Towards a Formal Language for Electronic Contracts

Gerardo Schneider

gerardo@ifi.uio.no

Joint work with Cristian Prisacariu (cristi@ifi.uio.no)

Department of Informatics,
University of Oslo

University of Edinburgh
17 of July 2007 – Edinburgh, Scotland

Contracts

- "A contract is a binding agreement between two or more persons that is enforceable by law." [Webster on-line]

Contracts

- "A contract is a binding agreement between two or more persons that is enforceable by law." [Webster on-line]

This deed of **Agreement** is made between:
1. **[name]**, from now on referred to as **Provider** and
2. the **Client**.

INTRODUCTION

3. The **Provider** is obliged to provide the **Internet Services** as stipulated in this **Agreement**.
4. DEFINITIONS

a) **Internet traffic** may be measured by both **Client** and **Provider** by means of Equipment and may take the two values **high** and **normal**.

OPERATIVE PART

1. The **Client** shall not 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**.

Contracts

- We call the above a *conventional contract*
- An e-contract (electronic contract) is a machine-readable contract

Two scenarios:

1. Obtain an e-contract from a conventional contract
   - Context: legal (e.g. financial) contracts
2. Write the e-contract directly in a formal language
   - Context: web services, components, OO, etc

We are interested in both:

Definition

A contract is a document which engages several parties in a transaction and stipulates their obligations, rights, and prohibitions, as well as penalties in case of contract violations.

Contracts

- We call the above a *conventional contract*
- An e-contract (electronic contract) is a machine-readable contract

Two scenarios:

1. Obtain an e-contract from a conventional contract
   - Context: legal (e.g. financial) contracts
2. Write the e-contract directly in a formal language
   - Context: web services, components, OO, etc

We are interested in both:

Definition

A contract is a document which engages several parties in a transaction and stipulates their obligations, rights, and prohibitions, as well as penalties in case of contract violations.

Outline

Aim

1. Give a formal language for specifying/writing contracts
2. Analyse contracts "internally"
   - Detect contradictions/inconsistencies statically
   - Determine the obligations (permissions, prohibitions) of a signatory
   - Detect superfluous contract clauses
   - ...
3. Monitor contracts
   - Run-time system to ensure the contract is respected
   - In case of contract violations, act accordingly
4. Develop a theory of contracts
   - Contract composition
   - Subcontracting
   - Conformance between a contract and the governing policies
   - *Meta-contracts*
   - ...

A Formal Language for Contracts

- A precise and concise syntax and a formal semantics
- Expressive enough as to capture natural contract clauses
- Restrictive enough to avoid the philosophical (deontic) paradoxes and be amenable to formal analysis
  - Model checking
  - Deductive verification
- Allow the representation of complex clauses especially conditional obligations, permissions, and prohibitions
- Allow the specification of (nested) contrary-to-duty (CTD) and contrary-to-prohibition (CTP)
  - CTD: when an obligation is not fulfilled
  - CTP: when a prohibition is violated
- We want to combine
  - The logical approach (e.g., dynamic, temporal, deontic logic)
  - The automata-like approach (labelled Kripke structures)

A Formal Language for Contracts

- A precise and concise syntax and a formal semantics
- Expressive enough as to capture natural contract clauses
- Restrictive enough to avoid the philosophical (deontic) paradoxes and be amenable to formal analysis
  - Model checking
  - Deductive verification
- Allow the representation of complex clauses especially conditional obligations, permissions, and prohibitions
- Allow the specification of (nested) contrary-to-duty (CTD) and contrary-to-prohibition (CTP)
  - CTD: when an obligation is not fulfilled
  - CTP: when a prohibition is violated
- We want to combine
  - The logical approach (e.g., dynamic, temporal, deontic logic)
  - The automata-like approach (labelled Kripke structures)

(Standard) Deontic Logic
Few Words

- Concerned with moral and normative notions
  - *obligation*, *permission*, *prohibition*, *optionality*, *power*, *indifference*, *immunity*, etc
- Focus on
  - The logical consistency of the above notions
  - The faithful representation of their intuitive meaning in law, moral systems, business organisations and security systems
- Difficult to avoid *puzzles* and *paradoxes*
  - Logical paradoxes, where we can deduce contradictory actions
  - "Practical oddities", where we can get counterintuitive conclusions
- Approaches
  - ought-to-do: expressions consider *names of actions*
    - "The Internet Provider *must send* a password to the Client"
  - ought-to-be: expressions consider state of affairs (results of actions)
    - "The average bandwidth *must be* more than 20kb/s"
- We'll only consider obligation, permission and prohibition over actions
  - Assertions define the "state of affairs"

