Specification and Analysis of Contracts
Lecture 8
Verification of 'Deontic' Contracts in $\mathcal{CL}$

Gerardo Schneider

gerardo@ifi.uio.no

http://folk.uio.no/gerardo/

Department of Informatics,
University of Oslo

SEFM School, Oct. 27 - Nov. 7, 2008
Cape Town, South Africa

Plan of the Course

Plan

Plan

Model Checking in a Nutshell

A model checker is a software tool that given:

- A model M (usually a **Kripke model**)
- A property ϕ (usually a **temporal logic formula**)

It decides whether

$$M \models \phi$$

- It returns YES if the property is satisfied,
- Otherwise returns NO and provides a counterexample

It is completely automatic!

Model Checking in a Nutshell

A model checker is a software tool that given:

- A model M (usually a **Kripke model**)
- A property ϕ (usually a **temporal logic formula**)

It decides whether

$$M \models \phi$$

- It returns YES if the property is satisfied,
- Otherwise returns NO and provides a counterexample

It is completely automatic!

Model Checking (1)

- Model checking is a technique for verifying **finite-state** concurrent systems
- Theoretically speaking, model checking consists of the following tasks:

1. Modeling the system
   - It may require the use of abstraction
   - Often using some kind of automaton
2. Specifying the properties the design must satisfy
   - It is impossible to determine all the properties the systems should satisfy
   - Often using some kind of temporal logic
3. Verifying that the system satisfies its specification
   - In case of a negative result: error trace
   - An error trace may be product of a specification error (*false negative*)

Model Checking (2)

The application of model checking in a design project typically consists of the following steps:

1. Choose the properties (correctness requirements) critical to the design
2. Build a verification model guided by the above correctness requirements
   - The model should be as smallest as possible
   - It should, however, capture everything which is relevant to the properties to be verified
3. Select the appropriate verification method based on the model and the properties
4. Refine the verification model and correctness requirements until all correctness concerns are adequately satisfied

State-explosion problem!

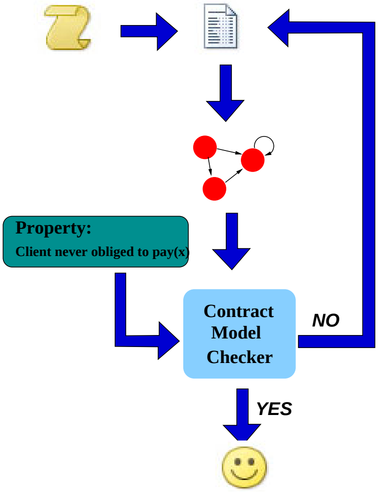
- Use abstraction
- Special techniques for infinte-state systems
- ...

Model Checking (3)

Important Decisions

- Branching vs Linear Time
- Symbolic vs Explicit Verification
- Breadth-First Search vs Depth-First Search
- Tarjan's SCC Algorithms vs Spin's Nested Depth-First Search
- Events vs States
- Real-time vs Timeless Verification
- Probabilities vs Possibilities
- Asynchronous vs Synchronous Systems
- Interleaving Semantics vs True Concurrency
- Open vs Closed Systems
- Backward vs Forward Reachability
- Compositional vs Non-compositional Verification
- Deductive vs Algorithmic Verification

Model Checking Contracts



Plan

Model Checking Contracts

1. Model the conventional contract (in English) as a $\mathcal{CL}$ expression
2. Translate the $\mathcal{CL}$ specification into $\mathcal{C}\mu$
3. Obtain a Kripke-like model (LTS) from the $\mathcal{C}\mu$ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
   1. Check the model is 'good'
   2. Check some properties about the client
   3. Check some properties about the provider
6. In case of a counter-example given by NuSMV, interpret it as a $\mathcal{CL}$ clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract

Model Checking Contracts

1. Model the conventional contract (in English) as a $\mathcal{CL}$ expression
2. Translate the $\mathcal{CL}$ specification into $\mathcal{C}\mu$
3. Obtain a Kripke-like model (LTS) from the $\mathcal{C}\mu$ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
    1. Check the model is 'good'
    2. Check some properties about the client
    3. Check some properties about the provider
6. In case of a counter-example given by NuSMV, interpret it as a $\mathcal{CL}$ clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract

