Solutions TEN1 2019-06-14 MMA130

1. In your own words,

a) explain the purpose of Gödel numbers

Answer: Since Peano arithmetic, which is the context within which one usually proves Gödel's Incompleteness Theorem, only works with natural numbers as data, it is for the purpose of speaking about formulae in logic necessary to somehow encode arbitrary formulae (and sequences of formulae) as natural numbers. That is what Gödel numbers do.

b) describe in detail some system of Gödel numbers for formulae,

Answer 1. The following system was used in E. Mendelson, "Introduction to Mathematical Logic". It is using the same principles as Gödel's original system (though details such as exact choices of numbers for symbols may differ).

Supported formula language:

| Symbol | Name | Number |
|---|---|---|
| ( | left parenthesis | 3 |
| ) | right parenthesis | 5 |
| , | comma | 7 |
| $\neg$ | negation | 9 |
| $\Rightarrow$ | implies | 11 |
| $\forall$ | forall | 13 |

There are six singleton symbols in the formula language, as listed in the table; all other connectives and quantifiers are shorthands for combinations of these.

There are also four countably infinite families of symbols, which are assigned odd numbers $\geq 15$ as Gödel numbers. For $k, n \geq 1$, these families are:

<u>Variables</u>: $x_k$ has number $13+8k$

<u>Constants</u>: $a_k$ has number $7+8k$

<u>Functions</u>: Are explicitly indexed by both arity $n$ and sequence number $k$, were $f_k^n$ has number $1+8\cdot 2^n\cdot 3^k$.

<u>Predicates</u>: Are doubly indexed like the functions; $A_k^n$ (the k'th predicate taking $n$ arguments) has number $3+8\cdot 2^n\cdot 3^k$.

A <u>formula</u> (not necessarily well-formed) is a sequence of symbols, and the corresponding sequence of Gödel numbers for these symbols get encoded into one number by taking them as exponents in the prime factorisation of the number: if the k'th symbol has Gödel number $m$ then the Gödel number for the whole formula has a factor $p_k^m$, where $p_k$ is the k'th prime.

A <u>sequence of formulae</u> (e.g. a proof) is similarly encoded as the number whose prime exponents are the Gödel numbers of the formulae in the sequence. (It is a feature of this system that Gödel numbers of symbols are always odd, Gödel numbers of formulae always even with an odd exponent on 2, and Gödel numbers of sequences of formulae always even with an even exponent on 2, but whether that (feature) is of practical use is less clear.)

<u>Answer 2</u>. The more computing-oriented author might pick a more contemporary solution: Just Use Unicode! Any formula is a sequence of (non-control) Unicode characters, and to turn it into a Gödel number we first transform it into an octet-sequence

according to UTF-8, and then take the bytes of that octet-sequence as radix-256 digits in an unsigned number, written in little-endian order. For sequences of formulae, we separate the elements of the sequence using New line (U+000A) characters.

To spell out some details of that, here is a partial symbol table:

| Character | Codepoint (hex) | UTF-8 (hex) | UTF-8 (decimal) |
|---|---|---|---|
| ∀ | 2200 | E2 88 80 | 226 136 128 |
| x | 0078 | 78 | 120 |
| ¬ | 00AC | C2 AC | 194 172 |
| ( | 0028 | 28 | 40 |
| S | 0053 | 53 | 83 |
| = | 003D | 3D | 61 |
| 0 | 0030 | 30 | 48 |
| ) | 0029 | 29 | 41 |

This approach begs some questions, for example how to distinguish between variables, constants, functions, and predicates, but that is technically more part of defining what it means for a formula to be well-formed (i.e., the syntax of formulae). One issue that the designer probably should address is that of how to provide an unbounded supply of variable names, since even the huge number of letters in Unicode may well turn out to be insufficient for expressing a Gödel sentence (they're really long). The Unicode purist could suggest using combining characters. A more mathematics-looking solution could be to declare that a variable name consists of a base letter followed by zero or more subscript digits (codepoints U+2080 through U+2089), so that $x$, $x_0$, and $x_{10}$

are all distinct variables.

c) explain the role of recursive functions in the proof of Gödel's incompleteness theorem.

