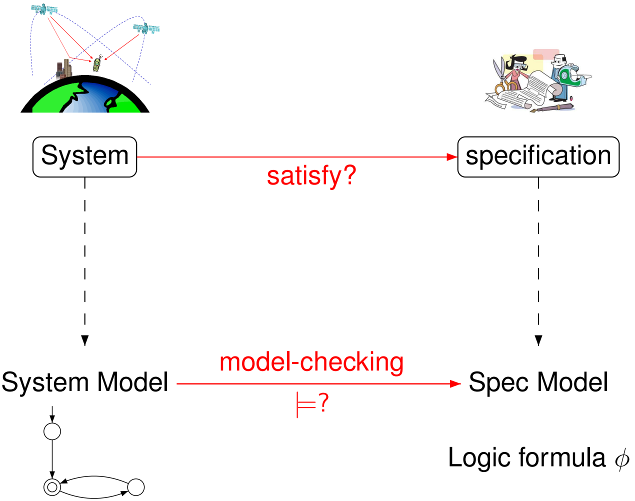
CS 228 : Logic in Computer Science

Krishna. S

Satisfaction, Validity

- Given a FO formula $\varphi(x_1, \ldots, x_n)$ over a signature τ, is it satisfiable/valid?
  - Satisfiable, if there exists a τ-structure $\mathcal{A}$ and an assignment α for $x_1, \ldots, x_n$ in $u(\mathcal{A})$ such that $\mathcal{A} \models_\alpha \varphi(x_1, \ldots, x_n)$
  - Valid, if for any τ-structure $\mathcal{A}$ and any assignment α for $x_1, \ldots, x_n$ in $u(\mathcal{A})$, $\mathcal{A} \models_\alpha \varphi(x_1, \ldots, x_n)$
- Assume we fix the type of the structure $\mathcal{A}$, say words
- FO over words (why words?)

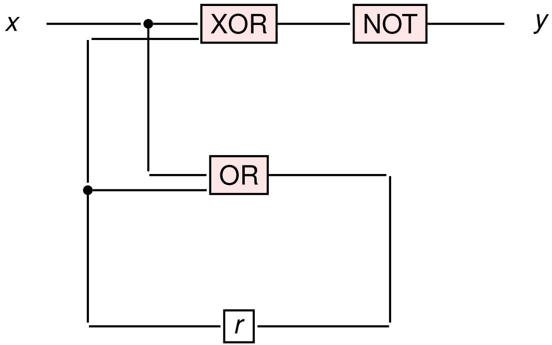
Verification through Model Checking



Model Checking

- Abstract the given system = code/circuit as a finite state transition system, G
- Behaviours of the system = sequence of actions taken by G (these are words, and the actions are the symbols of the alphabet)
- Write the property of interest in a chosen logic as formula φ
- Check $G \models \varphi$

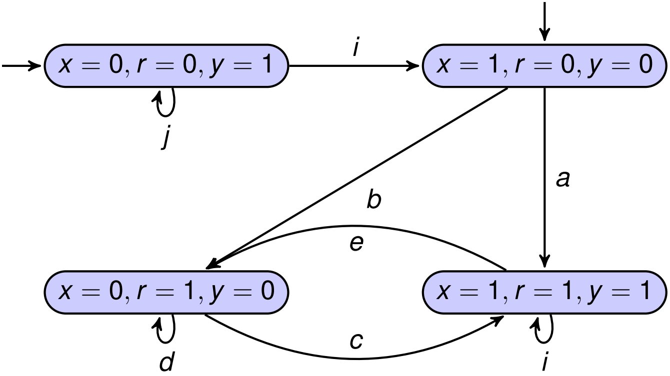
Sequential Circuits



- Input variable x, output variable y, register r
- Output $\neg(x \bigoplus r)$ and register evaluates to $x \vee r$

Transition System for the Circuit

Initially, assume $r = 0$



- Some possible behaviours : j j i ae, i b d d d
- Property : No two i actions $\neg\exists x \exists y(x \neq y \wedge Q_i(x) \wedge Q_i(y))$
- Property : Every i is followed by an a or b :
  $\forall x(Q_i(x) \Rightarrow \exists y(x < y \wedge [Q_a(y) \vee Q_b(y)]))$