Model Checking Contracts

1. Model the conventional contract (in English) as a $\mathcal{CL}$ expression
2. Translate the $\mathcal{CL}$ specification into $\mathcal{C}\mu$
3. Obtain a Kripke-like model (LTS) from the $\mathcal{C}\mu$ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
    1. Check the model is 'good'
    2. Check some properties about the client
    3. Check some properties about the provider
6. In case of a counter-example given by NuSMV, interpret it as a $\mathcal{CL}$ clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract

Model Checking Contracts

1. Model the conventional contract (in English) as a $\mathcal{CL}$ expression
2. Translate the $\mathcal{CL}$ specification into $\mathcal{C}\mu$
3. Obtain a Kripke-like model (LTS) from the $\mathcal{C}\mu$ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
    1. Check the model is 'good'
    2. Check some properties about the client
    3. Check some properties about the provider
6. In case of a counter-example given by NuSMV, interpret it as a $\mathcal{CL}$ clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract

Model Checking Contracts

1. Model the conventional contract (in English) as a $\mathcal{CL}$ expression
2. Translate the $\mathcal{CL}$ specification into $\mathcal{C}\mu$
3. Obtain a Kripke-like model (LTS) from the $\mathcal{C}\mu$ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
    1. Check the model is 'good'
    2. Check some properties about the client
    3. Check some properties about the provider
6. In case of a counter-example given by NuSMV, interpret it as a $\mathcal{CL}$ clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract

Model Checking Contracts

1. Model the conventional contract (in English) as a $\mathcal{CL}$ expression
2. Translate the $\mathcal{CL}$ specification into $\mathcal{C}\mu$
3. Obtain a Kripke-like model (LTS) from the $\mathcal{C}\mu$ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
    1. Check the model is 'good'
    2. Check some properties about the client
    3. Check some properties about the provider
6. In case of a counter-example given by NuSMV, interpret it as a $\mathcal{CL}$ clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract

Model Checking Contracts

1. Model the conventional contract (in English) as a $\mathcal{CL}$ expression
2. Translate the $\mathcal{CL}$ specification into $\mathcal{C}\mu$
3. Obtain a Kripke-like model (LTS) from the $\mathcal{C}\mu$ formulas
4. Translate the LTS into the input language of NuSMV
5. Perform model checking using NuSMV
    1. Check the model is 'good'
    2. Check some properties about the client
    3. Check some properties about the provider
6. In case of a counter-example given by NuSMV, interpret it as a $\mathcal{CL}$ clause and repeat the model checking process until the property is satisfied
7. In some cases rephrase the original contract

Case Study
A Contract Example

1. The **Client** shall not:
a) supply false information to the Client Relations Department of the **Provider**.
2. Whenever the Internet Traffic is **high** then the **Client** must pay [*price*] immediately, or the **Client** must notify the **Provider** by sending an e-mail specifying that he will pay later.
3. If the **Client** delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the **normal** level, and pay later twice $(2 * [price])$.
4. If the **Client** does not lower the Internet traffic immediately, then the **Client** will have to pay $3 * [price]$.
5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**'s web page to the Client Relations Department of the **Provider**.
6. **Provider** may, at its sole discretion, without notice or giving any reason or incurring any liability for doing so:
a) Suspend Internet Services immediately if **Client** is in breach of Clause 1;

1. Model the Contract in $\mathcal{CL}$

1. The **Client** shall not:
a) supply false information to the Client Relations Department of the **Provider**.
2. Whenever the Internet Traffic is **high** then the **Client** must pay [*price*] immediately, or the **Client** must notify the **Provider** by sending an e-mail specifying that he will pay later.
3. If the **Client** delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the **normal** level, and pay later twice ($2 * [price]$).
4. If the **Client** does not lower the Internet traffic immediately, then the **Client** will have to pay $3 * [price]$.
5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**'s web page to the Client Relations Department of the **Provider**.
6. **Provider** may, at its sole discretion, without notice or giving any reason or incurring any liability for doing so:
a) Suspend Internet Services immediately if **Client** is in breach of Clause 1;

