Modeling and Verification of Transmission Protocols:

A Case Study on CSMACD Protocol

SHI Ling and LIU Yan

NUS
National University
of Singapore

School of Computing

Outline

- Motivation
- Background
- Model for CSMA/CD Protocol
- Verification Properties and Experimental Results
- Conclusion & Future Works

Motivation

- Real-time systems are mission critical;
- Potential causes to real-time systems:

  -Environmental conditions, human errors

  Design errors

- Verification Methods:

  -Human inspection, Simulation, Testing

  No guarantee!

  - Model Checking and PAT

  ~ Potential guarantee correctness

Outline

- Motivation
- Background

  - Timed extension for CSP#

  - Timed refinement checking

  - The CSMA/CD protocol
- Model for CSMA/CD Protocol
- Verification & Results
- Conclusion & Future Works

Background(1) –Timed CSP#

```
P = Stop | Skip             – primitives
        | e -> P            – event prefixing
        |P [] Q | P<>Q      – general choice
        | P; Q              – sequential composition
        | P ||Q             – parallel composition
        | Wait[d]           – delay
        | P timeout[d] Q          – timeout
        | P interrupt[d] Q        – timed interrupt
        | P within[d]             – react within some time
        | P waituntil[d]          – wait until
        | P deadline[d]           – deadline
```

Background(2) –Timed Refinement

- Timed safety property an be proved by

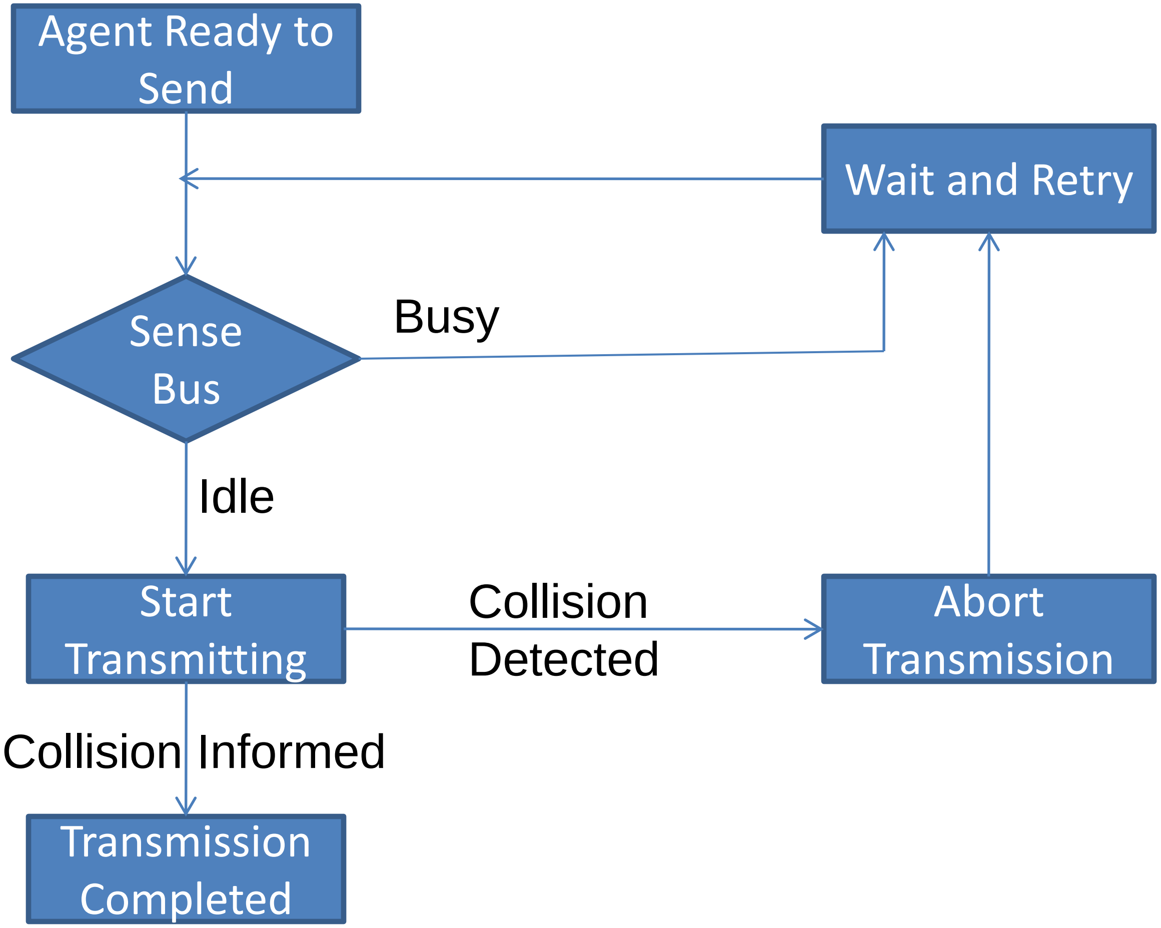
#assert implementation refines<T> specification;

- For example: a model I contains two events start and end, a specification S = start ->((end -> S) within[5])

#assert I *refines<T>* S *;*

Background(3) – The CSMA/CD Protocol

Abstract algorithm of CSMA/CD Protocol:



Outline

- Motivation
- Background
- Model for CSMA/CD Protocol
- Verification Properties and Experimental Results
- Conclusion & Future Works

Model for CSMA/CD Protocol

- Assumptions
  - ✓ Agents communicate in the 10Mbps Ethernet with a worst case for absence signal travel of *26 μsec*
  - ✓ Messages have a fixed length of 1024 bytes
  - ✓ Time for transmitting a complete message is assumed to be a constant time *808 μsec,* including propagation time
  - ✓ Backoff strategy for agent retrying is not modeled

