the halting problem paradox

Nicholas Swenson
dart200@gmail.com

```
halting function h(m) -> {
  true  : iff m halts
  false : otherwise
}
```

Let A: set of all describable programs

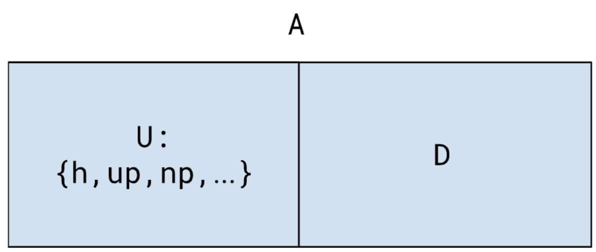
Let U: subset of A that contains programs undecidable by h,
cannot be mapped via function h

```
up = () -> h(up) && loop_forever()
  h(up) = ???

np = () -> !h(np) && loop_forever()
  h(np) = true and h(np) = false ???
```

∴ h(m) is also undecidable

Let D: subset of A that contains programs decidable by h,
can be mapped via function h



Presume U does not exist

We're left with D that does exist, which now should be decidable by function hd(m), the subfunction of h(m) where m exists in D

Does hd exist?

If hd does exist, then what about

```
upd = () -> hd(upd) && loop_forever()
```

∴ hd(m) is part of U, and does not exist

If hd does not exist, then how is D actually the decidable subset?

∴ there no subset of programs that is completely decidable, even trivially

Neither of these conclusions are acceptable, so what went wrong?