1. Model the Contract in $\mathcal{CL}$

1. The **Client** shall not:
a) supply false information to the Client Relations Department of the **Provider**.
2. Whenever the Internet Traffic is **high** then the **Client** must pay [*price*] immediately, or the **Client** must notify the **Provider** by sending an e-mail specifying that he will pay later.
3. If the **Client** delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the **normal** level, and pay later twice $(2 * [price])$.
4. If the **Client** does not lower the Internet traffic immediately, then the **Client** will have to pay $3 * [price]$.
5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**'s web page to the Client Relations Department of the **Provider**.
6. **Provider** may, at its sole discretion, without notice or giving any reason or incurring any liability for doing so:
a) Suspend Internet Services immediately if **Client** is in breach of Clause 1;

1. Model the Contract in $\mathcal{CL}$

1. $\Box F(fi)$

2. Whenever the Internet Traffic is **high** then the **Client** must pay [*price*] immediately, or the **Client** must notify the **Provider** by sending an e-mail specifying that he will pay later.
3. If the **Client** delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the **normal** level, and pay later twice $(2 * [price])$.
4. If the **Client** does not lower the Internet traffic immediately, then the **Client** will have to pay $3 * [price]$.
5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**'s web page to the Client Relations Department of the **Provider**.
6. **Provider** may, at its sole discretion, without notice or giving any reason or incurring any liability for doing so:
a) Suspend Internet Services immediately if **Client** is in breach of Clause 1;

1. Model the Contract in $\mathcal{CL}$

1. $\Box F(fi)$

2. Whenever the Internet Traffic is **high** then the **Client** must pay [*price*] immediately, or the **Client** must notify the **Provider** by sending an e-mail specifying that he will pay later.
3. If the **Client** delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the **normal** level, and pay later twice ($2 * [price]$).
4. If the **Client** does not lower the Internet traffic immediately, then the **Client** will have to pay $3 * [price]$.
5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**'s web page to the Client Relations Department of the **Provider**.
6. **Provider** may, at its sole discretion, without notice or giving any reason or incurring any liability for doing so:
a) Suspend Internet Services immediately if **Client** is in breach of Clause 1;

1. Model the Contract in $\mathcal{CL}$

1. $\Box F_{P(s)}(fi)$

2. Whenever the Internet Traffic is **high** then the **Client** must pay [*price*] immediately, or the **Client** must notify the **Provider** by sending an e-mail specifying that he will pay later.
3. If the **Client** delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the **normal** level, and pay later twice $(2 * [price])$.
4. If the **Client** does not lower the Internet traffic immediately, then the **Client** will have to pay $3 * [price]$.
5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**'s web page to the Client Relations Department of the **Provider**.

1. Model the Contract in $\mathcal{CL}$

1. $\Box F_{P(s)}(fi)$

2. $\Box[h](\phi \Rightarrow O(p + (d \& n)))$

3. If the **Client** delays the payment as stipulated in 2, after notification he must immediately lower the Internet traffic to the **normal** level, and pay later twice $(2 * [price])$.
4. If the **Client** does not lower the Internet traffic immediately, then the **Client** will have to pay $3 * [price]$.
5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**'s web page to the Client Relations Department of the **Provider**.

1. Model the Contract in $\mathcal{CL}$

1. $\Box F_{P(s)}(fi)$

2. $\Box[h](\phi \Rightarrow O(p + (d \& n)))$

3. $\Box([d\&n](O(l) \wedge [l]\Diamond O(p\&p)))$

4. If the **Client** does not lower the Internet traffic immediately, then the **Client** will have to pay $3 * [price]$.
5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**'s web page to the Client Relations Department of the **Provider**.

1. Model the Contract in $\mathcal{CL}$

1. $\Box F_{P(s)}(fi)$

2. $\Box[h](\phi \Rightarrow O(p + (d\&n)))$

3. $\Box([d\&n](O(l) \wedge [l]\Diamond O(p\&p)))$

4. $\Box([d\&n \cdot \bar{l}\,]\Diamond O(p\&p\&p))$

5. The **Client** shall, as soon as the Internet Service becomes operative, submit within seven (7) days the Personal Data Form from his account on the **Provider**'s web page to the Client Relations Department of the **Provider**.

