Post's Correspondence Problem

CS154
Chris Pollett
May 1, 2006.

Outline

- More on Post Correspondence Problem
- Mapping Reducibility

Undecidability of PCP

Theorem. *PCP* is undecidable.

Proof. We show via computation histories that if *PCP* is decidable so is A_{TM}, thus giving a contradiction. Given an input <*M, w*> for A_{TM} we will construct an instance of *PCP* which will have a match iff *M* accepts *w*. Further, the string of the match which be an accepting computation history of *M* on *w*.

To simplify our problem we will consider the following modifications to A_{TM} and *PCP*:

1. We assume *M* on input w never tries to move left off the tape.
2. If $w = \varepsilon$, we use the string '_' in place of *w* in the construction.
3. We modify *PCP* to require that a match starts with the first domino.

…

More Undecidability of PCP

Call the instance of *PCP* that we are building *P*.

- Put [#| $\#q_0w_1..w_n\#$] as the first domino of the *PCP* instance. Playing this domino first will force the string to look like an initial configuration of *M* on *w*.

Next we add dominos to handle the part of the configuration where that the TM might change

2. For every *a*,*b* in the tape alphabet, and every states *q*, *r* of *M* where $q \neq q_{reject}$, if $\delta(q, a) = (r, b, R)$, we put [*qa* | *br*] into *P*.
3. For every a,b,c in the tape alphabet and every states *q, r* in *M* where $q \neq q_{reject}$, if $\delta(q, a) = (r, b, L)$, we put [*cqa* | *rcb*] into *P*.

Next we add dominos to copy the unchanged parts of configurations and to copy the end of configuration markers

4. For every *a* in the tape alphabet, we put [*a* | *a*] into *P*.
5. Put [#|#] and [#|_#] into *P*. (The second is to handle if the simulate where the size of a configuration grows).

Lastly, we add dominos, so that once we get to an accept state we have a sequence of configurations with an ever smaller number of squares so we can "catch up" the top row with the bottom row:

6. For every tape symbol *a* we have dominos [aq_{accept} | q_{accept}] and [$q_{accept}a$ | q_{accept}] and to complete the match we have [$q_{accept}\#\#$ | $q_{accept}\#$]

Examples of these Dominos

- Suppose M on input w=0100 starts in state q_0.
- Suppose further in this state reading a 0 it goes into state q_7 writes a 2 and moves right.
- The first 5 kinds of dominos could be played to get the following partial configuration history:

$\#\ q_0\ 0\ 1\ 0\ 0\ \#$

$\#\ q_0\ 0\ 1\ 0\ 0\ \#\ 2\ q_7\ 1\ 0\ 0\ \#$

Eliminating the First Domino Condition

- Suppose we have an instance
$$P = \{[t_1 | b_1], [t_2 | b_2], .., [t_k | b_k]\}$$
of *PCP* where we'd like the first domino to be played and we like to make it into a "real" instance P^* of *PCP* without this condition.
- Let * and $ be new symbols not appearing in the t_i's and b_i's. Given a string $u = u_1..u_n$. Define $*u = *u_1* u_2*...*u_n$, $*u* = *u_1* u_2*...*u_n*$, and $u* = u_1* u_2*...*u_n*$.
- Then $P^* = \{[*t_1 | *b_{1*}], [*t_2 | b_2*] .., [*t_k | b_k*],$ $[*\$,\$]\}$ will have a match in the usual *PCP* iff *P* had a match in the modified PCP.

End of the PCP reduction

- So assume we had a decision procedure *D* for *PCP*.
- Consider the machine:

  S="On input $<M, w>$:

  1. Construct an instance $<P^*>$ of *PCP* following the construction of the last few slides.
  2. Run *D* on $<P^*>$.
  3. If *D* accepts, **accept**; else if *D* rejects, **reject**."
- This machine is a decision procedure for A_{TM}, which we know cannot exist.
- Therefore, the assumption *D* exists must be false.
- Therefore, *PCP* is undecidable.

Mapping Reducibility

- So far we have used the notion of reducibility to show many problems are undecidable.
- To study reduction more formally we are now going to work towards a more precise definition of reducibility called **mapping reducibility.**
- We need one definition before we define mapping reducibility

Computable Functions

Definition. A function f: $\Sigma^* \rightarrow \Sigma^*$ is a **computable function** if some Turing machine M, on every input w, halts with just $f(w)$ on its tape.

Example. The function $<m, n> \rightarrow m + n$ is computable.

Example. The function $<M> \rightarrow <M'>$ which if $<M>$ is the encoding of a TM maps this encoding to a new encoding of a TM for the same language but which does not try to move left off the tape.

Formal Definition of Mapping Reducibility

Definition. The language A is **mapping reducible** to the language B, written $A \leq_m B$, if there is a computable function $f: \Sigma^* \rightarrow \Sigma^*$, where for every w,

$$w \in A \text{ iff } f(w) \in B.$$

The function f is called the reduction of A to B.