Computability Theory and Ramsey Theory

An Exposition by William Gasarch

All of the results in this document are due to Jockusch [2]. For more results in computable combinatorics see the survey by Gasarch [1].

1 A Crash Course in Computability Theory

Notation 1.1

1. $M_1, M_2, \ldots$ is a standard list of Turing Machines (TMs). You can think of them as all Java programs.

2. We assume that from $e$ we can extract the code for $M_e$.

3. $M_{e,s}(x)$ means that we run $M_e$ for $s$ steps.

4. $M(x) \downarrow = a$ means that $M(x)$ halts and outputs $a$.

5. $M(x) = a$ means that $M(x)$ halts and outputs $a$ (we use the $\downarrow$ when we want to emphasize that $M(x)$ halts).

6. $M(x) \uparrow = a$ means that $M(x)$ does not halt.

7. A set $A$ is computable if there is a TM $M$ such that

\[
\begin{align*}
x \in A & \implies M(x) \downarrow = 1 \\
x \notin A & \implies M(x) \downarrow = 0
\end{align*}
\]

(Older books use the term recursive instead of computable.)

8. If $M$ is a TM such that on every input $x$, $M(x) \downarrow \in \{0, 1\}$ (so $M$ computes some set) then $L(M) = \{x \mid M(x) = 1\}$ (so $L(M)$ is the set that $M$ computes).
9. A set $A$ is *computably enumerable (c.e.)* if there is a TM $M$ such that

\[ x \in A \implies M(x) \downarrow \]
\[ x \notin A \implies M(x) \uparrow \]

(Older books use the term *recursively enumerable (r.e.)* instead of *computably enumerable (c.e.).*)

10. $W_e$ is the domain of $M_e$, that is, $W_e = \{ x | (\exists s)[M_{e,s}(x) \downarrow] \}$.

11. $W_{e,s} = \{ x | M_{e,s}(x) \downarrow \}$.

12. A function $f$ is *computable* if there is a TM $M$ such that, for all $x$, $M(x) \downarrow = f(x)$. (Older books use the term *recursive* instead of *computable.*)

Sets are classified in the Arithmetic hierarchy.

**Notation 1.2**

1. $A \in \Sigma_0$ if $A$ is computable.

2. $A \in \Pi_0$ if $A$ is computable.

3. $A \in \Sigma_1$ is there exists $B \in \Pi_0$ such that $A = \{ x | (\exists y)((x, y) \in B) \}$.

4. $A \in \Pi_1$ is there exists $B \in \Sigma_0$ such that $A = \{ x | (\forall y)((x, y) \in B) \}$.

5. Alternative definition: $A \in \Pi_1$ if $\overline{A} \in \Sigma_1$.

6. For $i \geq 1$ $A \in \Sigma_i$ is there exists $B \in \Pi_{i-1}$ such that $A = \{ x | (\exists y)((x, y) \in B) \}$

7. For $i \geq 1$ $A \in \Pi_i$ is there exists $B \in \Sigma_{i-1}$ such that $A = \{ x | (\forall y)((x, y) \in B) \}$

8. Alternative definition: $A \in \Pi_i$ if $\overline{A} \in \Sigma_i$. 

2
Examples and Facts

1. $HALT = \{(e, x) \mid (\exists s)(M_{e,s}(x) \downarrow) \in \Sigma_1 - \Sigma_0\}$

2. $W_0, W_1, \ldots$ is a list of all $\Sigma_1$ sets.

3. $FIN$ is the set of all $e$ such that $W_e$ is finite.

   $$FIN = \{e \mid (\exists x)(\forall y, s)(y > x \implies y \notin W_{e,s}) \in \Sigma_2 - \Pi_2.\}$$

   (The proof that $FIN \notin \Pi_2$ is not easy.)

4. $INF$ is the set of all $e$ such that $W_e$ is infinite. $INF \in \Pi_2 - \Sigma_2$. (The proof that $INF \notin \Sigma_2$ is not easy.)

5. $COF$ is the set of all $e$ such that $W_e$ is co-finite. We leave it to you to show that $COF \in \Sigma_3$.

   (The proof that $COF \notin \Pi_3$ is not easy.)

6. $\Sigma_0 \subset \Sigma_1 \subset \Sigma_2 \subset \cdots$.