1. Model the Contract in $\mathcal{CL}$

1. $\Box F_{P(s)}(fi)$
2. $\Box[h](\phi \Rightarrow O(p + (d\&n)))$
3. $\Box([d\&n](O(l) \wedge [l]\Diamond O(p\&p)))$
4. $\Box([d\&n \cdot \bar{l}\,]\Diamond O(p\&p\&p))$
5. $\Box([o]O(sfD))$

2. From $\mathcal{CL}$ into $\mathcal{C}\mu$

- $\Box F_{P(s)}(fi)$ is translated into

$$\nu Z.([fi]\mathcal{F}_{fi} \wedge [fi]\langle s\rangle\neg F_s \wedge [\mathbf{any}]Z)$$

  - This is done by applying the encoding function $f^{\mathcal{T}}$:

$$f^{\mathcal{T}}(\Box F_{P(s)}(fi)) = \nu Z.f^{\mathcal{T}}(F_{P(s)}(fi)) \wedge [\mathbf{any}]Z$$

where: $$f^{\mathcal{T}}(F_{P(s)}(fi)) = f^{\mathcal{T}}(F(fi) \wedge [fi]P(s)) = [fi]\mathcal{F}_{fi} \wedge [fi]\langle s\rangle\neg F_s$$

- Using the $\Box$ as syntactic sugar (which will be reduced to ν) we obtain:

1. $\Box[fi]\mathcal{F}_{fi} \wedge [fi]\langle s\rangle\neg F_s$
2. $\Box[h](\phi \implies (\langle p\rangle O_p \wedge \langle\{d,n\}\rangle(O_d \wedge O_n)))$
3. $\Box[\{d,n\}](\langle l\rangle O_l \wedge [l](\mu Z.\langle\{p,p\}\rangle O_p \vee ([\mathbf{any}]Z \wedge \langle\mathbf{any}\rangle\top)))$
4. $\Box[\{d,n\}][\bar{l}](\mu Z.\langle\{p,p,p\}\rangle O_p \vee ([\mathbf{any}]Z \wedge \langle\mathbf{any}\rangle\top))$
5. $\Box[o]\langle sfD\rangle O_{sfD}$

2. From $\mathcal{CL}$ into $\mathcal{C}\mu$

- $\Box F_{P(s)}(fi)$ is translated into

$$\nu Z.([fi]\mathcal{F}_{fi} \wedge [fi]\langle s\rangle\neg F_s \wedge [\mathbf{any}]Z)$$

  - This is done by applying the encoding function $f^{\mathcal{T}}$:

$$f^{\mathcal{T}}(\Box F_{P(s)}(fi)) = \nu Z.f^{\mathcal{T}}(F_{P(s)}(fi)) \wedge [\mathbf{any}]Z$$
$$\text{where:} \quad f^{\mathcal{T}}(F_{P(s)}(fi)) = f^{\mathcal{T}}(F(fi) \wedge [fi]P(s)) = [fi]\mathcal{F}_{fi} \wedge [fi]\langle s\rangle\neg F_s$$

- Using the $\Box$ as syntactic sugar (which will be reduced to ν) we obtain:

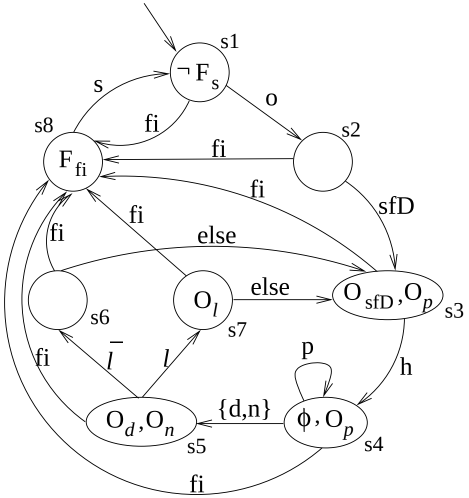
1. $\Box[fi]\mathcal{F}_{fi} \wedge [fi]\langle s\rangle\neg F_s$
2. $\Box[h](\phi \implies (\langle p\rangle O_p \wedge \langle\{d,n\}\rangle(O_d \wedge O_n)))$
3. $\Box[\{d,n\}](\langle l\rangle O_l \wedge [l](\mu Z.\langle\{p,p\}\rangle O_p \vee ([\mathbf{any}]Z \wedge \langle\mathbf{any}\rangle\top)))$
4. $\Box[\{d,n\}][\bar{l}](\mu Z.\langle\{p,p,p\}\rangle O_p \vee ([\mathbf{any}]Z \wedge \langle\mathbf{any}\rangle\top))$
5. $\Box[o]\langle sfD\rangle O_{sfD}$

