Lecture #13: Simulations and Simple Turing Machine Variants

Key Concepts

Recall that a ***simulation*** is something that can be presented to relate the power of two models of computation. In order to show that the machines described by a ***second*** model of computation are (in some sense) at least as powerful or efficient as the machines described by a ***first*** model of computation, we generally do the following:

(a) Consider an arbitrary machine M, of the type described by the ***first*** model of computation.

(b) Use M to define another machine $\widehat{M}$, of the type described by the ***second*** model of computation.

(c) Prove that $\widehat{M}$ solves the same problem as M.

Step (b) can be expanded as follows:

- Begin by describing — as clearly and completely as you can — how a configuration of the first machine, M can be represented when a machine, $\widehat{M}$, of the second type is being used.

  When $\widehat{M}$ is a kind of Turing machine this will, generally, including describing how many tapes $\widehat{M}$ has, and what they are used for. It might also include describing $\widehat{M}$'s tape alphabet, $\widehat{\Gamma}$.

This will, ideally, make it significantly easier, to describe the following:

- ***Initialization:*** Let Σ be the input alphabet for M — so that it must be the input alphabet for $\widehat{M}$, as well. Describe how $\widehat{M}$ begins with its initial configuration for an input string $\omega \in \Sigma^\star$ and moves to a representation of M's initial configuration for the same input string.

- ***Step-by-Step Simulation:***

  For configurations $\mathcal{C}_1$ and $\mathcal{C}_2$ of M, such that M moves from configuration $\mathcal{C}_1$ to configuration $\mathcal{C}_2$ using a single step, describe how $\widehat{M}$ moves from a representation of $\mathcal{C}_1$ to a representation of $\mathcal{C}_2$.

- ***Cleanup:***

  Describe anything more that $\mathcal{M}$ must do, to end its computation, after simulating M's final step.

  Note: For Turing machines that recognize (or decide) languages, there might not be anything, here, to describe.

If $\widehat{M}$ is a Turing machine then it is possible that $\widehat{M}$s's transition function, $\widehat{\delta}$, has been described in detail once these steps have been completed. Alternatively, if the simulation is more complex, then enough information has been given so that it *could* be completed, if you had time.

If it is not obvious then a ***proof of correctness*** of the simulation should be given. When M and $\widehat{M}$ are types of Turing machines, that each have an input alphabet Σ, then this should imply the following: For every string $\omega \in \Sigma^\star$ and for every non-negative integer t, the following property is satisfied: If the execution of M on input ω uses at least t steps, and M is in configuration $\mathcal{C}$ after the first t steps of its execution on input ω, then $\widehat{M}$'s execution on input ω includes a simulation of at least t steps of M and — after the simulation of t steps of M — $\widehat{M}$ is in a configuration that gives a representation of $\mathcal{C}$.