

EXAM 2: TAKEHOME – SOLUTIONS

PROBLEM 1

(a). Let C_1 be the collection of linear orders. Is C_1 an Δ -elementary class?

Proof. Here we just have to show that there is a collection of sentences such that this collection's models are precisely the set of linear orders. But this is trivial, as we defined linear orders as those things of which the following are true: (1) "for all x , not $x < x$ ", (2) "for all x, y, z , $x < y$ and $y < z$ implies $x < z$ ", (3) "for all x, y , either $x < y$, $y < x$, or $x = y$ ". But that just means that we have defined linear order to be the models of sentences (1) to (3). \square

(b). Let C_2 be the collection of finite linear orders. Is C_2 a Δ -elementary class?

Proof. Here the answer is no, by the compactness theorem. This proof was in fact sketched in class. Suppose for a contradiction that T is true of the finite linear orders, and only true of the finite linear orders. Now expand the language by new constants $\{c_i : i \in \mathbb{N}\}$, and expand T to $T' := T \cup \{c_i < c_j : i < j\}$. Now take finite subset of T_F of T' . T_F will only mention a finite number of c_i . Say the biggest i is n . Now take a linear order of length n , and name each element of this linear order by the constants c_1 to c_n . Clearly, this is a model of T_F , and thus there is a model of T' as well, and any model of T' is an infinite linear order that satisfies every sentence in T , a contradiction. \square

(c). Let C_3 be the collection of well-orders. (A well-order is a linear order with no infinite descending chain of elements. For example $(\mathbb{N}, <)$ but not $(\mathbb{Q}, <)$ or $(\mathbb{Z}, <)$.) Is C_3 a Δ -elementary class?

Proof. Again the answer is no, and again the proof is by the Compactness Theorem. For a contradiction, assume that there is some T such that the models of T are precisely the well-orders. Now expand the language with new constants c_i for $i \in \mathbb{Z}$. Expand T to T' by adding the sentences that $c_i < c_j$ for $i < j$. Any finite subset of T' will only mention finitely many constants. That such a finite subset is consistent can be witnessed letting those constants name a (finite) descending chain in \mathbb{N} . But it is equally clear that any model of T' has an infinite descending chain, for instance those named by the c_i such that $i < 0$. \square

(d). Let L be any language and let \mathfrak{M} be an infinite model in that language. Is the set of models isomorphic to \mathfrak{M} a Δ -elementary class.

Proof. I only asked the question for \mathfrak{M} infinite, but one might also ask the question for \mathfrak{M} finite. I'll give a proof for both cases.

If \mathfrak{M} is infinite, this is just an application of the Upward Löwenheim Skolem Theorem, which says that any theory with infinite models has models of arbitrarily large cardinality. In particular, any such theory will have models larger than \mathfrak{M} and hence not isomorphic to \mathfrak{M} .

On the other hand, if \mathfrak{M} is finite, then the set of models isomorphic to \mathfrak{M} is Δ -elementary. Suppose that the universe of \mathfrak{M} is $\{m_1, \dots, m_n\}$. The intuition is that since the model is finite, we can write down everything that happens in the model.

Consider the single sentence that begins “there exists x_1, \dots, x_n ” and then continues to say (1) that a k -ary relation, R_j holds of x_{i_1}, \dots, x_{i_k} whenever R_j holds of m_{i_1}, \dots, m_{i_k} , (2) that x_i equals the constant c_j whenever $m_i = c_j$, (3) that $f_j(x_{i_1}, \dots, x_{i_k}) = x_{i_0}$ whenever $f(m_{i_1}, \dots, m_{i_k}) = m_{i_0}$, and finally (4) that there are only n elements total.

Now suppose that one has another model \mathfrak{N} that also satisfies the sentence described above. Thus there is some way of assigning elements of N to $x_1 \dots x_n$ in some fashion that makes the sentence true. Now consider the map that sends m_i to the element in N corresponding to x_i . The things that one has to check to see that this map is an isomorphism are precisely the things that sentence in the previous paragraph says are true. □

PROBLEM 2

Let \mathfrak{K} be a Δ -elementary class. Let $\mathfrak{K}' \subseteq \mathfrak{K}$ be the class of models in \mathfrak{K} that are infinite. Show that \mathfrak{K}' is also Δ -elementary.