3. Handcrafting the Model (LTS)

ϕ	=	the Internet traffic is high
fi	=	client supplies false information to Client Relations Department
h	=	client increases Internet traffic to *high* level
p	=	client pays [price]
d	=	client delays payment
n	=	client notifies by e-mail
l	=	client lowers the Int. traffic
sfD	=	client sends the Personal Data Form to Client Relations Department
o	=	provider activates the Internet Service (it becomes operative)
s	=	provider suspends service

3. Handcrafting the Model (LTS)

ϕ	=	the Internet traffic is high
fi	=	client supplies false information to Client Relations Department
h	=	client increases Internet traffic to *high* level
p	=	client pays [price]
d	=	client delays payment
n	=	client notifies by e-mail
l	=	client lowers the Int. traffic
sfD	=	client sends the Personal Data Form to Client Relations Department
o	=	provider activates the Internet Service (it becomes operative)
s	=	provider suspends service



Few Words about NuSMV

- NuSMV is the successor of symbolic model checker SMV
- Symbolic model checking on encoding states using binary decision diagrams (BDD) or similar techniques
- It allows checking properties specified in CTL, LTL, or PSL
- More recently NuSMV has included *input variables* to specify LTS directly

4. Encoding into NuSMV

General Issues

- NuSMV uses **state variables** to identify states and **input variables** to specify labels of an LTS
- The number of states is determined by the product of the number of different values each state variable can take
- We have used one input variable for each atomic action of the $\mathcal{CL}$ specification
  - The type of the input variables is *boolean*
  - Unspecified variables are given any value: it creates a transition (or a state in case of state variables) for each value of the variable
- Concurrent actions ($p \& p$) are encoded with the type *range of integers*
  - If $p = 0$: the transition is not labelled with the action p
  - if $p = 1$: the transition is labelled with one action p
  - if $p = 2$ then we take the transition if $p \& p$

Encoding the Model

- Actions

```
IVAR
  d : boolean ;
  n : boolean ;
  p : 0 .. 3 ;
```

- States and deontic constants

```
VAR
  state : {s1,s2,s3,s4,s5,s6,s7,s8} ;
  high : boolean ;
  F_s : boolean ;  F_fi : boolean ;
  O_p : boolean ;  O_d : boolean ;  O_n : boolean ;
  O_l : boolean ;  O_sfD : boolean ;
```

4. Encoding into NuSMV

Encoding the Model

- Initial state and one of its outgoing transitions

```
INIT
  (state = s1) & !high &
  !F_fi & !O_p & !O_d & !O_n & !O_l & !O_sfD & !F_s ;
```

- transition from s_1 till s_6

```
TRANS
--state variables of the current state
  ((state = s1) & !high &
   !F_fi & !O_p & !O_d & !O_n & !O_l & !O_sfD & !F_s &
--input variables as the labels
   (!fi & p = 0 & !d & !n & !l & !negl & !sfD & o & !s) &
--the values of the state variables in the next states
   (next(state) = s6) & !next(high) &
   next(!F_fi & !O_p & !O_d & !O_n & !O_l & !O_sfD & !F_s))
```

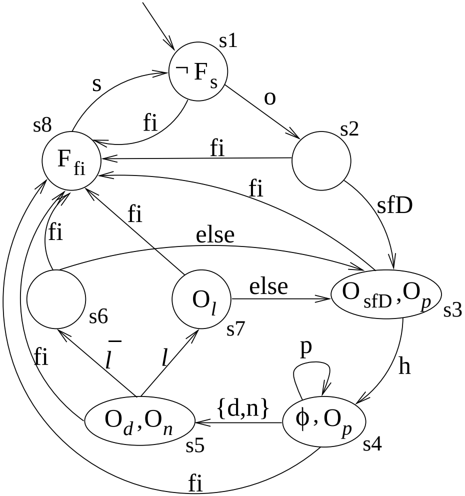
Encoding the Properties

- Properties are encoded into LTL
- Is it possible to encode deontic notions? Does it mean that finally LTL is enough?
  - Need a lot of hacking!
  - It works for the particular properties we are dealing with, not in general
- Out of the scope of this tutorial (too technical!)

