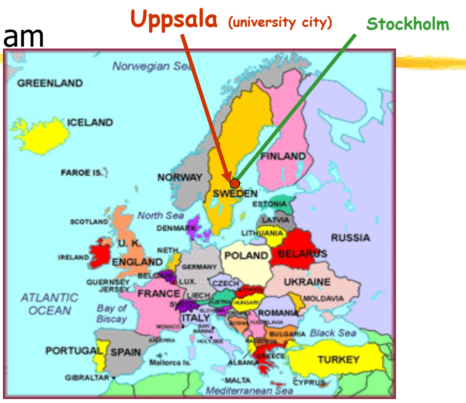
Modeling and Verification of Real Time and Embedded Systems

A tutorial on UPPAAL

Wang Yi
Uppsala University, Sweden, 2005

Here I am



UPPAAL: www.uppaal.com

- A model checker for real time systems developed jointly by Uppsala university and Aalorg university since 1993
- UPPsala + AALborg = UPPAAL
  - SWEDEN + DENMARK = SWEDEN
  - SWEDEN + DENMARK = DENMARK

TIMES: www.timestool.com

- Extended version of UPPAAL, towards a tool environment for the complete system development process:

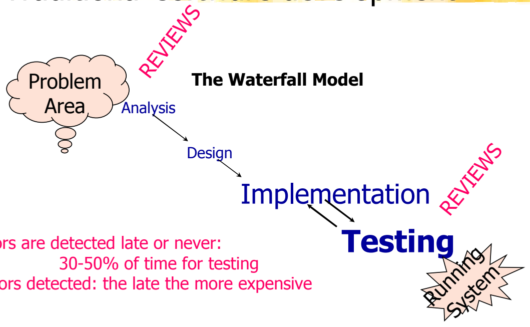
  from design to implementation

- TIMES = a Tool for Modeling and Implemenation of Embedded Systems

Main Goal of the tutorial

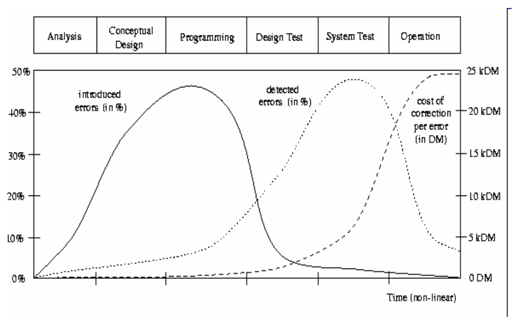
- What is inside the tools
  - UPPAAL
  - TIMES

Traditional software development



♦ Errors are detected late or never:
30-50% of time for testing
♦ Errors detected: the late the more expensive

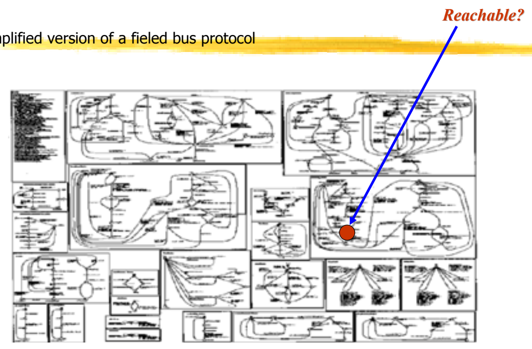
Introducing, Detecting and Correcting errors



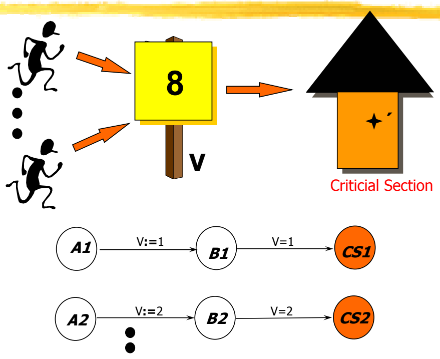
Finding errors as early as possible!

HOW?

