Undecidable Problems in Topology

Uri Andrews

May 10, 2005

1 Abstract

The purpose of this paper is to elucidate many of the important undecidability problems in topology. First, the paper will introduce the notion of computability (recursivity) and recursive enumerability. Then, via some algebraic undecidable problems and some basic algebraic topology we will be able to show some topological undecidable problems, culminating in Novikov's theorem on the diffeomorphism problem of n-manifolds.

2 A Short History of Decidability

In 1900, David Hilbert made his famous address to the International Congress of Mathematicians. In that speech, he presented 10 problems that were meant to be examples of the most important problems to solve that would further our knowledge of various disciplines in math. In the printed address, this list was extended to 23 problems. Many of his problems were later proven to be unsolvable in all interesting cases. For instance, Gödel showed Hilbert's second problem unsolvable. That problem was to prove (within logic) that the axioms of logic are consistent. Hilbert's 10^{th} problem was to find an algorithm that will decide whether an inputted diophantine equation has a solution. Hilbert's 10^{th} problem was shown unsolvable by Yuri Matiyasevich in 1970. Some of the better known problems to make Hilbert's list are the continuum hypothesis, and Riemann's hypothesis. In the 30s, a rigorous definition of "algorithm" was given. There are numerous equivalent formulations including that of a turing machine. This gave Hilbert's problems a rigorous definition and they became accessible.

Hilbert believed strongly in the formalization of mathematics. "Hilbert's Programme can be described thus: There are two systems, nowadays called formal theories, S and T of mathematics. S consists of the finite, meaningful statements and methods of proof and T the transfinite, idealised such statements and methods. The goal is to show that, for any meaningful assertion ϕ, if T $\vdash \phi$, then S $\vdash \phi$." (Smorynski 1). Here $T \vdash \phi$ means T proves ϕ to be true. This then implies the decidability of everything since one can simply run through all the (recursively enumerable) theorems of S, and see if ϕ or $\neg\phi$ is on the list. Gödel in 1930 proved this undoable. In fact, he showed that any sufficiently complex theory (recursively enumerable) is incomplete. This then leads to a number of undecidability issues. In 1935 Alonzo Church gave the first explicit undecidable problem. In 1936, Alan Turing, who the Turing machine is named after, showed that the halting problem (to decide if a turing machine will stop on an input) is unsolvable in finitely many steps by any turing machine. We'll see an argument for this later.

In 1910, before Gödel's incompleteness theorems, Max Dehn wanted to extend Hilbert's ideas of algorithmic decidability to geometric problems. He proposed three problems in group theory, each with a geometric interpretation. These problems were the word problem, the conjugacy problem, and the isomorphism problem. The word problem is to find an algorithm to determine whether a given word in the generators of a finitely presented group is trivial in that group. The conjugacy problem is to find an algorithm that decides for two words u and v in a group G (finitely presented) whether or not they are conjugate. The isomorphism problem is to find an algorithm to determine whether two finite presentations give isomorphic groups. The geometric motivations are: to see if a loop in a topological space (CW-complex) is contractible, to see if whether two loops are 'freely homotopic', and to see if one can algorithmically distinguish spaces based on their fundamental groups. A CW-complex yields an easy way to represent its fundamental group, and applying the above algorithms correspond to those motivations. We also get the question as to whether it is algorithmically determinable whether a CW-complex is simply connected, ie: if a finite presentation of a group can be determined trivial or not.

Until 1947, the only undecidability results were in Logic. At that time, the first unsolvable problem of regular mathematics was shown. Emil Post and Andrei Markov showed unsolvable a slight modification of Dehn's problem, that of the word problem for semi-groups. In 1954-1958, Novikov, Boone, and Britton all, independently succeeded in extending this result to groups. We'll see more of the word problem later. In fact, we'll prove the semi-group result below.

In 1950, Julia Robinson and Martin Davis started work on Hilbert's 10th problem, which was finally solved in 1970 by Yuri Matiyasevich. Matiyasevich proved the result that all recursively enumerable sets were diophantine. This reduced Hilbert's 10th problem back to the halting problem.