Model for CSMA/CD Protocol

National University of Singapore

School of Computing

Components	Name	Description
Global Definition	N	Constant: number of senders
	channel newMess 0	Sender gets messages to send
	channel begin 0	Sender starts sending message
	channel busy 0	Sender senses a busy bus
	channel cd 0	Sender detects a collision
	channel end 0	Sender completes its transmission
Sender Behavior	WaitFor(i)	Sender i is waiting for a message from the upper level
	Trans(i)	Sender i is sending a message
	Retry(i)	Sender i is waiting to retry after detecting a collision or a busy bus
Bus Behavior	Idle	Bus is free, no sender is transmitting
	Active	One sender starts transmitting and is detecting collision
	Active1	One sender is transmitting messages, bus is busy
	Collision	Collision occurs and bus broadcasts the collision information to all senders

Model for CSMA/CD Protocol

- Sender Behavior

```
WaitFor(i) = (cd?i -> WaitFor(i))
           [] (newMess!i -> ((begin!i -> Trans(i))
                            [] (busy?i -> Retry(i))
                            [] (cd?i -> Retry(i))));

Trans(i) = (cd?i -> Retry(i) within[0,52])
        [] (atomic{end!i -> Skip} within[808,808];
           WaitFor(i));

Retry(i) = newMess!i -> ((begin!i -> Trans(i) within[0, 52])
                [] (busy?i -> Retry(i) within [0, 52])
                [] (cd?i -> Retry(i) within[0, 52]));
```

Model for CSMA/CD Protocol(Cont.)

- Bus Behavior

```
Idle = newMess?i -> begin?i -> Active;

Active = (end?i -> Idle)
         [] (newMess?i ->
                ((begin?i -> Collision) timeout[26] (busy!i -> Active1)));

Active1 = (end?i -> Idle)
          [] (newMess?i -> busy!i ->Active1);

Collision = atomic{BroadcastCD(0)} within[0,26]; Idle;
```

Model for CSMA/CD Protocol(Cont.)

- *BroadcastCD process*

```
BroadcastCD(x) = if (x < N) {
                        (cd!x -> BroadcastCD(x+1))
                       []
                        (newMess?[i==x]i -> cd!x ->BroadcastCD(x+1))
                        }
                        else {
                        Skip
                        };
```

- *CSMACD Process*

```
CSMACD = (|||x :{0..N-1}@WaitFor(x))|||Idle;
```

Outline

- Motivation
- Background
- Model for CSMA/CD Protocol
- Verification Properties and Experimental Results
- Conclusion & Future Works

Verification Properties

- Deadlock Freeness (P0)
- Timed Divergence-free (P1)
- Collision detection in a given bounded delay (P2)
  - ✓ Use refinement model checking techniques
  - ✓ Build a model *Spec* which satisfies the property, then check whether *CSMACD* model satisfies *Spec* or not

Verification Properties (Cont.)

- Spec Model

```
Spec = (newMess.0 -> begin.0 -> Constrained1)
       [] (newMess.1 -> begin.1 -> Constrained2)
       [] Relaxed;
Constrained1 = ((newMess.1 -> begin.1 ->
                     ((cd.0 -> Skip [] cd.1 -> Skip) deadline[52])); Spec)
               [] Relaxed;
Constrained2 = ((newMess.0 -> begin.0 ->
                      ((cd.0 -> Skip [] cd.1 -> Skip) deadline[52])); Spec)
               [] Relaxed;
Relaxed = ([] x:{2..N-1} @ (newMess.x -> begin.x -> Spec))
          [] ([] x:{0..N-1} @ ((newMess.x -> (busy.x -> Spec [] cd.x -> Spec))
                                [] (cd.x -> Spec)
                              [] (end.x -> Spec)));
```

Experimental Results

Property	No. of Senders	Result	#States	#Transitions	Time(sec)
P0	4	Yes	787	1075	0.20
P0	5	Yes	2789	3847	0.60
P0	6	Yes	8851	12227	2.28
P0	7	Yes	26109	35991	8.43
P0	8	Yes	73123	100419	31.03
P0	9	Yes	196997	269319	108.69
P0	10	Yes	514915	700611	361.58
P1	4	Yes	787	1075	0.17
P1	5	Yes	2789	3847	0.66
P1	6	Yes	8851	12227	2.53
P1	7	Yes	26109	35991	9.79
P1	8	Yes	73123	100419	35.69
P1	9	Yes	196997	269319	123.24
P1	10	Yes	514915	700611	407.12
P2	4	Yes	787	1075	0.20
P2	5	Yes	2789	3847	0.90
P2	6	Yes	8851	12227	3.69
P2	7	Yes	26109	35991	14.74
P2	8	Yes	73123	100419	55.38
P2	9	Yes	196997	269319	196.35
P2	10	Yes	514915	700611	655.3

Testbed is a computer with 2.33GHz Intel(R) core(TM)2 Duo CPU and 3.25GB memory.

Outline

- Motivation
- Background
- Model for CSMA/CD Protocol
- Verification Properties and Experimental Results
- Conclusion & Future Works

Conclusion

- Specify a formal model for CSMA/CD protocol
- Verify the properties using PAT

On-going and Future Works

- Model back off strategy for agent retrying of CSMA/CD protocol
- Apply probabilistic model checking techniques to model more richer proportories of the protocol
- Improve PAT to efficiently deal with state explosion problems

Thanks & QA!

NUS
National University
of Singapore

School *of* Computing