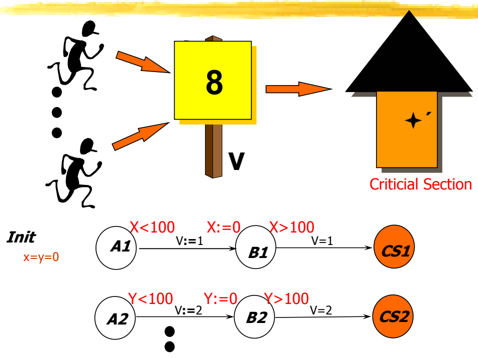
A simplified version of a fieled bus protocol



Example: Fischer's Protocol



Example: Fischer's Protocol



Example: Petersson's algorithm

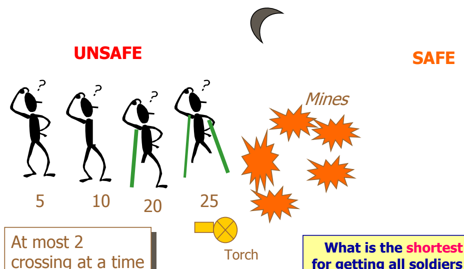
turn, flag1, flag2: shared variables

- Process 1
- Loop
- flag1:=1; turn:=2
- While (flag2 and turn=2) wait
- **CS1**
- flag1:=0
- End loop

- Process 2
- Loop
- flag2:=1; turn:=1
- While (flag1 and turn=1) wait
- **CS2**
- flag2:=0
- End loop

Question: no more than one process run in CS?

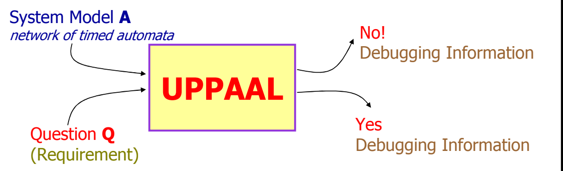
Example: the Soldiers Problem
Real time scheduling



At most 2
crossing at a time
Need torch

What is the shortest time for getting all soldiers on the safe side ?

UPPAAL = UPPsala + AALborg
A tool set for modelling and verification of real-time systems developed jointly by Uppsala and Aalborg University



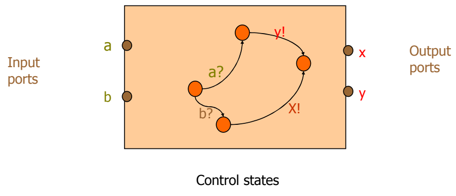
Prototypes
Executable Code
TIMES will do this for you!

Model Checking
in a Nutshell

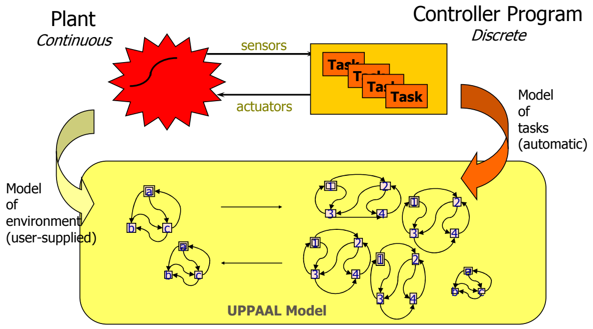
MODELING
How to construct Model ?

Modeling = programming+abstraction

Program as State Machine!