Proof. Let ϕ_n be the sentence that says “there exists x_1 to x_n with $x_i \neq x_j$ for $i \neq j$ ”. Since \mathfrak{K} is Δ -elementary, there is by definition some T such that \mathfrak{K} is precisely the class of models of T . Let $T' := T \cup \{\phi_n \mid n \in \mathbb{N}\}$. Then any infinite model in \mathfrak{K} satisfies T' , and none of the finite models do. Thus T' demonstrates that \mathfrak{K}' is Δ -elementary. □

PROBLEM 3

Fix a language, L , and a complete theory T in L . Let S be a collection of L formulas, all with free variable x . Say that S is consistent iff for all finite collection $\varphi_1(x), \dots, \varphi_n(x)$ taken from S one has that T proves $\exists x(\varphi_1(x) \wedge \dots \wedge \varphi_n(x))$. We define a *type* to be a consistent set of formulas with the same free variable.

(a). Let our language, L , be $\{+, \cdot, 0, 1, <\}$. Recall that if r is an element of \mathfrak{R} , a model elementarily equivalent to the real numbers, we have defined what it means for r to be infinitesimal. Show that “infinitesimally close to zero” can be expressed as a type. That is, show that there is a consistent set of formulas $p(x)$, each with free variable x , such that each formula in $p(x)$ is true of r whenever r is infinitesimal. Moreover, show that if r is an element of which every formula in the type is true then r is infinitesimal.

Proof. Let θ_n be the formula that says “ x is less than $\frac{1}{n}$ ”. Let θ_0 be the formula that says $x > 0$. Let $p = \{\theta_i \mid i \in \mathbb{N}\}$. First note that p is consistent. Take any finite set of formulas $\theta_{n_0}, \dots, \theta_{n_k}$ from p . Then these formulas will only mention finitely many rational numbers of the form $\frac{1}{n_i}$. Let r be some positive real number less than the smallest $\frac{1}{n_i}$ mentioned in the finite collection of formulas from S . Thus one sees that that $Th(\mathbb{R})$ proves that $\exists x(\theta_{n_0}(x) \wedge \dots \wedge \theta_{n_k}(x))$ since $\exists x(\theta_{n_0}(r) \wedge \dots \wedge \theta_{n_k}(r))$ holds.

Next take a model, \mathfrak{M} , elementarily equivalent to \mathbb{R} . Assume that some element, s , of \mathfrak{M} realizes the type p . Then s is less than $1/n$ for each n , and hence less than each rational. And s is greater than zero. Thus s is infinitesimal.

Now take an infinitesimal element. By definition, s is less than each positive rational number, and greater than zero. Thus it is easy to see that s satisfies every formula in p , and thus realizes p . □

(b). Let \mathfrak{M} be any model. Let $p(x)$ be a type. Show that the set of elements m in \mathfrak{M} of which $p(x)$ is true is fixed by automorphism.

Proof. Suppose that $p := \{\varphi_i\}$. Each φ_i defines a set D_i . Note that the set of realizations of p is just $\bigcap D_i$. Thus since we have proven that automorphisms fix each definable set, each D_i is fixed, and thus the intersection of the D_i is also fixed by any automorphism. □

PROBLEM 4

(a). Fix a register machine P . Consider the set of pairs x, y where P takes input x and outputs y . Is this set of pairs enumerable? Is it always decidable, never decidable, or does its decidability depend on P ?

Proof. Yes, it is enumerable, but its decidability depends on P .

First to prove that it is enumerable: Let P_0 operate in the following fashion: at stage n , P_0 generates the first n elements of A^* in lexicographic order, running P on each element for n steps, outputting the pair (x, y) whenever it finds an input x that causes P to output y . Then P_0 goes on to stage $n + 1$. Clearly, if P takes x and outputs y , P_0 will at some stage output (x, y) .

Now suppose P is the program that just consists of the command “Print y_0 ”. Then the set of pair (x, y) where x is an input and y is an output of P are all the pairs (x, y_0) for any $x \in A^*$, which is clearly decidable.