Abstract this!

```
#include <iostream>
using namespace std;
int main(void)
{
float a, b, c;
a=b=c=0;
while(b<10)
{
if(1 < c < 5) { a = a + c; }
else { a = |a-c|;}
b=b+0.00001;
input a value for c;
}
}
```

- Property to check : Can $a = 2$ and $b = 3$

First-Order Logic over Words

FO Over Words

- Given an FO sentence φ over words, is it satisfiable/valid?
- Satisfiable (Valid) iff some word (all words) satisfies φ
- There could be infinitely many words w satisfying φ
- $L(\varphi) = \{$ words $w \mid w \models \varphi\}$ is called the language of φ
- Given φ, write an algorithm to check $L(\varphi) = \emptyset$?

Expressiveness and Satisfiability

- Signature for words : $<$, S and Q_a for finitely many symbols a
- Given a FO formula over words, the signature is fixed
- Expressiveness
  - Given a set of words or a language L, can you write a FO formula φ such that $L(\varphi) = L$
  - If you can, FO is expressive enough to capture your language/specification/property
  - If you cannot, show that FO cannot capture your property.
- Satisfiability
  - Given a FO formula φ over words, is $L(\varphi)$ non-empty?

A Primer for Words

Alphabet

- An alphabet Σ is a finite set
  - $\Sigma = \{a, b, \ldots, z\}$
  - $\Sigma = \{+, \alpha, 100, B\}$
- Elements of Σ called letters or symbols
- A word or string over Σ is a finite sequence of symbols from Σ
- If $\Sigma = \{a, b\}$, then *abababa* is a word of length 7
- The length of a word w is denoted $|w|$
- There is a unique word of length 0 denoted ϵ, called the empty word
- $|\epsilon| = 0$

Notations for Words

- *aaaaa* denoted a^5
- $a^0 = \epsilon$
- $a^{n+1} = a^n.a = a.a^n$
- The set of all words over Σ is denoted Σ^*
  - $\{a, b\}^* = \{\epsilon, a, b, aa, ab, ba, bb, aaa, \dots\}$
  - $\{a\}^* = \{\epsilon, a, aa, aaa, \dots\} = \{a^n \mid n \geqslant 0\}$
- By convention, $\{\}^* = \{\epsilon\}$

Notations for Words

- Σ is a finite set
- Σ^* is the set of all finite words over alphabet Σ
- Σ^* is an infinite set
- Each $w \in \Sigma^*$ is a finite word
  - $\{a, b\} = \{b, a\}$ but $ab \neq ba$
  - $\{a, a, b\} = \{a, b\}$ but $aab \neq ab$
  - $\emptyset$ is the set consisting of no words
  - $\{\epsilon\}$ is a set having the single word ϵ
  - ϵ is a word

Operations on Words

- Concatenation of words : $x.y = xy$
  - Concatenation is associative : $x.(yz) = (xy).z$
  - Concatenation not commutative in general $x.y \neq y.x$
  - ϵ is the identity for concatenation $\epsilon.x = x.\epsilon = x$
  - $|x.y| = |x| + |y|$
- x^n : catenating word x n times
  - $(aab)^5 = aabaabaabaabaab$
  - $(aab)^0 = \epsilon$
  - $(aab)^* = \{\epsilon, aab, aabaab, aabaabaab, \dots\}$
  - $x^{n+1} = x^n x$

Operations on Words

- For $a \in \Sigma$ and $x \in \Sigma^*$,

$$|x|_a = \textit{number of times the symbol a occurs in the word x}$$

  - $|aabbaa|_a = 4$, $|aabbaa|_b = 2$
  - $|\epsilon|_a = 0$
- Prefix of a word $w \in \Sigma^*$ is an initial subword of w

$$Pref(w) = \{x \in \Sigma^* \mid \exists y \in \Sigma^*, w = x.y\}$$

  - $Pref(aaba) = \{\epsilon, a, aa, aab, aaba\}$
  - Proper prefixes = $\{a, aa, aab\}$
  - ϵ, $aaba$ improper prefixes