7. $\Pi_0 \subset \Pi_1 \subset \Pi_2 \subset \cdots$.

8. For all $i \geq 1$, $\Sigma_i$ and $\Pi_i$ are incomparable.

**Theorem 1.3** Every infinite $\Sigma_1$ set has an infinite computable subset.

**Proof:** Let $A = \{x \mid (\exists y)((x, y) \in B)\}$ where $B$ is computable. Assume $A$ is infinite. Let $M$ be the TM that decides $B$. We first write a program for a function that outputs a subset of the elements of $A$ in increasing order. Since we have a program, $f$ is computable.

Algorithms for function $f$.

1. Input($i$)
2. If \( i = 0 \) then compute \( M(0, 0), M(0, 1), M(1, 0) \ldots \) (go through all pairs until it stops) until you find an \((x, y)\) with \( M(x, y) = 1 \). Output \( x \).

3. If \( i \geq 1 \) then compute \( Z = \{f(0), \ldots, f(i-1)\} \). Let \( m \) be the max element of \( Z \).

4. Compute \( M(0, 0), M(0, 1), M(1, 0) \ldots \) (go through all pairs until it stops) until you find an \((x, y)\) with \( M(x, y) = 1 \). AND \( x > m \). Output \( x \).

Since \( A \) is infinite, for all \( f \), \( f(i) \) is defined. Note that the image of \( f \) is an infinite subset of \( A \).

We now show that the image of \( f \) is computable.

Algorithm that computes \( C \), an infinite subset of \( A \).

1. Input \( x \)

2. Compute \( f(0), f(1), \ldots \) until one of the following occurs.

   - You find an \( i \) such that \( f(i) = x \). Then output 1 and halt.
   - You find an \( i \) such that \( f(i) < x < f(i+1) \). Then output 0 and halt.

Clearly \( C \) is computable and is the image of \( f \), hence an infinite subset of \( A \). □

We now allow our TMs to have access to an oracle. That is, they can ask questions to some set \( X \). We can define Oracle TM (OTM) independent of the oracle, like writing a Java Program that calls a not-yet-defined-procedure.

Notation 1.4 \( X \) is a set throughout this definition.

1. \( M^{(i)}_1, M^{(i)}_2, \ldots \) is a standard list of OTM. You can think of them as all Java programs with a call to a non-yet-written subroutine that returns YES or NOT.

2. We assume that from \( e \) we can extract the code for \( M^{(i)}_e \).

3. \( M^{X}_{e,s}(x) \) means that we run \( M^{X}_e \) for \( s \) steps.
4. $M^X(x) \downarrow = a$ means that $M^X(x)$ halts and outputs $a$.

5. $M^X(x) = a$ means that $M(x)$ halts and outputs $a$ (we use the $\downarrow$ when we want to emphasize that $M^X(x)$ halts).

6. $M^X(x) \uparrow = a$ means that $M^X(x)$ does not halt.

7. A set $A$ is computable-in-$X$ if there is an OTM $M^0$ such that

$$x \in A \implies M^X(x) \downarrow = 1$$
$$x \notin A \implies M^X(x) \downarrow = 0$$

We also denote this by $A \leq_T X$. This is a very important definition. (Older books use the term recursive-in-$X$ instead of computable-in-$X$.)

8. If $M^0_i$ is a OTM such that on every input $x$, $M^X(x) \downarrow \in \{0, 1\}$ (so $M^X$ computes some set)
then $L(M^X) = \{x | M^X(x) = 1\}$ (so $L(M^X)$ is the set that $M^X$ computes).

9. A set $A$ is computably enumerable-in-$X$ (c.e.-in-$X$) if there is a OTM $M^0$ such that

$$x \in A \implies M^X(x) \downarrow$$
$$x \notin A \implies M^X(x) \uparrow$$

10. $W^X_e$ is the domain of $M^X_e$, that is, $W^X_e = \{x | \exists s [M^X_{e,s}(x) \downarrow]\}$.

11. $W^X_{e,s} = \{x | M^X_{e,s}(x) \downarrow\}$.

12. A function $f$ is computable-in-$X$ if there is a OTM $M^0$ such that, for all $x$, $M^X(x) \downarrow = f(x)$.

(Older books use the term recursive-in-$X$ instead of computable-in-$X$.)