Answer. The serve as a (primitive) programming language, in which one may express the many helper functions that are needed to construct claims such as "... is a proof of ..." about formulae encoded as Gödel numbers.

d) Encode the formula $\forall x \neg(Sx = 0)$ (first axiom of Peano arithmetic: 0 is not the successor of any natural number) using the Gödel number system you described above.

Solution 1 (Mendelson system). Here it must first be observed that Mendelson cannot write the formula quite like that, because he has all the variables, constants, functions, and predicates indexed: instead of x, one would have to use "variable no. 1" $x_1$, whose Gödel number is 21. Likewise 0 is $a_1$ (constant 1), S is $f_1^1$ (unary function no. 1) and = is $A_1^2$ (binary predicate no. 1), so the formula to encode is technically

$$(\forall x_1) \neg A_1^2(f_1^1(x_1), a_0)$$

(for some reason Mendelson encloses quantifiers in a parenthesis, and arguably too many symbols in this formula language are A's) and the sequence of symbol Gödel numbers are thus 3, 13, 21, 5, 9, $3+8\cdot2^2\cdot3^1=99$, 3, $1+8\cdot2^1\cdot3^1=55$, 3, 21, 5, 7, $7+8\cdot1=15$, 5. Therefore the Gödel number for the formula is

**Answer:** $2^{3} \cdot 3^{13} \cdot 5^{21} \cdot 7^{5} \cdot 11^{9} \cdot 13^{99} \cdot 17^{3} \cdot 19^{55} \cdot 23^{3} \cdot 29^{21} \cdot 31^{5} \cdot$
$\cdot 37^{7} \cdot 41^{15} \cdot 43^{5}$

Solution 2 (Unicode system). The sequence of bytes to encode are 226, 136, 128, 120, 194, 172, 40, 83, 120, 61, 48, 41, so the Gödel number for the formula is

**Answer:** $226 + 136 \cdot 256 + 128 \cdot 256^{2} + 120 \cdot 256^{3} + 194 \cdot 256^{4} +$
$+ 172 \cdot 256^{5} + 40 \cdot 256^{6} + 83 \cdot 256^{7} + 120 \cdot 256^{8} + 61 \cdot 256^{9} +$
$+ 48 \cdot 256^{10} + 41 \cdot 256^{11} = 12\,747\,204\,125\,973\,623\,167\,067\,916\,514$

2. Give a natural deduction proof that $p \wedge q, \neg r \to \neg p \vdash r$.

Solution. $q$ is of no use at all — the real argument is about $p$ and $r$. That main part $p, \neg r \to \neg p \vdash r$ looks a lot like Modus Tollens (MT, rule 11), but the negations are in the wrong places (in the implication, not in the other formulae). However we can still use MT if we first pad with double negations. This leads to the following proof

| | | |
|---|---|---|
| 1. | $p \wedge q$ | Premise |
| 2. | $p$ | $(\wedge e)$ on 1 |
| 3. | $\neg\neg p$ | $(\neg\neg i)$ on 2 |
| 4. | $\neg r \to \neg p$ | Premise |
| 5. | $\neg\neg r$ | (MT) on 3, 4: $\phi$ is $\neg r$, $\psi$ is $\neg p$ |
| 6. | $r$ | $(\neg\neg e)$ on 5 |

Alternative solution. Proof by contradiction could also be used to produce an $r$ without negation.

| | | |
|---|---|---|
| 1. | $p \wedge q$ | Premise |
| 2. | $\neg r \to \neg p$ | Premise |
| 3. | $\neg r$ | Assumption |
| 4. | $\neg p$ | MP ($\to e$) on 2, 3 |
| 5. | $p$ | ($\wedge e$) on 1 |
| 6. | $\bot$ | ($\neg e$) on 4, 5 |
| 7. | $r$ | PBC on 3–6. |

3. Russell's paradox in set theory starts with considering the set $R$ of all sets that do not contain themselves as elements.

a) Using set membership $\in$ as only predicate, write down a predicate logic formula expressing the claim that $R$ is the set of all sets that do not contain themselves.

