V: Introspection, and Gödel's Second Incompleteness Theorem

In this section, we present Gödel's Second Incompleteness Theorem (G2T). Along the way, we state a stronger version of G1T which doesn't require the semantic notion of **N**-consistency. The key idea for both of these is that TNT, like us, is able to prove things about what it (and its strengthenings) can prove...

Notation: If **S** is a formal system in an alphabet including that of TNT, we say "**S _proves_ \sigma**" and write
S |- \sigma
to mean that **\sigma** is a TNT-sentence which is also an **S**-theorem.

Definition:
S is _logically adequate_ if
whenever **S** proves some sentences **\sigma_1,...,\sigma_n** which together necessitate a sentence **\tau**, then **S** also proves **\tau**.
Symbolically:
if { **\sigma_1**, ..., **\sigma_n** } **|= \tau**
and if **S |- \sigma_i** for **i=1**,...,**n**,
then **S |- \tau**.

(So PRED is logically adequate, as is TNT)

S is _a strengthening of TNT_ if **S** proves every TNT-sentence which TNT does, i.e. TNT **|- \sigma => S |- \sigma**.

For the remainder of this section, we assume that **S** is a logically adequate strengthening of TNT.

In particular, **S** could be (Formal)TNT itself.

Definition:
S is _**(negation-)consistent**_ iff for no TNT-sentence **\sigma** does it hold
S |- \sigma and **S |- ~\sigma**.
S is _**(negation-)complete**_ iff for every TNT-sentence **\sigma**
S |- \sigma or **S |- ~\sigma**.

Remark:
N-soundness **=>** consistency
N-completeness **=>** completeness

If **S** is inconsistent, then **S |- \sigma** for all **\sigma**.

Fact 1:
TNT proves all true sentences of the form
Theorem_S(n)
Idea of proof:
If **Theorem_S(n)** is true, then **ProofPair(m,n)** is true for some m; similarly, all the existentials involved in the (lengthy) definition of ProofPair have natural number witnesses, and it becomes a matter of checking that TNT can prove simple (quantifier-free) arithmetical truths about terms. It can. Induction isn't needed.

Let **G := G_S** be the Gödel sentence of **S**.

Fact 2:
TNT "knows what **G** is":
TNT **|- < <G =) ~Theorem_S(G)> /\ <~Theorem_S(G) =) G> >**

[
Idea of proof (omitted in lectures):
This is morally straightforward, but technically less so.
Recall that **G** is **Ex:<x=[U] /\ U>**
where **U** is **Ey:<Arithmoquine(x,y) /\ ~Theorem_S(y)>**.
Now one can check, as in Fact 1, that
TNT **|- Arithmoquine([U],[G])**.
Furthermore, TNT **|- Ay:<Arithmoquine([U],y) =) y=[G]>**;
i.e. it understands (case-by-case) that Arithmoquine is a function.
This, perhaps surprisingly, is actually the most painful bit - see the notion of "**captures***" in chapter 9 of Peter Smith's "Gödel Without Tears" (linked from website).

The rest is basic logic, which TNT can do by Gödel's completeness theorem.
]

[
Actually, for Facts 1 and 2 we don't need much of TNT at all - just TNT' and the single TNT-theorem **Ax:<x=0 \/ Ey:x=Sy>**. This system
Q := TNT'+{Ax:<x=0 \/ Ey:x=Sy>}
is known as "Robinson arithmetic" (after Raphael Robinson, who isn't Abraham or Julia!).

Technically, to get Fact 2 in **Q** you have to modify the definition of Arithmoquine, to **Arithmoquine'(x,y) :=**
< Arithmoquine(x,y) /\ Ay':< y'<y =) ~Arithmoquine(x,y') > >.
They're equivalent in **N**, and TNT proves the equivalence, but **Q** doesn't.
]

Lemma 1:
If **S** is consistent, then **S |/- G**.

i.e. **N |= <Con(S) =) ~Theorem_S([G])>**
Proof:
Suppose **S |- G**.

Then by Fact 1, since **S** strengthens TNT,
S |- Theorem_S([G]).