Skipping a few years and moving up to a couple of geometric modern results. For n$\geq$3, there is no algorithm to determine whether a given knot f:$S^n \rightarrow R^{n+2}$ is trivial or not was shown in 1996 by Nabutovsky and Weinberger. More recently, they showed a way to relate computable sets to Riemannian metrics on a manifold M. Moreover, they related the difficulty of computation on those sets to the behavior of the diameter function on the space of Riemannian metrics on M. This inspires the question: How much can computability of sets really interact with purely geometric ideas?

3 Making precise the notion of decidability

There are three main things to present here: The notion of an r.e. set, the notion of a recursive set, and that they are non-equal.
To do this, we will need to define a Turing machine. You can picture a turing machine as a box with a tape running through it. On the tape, there are squares of data. There are finitely many symbols (including the empty-symbol) that may be on any given square. The machine may be in any of finitely many states. One state is specified as the stopping-state. There is no command after that state other than: stop! The value on the tape in the machine at that point can

be viewed as an output. The non-stopping commands say one of the following:
1)If you're in state Q and reading symbol S, then replace the symbol on that square by S_2, move into state Q_2 and scan the same square;
2)If you're in state Q and reading symbol S, move into state Q_2 then move one square to the right and scan that;
3)If you're in state Q and reading symbol S, move into state Q_2 then move one square to the left and scan that.
The machine then moves according the the next command. These tapes are technically finite in length, but when the machine moves to the end of the tape, we adjoin it with a new square with the empty-symbol in it.
Okay, so the natural question now is: Does this really represent what we consider to be a computation? The answer is a resounding: We think so. It's been equivalent to every other form of reasonable computation that's been suggested, so it seems to be the right thing to look at.

Definition 3.1 *We define a subset of* $\mathbf{N}$ *to be Recursively Enumerable if there is a Turing machine that stops iff it is inputted a tape that represents a member of this set.*

We like to restrict the symbols to: empty, 0, or 1. This makes the Computer-Scientists happy. Then, we just encode a natural number on a finite tape in its base-2 form. At first, one would think that the encoding chosen might change which sets were recursively enumerable (r.e. from here on), but all reasonable ways turn out to give an equivalent description.

Definition 3.2 *A set is called recursive if both it and its complement in* $\mathbf{N}$ *are r.e..*

Intuitively this just says that to figure out if x is a member of a recursive set S, then you just run the two machines describing S and $\mathbf{N} - \mathbf{S}$ simultaneously. One will stop in finite time, and then you know whether x is in S or not. We call a property algorithmically decidable or recursively decidable iff the set of elements satisfying the property is a recursive set. As turing machines are allowed to have any symbols as inputs, there's no real reason to restrict this definition only to the naturals. Indeed, given any set encoded by finite strings of symbols, we can make the same definitions. ie: it's r.e. iff there's a turing machine taking those finite strings of symbols as inputs that stops iff its input was in the set. Similarly, recursive means that it and its negation in this set are r.e.
The next important thing to see, which I suppose is intuitively obvious, is that there are r.e. sets that are not recursive. That means that there are r.e. sets whose negation is not r.e.

Proposition 3.3 *There is a recursively enumerable subset of* $\mathbf{N}$ *that is not recursive.*

Proof:
There are only countably many Turing machines on the natural numbers. This is because there are only finitely many states and finitely many symbols for any machine. A Turing Machine can be encoded by its set of commands, which take the form $q_i s_j s_k q_l$ for a move of the first type above, $q_i s_j R q_k$ for a move of the second type above, and $q_i s_j L q_k$ for a move of the third type. The first can be

