Jeffrey Shallit  
School of Computer Science  
University of Waterloo  
shallit@uwaterloo.ca  
https://cs.uwaterloo.ca/~shallit
Recall that we saw how to encode things like DFA’s into a binary string.

The properties we need are

(a) Every DFA has an encoding (it might have more than one)

(b) You can decide if something is a valid encoding of a DFA with a TM

(c) You can uniquely decode an encoding to get the DFA it represents

(d) Given two encodings that are concatenated together, you can tell where one encoding ends and the other begins.
Now we want to be able to encode a DTM into a binary string.

The encoding should have the same four properties.

Remember that a TM can have an arbitrary input alphabet $\Sigma$ and an arbitrary tape alphabet $\Gamma$.

We will assume that both these alphabets are subsets of a universal alphabet $\mathcal{U} = \{a_1, a_2, \ldots\}$ like we did before.
What are the parts of a DTM?

They are $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{acc}}, q_{\text{rej}})$, where

- $Q$ is a finite set of states;
- $\Sigma$ is the finite input alphabet;
- $\Gamma$ is the finite tape alphabet (and $\Sigma \subseteq \Gamma$);
- $q_0$ is the initial state;
- $q_{\text{acc}}$ is the accepting state;
- $q_{\text{rej}}$ is the rejecting state.
- $\delta$ is the transition function.
The domain of $\delta$ is $(Q - \{q_{\text{acc}}, q_{\text{rej}}\}) \times \Gamma$ and the range of $\delta$ is $Q \times \Gamma \times \{←, →\}$.

We can encode each of these pieces.

We can assume the states are numbered $q_0, q_1, \ldots, q_n$, where $q_{n-1}$ is the accept state and $q_n$ is the reject state.

Let us define $e(q_i) = 10^{i+1}$, $e(a_i) = 10^i$, and $e(←) = 10$, $e(→) = 100$. 
Encoding a DTM

For example, to encode $\Sigma = \{ a_{i_1}, \ldots, a_{i_r} \}$ we can write it as a string

$$[e(a_{i_1})\#e(a_{i_2})\# \cdots \#e(a_{i_r})]$$

over the alphabet $\{0, 1, [, ], \#\}$ and then re-encoding this into binary as we did before.

To encode $\delta$, we can do the following: each move looks like

$\delta(q_i, a_k) = (q_j, a_\ell, D)$, where $D$ is a direction.

We can encode this move by first writing it as a string of the form

$$[e(i)\#e(k)\#e(j)\#e(\ell)\#e(D)]$$

over the alphabet $\{0, 1, [, ], \#\}$ and then re-encoding this into binary as we did before.

Then to encode all of $\delta$ we simply concatenate these encodings together. This gives $e(\delta)$. 
Finally, to encode $M$ we can use, where $n$ is the number of states:

$$e(n) \ e(\Sigma) \ e(\Gamma) \ e(\delta).$$

This has all the properties we want.

The only one that is slightly hard to check is that $e(\delta)$ is a valid encoding of a transition function.

What could go wrong?
Encoding a DTM

Well, it could have transitions on symbols not in $\Gamma$, so we would need to check that every symbol listed is actually in $\Gamma$.

It could have transitions to or from states that are not in $Q$, so we would have to check that all state numbers are between 0 and $n$.

It could have multiple transitions on the same state-input pair, which would make it nondeterministic, so we would have to check that, too.

I claim all these checks can be made with a TM.

(Furious hand-waving as justification...