Examples and Facts

1. $HALT^X = \{(e, x) | \exists s [M^X_{e,s}(x) \downarrow] \in \Sigma^X_1 - \Sigma^X_0$
2. \( W_0^X, W_1^X, \ldots \) is a list of all \( \Sigma_1^X \) sets.

3. \( FIN^X \) is the set of all \( e \) such that \( W_e^X \) is finite.

\[
FIN^X = \{ e \mid (\exists x)(\forall y, s)[y > x \implies y \notin W_{e,s}^X] \in \Sigma_2^X - \Pi_2^X. \}
\]

(The proof that \( FIN \notin \Pi_2^X \) is identical to the proof that \( FIN \notin \Pi_2 \).

4. \( INF^X \) is the set of all \( e \) such that \( W_e^X \) is infinite. \( INF^X \in \Pi_2^X - \Sigma_2^X. \) (The proof that \( INF \notin \Sigma_2^X \) is identical to the proof that \( INF \notin \Sigma_2 \) (Proving that \( INF \notin \Sigma_2 \) is not easy.)

5. \( COF^X \) is the set of all \( e \) such that \( W_e^X \) is co-finite. We leave it to you to show that \( COF^X \in \Sigma_3^X \). (The proof that \( COF^X \notin \Pi_3^X \) is identical to the proof that \( COF \notin \Pi_3 \).

6. \( \Sigma_0^X \subset \Sigma_1^X \subset \Sigma_2^X \subset \cdots \).

7. \( \Pi_0^X \subset \Pi_1^X \subset \Pi_2^X \subset \cdots \).

8. For all \( i \geq 1, \Sigma_i^X \) and \( \Pi_i^X \) are incomparable.

**Lemma 1.5** If \( A \in \Sigma_1 \) or \( A \in \Pi_1 \) then \( A \leq_T HALT \). The OTM is very simple in that in asks \( HALT \) only one question. (We use this in the following form: \( HALT \) can be used to answer a any question of the form \( (\exists z)[z \in B] \) or \( (\forall z)[z \in B]. \)

**Proof:** Let \( A = \{ x \mid (\exists y)[(x, y) \in B] \} \) where \( B \) is computable. Let \( B \) be computed by TM \( M \).

The following OTM with oracle \( HALT \) decides \( A \)

1. Input \( x \)

2. CREATE (but DO NOT RUN) a TM that does the following
• For $y = 0, 1, \ldots$ until you find a $z$ such that $M(x, y) = 1$ (if this never happens then the program will diverge)

3. Let $e$ be such that the program above is $M_e$.

4. ASK $e \in HALT$. If YES then output 1, if NO then output 0.

Since $\Pi_1$ sets are the complements of $\Sigma_1$ sets, one can easily get that $\Pi_1$ sets are $\leq_T HALT$.

\textbf{Theorem 1.6} Every infinite $\Sigma_2$ set has an infinite subset $X \leq_T HALT$. (There is a statement about every $\Sigma_i$ set has an infinite subset with some properties but it is not needed here and would take us too far afield.)

\textbf{Proof:} Let $A = \{x \mid (\exists y)(\forall z)[(x, y, z) \in B]\}$ where $B$ is computable. Assume $A$ is infinite. Let $M$ be the TM that decides $B$. We first write a program using oracle $HALT$ for a function that outputs a subset of the elements of $A$ in increasing order. Since we have an oracle-program with oracle $HALT$, $f \leq_T HALT$.

Algorithms using oracle $HALT$ for function $f$.

1. Input($i$)

2. If $i = 0$ then

   using the oracle $HALT$ ask the questions (using Lemma 1.5).

   $(\forall z)[M(0, 0, z)]$

   $(\forall z)[M(0, 1, z)]$

   $(\forall z)[M(1, 0, z)]$

   $(\forall z)[M(1, 1, z)]$
(go through all pairs until you stop) until you find an \((x, y)\) such that the answer is YES.

Output \(x\).

3. If \(i \geq 1\) then compute \(Z = \{f(0), \ldots, f(i-1)\}\). Let \(m\) be the max element of \(Z\).

4. using the oracle \(HALT\) ask the questions (using Lemma 1.5)

\[
(\forall z)[M(0, 0, z)]
\]
\[
(\forall z)[M(0, 1, z)]
\]
\[
(\forall z)[M(1, 0, z)]
\]
\[
(\forall z)[M(1, 1, z)]
\]

(go through all pairs until you stop) until you find an \((x, y)\) such that the answer is YES AND \(x > m\). Output \(x\).