(Standard) Deontic Logic
Few Words

- Concerned with moral and normative notions
  - *obligation*, *permission*, *prohibition*, *optionality*, *power*, *indifference*, *immunity*, etc
- Focus on
  - The logical consistency of the above notions
  - The faithful representation of their intuitive meaning in law, moral systems, business organisations and security systems
- Difficult to avoid *puzzles* and *paradoxes*
  - Logical paradoxes, where we can deduce contradictory actions
  - "Practical oddities", where we can get counterintuitive conclusions
- Approaches
  - ought-to-do: expressions consider *names of actions*
    - "The Internet Provider *must send* a password to the Client"
  - ought-to-be: expressions consider state of affairs (results of actions)
    - "The average bandwidth *must be* more than 20kb/s"
- We'll only consider obligation, permission and prohibition over actions
  - Assertions define the "state of affairs"

(Standard) Deontic Logic
Few Words

- Concerned with moral and normative notions
  - *obligation*, *permission*, *prohibition*, *optionality*, *power*, *indifference*, *immunity*, etc
- Focus on
  - The logical consistency of the above notions
  - The faithful representation of their intuitive meaning in law, moral systems, business organisations and security systems
- Difficult to avoid *puzzles* and *paradoxes*
  - Logical paradoxes, where we can deduce contradictory actions
  - "Practical oddities", where we can get counterintuitive conclusions
- Approaches
  - ought-to-do: expressions consider *names of actions*
    - "The Internet Provider *must send* a password to the Client"
  - ought-to-be: expressions consider state of affairs (results of actions)
    - "The average bandwidth *must be* more than 20kb/s"
- We'll only consider obligation, permission and prohibition over actions
  - Assertions define the "state of affairs"

- Concerned with moral and normative notions
  - *obligation*, *permission*, *prohibition*, *optionality*, *power*, *indifference*, *immunity*, etc
- Focus on
  - The logical consistency of the above notions
  - The faithful representation of their intuitive meaning in law, moral systems, business organisations and security systems
- Difficult to avoid *puzzles* and *paradoxes*
  - Logical paradoxes, where we can deduce contradictory actions
  - "Practical oddities", where we can get counterintuitive conclusions
- Approaches
  - ought-to-do: expressions consider *names of actions*
    - "The Internet Provider *must send* a password to the Client"
  - ought-to-be: expressions consider state of affairs (results of actions)
    - "The average bandwidth *must be* more than 20kb/s"
- We'll only consider obligation, permission and prohibition over actions
  - Assertions define the "state of affairs"

(Standard) Deontic Logic
Few Words

- Concerned with moral and normative notions
  - *obligation*, *permission*, *prohibition*, *optionality*, *power*, *indifference*, *immunity*, etc
- Focus on
  - The logical consistency of the above notions
  - The faithful representation of their intuitive meaning in law, moral systems, business organisations and security systems
- Difficult to avoid *puzzles* and *paradoxes*
  - Logical paradoxes, where we can deduce contradictory actions
  - "Practical oddities", where we can get counterintuitive conclusions
- Approaches
  - ought-to-do: expressions consider *names of actions*
    - "The Internet Provider *must send* a password to the Client"
  - ought-to-be: expressions consider state of affairs (results of actions)
    - "The average bandwidth *must be* more than 20kb/s"
- We'll only consider obligation, permission and prohibition over actions
  - Assertions define the "state of affairs"

Outline

The Contract Specification Language $\mathcal{CL}$

$$
\begin{array}{rcl}
\mathit{Contract} & := & \mathcal{D}\ ;\ \mathcal{C} \\
\mathcal{C} & := & \phi \mid \mathcal{C}_O \mid \mathcal{C}_P \mid \mathcal{C}_F \mid \mathcal{C} \wedge \mathcal{C} \mid [\alpha]\mathcal{C} \mid \langle\alpha\rangle\mathcal{C} \mid \mathcal{C}\,\mathcal{U}\,\mathcal{C} \mid \bigcirc\,\mathcal{C} \mid \Box\mathcal{C} \\
\mathcal{C}_O & := & O(\alpha) \mid \mathcal{C}_O \oplus \mathcal{C}_O \\
\mathcal{C}_P & := & P(\alpha) \mid \mathcal{C}_P \oplus \mathcal{C}_P \\
\mathcal{C}_F & := & F(\delta) \mid \mathcal{C}_F \vee [\alpha]\mathcal{C}_F
\end{array}
$$

- ϕ denotes assertions and ranges over Boolean expressions
- $O(\alpha)$, $P(\alpha)$, $F(\delta)$ specify obligation, permission (rights), and prohibition (forbidden) over actions
- α and δ are actions given in the definition part $\mathcal{D}$
- $[\alpha]$ and $\langle\alpha\rangle$ are the action parameterised modalities of dynamic logic
- $\mathcal{U}$, $\bigcirc$, and $\Box$ correspond to temporal logic operators
- $\wedge$, $\vee$, and $\oplus$ are conjunction, disjunction, and exclusive disjunction

