the weak form of Rice's Theorem associated with ranges and is therefore undecidable.

c.) Show that **TOT** $\leq_m$ **NAT**, where **TOT = { f | ∀x φ_f(x)↓ }**.

Let f be arbitrary. Define an algorithmic mapping G_f from indices to indices as G_f (x) = f(x)-f(x)+x.

Now, G_f (x) = I(x) (the Identity function) iff f ∈ TOTAL and ∃x x ∉ range(G_f) iff f ∉ TOTAL This will be any x where φ_f(x) ↑.
Thus, f is in TOT iff G_f is in NAT. Thus, TOTAL $\leq_m$ NAT.

8. Why does Rice's Theorem have nothing to say about the following? Explain by showing some condition of Rice's Theorem that is not met by the stated property.
AT-LEAST-LINEAR = { f | ∀y φ_f(y) converges in no fewer than y steps }.
We can deny the 2nd condition of Rice's Theorem since
Z, where Z(x) = 0, implemented by the TM R converges in one step no matter what x is and hence is not in AT-LEAST-LINEAR
Z', defined by the TM $\mathscr{L}\mathscr{R}$ R, is in AT-LEAST-LINEAR
However, ∀x [Z(x) = Z'(x)], so they have the same I/O behavior and yet one is in and the other is out of AT-LEAST-LINEAR, denying the 2nd condition of Rice's Theorem

9. Consider the following set of independent tasks with associated task times:
(T1,4), (T2,5), (T3,2), (T4,7), (T5,1), (T6,4), (T7,8)
Fill in the schedules for these tasks under the associated strategies below.

Greedy using the list order above:

T1	T1	T1	T1	T3	T3	T5	T6	T6	T6	T6	T7	T7	T7	T7	T7	T7	T7	T7			
T2	T2	T2	T2	T2	T4	T4	T4	T4	T4	T4	T4										

Greedy using a reordering of the list so that longest running tasks appear earliest in the list:

T7	T7	T7	T7	T7	T7	T7	T7	T1	T1	T1	T1	T6	T6	T6	T6						
T4	T4	T4	T4	T4	T4	T4	T2	T2	T2	T2	T2	T3	T3	T5							

10. We described the proof that **3SAT** is polynomial reducible to **Subset-Sum**. You must repeat that.

a.) Assuming a **3SAT** expression **(a + a + ~b) (~a + b + c)**, fill in all omitted values (zeroes elements can be left as omitted) of the reduction from **3SAT** to **Subset-Sum**.

	a	b	c	a + a + ~b	~a + b + c
a	1			1 or 2	
~a	1				1
b		1			1
~b		1		1	
c			1		1
~c			1		
C1				1	
C1’				1	
C2					1
C2’					1
	1	**1**	**1**	**3**	**3**

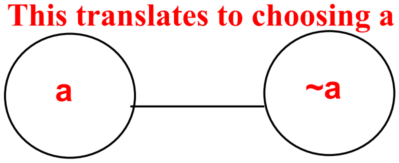
b.) List some subset of the numbers above (each associated with a row) that sums to **1 1 1 3 3**. Indicate what the related truth values are for **a**, **b** and **c**.

a = T ; b = T ; c = T

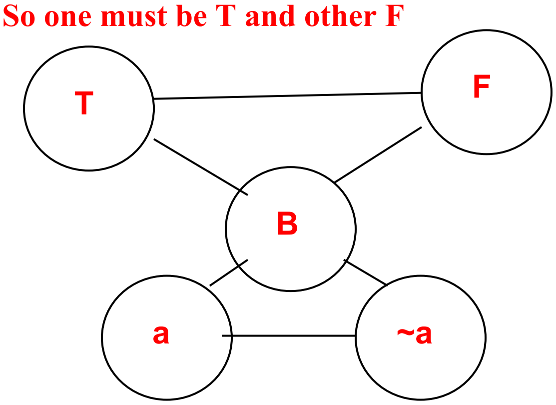
1 0 0 1 0 or 1 0 0 2 0
0 1 0 0 1 0 1 0 0 1
0 0 1 0 1 0 0 1 0 1
0 0 0 1 0 0 0 0 1 0
0 0 0 1 0 0 0 0 0 1
0 0 0 0 1

11. Present a gadget used in the reduction of **3-SAT** to some graph theoretic problem where the gadget guarantees that each variable is assigned either **True** or **False**, but not both. Of course, you must tell me what graph theoretic problem is being shown **NP-Complete** and you must explain why the gadget works.