5. Model Checking the Contract

5.1. Check the Model is Good

1. $\Box F_{P(s)}(fi)$
2. $\Box [h](\phi \Rightarrow O(p + (d\&n)))$
3. $\Box([d\&n](O(l) \wedge [l]\Diamond O(p\&p)))$
4. $\Box([d\&n \cdot \bar{l}\,]\Diamond O(p\&p\&p))$
5. $\Box([o]O(sfD))$

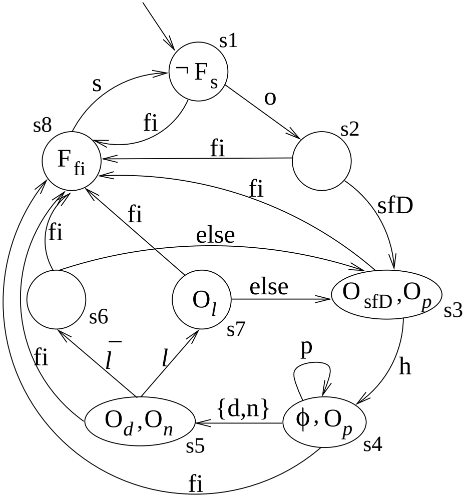


5. Model Checking the Contract

5.1. Check the Model is Good

1. $\Box F_{P(s)}(fi)$
2. $\Box[h](\phi \Rightarrow O(p + (d\&n)))$
3. $\Box([d\&n](O(l) \wedge [l]\Diamond O(p\&p)))$
4. $\Box([d\&n \cdot \bar{l}\,]\Diamond O(p\&p\&p))$
5. $\Box([o]O(sfD))$

1, 2, and 4: OK

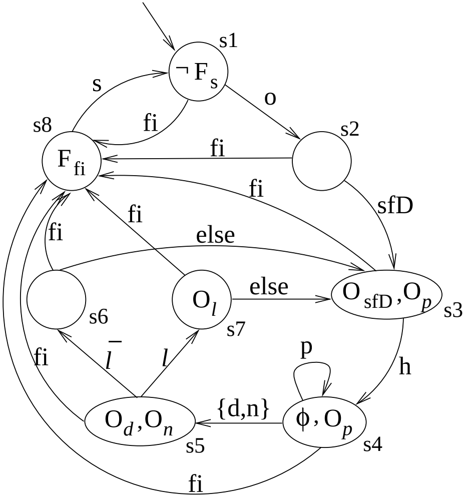


5. Model Checking the Contract

5.1. Check the Model is Good

1. $\Box F_{P(s)}(fi)$
2. $\Box [h](\phi \Rightarrow O(p + (d\&n)))$
3. $\Box([d\&n](O(l) \wedge [l]\Diamond O(p\&p)))$
4. $\Box([d\&n \cdot \bar{l}]\Diamond O(p\&p\&p))$
5. $\Box([o]O(sfD))$

1, 2, and 4: OK
3 and 5: FAIL!



Failure of 3. It fails since there is a dependency with clause 2

- We need to combine clauses 2 and 3: it model checks!

Failure of 3. It fails since there is a dependency with clause 2

- We need to combine clauses 2 and 3: it model checks!

Failure on our formalization in $\mathcal{CL}$!

5.1. Check the Model is Good

Failure of 3. It fails since there is a dependency with clause 2

- We need to combine clauses 2 and 3: it model checks!

Failure on our formalization in $\mathcal{CL}$!

Failure of 5. $(\Box([o]O(sfD)))$

- The system should become operative only once

5. Model Checking the Contract

5.1. Check the Model is Good

Failure of 3. It fails since there is a dependency with clause 2

- We need to combine clauses 2 and 3: it model checks!

Failure on our formalization in $\mathcal{CL}$!

Failure of 5. $(\Box([o]O(sfD)))$

- The system should become operative only once

1. We rewrite the original contract
2. This is formulated in $\mathcal{CL}$, written in NuSMV, and it model checks!

Failure of 3. It fails since there is a dependency with clause 2

- We need to combine clauses 2 and 3: it model checks!

Failure on our formalization in $\mathcal{CL}$!