The Contract Specification Language $\mathcal{CL}$

$$
\begin{array}{rcl}
\mathcal{C}ontract & := & \mathcal{D} \ ; \ \mathcal{C} \\
\mathcal{C} & := & \phi \mid \mathcal{C}_O \mid \mathcal{C}_P \mid \mathcal{C}_F \mid \mathcal{C} \wedge \mathcal{C} \mid [\alpha]\mathcal{C} \mid \langle\alpha\rangle\mathcal{C} \mid \mathcal{C}\,\mathcal{U}\,\mathcal{C} \mid \bigcirc\mathcal{C} \mid \Box\mathcal{C} \\
\mathcal{C}_O & := & O(\alpha) \mid \mathcal{C}_O \oplus \mathcal{C}_O \\
\mathcal{C}_P & := & P(\alpha) \mid \mathcal{C}_P \oplus \mathcal{C}_P \\
\mathcal{C}_F & := & F(\delta) \mid \mathcal{C}_F \vee [\alpha]\mathcal{C}_F
\end{array}
$$

- ϕ denotes assertions and ranges over Boolean expressions
- $O(\alpha)$, $P(\alpha)$, $F(\delta)$ specify obligation, permission (rights), and prohibition (forbidden) over actions
- α and δ are actions given in the definition part $\mathcal{D}$
- $[\alpha]$ and $\langle\alpha\rangle$ are the action parameterised modalities of dynamic logic
- $\mathcal{U}$, $\bigcirc$, and $\Box$ correspond to temporal logic operators
- $\wedge$, $\vee$, and $\oplus$ are conjunction, disjunction, and exclusive disjunction

Actions

- Actions are denoted by α and are constructed using the operators:
  - $+$ choice
  - $\cdot$ concatenation (sequencing)
  - $\&$ concurrent execution
- Tests as actions: $\phi?$
  - The behaviour of a test is like a *guard*; e.g. $\varphi? \cdot a$ *if the test succeeds then action a is performed*
  - Tests are used to model implication: $[\varphi?]\mathcal{C}$ is the same as $\varphi \Rightarrow \mathcal{C}$
- Action negation $\overline{\alpha}$
  - It represents all immediate traces that take us outside the trace of α
  - Involves the use of a *canonic form* of actions
  - E.g.: consider two atomic actions a and b then $\overline{a \cdot b}$ is $b + a \cdot a$

Actions

- Actions are denoted by α and are constructed using the operators:
  - $+$ choice
  - $\cdot$ concatenation (sequencing)
  - $\&$ concurrent execution
- Tests as actions: $\phi?$
  - The behaviour of a test is like a *guard*; e.g. $\varphi? \cdot a$ *if the test succeeds then action a is performed*
  - Tests are used to model implication: $[\varphi?]\mathcal{C}$ is the same as $\varphi \Rightarrow \mathcal{C}$
- Action negation $\overline{\alpha}$
  - It represents all immediate traces that take us outside the trace of α
  - Involves the use of a *canonic form* of actions
  - E.g.: consider two atomic actions a and b then $\overline{a \cdot b}$ is $b + a \cdot a$

Actions

- Actions are denoted by α and are constructed using the operators:
  - $+$ choice
  - $\cdot$ concatenation (sequencing)
  - $\&$ concurrent execution
- Tests as actions: $\phi?$
  - The behaviour of a test is like a *guard*; e.g. $\varphi? \cdot a$ *if the test succeeds then action a is performed*
  - Tests are used to model implication: $[\varphi?]\mathcal{C}$ is the same as $\varphi \Rightarrow \mathcal{C}$
- Action negation $\overline{\alpha}$
  - It represents all immediate traces that take us outside the trace of α
  - Involves the use of a *canonic form* of actions
  - E.g.: consider two atomic actions a and b then $\overline{a \cdot b}$ is $b + a \cdot a$

- *a&b*
- "The client must pay immediately, or the client must notify the service provider by sending an e-mail specifying that he delays the payment"

$$O(p) \oplus O(d\&n)$$

- $O(d\&n) \equiv O(d) \wedge O(n)$
- Action algebra enriched with a conflict relation to represent incompatible actions
  - $a =$ "increase Internet traffic" and $b =$ "decrease Internet traffic"
    - $a \#_{\mathcal{C}} b$
    - $O(a) \wedge O(b)$ gives an inconsistency

