More Undecidable Problems

We prove that determining whether a PDA accepts all strings is undecidable, as is Post's Correspondence Problem (PCP).

Totality for PDAs is Undecidable

We now prove our earlier claim that totality for context-free languages is undecidable.

We need the fact that a PDA can decide whether one configuration follows from another:

Fact. *Given a deterministic TM M, one can build a deterministic PDA that accepts the following language: $\{{c_1}^R \# c_2 : c_1, c_2$ are configurations of M such that c_2 follows from $c_1\}$.*

Proof of Fact

The PDA starts by pushing c_1 onto stack. Then, as it reads c_2, it compares the two. They are almost identical except around state symbol. So PDA reads symbol, pops symbol, checks they match, and discards; until it hits first state symbol.

Then PDA checks whether implied transition is correct for M (M is hard-coded into the PDA). If string passes test around state, then PDA reads and pops to the end, checking the symbols match as before.

Totality for PDAs is Undecidable

Theorem. *It is undecidable whether a PDA accepts every string.*

Proof. We reduce from A_{tm} by designing a PDA that accepts all strings that are *not* “computation strings”. The key is that:

> *nondeterminism simplifies checking that the string is invalid since one only has to find one flaw.*

Altered Computation Strings

For $\langle M, w\rangle$, define language of altered computation strings:

$$N_{M,w} = \{\, c_1 \# c_2^R \# c_3 \# c_4^R \# \ldots \# c_k \,\}$$

where c_1 is start configuration, c_k is accepting configuration, and each c_i follows from c_{i-1}. That is, the configurations are written down with every second one reversed. We will build a PDA $P_{M,w}$ for the *complement* of $N_{M,w}$.

Finding Flaws

Recall that to check whether string is computation string for M on w, one checks three things:

- the first configuration is correct;
- the last configurations is accepting; and
- every configuration follows from previous one.

This PDA will guess which flaw to find.

Finding Flaws

Checking the first or last configuration is easy.

If PDA guesses that some pair of configurations conflict, then it uses nondeterminism to guess which pair, and the above fact to do the checking.

The PDA accepts whenever it finds a problem; thus it accepts $\overline{N_{M,w}}$.

Punch Line

Now: if M does not accept w, then there is no computation string. That is, $N_{M,w}$ is empty. Hence $P_{M,w}$ accepts every string. If M does accept w, then there is a computation string and so $P_{M,w}$ does not accept every string.

That is, being able to answer the totality question for PDAs would enable one to answer the acceptance question for TMs. Since the latter is undecidable, so is the former.

Post's Correspondence Problem

We have a set of tiles where each ***tile*** has a top string w_i and a bottom string x_i. A ***solution*** is to choose a subset of the tiles, and order them, such that string reading across the top is the same as string reading across the bottom. This is ***Post's Correspondence Problem.***

Example

Suppose the tiles are $(\texttt{0}, \texttt{101})$, $(\texttt{10101}, \texttt{10})$, $(\texttt{11}, \texttt{00000})$ and $(\texttt{1}, \texttt{01})$. Then one solution is:

$$\boxed{\begin{array}{c}10101\\10\end{array}} \qquad \boxed{\begin{array}{c}0\\101\end{array}} \qquad \boxed{\begin{array}{c}1\\01\end{array}}$$

MPCP is Undecidable

Define ***Modified PCP*** (MPCP) as PCP problem where one tile is designated the start tile.

A_{tm} reduces to MPCP. Hence MPCP is undecidable.

Rough proof sketch.

Consider a $\langle M, w\rangle$ that might be in A_{tm}. The idea is:

> *design a set of tiles such that a solution is a computation string for M on w.*

That is, reading across top/bottom, one sees sequence of *configurations* leading to acceptance.

Building the Tile Set

Recall that configuration of TM is written as $t_L\, S\, t_R$ where t_L is the used tape to the left of the head, S the current state, and t_R the used tape to the right of the head. So the start configuration is $q_0 w$.

The solution will be such that, for the most part, *the bottom of the solution is one configuration ahead of the top.*

The Tiles

These are five kinds of tiles:

1) The start tile is $(\texttt{\$}, \texttt{\$}q_0w\texttt{\$})$ where $\texttt{\$}$ is a special symbol.

2) For the tiles that describe the action of M, iterates through the transitions: if q is a state and $\texttt{b}$ a tape symbol, then

- if $\delta(q, \texttt{b}) = (r, \texttt{c}, \texttt{L})$, then add tiles $(\texttt{a}q\texttt{b}, r\texttt{ac})$ for all symbols $\texttt{a}$;
- if $\delta(q, \texttt{b}) = (r, \texttt{c}, \texttt{R})$, then add tile $(q\texttt{b}, \texttt{c}r)$.

More Tiles

3) To deal with M moving to a cell not previously visited, add tiles $(\mathtt{\$}, \Delta\mathtt{\$})$ and $(\mathtt{\$}, \mathtt{\$}\Delta)$.

4) To allow for copying down of rest of the configuration, add tiles $(\mathtt{a}, \mathtt{a})$ for every symbol.

5) Finally, add tiles that allow top row to catch up provided that bottom row has reached h_a. For every tape symbol $\mathtt{b}$, add tiles $(\mathtt{b}h_a, h_a)$, $(h_a\mathtt{b}, h_a)$ and $(h_a\mathtt{\$}, \varepsilon)$.

Apart from start tile, there are unlimited copies of each tile.

The Conclusion

The fundamental point is that there is a solution to the resultant MPCP exactly when the TM accepts the input string. Since the latter is undecidable, so is the former.

(We omit a formal proof that tiles have desired property.)

One can also show that MPCP reduces to PCP: this is more a trick (see exercises).

Summary

Reductions from the acceptance problem A_{tm} show that determining whether a PDA accepts all strings is undecidable, as is Post's Correspondence Problem (PCP).