Failure of 5. $(\Box([o]O(sfD)))$

- The system should become operative only once

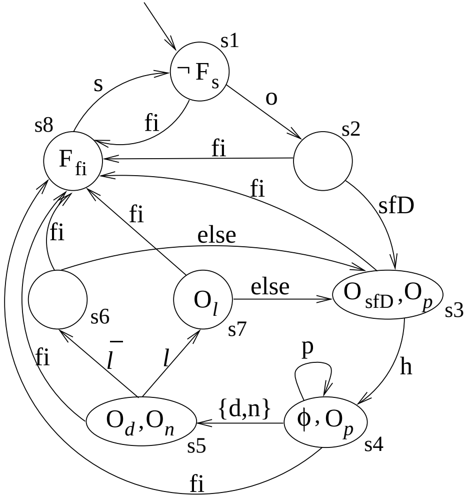
1. We rewrite the original contract
2. This is formulated in $\mathcal{CL}$, written in NuSMV, and it model checks!

'Failure' on the original contract!

5. Model Checking the Contract

5.2. Verifying a property about client obligations

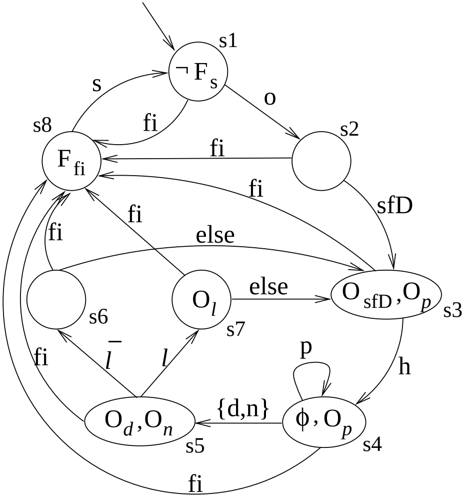
- "It is always the case that whenever the Internet traffic is high, if the clients pays immediately, then the client is *not* obliged to pay again immediately afterward"



5. Model Checking the Contract

5.2. Verifying a property about client obligations

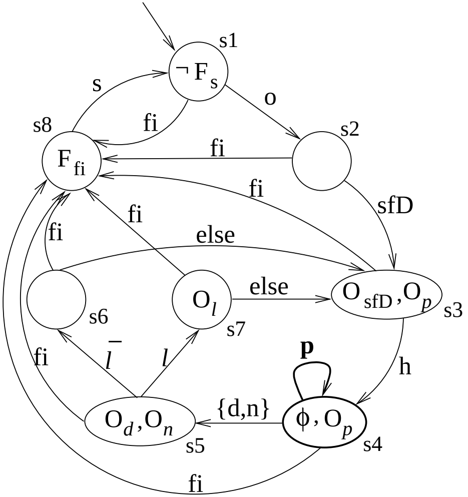
- "It is always the case that whenever the Internet traffic is high, if the clients pays immediately, then the client is *not* obliged to pay again immediately afterward"
- It fails!



5. Model Checking the Contract

5.2. Verifying a property about client obligations

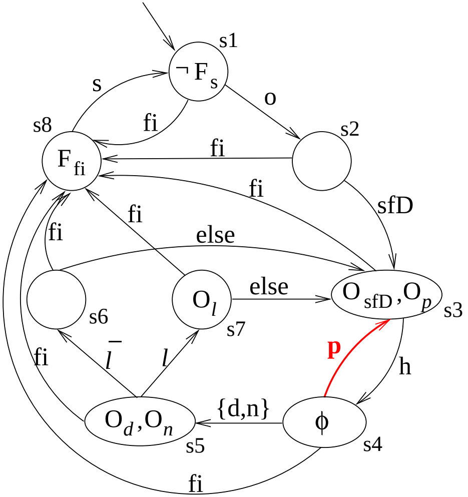
- "It is always the case that whenever the Internet traffic is high, if the clients pays immediately, then the client is *not* obliged to pay again immediately afterward"
- It fails!
- We get a counter-example –Problem: state $s4$



5. Model Checking the Contract

5.2. Verifying a property about client obligations

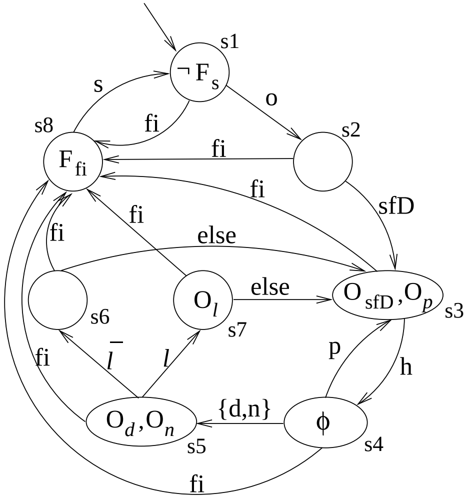
- "It is always the case that whenever the Internet traffic is high, if the clients pays immediately, then the client is *not* obliged to pay again immediately afterward"
- It fails!
- We get a counter-example –Problem: state *s4*
- We modify the original contract to capture the above more precisely



