Learning finite state environments without a reset

Say we start in state q and perform action sequence h. Define

- $obs(q, h)$ = sequence of observations seen.
- qh = final state.

We say that h is a *distinguishing sequence* if

$$obs(q, h) = obs(q', h) \;\Rightarrow\; q = q'$$

We say that h is a *homing sequence* if

$$obs(q, h) = obs(q', h) \;\Rightarrow\; qh = q'h$$

Does every DFA have a distinguishing sequence?

Does every DFA have a homing sequence?

Every DFA has a homing sequence

Claim: every n-state DFA has a homing sequence of length $O(n^2)$.

Proof:

- Start with $h = \lambda$.
- Partition states based on observation sequence.
- If there exist q, q' in same partition where $qh \neq q'h$,
  - Pick A, $|A| \leq n$, such that $obs(qh, A) \neq obs(q'h, A)$.
  - $h \leftarrow hA$.
- Each addition to h increases number of partitions by at least 1.

Can we hope to get $O(n)$? $o(n^2)$?

Learning if we have a homing sequence

- Run h and record the observation σ.
- Instantiate a new copy of Learn-DFA, indexed by the observation seen.
- When you need to reset, run h instead.
  - If you see a new observation, start up a new copy of the algorithm.
  - If you see an old observation, go to the associated copy.

There will be at most n algorithms running (interleaved) in total and every time through we make one unit of progress on one of them. So, just blows up running time by factor of n.

Learn homing sequence as we learn DFA

Idea (high level):

- Start with $h = \lambda$.
- Run approach from last slide.
- If it fails, use this to improve h.

Two failure modes:

1. Some copy discovers inconsistent behavior.
2. Some copy grows too many states.

What if some copy has inconsistent behavior?

LearnDFA$_\sigma$ finds that two runs of action sequence A have produced two different outputs.

$\Rightarrow$ A is a distinguishing sequence for two states that were in the same partition σ under h.

$\Rightarrow$ hA is a better homing sequence.

For better performance, build “homing tree” (aka “adaptive homing sequence”). I.e., run h, then only do A if we saw σ. This way we don’t need to restart all LearnDFAs, just LearnDFA$_\sigma$.

What if some copy grows a model with $> n$ states?