Answer: $\forall x : (x \in R \leftrightarrow \neg(x \in x))$

b) Write down a proof, with justification of all steps, that the Russell set $R$ does not exist.

Solution. This requires some planning. That $R$ does not exist is $\neg \exists R : \forall x : (x \in R \leftrightarrow \neg(x \in x))$, so we want to end with negation introduction ($\neg i$), which means we assume the $\exists R : \ldots$ subformula and seek a contradiction. That's enough to get started.

1. $\exists R: \forall x: ((x \in R \to \neg(x \in x)) \wedge (\neg(x \in x) \to x \in R))$ Hyp for $(\neg i)$

2. $R_0$ $\forall x: ((x \in R_0 \to \neg(x \in x)) \wedge (\neg(x \in x) \to x \in R_0))$ Hyp for $(\exists e)$

Then what? The hint says consider $R \in R$, which now is $R_0 \in R_0$, but how does one get that (as a subformula)? By using $(\forall e)$ to substitute $R_0$ for $x$!

3. $(R_0 \in R_0 \to \neg(R_0 \in R_0)) \wedge (\neg(R_0 \in R_0) \to R_0 \in R_0)$ $(\forall e)$ on 2

4. $R_0 \in R_0 \to \neg(R_0 \in R_0)$ $(\wedge e)$ on 3

Now we are ready to introduce the hypothesis $R_0 \in R_0$, and can expect it to generate a contradiction, so we're in $(\neg i)$ territory.

5. $R_0 \in R_0$ Hyp

6. $\neg(R_0 \in R_0)$ $(\to e)$ on 4,5

7. $\bot$ $(\neg e)$ on 5,6

8. $\neg(R_0 \in R_0)$ $(\neg i)$ on 5–7

9. $\neg(R_0 \in R_0) \to R_0 \in R_0$ $(\wedge e)$ on 3

10. $R_0 \in R_0$ $(\to e)$ on 8,9

11. $\bot$ $(\neg e)$ on 8,10

12. $\bot$ $(\exists e)$ on 1, 2–11

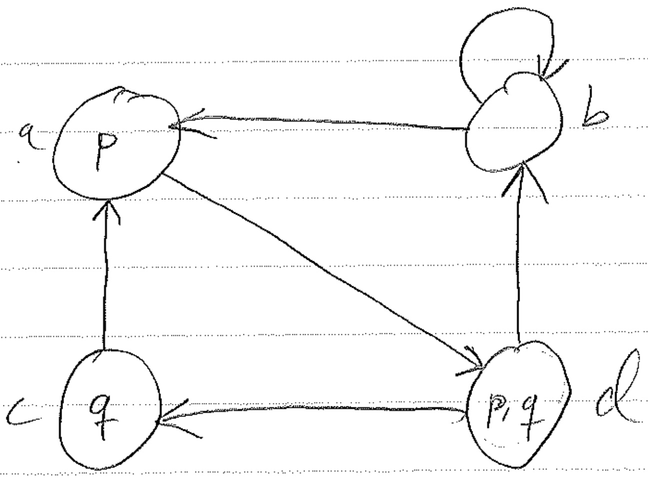
13. $\neg \exists R: \forall x: ((x \in R \to \neg(x \in x)) \wedge (\neg(x \in x) \to x \in R))$ $(\neg i)$ on 1–12.