Since \(A\) is infinite, for all \(f\), \(f(i)\) is defined. Note that the image of \(f\) is an infinite subset of \(A\).

We now show that the image of \(f\) is computable.

Algorithm with oracle \(HALT\) that computes \(C\), an infinite subset of \(A\).

1. Input \(x\)

2. Compute \(f(0), f(1), \ldots\) until one of the following occurs.

   - You find an \(i\) such that \(f(i) = x\). Then output 1 and halt.
   - You find an \(i\) such that \(f(i) < x < f(i + 1)\). Then output 0 and halt.

Clearly \(C \leq_T HALT\) and is the image of \(f\), hence an infinite subset of \(A\).

\textbf{Theorem 1.7} \(A \in \Sigma_2\) iff \(A\) is c.e.-in-\(HALT\).
Proof:

1) $A \in \Sigma_2$ implies $A$ is c.e.-in-$HALT$:

If $A \in \Sigma_2$ then there exists a TM $M$ that always converges such that

$$A = \{x \mid (\exists y)(\forall z)[M(x, y, z) = 1]\}.$$

Let $M^{HALT}$ be the TM that does the following:

1. Input($x$, $y$).

2. Ask $HALT$ $(\forall z)[M(x, y, z) = 1]$. (Can rephrase as $(\exists z)[M(x, y, z) = 0]$.)

3. If YES answer YES, if NO then answer NO.

$$A = \{x \mid (\exists y)[M^{HALT}(x, y) = 1]\}.$$

Hence $A$ is c.e.-in-$HALT$.

2) $A$ c.e.-in-$HALT$ implies $A \in \Sigma_2$.

$A$ is c.e.-in-$HALT$. So

$$A = W_e^{HALT} = \{x \mid (\exists s)(\forall t)[t \geq s \implies x \in W_{e,t}^{HALT}]\}.$$ 

So $A$ is $\Sigma_2$.  

2 A Computable Coloring With No Infinite $\Sigma_2$ Homog Set

Def 2.1

1. $HALT_s = \{(e, x) \mid 0 \leq e, s \leq s \land M_{e,s}(x) \downarrow\}$ Note that $HALT_s$ is a finite set which can be determined given $s$. 
2. Let $M_{e,s}^{HALT}(x)$: compute $HALT_s$, then use it as an oracle in the $M_{e,s}$ calculation. If it halts normally, GREAT output what it outputs. If not then DIVERGE.

We first show there is a computable coloring with no homog set $X \leq_T HALT$.

**Theorem 2.2** *There exists a computable COL : $(\mathbb{N}_2) \to [2]$ such that there is no infinite homog set $X$ with $X \leq_T HALT$.*

**Proof:** We use that $L(M_0^{HALT}), L(M_1^{HALT}), \ldots$ is a list that contains all sets $X \leq_T HALT$.

We construct computable $COL : (\mathbb{N}_2) \to [2]$ to satisfy the following requirements (NOTE-requirements is the most important word in computability theory.)

\[ R_e : L(M_e^{HALT}) \text{ infinite } \implies L(M_e^{HALT}) \text{ NOT a homog set}. \]

**CONSTRUCTION OF COLORING**

*Stage 0:* $COL$ is not defined on anything.

*Stage s:* We define $COL(0, s), \ldots, COL(s - 1, s)$. For $e = 0, 1, \ldots, s$:

If this occurs: ($\exists x < y \leq s - 1$) such that

- $COL(x, s)$ and $COL(y, s)$ have not been colored (note that they may have been colored by some $R_i$ with $i < e$).
- $x \in L(M_{e,s}^{HALT}(x))$.
- $y \in L(M_{e,s}^{HALT}(y))$.

then take the LEAST two $x, y$ for which this is the case and do the following:

- $COL(x, s) = RED$
- $COL(y, s) = BLUE$. 
(Note that IF $M_e^{HALT} = 1$ (which we do not know at this time) then $R_e$ would be satisfied.)

After you go through all of the $0 \leq e \leq s$ define all other $COL(x, y)$ where $0 \leq x < y \leq s$ that have not been defined by $COL(x, y) = RED$. This is arbitrary. The important things is that ALL $COL(x, s)$ where $0 \leq x \leq s - 1$ are now defined. This is why $COL$ is computable—at stage $s$ we have defined all $COL(x, y)$ with $0 \leq x < y \leq s$.