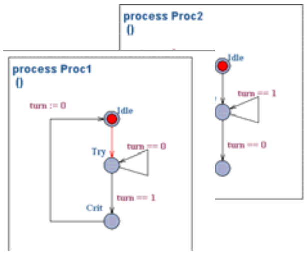


Construction of Models: Concurrency

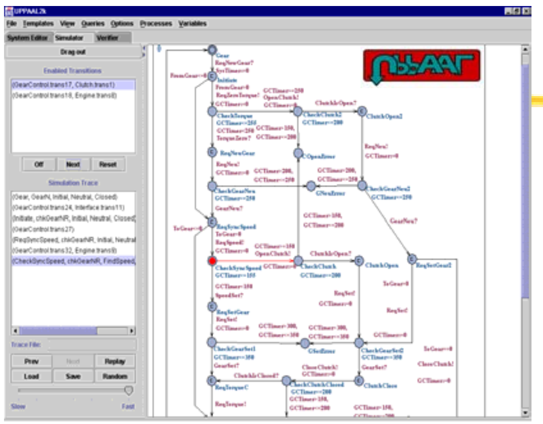


Modeling in UPPAAL: Example

```
P1 :: while True do
      T1 : wait(turn=1)
      C1 : CS1; turn:=0
      endwhile
||
P2 :: while True do
      T2 : wait(turn=0)
      C2 : CS2; turn:=1
      endwhile
```



Mutual Exclusion Program



SPECIFICATION

How to ask questions: Specs ?

Specification=Requirement, Lamport 1977

- Safety
  - Something (bad) will not happen
- Liveness
  - Something (good) must happen

Specification=Requirement, Lamport 1977

- Safety
  - Something (bad) will not happen
- Liveness
  - Something (good) must happen
- Realizability (Schedulability etc)
  - Can we implement the specs with given resources?

Specification: Examples

- AG not (CS1 and CS2)
  - never CS1 and CS2
  - Safety property
- AG (a $\rightarrow_{<=10}$ b)
  - if a then b within 10
  - Bounded liveness property
- EF p.test
  - Useful for debugging
- EF false
  - Generate the whole state space
  - Report deadlocks etc.
- AG (try => AF critical-section) (liveness)

VERIFICATION

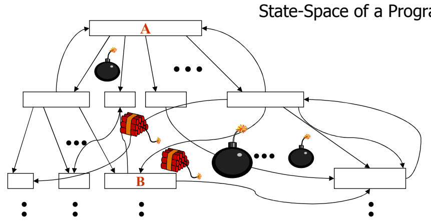
Model meets Specs ?

Verification

- Semantics of a system
  - = all states + state transitions
  - (all possible executions)
- Verification
  - = state space exploration + examination

Verificatioin = *Searching*

State-Space of a Program



(1) Is it possible to fire the bombs?
(2) Is it possible to go from A to B within 10 sec?

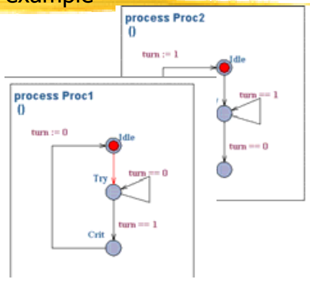
Two basic verification algorithms

- Reachability analysis
  - Checking safety properties
- Loop detection
  - Checking liveness properties

Modelling in UPPAAL: example

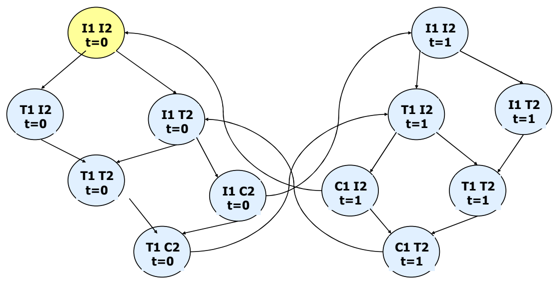
```
P1 :: while True do
      T1 : wait(turn=1)
      C1 : CS1; turn:=0
      endwhile
||
P2 :: while True do
      T2 : wait(turn=0)
      C2 : CS2; turn:=1
      endwhile
```

Mutual Exclusion Program



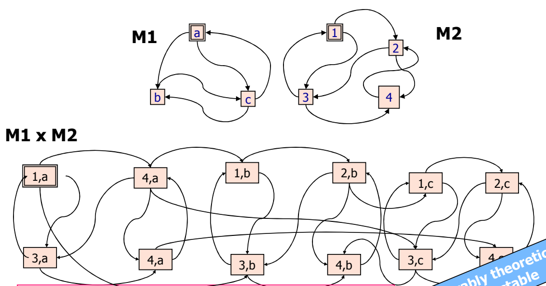
Is it possible that P1 and P2 reach C1 and C2 simultaneously?