But by Fact 2, since **S** strengthens TNT and is logically adequate,
S |- ~Theorem_S([G]).

So **S** is inconsistent.

Historical remark:
Lemma 1 formed half of Gödel's original proof of his First Incompleteness Theorem. Assuming a stronger form of consistency ("omega-consistency"), he proved that also **S |/- ~G**, showing that **S** is incomplete. **A** few years later, Rosser found a way to remove the assumption of omega-consistency, yielding:

Gödel-Rosser First Incompleteness Theorem [G1T]:
If **S** is consistent, it is incomplete.
Proof: omitted.

Definition:
Let **Con(S)** be the sentence
~Theorem_S([~0=0])
So **Con(S)** is true iff **S** is consistent.

Fact 3: TNT is able to prove Lemma 1:
TNT **|- <Con(S) =) ~Theorem_S([G])>**
Idea of proof:
Follow the proof of Lemma 1 (and hence the proofs of Facts 1 and 2), and see that TNT can do them... for example, we have to prove an "introspective version" of Fact 1:
TNT **|- Ax:<Theorem_S(x) =) "Theorem_TNT([Theorem_S(x)])">**
(where a little work is needed even to express the bit in quotes)

[
We need more of TNT for this than for the first two facts. We still don't need all of TNT, though: we only need induction axioms for "**\Sigma_1**-formulae", being those formulas which start with zero or more existential quantifiers, then have no more quantifiers of any kind.
]

Theorem [G2T]:
If **S** is consistent, **S |/- Con(S)**
Proof:
Suppose **S |- Con(S)**.

Then by Fact 3, **S |- ~Theorem_S([G])**.

So by Fact 2, **S |- G**.

But this contradicts Lemma 1.

Appendix **A**: some remarks on Hofstadter's presentation of Gödel's theorems

It's a bit of a mess, really.

He spends a whole chapter on the difference between primitive recursion and recursion, so that he can say that TNT represents primitive recursive relations. That's true − although **Q** also represents general recursive relations, and since he isn't even sketching a proof it seems strange to mention only the easier result. But more to the point, despite making a big deal of this representability, it's never actually used in the Gödel statement he proves! He sketches a proof of only the semantic version of G1T, for which expressibility is enough. Of course, the representability is relevant to G2T, or to Syntactic G1T, but he doesn't even try to prove those. Relatedly, he seems to use soundness and consistency interchangeably.

I have great affection for the book, and on balance I don't regret using it for this course, but it has meant that the overlap between the course and the book is rather smaller than I'd have liked.

Appendix **B**: some remarks on Bays's presentation of Gödel's theorems

(Since we're on the subject of introspection...)

I hope it isn't a mess. Really.

The approach to Gödel's theorems this course has ended up taking is fairly idiosyncratic. To summarise: taking more seriously than he could possibly have intended a throwaway parenthetical at the start of Hofstadter's GEB, we build everything around the notion of a Post formal system. This makes a nice introduction to semantics of syntax, leading to the propositional and predicate calculi. From the point of view of Gödel, the main advantage of using Post systems is that, given a Post system **S**, the formula expressing "is an **S**−theorem" (i.e. **S**−production) is relatively simple. We were able to describe it essentially in full detail. This not only makes it about as clear as it could be that theoremhood is definable, but also makes it relatively easy to believe that **Q** can do the proofs.

This is to be contrasted with a more conventional approach, which would ***first*** show that recursive functions **f** are definable and that **Q** can prove true statements of the form **f(n)=k**, and then argue that since theoremhood is clearly **r**.e. (which is only clear once you've built up a solid appreciation for Turing completeness!) we get the same results.

Of course, we still need to talk about computability if we want to say that **Th(N)** is undecidable, but the Turing completeness of Post systems gives us an avenue in to that.

The main downside of avoiding talking directly about recursive functions is that we can't extract a clear statement that **Q** proves all true statements of the form **\phi(n)** for **\phi** an **r**.e. predicate, nor that **r**.e. corresponds to **\Sigma_1**.