QED

4. Consider the Kripke model $M = (W, R, L)$ where $W = \{a, b, c, d\}$, $L(a) = \{p\}$, $L(b) = \emptyset$, $L(c) = \{q\}$, $L(d) = \{p, q\}$, and $R = \{(a,d), (b,a), (b,b), (c,a), (d,b), (d,c)\}$.

a) Draw a graph for $M$.

Solution:




b) Determine the set of words where $p \to q$ is satisfied

Solution. By table.

| | $p$ | $q$ | $p \to q$ |
|---|---|---|---|
| a | T | F | F |
| b | F | F | T |
| c | F | T | T |
| d | T | T | T |

Answer: $\{b, c, d\}$

c) Determine the set of worlds where $\Diamond(p \to q)$ is satisfied.

Solution. Continuing the above, this would be the set of worlds which have b, c, or d as some successor, i.e., a (because d), b (because b), not c (only successor is a), but d (because c, or b).
Answer: $\{a, b, d\}$.

d) Determine the set of worlds where $\Box p \to q$ is satisfied.

Solution. The old table is of little use, so let's write a new one.

| | $p$ | $\Box p$ | $q$ | $\Box p \to q$ |
|---|---|---|---|---|
| a | T | T | F | F |
| b | F | F | F | T |
| c | F | T | T | T |
| d | T | F | T | T |

Answer: $\{b, c, d\}$

e) Determine the set of worlds where $Fp$ is satisfied

Solution. $Fp$ is an LTL formula; to ask whether it is satisfied in a world (state) is to ask whether it is satisfied for all paths which begin at that state. a and d trivially satisfy $Fp$, because $p$ is satisfied there. c satisfies $Fp$ because the only next state is a. b will however not satisfy $Fp$, because on the path that loops at b forever $p$ never goes true.

Answer: $\{a, c, d\}$.

8. Check the validity in the non-classical logic $L_2$ of the following statements.

a) $\vDash p \to \neg\neg p$

b) $\vDash \neg\neg p \to p$

c) $\vDash p \vee \neg p$

d) $p \vDash \neg p \to \bot$

e) $\vDash (p \wedge \neg p) \to \bot$

f) $p \to r, q \to r \vDash (p \vee q) \to r$

Solution. This is again doable much by table, but now with three logic values $0, \frac{1}{2}$, and $1$:

| | | | (a) | (b) | (c) | | (e) |
|---|---|---|---|---|---|---|---|
| $p$ | $\neg p$ | $\neg\neg p$ | $p \to \neg\neg p$ | $\neg\neg p \to p$ | $p \vee \neg p$ | $p \wedge \neg p$ | $(p \wedge \neg p) \to \bot$ |
| $0$ | $1$ | $0$ | $1$ | $1$ | $1$ | $0$ | $1$ |
| $\frac{1}{2}$ | $\frac{1}{2}$ | $\frac{1}{2}$ | $1$ | $1$ | $\frac{1}{2}$ | $\frac{1}{2}$ | $\frac{1}{2}$ |
| $1$ | $0$ | $1$ | $1$ | $1$ | $1$ | $0$ | $1$ |

For the claims with premises, the method is slightly different: putting something on the left of an $\vDash$ means we only consider cases where that is true (1). For (d), that means

$$V(p) = 1, \ V(\neg p) = 0, \ V(\bot) = 0, \ V(\neg p \to \bot) = \min\{1, 1 - 0 + 0\} = 1.$$

For (f), $V(p \to r) = V(q \to r) = 1$ implies $V(p), V(q) \leq V(r)$

$$V((p \vee q) \to r) = \min\{1, 1 - V(p \vee q) + V(r)\} = \min\{1, 1 - \max\{V(p), V(q)\} + V(r)\} =$$
$$= \min\{1, 1 + \min\{-V(p), -V(q)\} + V(r)\} = \min\{1, 1 - V(p) + V(r), 1 - V(q) + V(r)\} = 1.$$

Answer: (a), (b), (d), and (f) are valid in $L_2$, (c) and (e) are not.