Verification: example



(C1 C2) is never reachable!

Problem with verification: 'State Explosion'



All combinations = exponential in no. of components

Provably theoretical intractable

EXAMPLE

10 components and each with10 states

# of control states = 10,000,000,000 =10 G
Each state needs 4*(10 x 10) = 400 B

Worst case memory usage >> 4000 GB

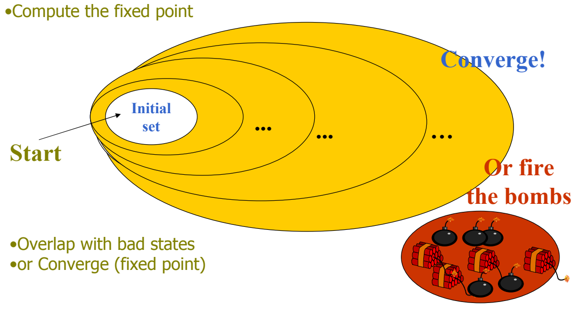
Solutions

- Theorem provers
- Symbolic Techniques e.g. BDD [Bryant 86]
- Abstraction techniques [Cosot and Cosot]
- Approximation methods [Holzman, Wang-Toi ...]
- On-the-fly verification [Holzman]
- Partial order reduction [Wolper et al]
- Compositional verification [too many]
- Combining theorem provers and model checkers
- ... ...

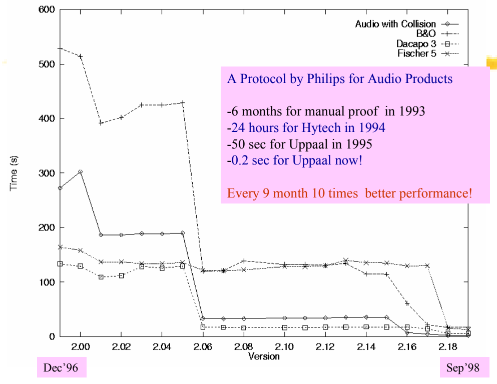
Symbolic Techniques:

Compute Sets of States instead of one-by-one

- Use formulas to represent sets of states
- Compute the fixed point



- Overlap with bad states
- or Converge (fixed point)

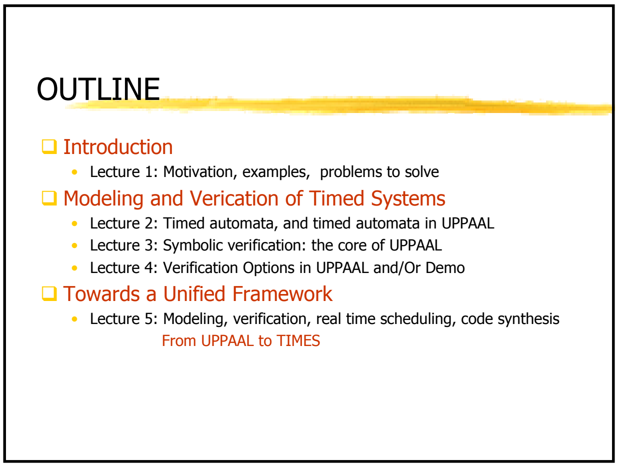


A Protocol by Philips for Audio Products

-6 months for manual proof in 1993
-24 hours for Hytech in 1994
-50 sec for Uppaal in 1995
-0.2 sec for Uppaal now!

Every 9 month 10 times better performance!

End of INTRODUCTION



OUTLINE

- Introduction
  - Lecture 1: Motivation, examples, problems to solve
- Modeling and Verication of Timed Systems
  - Lecture 2: Timed automata, and timed automata in UPPAAL
  - Lecture 3: Symbolic verification: the core of UPPAAL
  - Lecture 4: Verification Options in UPPAAL and/Or Demo
- Towards a Unified Framework
  - Lecture 5: Modeling, verification, real time scheduling, code synthesis
    From UPPAAL to TIMES