Vertex Cover
Must Cover each Edge
Set goal to min vertices
Must choose one but not both are needed
This translates to choosing a or ~a



3-Color
Cannot choose B for either a or ~a
So one must be T and other F



12. Let **Q** be some problem (an optimization or decision problem). Assuming **≤ p** means many-one reducible in polynomial time and **≤ tp** means Turing-reducible in polynomial time, categorize **Q** as being in one of **P**, **NP**, **co-NP**, **NP-Complete**, **NP-Easy**, **NP-Hard**, or **NP-Equivalent** (see first two pages for definitions of each of these concepts). For each case, choose the most precise category. I filled in one answer already.

Description of Q	**Category**
Q is decidable in deterministic polynomial time	**P**
For some **R** in **NP**, **Q ≤ tp R**	**NP-Easy**
Q is both **NP-Easy** and **NP-Hard**	**NP-Equivalent**
Q is in **NP** and if **R** is in **NP** then **R ≤ p Q**	**NP-Complete**
A solution to **Q** is verifiable in deterministic polynomial time	**NP**
Q's complement is in **NP**	**Co-NP**

13. A graph **G** is **k-Colorable** if its vertices can be colored using just **k** (or fewer colors) such that adjacent vertices have different colors. The **Chromatic Number** of a graph **G** is the smallest number **k** for which **G** is **k-Colorable**. **k-Colorable** is a decision problem that has parameters (**G**, **k**), whereas the **Chromatic Number** problem is a function with a single parameter **G**. In all cases, assume **G** has **n** vertices.

a.) Show that **k-Colorable ≤ tp Chromatic Number** (**≤ tp** means Turing reducible in polynomial time).

G is k-Colorable iff its Chromatic Number is some j ≤ k

This can be checked by just one invocation of the Oracle for Chromatic Number

b.) Show that **Chromatic Number ≤ tp k-Colorable** (**≤ tp** means Turing reducible in polynomial time).

G's Chromatic Number is no worse than n, the number of vertices. Doing a binary search, we can make at most $\log_2 n$ calls to the oracle for k-Colorable to determine the smallest number for which G is k-colorable

14. Partition refers to the decision problem as to whether some set of positive integers **S** can be partitioned into two disjoint subsets whose elements have equal sums. **Subset-Sum** refers to the decision problem as to whether there is a subset of some set of positive integers **S** that precisely sums to some goal number **G**.

a.) Show that **Partition ≤p Subset-Sum**.

Look at notes

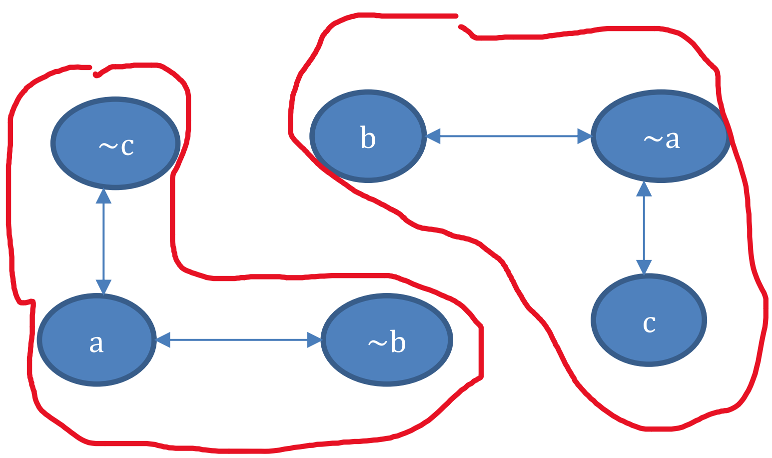
b.) Show that **Subset-Sum ≤p Partition**.