read as: when in state q_i and reading s_j, write symbol s_k and move to state q_l. The second and third mean move to the Right or Left and switch to state q_l. We can then assign to each state q a natural number (avoiding 0,1,2, and 3). Make sure to define R to 2 and L to 3, the symbol 0 to 0 and 1 to 1. After all, choosing to only allow machines with symbols 0 and 1 doesn't limit us at all, but makes the encoding smoother. There are finitely many commands. Say there are n commands describing T. For a given machine T, we can take the set of n commands and just write them one after another. Call this word W. W depends on the order we write the commands, so we will be encoding each turing machine many times. Then define the Gödel number of T by

$$G(T) = \prod_{i=1}^{4n} p_i^{e_i}$$

where p_i is the i^{th} prime, and e_i is the number corresponding to the i^{th} letter of the word W. So, this gives us a way to, in a recursive way, index the turing machines by the naturals.
Keep in mind that there is a way to program a Turing machine to decode the natural number x. That machine just factors x by running through the values less than x and seeing which divides x. From that information, it can retrieve the e_i's. Then 2 means R, 3 means L, each 0 means the symbol 0 and each 1 means the symbol 1. Each other number encodes a distinct state. If x does encode a Turing machine, then from this it's easy for a machine to re-create the commands for the machine x. If the commands decoded don't make sense, ie: x doesn't code a Turing machine, then our de-coding machine assigns to x the machine that immediately stops on any input.
Now, we can create a Turing machine that stops on x iff the x^{th} Turing machine stops on input x. To write out the program for such a machine is a mess, but intuitively it just decodes x to determine what commands to run, then it runs those commands on the input x.
Then, we use an argument similar to Cantor's proof that the reals are uncountable. We need to show that it's complement is not r.e.. So, suppose otherwise: there is a turing machine T s.t. T stops on x iff the x^{th} turing machine does not stop on x. Well, we enumerated all the turing machines, so this must be machine n for some n. But then T stops on n iff the n^{th} turing machine does not stop on n. This is a contradiction. So, indeed this set is r.e., but it is not recursive.

Corollary 3.4 *The halting problem is undecidable. That is, given a turing machine and an input, we can't recursively decide if it stops.*

Proof: We can't decide if the n^{th} turing machine stops on the n^{th} input. The halting problem would give us a way to do that.

4 Decision Problems on Groups

On groups, there are three classic decision problems: the word problem, the triviality problem, and the conjugacy problem. These problems are unsolvable, meaning there is no recursive algorithm that decides them. The triviality problem is a sub-problem of the isomorphism problem. The triviality problem is to

determine whether a finite presentation describes the trivial group, whereas the isomorphism problem is to determine for any two finite presentations whether they represent isomorphic groups.
The ones that we will be particularly interested in are the word problem and the triviality problem. I will give the intuitive understanding behind why the word problem is undecidable first. To do so, I'll prove it for semi-groups, then I'll appeal to the algebraic similarity of semi-groups and groups.

Theorem 4.1 *(Markov-Post 1947) There is a semi-group which has an algorithmically unsolvable word problem.*

Proof:
Given a Turing machine T with symbols $S = \{s_1, s_2, ...s_n\}$ and states $Q = \{q_0, q_1..., q_m\}$, the stopping state being q_0. We can take a semi-group generated by $S \cup \{h, q\}$, where we'll use h to denote the end of the tape, and q to denote that the machine has finished. Then a word of the form h$\sigma q_i \tau$h where σ and τ are words in S describes the machine at one moment while it works. The machine is at state q_i and is reading the first symbol in τ. The rest of the tape is described by the rest of the word. Then, impose the relations on this generating set that says '$\alpha = \beta$ whenever α describes a position that our machine T moves to β from.' This can be written in finitely many relations as T has finitely many states and a T-move only effects 2 consecutive members on the tape. If we could say that for a word α that $\alpha = q$, then we would know that if α represents a state of T, then T stops from there. In particular, if we can say that for a word α representing an input-position of T, $\alpha = q$, then we know T stops from that input. So, a solution to the word-problem for the semi-group described yields a solution for the halting problem for T. Letting T be a turing machine defining a recursively enumerable set that is not recursive, we see that the halting problem for T is unsolvable. This is because the values T stops on is not recursive. QED

It's a fact that uses some non-trivial amount of algebra to reduce the problem for groups to that for semi-groups. So, I'll state without proof:

Theorem 4.2 *(Novikov-Boone-Britton, 1954-1958) There is a group U such that U has an algorithmically unsolvable word problem.*

For a full proof, see (Rotman, 1995 pp:430-447). I believe that the proof for semi-groups contains almost all of the intuition behind the proof, and the rest is simply encoding the above semi-group in a group.