END OF CONSTRUCTION

We show that, for all $e$, $R_e$ is satisfied.

If $L(M_e^{HALT})$ is finite then $R_e$ is satisfied.

We assume $L(M_e^{HALT})$ is infinite. Let

$$x_1 < x_2 < \cdots < x_{2e+2}$$

be the first $2e + 2$ elements of $L(M_e^{HALT})$. Let $s_0$ be such that for all $t \geq s_0$, for $1 \leq j \leq 2e + 2$, the computation $M_{e,t}^{HALT}(x_j)$ is legit. Let $s_1 \geq t$ be such that $s_1 \in L(M_e^{HALT})$ (note that $s_1$ is much bigger than $x_{2e+2}$). Note that at state $s_1$ it is not known that $s_1 \in L(M_e^{HALT})$.

Lets look at stage $s_1$. KEY: requirements $R_0, \ldots, R_{e-1}$ will color at most $2e$ of the edges $COL(x_1, s_1), COL(x_2, s_1), \ldots, COL(x_{2e+2}, s_1)$. So when $R_e$ gets to act there will be an $x_{j_1} < x_{j_2}$ such that $COL(x_{j_1}, s_1)$ and $COL(x_{j_2}, s_1)$ have not been colored. So $COL(x_{j_1}, s_1) = RED$ and $COL(x_{j_2}, s_1) = BLUE$. Since $s_1 \in L(M_e^{HALT})$ (though that is not known yet). $R_e$ will be satisfied.

Theorem 2.3 There exists a computable $COL : \binom{\mathbb{N}}{2} \rightarrow [2]$ such that there is no infinite homog set $X$ with $X$ a $\Sigma_2$ set.

Proof: Let $COL$ be the coloring from Theorem 2.2. If there was an infinite $\Sigma_2$-homog set $X$ then, by Theorem 1.6 there would be an infinite $Y \subseteq X$ such that $Y \leq_T HALT$. But by Theorem 2.2 this is impossible.
3 Every Computable Coloring has an Infinite $\Pi^0_2$ Homog set

Take the standard proof of the infinite 2-ary Ramsey Theorem. Let $COL$ be the given coloring of $\mathbb{N}^2$. Assume $COL$ is computable.

The function $COL'$ from $\mathbb{N}$ to $\{R, B\}$ can be computed by asking $\Pi^0_2$ questions. Hence we say informally $COL' \leq^T \Pi^0_2$. One can show that using this all three sets: $R$, $B$, and $DEAD$ are $\Sigma^0_3$.

We now have a subtle point. If all we want to know is the complexity of a homog set we can say that ONE OF $R$ or $B$ is infinite, hence there IS a $\Sigma^0_3$-homog set. And this is the answer we will give. But notice that we do not know which of $R$ or $B$ is the homog set. That would require a $\Sigma^0_4$-question.

Can we do better? YES! See the next section.

4 Every Computable Coloring has an Infinite $\Pi^0_2$ Homog set

We obtain this with a modification of the usual proof of Ramsey’s theorem. the key is that we don’t really toss things out- we guess on what the colors are and change our mind.

**Theorem 4.1** For every computable coloring $COL : (\mathbb{N}^2) \to [2]$ there is an infinite $\Pi^0_2$ homog set.

**Proof:**

We are given computable $COL : (\mathbb{N}^2) \to [2]$.

CONSTRUCTION of $x_1, x_2, \ldots$ and $c_1, c_2, \ldots$

NOTE: at the end of stage $s$ we might have $x_1, \ldots, x_i$ defined where $i < s$. We will not try to keep track of how big $i$ is. Also, we may have at stage (say) 1000 a sequence of length 50, and then at stage 1001 have a sequence of length only 25. The sequence will grow eventually but do so in fits and starts.
\[ x_1 = 1 \]

\[ c_1 = \text{RED} \] We are guessing. We might change our mind later

Let \( s \geq 2 \), and assume that \( x_1, \ldots, x_{s-1} \) and \( c_1, \ldots, c_{s-1} \) are defined.