5. Model Checking the Contract

5.3. Verifying a property for the Provider's Benefit

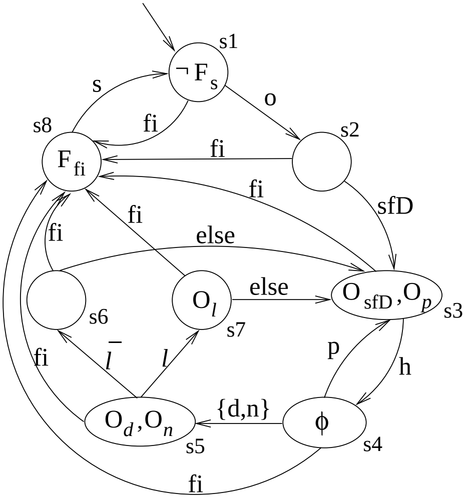
- "It is always the case that whenever Internet traffic is high, if the client delays payment and notifies, and afterward lowers the Internet traffic, then the client is forbidden to increase Internet traffic until he pays twice"



5. Model Checking the Contract

5.3. Verifying a property for the Provider's Benefit

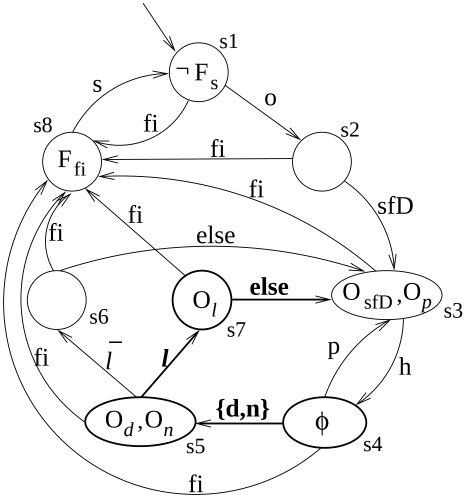
- "It is always the case that whenever Internet traffic is high, if the client delays payment and notifies, and afterward lowers the Internet traffic, then the client is forbidden to increase Internet traffic until he pays twice"
- It fails!



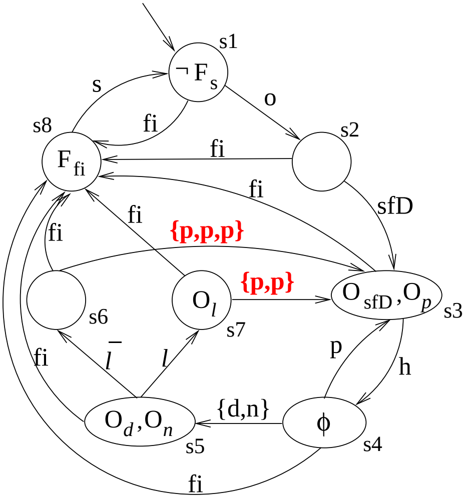
5. Model Checking the Contract

5.3. Verifying a property for the Provider's Benefit

- "It is always the case that whenever Internet traffic is high, if the client delays payment and notifies, and afterward lowers the Internet traffic, then the client is forbidden to increase Internet traffic until he pays twice"
- It fails!
- Counter-example: From s_4 (ϕ holds), after $d\&n \cdot l$, it is possible to increase Internet traffic in state s_7, so neither $F(h)$ nor $\text{done}_{p\&p}$ hold



- "It is always the case that whenever Internet traffic is high, if the client delays payment and notifies, and afterward lowers the Internet traffic, then the client is forbidden to increase Internet traffic until he pays twice"
- It fails!
- Counter-example: From s_4 (ϕ holds), after $d\&n \cdot l$, it is possible to increase Internet traffic in state s_7, so neither $F(h)$ nor $\text{done}_{p\&p}$ hold
- Add to the original contract the clause above!



Further Reading

- G. Pace, C. Prisacariu, and G. Schneider. **Model checking contracts -a case study.** In ATVA'07, vol. 4762 of LNCS, pp. 82-97, 2007