

Cpts 355 4 Sep 2009

Functions: total functions, partial functions and
 computable functions

A function f is a set of ordered pairs $\langle a, b \rangle \in A \times B$

$\langle 1, 3 \rangle \in \text{Int} \times \text{Int}$

$\langle 1, 'a' \rangle \in \text{Int} \times \text{Str}$

$\{ \langle 1, 3 \rangle, \langle 2, 4 \rangle, \langle 3, 5 \rangle, \dots \}$

$f(x) = x + 2$

→ Such that

2)

Function:

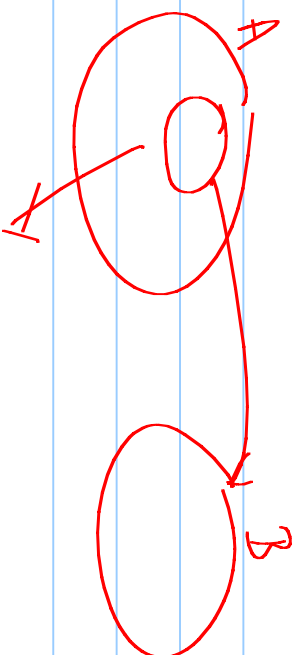
- \mathbb{R} →
- 1) If $\langle x, y \rangle \in f$ and $\langle x, z \rangle \in f$ then $y = z$
 - 2) For every $x \in A \exists y \in B$ with $\langle x, y \rangle \in f$

Total function:



What if I give you code 2.

There can be things in A that don't map to things in B.



A: $\text{Int} \times \text{Int}$

B: Int

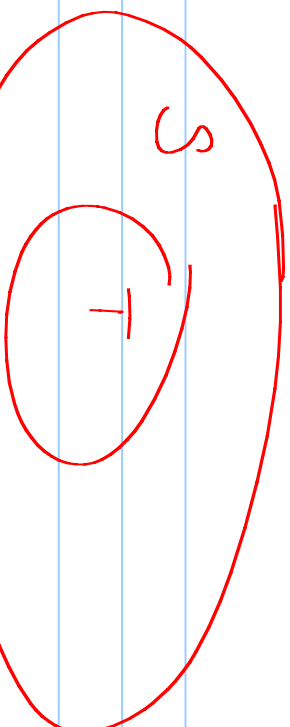
Total functions:

$f(x, y) = x + y$ is a partial function

$g(x, y) = x \cdot y$

+ is total

Set of all total functions



Set of all partial functions

In computing we work with functions like

```
def f(x, y): return x/y
```

$f(x, 0)$ ← error is one way to be partial

{ <0,0>, <2,0>, ... }

```
def f(x):
```

```
    if x == 0: return 0
```

```
    else: return f(x-2)
```

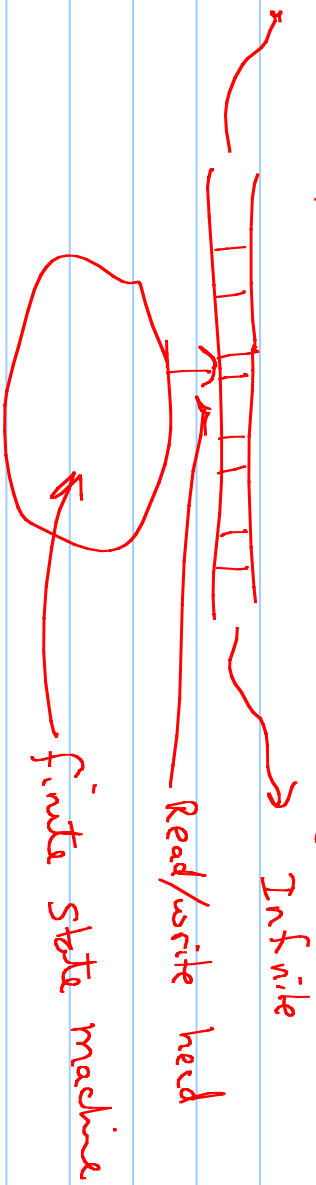
↙ does not halt
 $f(1), f(-2)$

Can we write a program for any partial function?
No! - Halting problem

Kurt Gödel solved the completeness of logic problem: No.

Alan Turing - No

Simple abstract computer: Turing machine



Turing machine model is as powerful as any form of computing that anybody has invented thus far.

Alonzo Church hypothesis

Can a Turing machine be designed to compute an arbitrary partial function? No.

||'

Does an arbitrary program halt?

as a function Q is total function $f(x) =$ if $x == 0$ then 0
that cannot be computed. else $f(x-2)$

Suppose program Q solves the halting problem:

$$Q(P, x) = \begin{cases} \text{true} & \text{if } P(x) \text{ halts} \\ \text{false} & \text{if } P(x) \text{ doesn't hal} \end{cases}$$

Now define function D

$$D(P) = \text{if } Q(P, P) \text{ then } f(-1) \text{ else } 0$$

what is $D(D)$: subst def. of D

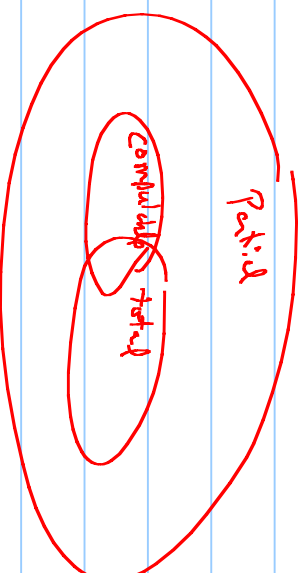
$$D(D) = \text{if } Q(D, D) \text{ then } f(-1) \text{ else } 0 \quad \text{subst def. of } Q$$

if $D(D)$ halts then $f(-1)$ else 0

Mathematical contradiction

Go back and look at assumptions:

- write programs as strings
- existence of program to compute function Ω — wrong —
There can be no such program!



Practical implications:

no ^{small} program to tell whether P_{gen} holds
memory is inaccessible

in many special cases we solve these
problems all the time