1. Ask \textit{HALT} \textit{Does there exists } \( x \geq x_{s-1} \text{ such that, for all } 1 \leq i \leq s-1, COL(x_i, x) = c_i \)?

2. If YES then (using that \( COL \) is computable) find the least such \( x \).

\[ x_i = x \]

\[ c_i = \text{RED} \] We are guessing. We might change our mind later

We have implicitly tossed out all of the numbers between \( x_{i-1} \) and \( x_i \).

3. If NO then we ask \textit{HALT} how far back we can go. More rigorously we ask the following sequence of questions until we get a YES.

- \textit{Does there exists } \( x \geq x_{s-1} \text{ such that, for all } 1 \leq i \leq s-2, COL(x_i, x) = c_i \)?
- \textit{Does there exists } \( x \geq x_{s-1} \text{ such that, for all } 1 \leq i \leq s-3, COL(x_i, x) = c_i \)?
- \ldots
- \textit{Does there exists } \( x \geq x_{s-1} \text{ such that, for all } 1 \leq i \leq 2, COL(x_i, x) = c_i \)?
- \textit{Does there exists } \( x \geq x_{s-1} \text{ such that, for all } 1 \leq i \leq 1, COL(x_i, x) = c_i \)?

(One of these must be a YES since (1) if \( c_1 = \text{RED} \) and there are NO red edges coming out of \( x_1 \) then there must be an infinite number of \( \text{BLUE} \) edges, and (2) if \( c_1 = \text{BLUE} \) its because there are only a finite number of \( \text{RED} \) edges coming out of \( x_1 \) so there are an infinite number
of BLUE edges. Let \( i_0 \) be such that There exists \( x \geq x_{s-1} \) such that, for all \( 1 \leq i \leq i_0 \),
\[ \text{COL}(x_i, x) = c_i \] Do the following:

(a) Change the color of \( c_{i+1} \). (We will later see that this change must have been from RED to BLUE.

(b) Wipe out \( x_{i+2}, \ldots, x_{s-1} \).

(c) Search for the \( x \geq x_{s-1} \) that the question asked says exist.

(d) \( x_{i+2} \) is now \( x \).

(e) \( c_{i+2} \) is now RED.

END OF CONSTRUCTION of \( x_1, x_2 \ldots \) and \( c_1, c_2, \ldots \).

We need to show that there is a \( \Pi_2 \) homog set.

Let \( X \) be the set of \( x_i \) that are put on the board and stay on the board.

Let \( R \) be the set of \( x_i \in X \) whose final color is RED.

Claim 1: Once a number turns from RED to BLUE it can’t go back to RED again.

Proof:

If a number is turned BLUE its because there are only a finite number of RED edges coming out of it. Hence there must be an infinite number of BLUE edges coming out of it. Hence it will never change color (though it may be tossed out).

End of Proof

Claim 1: \( X, R \in \Pi_2 \).

Proof:

We show that \( X \in \Sigma_2 \). In order to NOT be in \( X \) you must have, at some point in the construction, been tossed out.

\[ X = \{ x \mid (\exists x)[\text{at stage } s \text{ of the construction } x \text{ was tossed out }] \}. \]
Note that the condition is computable-in-\textsc{HALT}. Hence $\overline{X}$ is c.e.-in-\textsc{HALT}. By Theorem 1.7 $\overline{X} \in \Sigma_2$.

We show that $\overline{R} \in \Sigma_2$. In order to NOT be in $R$ you must have to either NOT be in $X$ or have been turned blue. Note that once you turn at some point in the construction, been tossed out.

$$\overline{R} = \overline{X} \cup \{x \mid (\exists x)[\text{at stage } s \text{ of the construction } x \text{ was turned BLUE}]\}.$$ 

Note that the condition is computable-in-\textsc{HALT}. Hence $\overline{R}$ is c.e.-in-\textsc{HALT}. By Theorem 1.7 $\overline{R} \in \Sigma_2$.

End of Proof

We have shown $X, R$ are $\Pi_2$ but have not shown that $B$ is- and in fact $B$ might not be. But we show that $B$ is $\Pi_2$ when we need it to be.

There are two cases:

1. If $R$ is infinite then $R$ is an infinite homog set that is $\Pi_2$.

2. If $R$ is finite then $B$ is $X$ minus a finite number of elements. Since $X$ is $\Pi_2$, $B$ is $\Pi_2$.

References