We'll now set our sights on the triviality problem. Though the triviality problem is a good goal in its own right, we'll prove a far more general theorem stating that any Markov Property is undecidable over finite presentations of groups. A Markov Property is one that is satisfied by some finitely presented group, call it G_+, and there is a finitely presented group G_- such that G_- being a subgroup of H implies H does not have the property. This is a very general and powerful theorem, since many things are markov properties.
Being the trivial group is a markov property. We see this by letting G_+ be the trivial group (or technically it should be your favorite finite presentation of it), and letting G_- be any non-trivial finitely presented group. This shows many

other things algorithmically unsolvable as well. For instance, it is algorithmically unsolvable whether a finite presentation represents an abelian group. Here, any abelian group works for G_+ and any non-abelian group works for G_-.

Theorem 4.3 *(Adian and Rabin) For any Markov Property P, the problem of determining which finite presentations represent groups that have the property P is algorithmically unsolvable.*

Proof:
Fix P a markov property, G_+ and G_- showing P to be a markov property and U, a group with an unsolvable word problem. We will construct a sequence of groups H_w for each w a word in U. This will be done in such a way that H_w has the property P iff w is trivial in U. To do this, we'll use a technical lemma that is well in the range of algebra.

Lemma 4.4 *Let K be a group with a specified finite presentation: K=$< X \mid R >$. X=$\{x_1, x_2 \ldots x_n\}$ Fix a word w in the generators of K. Let S be the following set of relations:*

$$a^{-1}ba = c^{-1}b^{-1}cbc$$

$$a^{-2}b^{-1}aba^2 = c^{-2}b^{-1}cbc^2$$

$$a^{-3}[w, b]a^3 = c^{-3}bc^3$$

$$a^{-(3+i)}x_iba^{3+i} = c^{-(3+i)}bc^{3+i}, i = 1, 2, \ldots, n$$

where [w,b] is the commutator. Then, define

$$L_w =< \{a, b, c\} \cup X \mid S \cup R >$$

Then:
1)L_w is generated by two elements: b and ca^{-1}
2)If w=1 in K, then L_w is trivial.
3)If w≠1 in K, then K is embedded in L_w via the inclusion of generators.

Proof: This is well in the range of algebra, so I won't prove this here. For a proof, see (Miller pg. 14).
Now, we'll use this to generate the sequence of groups H_w as above. Let K=U$*G_-$, the coproduct of U with G_-. The presentation of this group K can be written easily as it is generated by the disjoint union of the generating sets of U and G_- and has relations the disjoint union of the relation-sets for U and G_-. The fact that the coproduct keeps us in the range of finitely presented groups is very important. Now, for any word w in U, consider w as a word in K and construct L_w as in the lemma. Then, let $H_w = L_w * G_+$. This is again a coproduct construction. So, all along, the constructions have been easy and could easily have been done by a machine inputted w alone and having the finite presentations of U, G_- and G_+ programmed in. We see that If w=1 in U, then w=1 in K, and L_w is trivial. That makes $H_w = G_+$, which has property P. If w≠1 in U, then w≠1 in K, and K embeds in L_w. Therefore, K embeds in H_w. But, G_- embeds in K. This yields G_- embeds in H_w, which implies H_w does not have the property P. So, a recursive way to decide whether H_w has the property P would solve the word problem for U, which is impossible. QED

Theorem 4.5 *For any recursively enumerable set A, we can effectively (recursively) get a sequence of finite presentations of groups G_k such that*

$$k \in A \Longleftrightarrow G_k =< e >$$

Proof:
Let's trace where this all came from. We got a way to encode a turing machine's stopping set in a semi-group, then a way to encode that in a group G. This gave the word problem for a group. A word in this group can encode the starting state of that original turing machine. Then from the word we created a new finite presentation of a group H_w so that this new group is trivial iff that word is trivial in G. For any turing machine T, and any input k, we can follow this path and get a finite presentation of a group $H_{w(k)}$ that is trivial iff T stops on the input k. QED.
This theorem intuitively just says that the triviality problem is 'as hard a computation as a turing machine can do', since any other computation can be reduced to this one. The following is now trivial, but important to state.