Look at notes

15. QSAT is the decision problem to determine if an arbitrary fully quantified Boolean expression is true. Note: **SAT** only uses existential, whereas **QSAT** can have universal qualifiers as well so it includes checking for Tautologies as well as testing Satisfiability. What can you say about the complexity of **QSAT** (is it in **P**, **NP**, **NP-Complete**, **NP-Hard**)? Justify your conclusion.

QSAT is NP-Hard. This is so since SAT trivially reduces to QSAT (it is a subproblem of QSAT). Since SAT is known to be NP-Complete then some NP-Complete problem polynomially reduces to QSAT. This makes QSAT NP-Hard. As we cannot (at least not yet) show QSAT is in NP, then NP-Hard is the best we can do.

16. Given the following instance of 2SAT, E=(a ∨ b) ∧ (¬a ∨ ¬b) ∧ (¬a ∨ ¬c) ∧ (a ∨ c), display the associated implication graph, show its strongly connected components and then show how this leads to an assignment of variables that satisfies E.

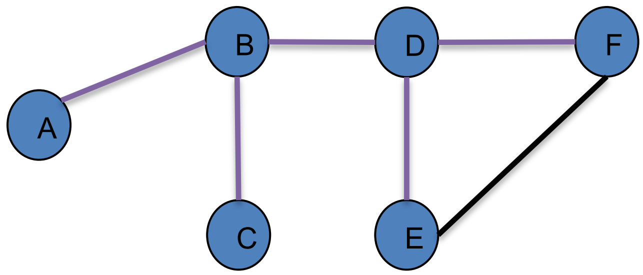


a = T, b = F, c = F or a=F, b=T, c=T

17. Specify True (**T**) or False (**F**) for each statement.

Statement	**T or F**
Every Regular Language is also a Context Free Language	***T***
Phrase Structured Languages are the same as RE Languages	***T***
The Context Free Languages are closed under Complement	***F***
A language is recursive iff it and its complement are re	***T***
PCP is undecidable even for one letter systems	***F***
Membership in Context Sensitive Languages is undecidable	***F***
Every RE language is Turing reducible to its complement	***T***
Emptiness is undecidable for Context Sensitive Languages	***T***
The complement of a trace language is Context Free	***T***
The word problem for two-letter Semi-Thue Systems is decidable	***F***

18. Consider the following graph. We wish to show it has vertex cover solution of 3. Our approach is to reduce this to the 2SAT related problem of determine if we can satisfy some associated positive 2SAT expression, S, so that the minimum solution for S involves at most 3 variables being set to true? What is that corresponding positive 2SAT expression S? What is a minimal positive solution for the expression S and the vertex cover solution for the graph with which we started?



S = (A∨B),(B∨D),(D∨E),(D∨F),(E∨F)
Solutions are same for S and graph. They are B, D, E or B, E, F or B, D, F.
For S the choice of a variable means it is set to true.

19. Let L be an arbitrary CFL. Show that $\mathbf{L = L^2}$ is undecidable by reducing $\mathbf{L = \Sigma^*}$ to $\mathbf{L = L^2}$.

Claim is that $L = \Sigma^*$ iff

(1) $\Sigma \cup \{\lambda\} \subseteq L$; and

(2) $L = L^2$

Clearly, if $L = \Sigma^*$ then (1) and (2) trivially hold.

Conversely, we have $\Sigma^* \subseteq L^* = \cup_{n\geq 0} L^n \subseteq L$

first inclusion follows from (1); second from (2)

20. Let $\mathbf{P} = \langle\langle x_1,x_2,\ldots,x_n\rangle, \langle y_1,y_2,\ldots,y_n\rangle\rangle$, $x_i,y_1 \in \Sigma^+$, $\mathbf{1 \leq i \leq n}$, be an arbitrary instance of **PCP**. We can use **PCP's undecidability** to show the undecidability of the problem to determine if a **Context Sensitive Grammar** generates a non-empty language. I will start the grammar, **G**. You must complete it so it maps an instance of PCP to the non-emptiness problem for this $\mathcal{L}$**(G)**.