So in retrospect, I think the Post−approach works quite well. I consider it a happy medium between (a) the truly fluffy approach of just asserting that computability is something **Q** can do, and syntactic stuff is obviously computable so duh, and (**b**) the fully Proper approach going via (primitive) recursive functions and properly demonstrating that the relevant syntactic operations are recursive (for which you might well want to go via some more appropriate Turing complete system, such as... Post systems, say?).

One regret I have regarding the use of Post systems, however, is the terminology, which I lifted from Hofstadter and kept unchanged throughout the course. Calling productions "theorems", and initial strings "axioms", is cute but not really appropriate in general. Better would be to have made a distinction at the stage when we defined 'well−formed' strings as the interpreted ones to which we assign meaning. "**S**−theorem" should have been

defined as "well-formed **S**-production", explicitly leaving open the possibility of going via non-well-formed strings, perhaps using auxiliary symbols, to arrive at our theorems. This would nicely foreshadow the statement of Turing completeness of Post systems, and would be appropriately suggestive when we considering extending TNT by going outside of number theory, e.g. to ZFC; making use of auxiliary symbols (which we ***could*** optionally also interpret, yielding a consistency argument analogous to that for TNT (and a proof of **Con(ZFC)** from existence of inaccessible cardinals...)).

Appendix **C**: Idiosyncrasies

* TNT is usually known as PA.
* TNT' is almost Robinson's Q; more precisely, **Q** is TNT' along with an axiom saying that every non-zero number is a successor, **Ax:<~x=0 =)** Ey: **x=Sy>**. The point of **Q** is that it's enough for the syntactic Gödel-Rosser version of G1T we mentioned to go through: no strengthening of **Q** is both consistent and complete.
* PRED is one of numerous possible proof systems for "first-order predicate logic" in the language of arithmetic (and would not have to be changed much to accomodate other languages)
* The particular system PROP for propositional calculus was, as I hope was clear, not standard. But it's as good as any other. 'Detachment' is normally called 'Modus Ponens'. 'Switcheroo' doesn't have a common name to my knowledge, though it ought to.
* "formal system" is not normally synonymous with "Post system" as it was for us. There is no very precise definition of "formal system" in general currency, but the key idea is that the set of theorems provable in a formal system should be recursively enumerable. So by the Turing completeness of Post systems I stated, any formal system can be turned into a Post formal system.
* Relatedly, our terminology for Post systems, borrowed from Hofstadter, is non-standard, since they're not normally thought of as proof systems. What we called an 'axiom' would normally be called something like an 'initial word', and what we called a 'theorem' would normally be a 'production'. Also, they should really be called "Post canonical systems", or just "Post systems". They aren't actually particularly well-known, other theoretically simpler but less intuitive string-rewriting systems being more common, but they suited our purposes.
* "Semantic G1T" isn't common terminology; I nabbed it from Peter Smith. Normally when people (who know what they're talking about) talk about Gödel's First Theorem they're referring to a version I never even mentioned, because it was superceded by Gödel-Rosser. The basic version, due mainly to Gödel, is this: no logically adequate strengthening of **Q** is both complete and **\omega**-consistent. Recall that 'complete' means 'negation-complete' means that for any **\sigma** it proves **\sigma** or proves **~\sigma**. We didn't define **\omega**-consistent, I'll do so now: it means that if for each **n** the system proves **\phi(n) (where that n has a bar on it)**, then the system doesn't prove **~Ax:\phi(x)**. Of course this implies that the system is consistent, because an inconsistent system proves everything, so this version is weaker than the Gödel-Rosser version which drops the "**\omega**". But people still sometimes talk about this original version anyway, essentially just for historical reasons.
* Also, it isn't normal to state G2T in terms of formal systems as we did; but doing so loses nothing. **A** more standard statement with the same power (I won't explain what all the terminology means, though... avoiding having to do so was why I only gave the Postal version!): let **T** be a recursively axiomatised theory in some language **L**, suppose **T** interprets a structure **N** in the language of arithmetic, and suppose the induced theory extends **Q**. Then if **T** is consistent, **T** does not include **Con(T)**, where the latter is considered as a sentence in **N**.