Naïve smooth relaxation friendly Turing machines

Adrian K. Xu, supervised by Daniel Murfet

January 2021

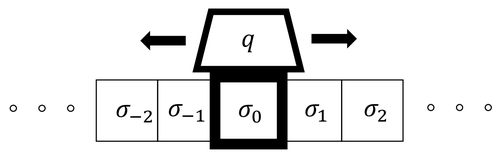
In [CM19], Clift and Murfet arrive at a notion of the "derivative of a Turing machine (TM)", built on work surrounding Jean-Yves Girard's linear logic, and emerging from which is the idea of a *smooth Turing machine*.

Turing machines

Recall: the TM model comprises an infinite tape with alphabet Σ, and a tape head with state in Q. In each transition, the tape head reads the current symbol, write a new symbol, updates its state and moves left, right, or stays put, as specified by the transition function $\delta: \Sigma \times Q \to \Sigma \times Q \times \{L, S, R\}$.

We restrict the space of classical tape configurations, $\Sigma_{\sqcup}^{\mathbb{Z}}$, to those containing a finite number of non-zero entries, and encode the tape head position via the index shift.

Every TM computes a function $F^t\colon \Sigma_{\sqcup}^{\mathbb{Z}} \to \Sigma_{\sqcup}^{\mathbb{Z}}$ by running for t-steps, and a partial function $F: \Sigma_{\sqcup}^{\mathbb{Z}} \hookrightarrow \Sigma_{\sqcup}^{\mathbb{Z}}$ defined for inputs on which it halts (reaches an idle state).



Smooth relaxation

A *smooth relaxation* is obtained by allowing uncertainties in the input (therefore output) and propagating the uncertainty in some way to obtain a smooth function $\Delta F\colon (\Delta\Sigma)_{\sqcup}^{\mathbb{Z}} \hookrightarrow (\Delta\Sigma)_{\sqcup}^{\mathbb{Z}}$, where $\Delta\Sigma$ is the set of probability distributions over Σ, or, geometrically, the standard Σ-simplex, $\{ \sum_{\sigma\in\Sigma} \alpha_\sigma \cdot \boldsymbol{\sigma} \mid \alpha_\sigma \in \mathbb{R}_{\geq 0}, \sum \alpha_\sigma = 1 \} \subset \mathbb{R}\Sigma$.

Intuitively, we may identify the derivative with the ratio between output and input uncertainty; however, standard probability does not work here, as the output uncertainty would be dependent only on the function F and not on the internal design of the machine.

Instead, we propagate the uncertainty with the assumption that, after each transition, the machine state, each tape entry, the tape head movement direction and write symbol are all independent random variables—this we call the *naïve Bayesian observer*, and leads to the naïve smooth relaxation of a Turing machine to which we refer in what follows.

Motivation

The modern arsenal of differential methods in modern machine learning relies on equipping the space of models with a smooth manifold structure. In order to port these methods to the general problem of program synthesis, we must endow the entire space of computable functions with such a structure; [CM19] and [CMW21] are key steps towards this vision.

However, if the (naïve Bayesian) smooth Turing machine is to be thought of as a notion fundamental to the theory of computation, then one would expect it to be independent of the particular model of computation adopted. In particular, constructions given on multi-tape machines (which, for convenience, they often are) in contexts where properties of the smooth relaxation are of interest, should admit equivalent constructions on single tape machines which respect those properties. Similarly, smooth universal Turing machines (which simulate any other TM given a suitable encoding) should propagate uncertainty through their simulations in a manner equivalent to that of the simulated machines. It is this issue that we proceed to clarify and verify.

Formalities

Let SIM be a single tape with alphabet Σ_{SIM}, associated functions F_{SIM}^t, $MULTI$ an n-tape TM with states Q, alphabet Σ, associated functions F_{MULTI}^t.

For simplicity, we assume Σ_{SIM} contains Q and Σ.

Definition A (Simulation). SIM is said to *simulate* $MULTI$ iff there exists a simulation timer $T_{SIM}\colon \mathbb{Z}_{\geq 0} \to \mathbb{Z}_{\geq 0}$, TMs $\{ GET_i \}_{i=1}^n$ and $INIT$ with associated partial functions $\left\{ F_{GET_i}\colon (\Sigma_{SIM})_{\sqcup}^{\mathbb{Z}} \hookrightarrow (\Sigma_{SIM})_{\sqcup}^{\mathbb{Z}} \right\}_{i=1}^n$ and $F_{INIT}\colon (\Sigma_{SIM})_{\sqcup}^{\mathbb{Z}} \hookrightarrow (\Sigma_{SIM})_{\sqcup}^{\mathbb{Z}}$ defined whenever they halt, such that:

1. $x_{SIM} \coloneqq F_{INIT}(x)$ is defined for all $x \in \Sigma_{\sqcup}^{\mathbb{Z}}$, and for all such x_{SIM}, F_{GET_i} is defined on $F_{SIM}^{T_{SIM}(t)}(x_{SIM})$ and maps into $\Sigma_{\sqcup}^{\mathbb{Z}}$ for $i = 1, \dots, n$ and $t \geq 0$
2. $F_{GET_i}\left(F_{SIM}^{T_{SIM}(t)}\left(F_{INIT}(x)\right)\right) = \Pi_i F_{MULTI}^t(x)$ for $i = 1, \dots, n$ and $t \in \mathbb{Z}_{\geq 0}$

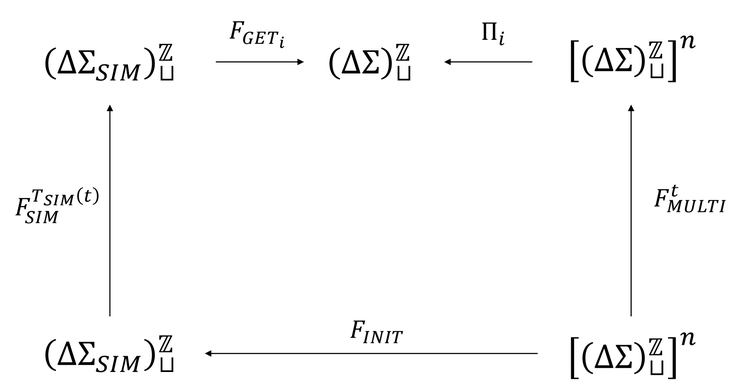
Definition B (Smooth rel. preserving simulation). Extend the alphabets in the above to their standard simplexes. The halt criterion for GET_i and $INIT$ needs modification.

It is a quirk of our particular choice of smooth relaxation that machines which halt do not necessarily have convergent smooth relaxations. In general, given a machine with associated smooth functions $\Delta \mathrm{F}^{\mathrm{t}}\colon \Delta\Sigma_{\sqcup}^{\mathbb{Z}} \to \Delta\Sigma_{\sqcup}^{\mathbb{Z}}$, one can define $\Delta F\colon \Delta\Sigma_{\sqcup}^{\mathbb{Z}} \to \Delta\Sigma_{\sqcup}^{\mathbb{Z}}$ sending

$$x \longmapsto \lim_{t\to\infty} \Delta \mathrm{F}^{\mathrm{t}}(x)$$

where the limit receives some sensible definition. This notion of stability is a stronger criteria than the requirement that the machine halt in the classical sense, and we use it to amend the definitions of F_{GET_i} and F_{INIT} in the smooth case.

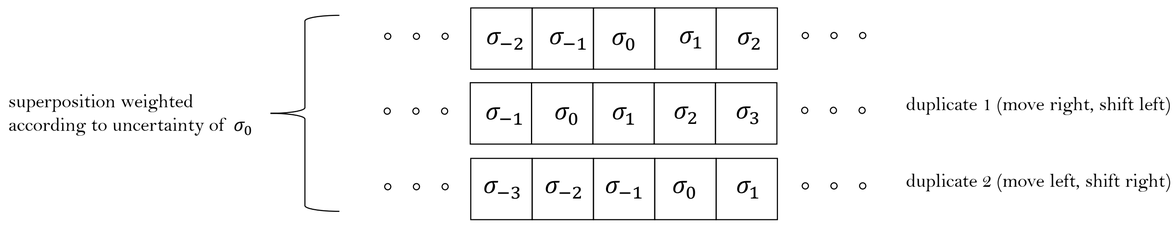
Our definition yields the following commutative diagram:



Theorem: For every machine $MULTI$, there indeed exists a machine SIM as in Definition B.

Of course, the classical case of Definition A is well known to many a theoretical computer science student. On the other hand, the smooth relaxation case of Definition B is novel. The question of its existence arose a couple weeks back from the time of writing in discussions with the authors of [CM19]. A working construction must surmount the following apparent engineering hurdle: that any such simulator must never have ambiguity in the direction the tape head moves. Why? Read on.

In a classical TM, the tape head reads the current symbol, writes a symbol to the tape and moves either left or right (or stays in some definitions). In a smooth TM, one should imagine the tape (with the new symbol written) being duplicated twice, one copy shifted left and the other copy shifted right. The resultant tape configuration is the superposition of the three copies, weighted according to the uncertainty in the symbol that was read (propagating to uncertainty in the move direction).

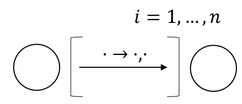


The reason the usual constructions fail to be smooth relaxation preserving is because the effect of this superposition is essentially to "smudge" every entry on the tape—when the simulation is all on one tape, this means ambiguity in one simulated tape will "contaminate" all the other simulated tapes.