Define **G = ({S, T} ∪ Σ, {*}, S, R)**, where **R** is the set of rules:

$\mathbf{S \rightarrow x_i\, S\, y_i^R \mid x_i\, T\, y_i^R}$ $\quad \mathbf{1 \leq i \leq n}$ **(Note: the superscripted R means Reversed)**

// Write the rules for the rest of this CSG.

$$
\begin{array}{lll}
a\,T\,a & \rightarrow \; *\,T\,* & \forall a \in \Sigma \\
\,a & \rightarrow \; a\, & \forall a \in \Sigma \\
a\,* & \rightarrow \; *\,a & \forall a \in \Sigma \\
T & \rightarrow \; * &
\end{array}
$$

21. Let set **A** be a **non-empty Regular Language**, and let **B** be a **non-Regular Context-Free Language**. For each of **(REG) Regular**, and **(CFL) non-Regular Context-Free**, specify if the language **C** can or cannot be of the given language type and prove your assertions. As always, assume **A**, **B** and **C** are over some finite alphabet, **Σ**.

C = B/A = { x | w = xy, where w ∈ B and y ∈ A }

Note: / is called **Quotient** and was extensively discussed in Class.
You may use any well-known Regular and Context-Free Languages. E.g., every language described by a Regular Expression is **Regular** and the set **{ a^nb^n | n>0 }** is a **CFL**.

REG: (**Big Hint: C** can be **Regular** so show **A** and **B** where **B/A** is **Regular**.
Explicitly describe languages **A**, **B** and **C** and you are done).

B = { a^n b^n | n>0 } ***A = {c}*** ***C = ∅, which is clearly regular***

CFL: (**Big Hint: C** can be a **CFL**, so show **A** and **B** where **B/A** is a **CFL**.
Explicitly describe languages **A**, **B** and **C** and you are done).

B = { a^n b^n | n>0 } ***A = {λ}*** ***C = { a^n b^n | n>0 }, which is clearly a non-regular CFL***

Alternative 1 to #5

5. Using the definition that **S** is a recursively enumerable, non-empty set iff **S** is the range of some algorithm $\mathbf{f_S}$, prove that if both **S** and its complement **~S** are recursively enumerable (using enumerating algorithms $\mathbf{f_S}$ and $\mathbf{f_{\sim S}}$) then **S** is decidable. To get full credit, you must show the characteristic function for **S**, χ_S, in all cases. Also, be sure to discuss why your χ_S works

Define $\chi_S(x) = f_S(\ \mu y\ [\ f_S(y)=x \text{ or } f_{\sim S}(y)=x]\ ) == x$

As f_S and $f_{\sim S}$ are both algorithms, each converges on all y. Since S and ~S are mutually exclusive and their union is all natural numbers, one and only one of them produces x for some input y. If that one is f_S then $\chi_S(x)$ = 1 (true); else $\chi_S(x)$ = 0 (false)

Thus, $\chi_S(x)$ meets our requirements.

Alternative 2 to #5

5. Using the definition that **S** is a recursively enumerable, non-empty set iff **S** is the domain of some effective procedure $\mathbf{f_S}$, prove that if both **S** and its complement **~S** are recursively enumerable (using the domains of procedures $\mathbf{f_S}$ and $\mathbf{f_{\sim S}}$) then **S** is decidable. To get full credit, you must show the characteristic function for **S**, χ_S, in all cases. Also, be sure to discuss why your χ_S works

Define $\chi_S(x) = STP(f_S, x, \mu\ t\ [(STP(f_S,x,t) \text{ or } (STP(f_{\sim S},x,t)])$

Since S and ~S are mutually exclusive and their union is all natural numbers, $\mu\ t\ [(STP(f_S,x,t)$ or $(STP(f_{\sim S},x,t)]$ will converge for some value of t. As one and only one of $f_S(x)$ and $f_{\sim S}(x)$ converges then just one of these STP functions ever returns true. If that one is for f_S then $f_S(x)\downarrow$ and $\chi_S(x)$ = 1 (true); else $f_{\sim S}(x)\downarrow$ and $\chi_S(x)$ = 0 (false)

Thus, $\chi_S(x)$ meets our requirements.