Theorem 4.6 *The isomorphism problem for finitely presented groups is unsolvable.*

Proof:
We can't solve whether or not a group is isomorphic to the trivial group. ie: The triviality problem is a sub-problem of the isomorphism problem.

5 Geometric Decision Problems

We will use extensively the undecidability of the word problem and isomorphism problem and discuss how we can get geometric meaning from this. Remember that the original motivation for asking this question was geometric in nature. We wanted to decide whether loops in topological spaces were or were not trivial up to homotopy. We clearly see that this is impossible, but indeed, there is more.

Theorem 5.1 *The problem of deciding for 2-dimensional CW-complexes which are simply connected is algorithmically unsolvable.*

Proof:
For a given finitely presented group G= <X|Y>, we can construct a 2-dimensional CW-complex with fundamental group=G. Start by taking a wedge of circles, one for each x$\in X$. Then attach 2-cells as prescribed by any given member of Y. The attaching map should follow the word y_i going once around the x^{th} loop if the letter is x, and going around it in the opposite direction if the letter is x^{-1}. This has G as the fundamental group.
Now, this means that a Turing way of determining which 2-dimensional CW-complexes are simply connected would give a Turing way of determining which finitely presented groups are trivial, which we know is undecidable.

Theorem 5.2 *The problem of deciding, for 2-dimensional CW-complexes which loops are contractible is algorithmically unsolvable.*

Proof: This amounts to the same as finding which words in the fundamental group are trivial. The proof of the previous theorem implies that an algorithmic solution to this problem would yield an algorithmic solution to the word problem for groups.

Theorem 5.3 *The problem of deciding for n-manifolds with n≥4 which are simply connected is algorithmically unsolvable.*

Proof:
Take the above-mentioned 2-dimensional CW-complex, and look at an ϵ-neighborhood of it embedded in R^n for n≥5. You get an object whose fundamental group is the same as the 2-complex. Moreover, the boundary of this object has the same fundamental group. How can we see this? The boundary of this neighborhood forms an S^{n-3} bundle over the original CW-complex. For n≥5, n-3≥2. So, S^{n-3} has a trivial fundamental group. The end of the long exact sequence of homotopy for this bundle then dictates that the boundary of this neighborhood should have the same fundamental group as the 2-complex.
This boundary is an (n-1)-dimensional manifold. Therefore, we have described a way of giving a k-manifold with fundamental group G for G finitely presented. Therefore, a solution to the problem of determining which k-manifolds are simply connected would give a solution to the triviality problem.
Similarly, the problem of determining which loops are contractible amounts to the word problem for groups, and thus is unsolvable for n-manifolds with n≥4.

Markov, working on a certain class of operations of finite presentations of groups, called Markov operations, found that if P' is the image of P under a Markov operation, then the n-manifolds corresponding to P, P' are homeomorphic. From this, he was able to find that a solution to the homeomorphism problem for manifolds would give a solution to the triviality problem for finitely presented groups. This yields the next theorem.

Theorem 5.4 *(Markov, 1958) The homeomorphism problem for n-manifolds is algorithmically unsolvable for n≥4.*

For a complete proof, look to (Haken, 1973 pp. 432-433) Now that we have that the homeomorphism problem is unsolvable, we can ask if the diffeomorphism problem is unsolvable. Indeed, Novikov proved a much stronger theorem. Note that Markov showed only that there is no algorithm deciding for a pair of manifolds whether they are homeomorphic. Novikov proved that for any fixed manifold M, there is no algorithm to decide whether another is diffeomorphic to it.