Operation on Sets

Given a finite alphabet Σ, denote by $A, B, C, \ldots$ subsets of Σ^*

- Subsets of Σ^* are called languages
- $A \cup B = \{x \in \Sigma^* \mid x \in A \text{ or } x \in B\}$
  - $A = a^*$, $B = \{b, bb\}$, $A \cup B = a^* \cup \{b, bb\}$
- $A \cap B = \{x \in \Sigma^* \mid x \in A \text{ and } x \in B\}$
  - $A = (ab)^*$, $B = \{abab, \epsilon, bb\}$, $A \cap B = \{\epsilon, abab\}$
- $\overline{A} = \{x \in \Sigma^* \mid x \notin A\}$
  - For $\Sigma = \{a\}$ and $A = (aa)^*$, $\overline{A} = \{a, a^3, a^5, \ldots\}$
- $AB = \{xy \mid x \in A, y \in B\}$
  - $A = \{a, ba\}$, $B = \{\epsilon, aa, bb\}$
  - $AB = \{a, a^3, abb, ba, ba^3, babb\}$
  - $BA = \{a, ba, a^3, aaba, bba, bbba\}$

Operation on Sets

For a set $A \subseteq \Sigma^*$,

- $A^0 = \{\epsilon\}$
- $A^{n+1} = A.A^n$
  - $\{a, ab\}^2 = \{a, ab\}.\{a, ab\} = \{aa, aab, aba, abab\}$
  - $\{a, b\}^n = \{x \in \{a, b\}^* \mid |x| = n\}$
- $A^* = A^0 \cup A \cup A^2 \cup \cdots = \bigcup_{i \geqslant 0} A^i$
- $A^+ = AA^* = A \cup A^2 \cup \cdots = \bigcup_{i \geqslant 1} A^i$
- Union : Associative, commutative
- Concatenation : Associative, Non commutative
- $A \cup \emptyset = \emptyset \cup A = A$
- $\{\epsilon\}A = A\{\epsilon\} = A$
- $\emptyset A = A\emptyset = \emptyset$

Operation on Sets

- Union, Intersection distribute over union
  - $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$
  - $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
- Concatenation distributes over union
  - $A(\cup_{i \in I} B_i) = \cup_{i \in I} AB_i$
  - $(\cup_{i \in I} B_i)A = \cup_{i \in I} B_i A$
- Concatenation does not distribute over interesection
  - $A = \{a, ab\}, B = \{b\}, C = \{\epsilon\}$
  - $A(B \cap C) \neq AB \cap AC$

FO for Languages

Formalize in FO

Write FO formulae φ_i such that $L(\varphi_i) = L_i$ for $i = 1, \ldots, 5$.

- L_1 = Words that have exactly one occurrence of the letter c
- L_2 = Words that begin with a and end with b
- L_3 = Words that have no two consecutive a's
- L_4 = Words in which any a is followed immediately by a b
- L_5 = Words in which whenever an a occurs, it is followed eventually by a b, and no c occurs in between the a and the b *aabbabab*, *aabbcbccaab* $\in L_5$, *aacaab* $\notin L_5$.

$\varphi_1 = \exists x (Q_c(x) \wedge \forall y \forall z (Q_c(y) \wedge Q_c(z) \Rightarrow y = z))$

$\varphi_3 = \forall x \forall y \, (Q_a(x) \wedge Q_a(y) \Rightarrow \neg S(x, y))$

$\varphi_4 = \forall x \, (Q_a(x) \Rightarrow \exists (Q_b(y) \wedge S(x, y)))$

Satisfiability of FO over Words

$\varphi_5 : \neg \exists x, y \left(Q_a(x) \wedge Q_c(y) \wedge S(x, y) \right)$

- Recall : Given an FO sentence φ over words, is $L(\varphi) = \emptyset$?
- Algorithm?
- Given φ, can we easily convert φ into some other mechanism M, which we know how to deal with?