On the other hand, suppose P is the program that outputs “yes” when given as an input a program that halts. Suppose that P does not halt otherwise. Then the set under consideration is $\{(x, y) \mid x \in \Pi_{halt}, y = \text{“yes”}\}$. Suppose that P_1 decides this set. Then one could take a program, P' , run P_1 on the pair (P', yes) , and get an answer as to whether P' halts, a contradiction. □

(b). Fix a pair x, y . Let S be the set of programs P that output y when given input x . Is this set enumerable? Is it always decidable, never decidable, or does its decidability depend on the pair x, y ? (To be precise, I should ask this question not of S , but of $S' = \{w_P : P \in S\}$. That is, I should ask it about the Gödel numbers of the register machines not the register machines themselves, but I think that this is one of those cases where extra precision does not make the problem clearer.)

Proof. Again, this set is enumerable. Fix (x, y) . Let P_0 do the following: at stage n , P_0 generates the first n programs in lexicographical order, runs each one on x for n steps, and output the ones that give y as output.

Now we want to show that for any pair (x, y) , the set of programs P that take x as input and output y is not decidable. Suppose for a contradiction that P_1 decides this set. We will show how to decide the halting problem using P_1 . Take an arbitrary program, and remove all of the “PRINT” commands in the program, and insert a new command to PRINT y just before it halts. Now the new program

outputs y (on any input) iff the original program halted. Now, to tell if the original program halts, one can simply use P_1 to determine if the modified program outputs y . Since it is impossible to come up with a decision procedure for the halting problem, P_1 cannot exist. \square

PROBLEM 5

Let W be the set of P (or more precisely, the Gödel numbers for P) such that P gives the same output whenever it halts (no matter what the input was). Is W decidable? Is it enumerable?

Proof. W is neither decidable nor enumerable. We will prove this by showing that the complement of W is enumerable, but not decidable. Once we show this, we will be able to conclude that W is not enumerable, because if both W and W^C were enumerable, they would both be decidable.

First to show that W^C is enumerable. W^C is the set of programs for which there are inputs w_1 and w_2 that give different outputs. Let P_0 be as follows: at stage n , P_0 generates the first n programs, runs them for n steps, and outputs any program that gives two distinct outputs during those n steps. Then P_0 goes to stage $n + 1$.

Now to show that W^C is not decidable. Suppose that P_1 is a decision procedure for W^C . Again we will show how to use P_1 to decide $\Pi_{halt} := \{w_P | P : \square \rightarrow halt\}$. Take a program P . We want to modify P to get P' so that P' is in W^C iff P is in Π_{halt} . If we can do this, we will have the contradiction we are looking for, since one could then decide Π_{halt} by taking a program P modifying it to get P' and then running P_1 on P' .

We will modify P as follows: Pick some $w_0 \neq \square$ in A^* . P' will begin with the sequence of instructions that cause P' , (1) on input w_0 , to print w_0 , goto the end of the program, and halt, and (2) on any input other than w_0 or \square , P' goes into an infinite loop. After this P' runs exactly as P , but skipping any PRINT commands in P . Pick w_1 not equal to w_0 , and replace the HALT command in P with the pair of commands that first prints w_1 and then halts. Thus P' never prints anything or halts on any input other than w_0 or \square , always prints w_0 on input w_0 , and prints w_1 if and only if $P : \square \rightarrow halt$. Thus P' prints out two distinct outputs iff $P \in \Pi_{halt}$, as desired. \square

PROBLEM 6

Say a register machine P' extends P iff on any input on which P halts, so does P' and, moreover, they both give the same output. (In other words, if you think of programs as functions from inputs to outputs, P' extends P in the sense of functions.)

Let P_0 be the program that given another program P runs it until it halts, and then outputs the number of steps that it took P to halt (and never halts if P does not halt). Show that there is no extension P' of P_0 that always halts.

Proof. Suppose that P' extends P_0 . We will show how to use P' to decide the halting problem. Note that P' , given a program that halts, outputs the number of steps it takes the program to halt. Take some program P , and run P_1 on P . By assumption, P_1 outputs something. It may not even be a number. If it isn't a number, than P does not halt. If P_1 does output a number, k , run P for k steps. If it halts, obviously P is in Π_{halt} . If it doesn't halt after k steps, P will never halt,

(because if it halted after some other number of steps, P' would have given that other number as an output instead of k), and thus P is not in Π_{halt} . \square