## 1 Introduction to logic

What happens when people don't use logic.

Example: pentium floating point bug. -> only testing
There was a huge slowdown because they did not assess it logically.
Originally people thought that testing is enough.
But verification is also important.

Example: Ariane 5 overflow bug

Example: Knight Capital Group trading glitch

How can you test an infinite object in finite steps?

problem example:
Coffee machine state machine -> finite state automata
dots -> states
arrows -> change of states

$S$ -> system (All possible computation)
$\phi$ -> all desired computations

$Lang(S) = \{10\text{c}, \text{reset,5c,5c,10c}, \dots\}$

---

A superior might have given us specification e.g.:

$\phi = \text{'10c or 5c' until (coffe,reset,kick)}$

do all model executions satisfy the delivered specifications?

Containment problem:
$Lang(S) \subseteq Lang(\phi)$

Create Negated specification and check for emptiness of product

$\text{not } \phi = \text{Forever(10c or 5c)}$

### 1.1 What is a logic

a language to express object properties or systems
or
a set of tool to reason about properties of systems

A language has three properties:

1. vocabulary -> The symbols that can be used

2. Syntax -> grammar how to combine them so that something has meaning

3. semantics -> meaning of formulas

Example:
Q: Are you two married
A: Depends

1. meaning: the people who are there married to people
2. or meaning: are the two people present married

Other example:
If all humans are mortal , and
Socrates is human ,
then Socrates is mortal .
$((\forall x A(x) \rightarrow B(x)) \wedge A(y_0)) \rightarrow B(y_0)$

The above formular is valid (i.e. true in every structure/interpretation). We call this a tautology

---

Other example:

There exists a set that
contains all and only
the sets that do not
contain themselves.
$\exists x \forall y (C(x,y) \iff \neg C(y,y))$

The formula is unsatisfiable (false in every structure/interpretation).
We call it a contradiction

---

Every incoming order is eventually processed

$\forall o \forall t (A(o,t) \rightarrow \exists t' \; t < t' \wedge B(o,t')$

---

If a non-deterministiac programs
has infinitely many configurations,
then it has an infinite execution.

if model is a tree,
is finitely-branching,
has infinitely many nodes,
Then
it has a infinite path
Todo: definitinoan

Here we use two different sorts of variables:

- first order varibles (e.g. x,y,z) interpretetd by single elements
- second-order variables (e.g$Y_{finite}$ or $Z_{infinite}$) interpreted as finite or infinite sets.

Path
Has alternating a node and a connection i.e.
n -> n ->n

Chain
Can have multiple connections:
n -> -> n -> n -> -> -> n

As a formal tool to reason about properties of a system

- What can be mechanized?
  (descision problem, algorithms, reduction from Halting, Domino)
- How hard is it to mechanize?
  (complexity, expressiveness)

What is a reduction?

A reduction from P to Q is an algorithm F that solves P using an oracle that returns solutions to Q

What is complexity?
Alg is Time/Space bounded by some function $O(f)$ $f : N \rightarrow N$
if Alg(input) uses <= $c \cdot f(|input|) + d$ units of time/space for some coefficients c,d

What is expressiveness
It is a competition Expressiveness vs Decidability
Which properties can be expressed in a given logic?
Is this logic more or less expressive than another logic?
Does it express undecidable properties.

Succinctness vs Efficiency
How complex is it to express a certain property
Which logic is more Succinctness -> using less tools only using nands is more succinct than using all gates
Which logic has more efficient

**1.2 Model checking problem**

Does a model hold using a structure

formula $\phi \rightarrow$ checker
structure $S \rightarrow$ checker
checker $\rightarrow$ yes/no depending on whether $\phi$ holds over S

validity satifiability

formula $\phi \rightarrow$ checker \rightarrow yes no

$\phi$ needs to hold over all S and is more strict than