Theorem 5.5 *(Novikov) For each n-manifold, n≥5, it is algorithmically unsolvable for an n-manifold P whether or not $M\approx_{diff}P$ (M is diffeomorphic to P).*

I will present the idea of the proof for M=S^n. For a more detailed proof see (Nabutovsky 1995, pg. 86). The idea is simple, but the details are very difficult. The idea is this: First, one must prove the triviality problem unsolvable for the class of groups with trivial first and second homology groups. This is done by showing a way that from a given finite presentation of a group G, we can get a finite presentation of a group H such that H has trivial first and

second homology groups and H is trivial iff G is trivial. This then reduces the problem to the triviality problem for groups. Then, For any finitely presented group with trivial first and second homology groups, we find a homology sphere P (meaning it has the same homology groups as S^n) embedded in R^{n+1} so that P has fundamental group G. Moreover, we can do this in a way so that P is diffeomorphic to S^n iff P has trivial fundamental group.
In fact, for P a homology n-sphere smoothly embedded in R^{n+1}, P is diffeomorphic to S^n iff P is simply connected. This follows from the following two theorems that are very involved and I cannot write up the proofs here because they are too long, and at least in the first case, too complex for me to truly understand.

Theorem 5.6 *(The Generalized Poincare Conjecture) (Smale and Stallings) For all n$\geq$5 a homology n-sphere with trivial fundamental group is homeomorphic to S^n*

Theorem 5.7 *Let P be an n-manifold smoothly embedded in R^{n+1} that is a homology n-sphere with trivial fundamental group, then P is diffeomorphic to S^n iff P is homeomorphic to S^n.*

Note, the proof of Novikov's theorem was once again completed by showing that algorithmically solving this problem would solve the triviality problem for groups. In so doing, we showed that the triviality problem for a group G reduced to the diffeomorphism problem of a manifold P. Therefore, the triviality problem for groups reduced to this problem for manifolds.

Theorem 5.8 *For any recursively enumerable set A, there is a way to find finite presentations of groups G_k and manifolds P_k such that*

$$k \in A \Longleftrightarrow G_k =< e > \Longleftrightarrow P_k \approx_{diff} S^n$$

It seems to me that all undecidability results have arisen from the fact that not all r.e. sets are recursive. This was encoded with no loss of generality in the semi-group word problem. I think very little loss of generality was lost in going from there to groups. So, it seems that the strength of undecidability results in geometry and topology relies largely or entirely on the connections to algebra. Indeed, it is directly through the algebraic topological invariants that we have been able to prove undecidability of some topological problems.

[Haken, 1973]
W. Haken, Connections between topological and group theoretical decision problems, In: Word problems: decision problems and the Burnside problem in group theory, Eds. W.W. Boone, F.B. Cannonito, and R.C. Lyndon, Studies in Logic and the Foundations of Mathematics, vol. 71, North-Holland, Amsterdam, London, 1973, 427-441

[Miller, 1992]
C.F. Miller, Decision Problems for Groups: Survey and Reflections, in Algorithms and Classification in Combinatorial Group Theory, MSRI publications No. 23, Springer-Verlag, 1992 1-59

[Nabutovsky, 1995]
A. Nabutovsky, Einstein Structures: Existence Versus Uniqueness, Geometric and Functional Analysis, 5, No. 1, (1995), 76-91

[Rotman, 1995]
J. J. Rotman, An Introduction to the Theory of Groups, Fourth Edition, Springer-Verlag, New York, 1995

[Smorynski, 1991]
C. Smorynski, Logical Number Theory, Springer-Verlag, Berlin, Germany, 1991

[Smorynsti, 1985]
C. Smorynski, Self-Reference and Modal Logic, Springer-Verlag, New York, 1985

[Soare, 2004]
R. I. Soare, Computability Theory and Differential Geometry, http://www.cs.uchicago.edu/∼soare