To give a full working construction is involved and at times tedious, so here we merely illustrate a couple of the key ingredients.

It is convenient to introduce some syntactic sugar for notating collections of state paths bifurcating from a shared source state and converging at a shared target state, parametrised by the write symbol α of the opening wedge. The transitions appearing on and between the wedges are understood to be cloned, one path for each value of the parameter, with all appearances of the parameter in the transitions replaced with its value for that path.

Moreover, we notate transitions that appear in a fixed number of (possibly parametrised) sequential repeats via:

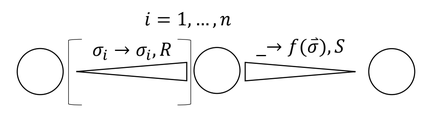


Loading uncertainties into state

Suppose we have a sequence of n adjacent symbols with uncertainty, their distributions given as vectors in the standard Σ-simplex, $s_i = \sum_{\sigma\in\Sigma} p_i(\sigma) \cdot \sigma$ where $p_i(\sigma)$ is the probability the i-th symbol reads σ. We want to write in the next tape entry over some function of this sequence, $f(\sigma_1, \dots, \sigma_n)$. One can think of the probabilistic extension of this function as a linear map

$$\Delta f\colon \Delta\Sigma^n \subset \mathbb{R}\Sigma \otimes \cdots \otimes \mathbb{R}\Sigma \to \Delta\Sigma$$

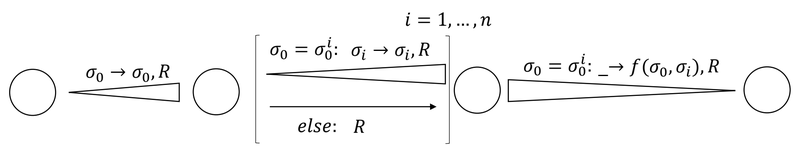
sending the basis vectors in the tensor product to the corresponding basis vectors in the destination space. The input vector can be "loaded" into the state of the TM via the following construction; the opening wedges correspond to tensor product operations to obtain the desired input distribution, and the final write instruction corresponds to the application of the above map. Note the final wedge closes all of the opening wedges. (That is, all the nested paths converge.)



Handling conditional uncertainties

Now suppose that we have $n+1$ adjacent symbols, where the first symbol $s_0 = \sum_{k=1}^n p_0(\sigma_0^k) \cdot \sigma_0^k$ is a distribution over n possible symbols, say $\Sigma_0 = \{\sigma_0^1, \dots, \sigma_0^n\} \subset \Sigma$, and the next n symbols are to be thought of as distributions *conditioned on* the value of the first symbol. f is now a two input function, the distribution of the second input being conditioned on the first. The input distribution to Δf is no longer computed via iterated tensor products; we must map each basis vector of $\mathbb{R}\Sigma_0$ individually to $\Delta(\Sigma_0 \times \Sigma)$ according to $\sigma \mapsto \sigma \otimes s_i = \sum_{\sigma'\in\Sigma} p_i(\sigma') \cdot (\sigma \otimes \sigma')$.

The following construction achieves this. Here we prefix a transition to refer to a particular state path. It is this kind of construction that is used to achieve the superposition operation described above, whilst circumventing the problems associated with tape head movement ambiguity. After the initial opening wedge, each subsequent wedge corresponds to the bifurcation of a particular path and equivalently to an application of the map just described.



Notes on the actual construction

The actual construction employs the above patterns extensively. As is often done in the classical case, the representations of the simulated tapes are interleaved on the single tape; however, an extra interleaved sequence is added, to store the current simulated state, and write the next state, write symbols and move directions. The simulated tape representations are spaced out with blank entries on either side of each symbol to allow the two duplicates described above to be "staged" before the technique described just above is used to compute the superposition.

Alternative design for universal Turing machine

The same ideas can be used to construct a universal Turing machine which is smooth relaxation preserving in the sense of Definition B. The resultant update rules when running codes with uncertainty are an (arguably) natural generalisation of those for the smooth relaxation of classical codes, with the transition function yielding the probabilistic extension $\Delta\delta\colon \Sigma \times Q \to \Delta\Sigma \times \Delta Q \times \Delta\{L, S, R\}$.

> *Dear Theorem,*
>
> *Lest you have an existential crisis, be reminded that you are (beyond the opening remarks) what Dan, my supervisor, (generously, and to whom I owe immense gratitude) likened to the "virtuosity" of a super Mario speed run.*
>
> *That is, you are among the many things in math, life and beyond which can and should be done simply for the hell of it.*
>
> *Regards,*
> *Your Maker*

References

[CM19] James Clift and Daniel Murfet. Derivatives of Turing machines in Linear Logic. 2019.
[CMW21] James Clift, Daniel Murfet and James Wallbridge. Geometry of Program Synthesis. 2021.