- $a \& b$
- "The client must pay immediately, or the client must notify the service provider by sending an e-mail specifying that he delays the payment"

$$O(p) \oplus O(d \& n)$$

- $O(d \& n) \equiv O(d) \wedge O(n)$
- Action algebra enriched with a conflict relation to represent incompatible actions
  - $a =$ "increase Internet traffic" and $b =$ "decrease Internet traffic"
    - $a \#_{\mathcal{C}} b$
    - $O(a) \wedge O(b)$ gives an inconsistency

- Expressing contrary-to-duty (CTDs)

$$O_{\mathcal{C}}(\alpha) = O(\alpha) \wedge [\overline{\alpha}]\mathcal{C}$$

- Expressing contrary-to-prohibition (CTPs)

$$F_{\mathcal{C}}(\alpha) = F(\alpha) \wedge [\alpha]\mathcal{C}$$

More on the Contract Language

Example

"In case the client delays the payment, after notification he must immediately lower the Internet traffic to the *low* level, and pay later twice. If the client does not lower the Internet traffic immediately, then the client will have to pay three times"

In $\mathcal{CL}$:

$$\Box([d\&n](O_{\mathcal{C}}(l) \wedge [l]\Diamond(O(p\&p))))$$

where

$$\mathcal{C} = \Diamond O(p\&p\&p)$$

Outline

Why μ-calculus?

- Expressive – embeds most of the used temporal and process logics
- Well studied – has a complete axiomatic and proof system
- Mathematically well founded on fix-point theory
- Possible to represent actions in the modal variant of μ-calculus
- Efficient algorithms for model checking
  - Tools

- The syntax of the $\mathcal{C}\mu$ logic
  $\varphi \;:=\; P \mid Z \mid P_c \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid [\gamma]\varphi \mid \mu Z.\varphi(Z)$

Main differences with respect to the classical μ-calculus:

1. P_c is set of propositional constants O_a and $\mathcal{F}_a$, one for each basic action a
   - Semantic restriction: $\|\mathcal{F}_a\|^{\mathcal{T}}_{\mathcal{V}} \cap \|O_a\|^{\mathcal{T}}_{\mathcal{V}} = \emptyset, \quad \forall a \in \mathcal{L}$
2. Multisets of basic actions: i.e. $\gamma = \{a, a, b\}$ is a label
3. Restricted non-determinism (more later)

- The syntax of the $\mathcal{C}\mu$ logic
  $\varphi \;:=\; P \mid Z \mid P_c \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid [\gamma]\varphi \mid \mu Z.\varphi(Z)$

Main differences with respect to the classical μ-calculus:

1. P_c is set of propositional constants O_a and $\mathcal{F}_a$, one for each basic action a
   - Semantic restriction: $\|\mathcal{F}_a\|^{\mathcal{T}}_{\mathcal{V}} \cap \|O_a\|^{\mathcal{T}}_{\mathcal{V}} = \emptyset, \quad \forall a \in \mathcal{L}$
2. Multisets of basic actions: i.e. $\gamma = \{a, a, b\}$ is a label
3. Restricted non-determinism (more later)

$$\|\top\|^{\mathcal{T}}_{\mathcal{V}} = \mathcal{S} \quad ; \quad \|P\|^{\mathcal{T}}_{\mathcal{V}} = \mathcal{V}_{Prop}(P)$$

$$\|Z\|^{\mathcal{T}}_{\mathcal{V}} = \mathcal{V}(Z) \quad ; \quad \|P_c\|^{\mathcal{T}}_{\mathcal{V}} = \mathcal{V}_{Prop}(P_c)$$

$$\|\neg\varphi\|^{\mathcal{T}}_{\mathcal{V}} = \mathcal{S} \setminus \|\varphi\|^{\mathcal{T}}_{\mathcal{V}}$$

$$\|\varphi \wedge \psi\|^{\mathcal{T}}_{\mathcal{V}} = \|\varphi\|^{\mathcal{T}}_{\mathcal{V}} \cap \|\psi\|^{\mathcal{T}}_{\mathcal{V}}$$

$$\|[\gamma]\varphi\|^{\mathcal{T}}_{\mathcal{V}} = \{s \mid \forall t \in \mathcal{S}.\ (s,t) \in R_\gamma \Rightarrow t \in \|\varphi\|^{\mathcal{T}}_{\mathcal{V}}\}$$

$$\|\nu Z.\varphi\|^{\mathcal{T}}_{\mathcal{V}} = \bigcup\{S \subseteq \mathcal{S} \mid S \subseteq \|\varphi\|^{\mathcal{T}}_{\mathcal{V}[Z:=S]}\}$$

$$\|\varphi \vee \psi\|^{\mathcal{T}}_{\mathcal{V}} = \|\varphi\|^{\mathcal{T}}_{\mathcal{V}} \cup \|\psi\|^{\mathcal{T}}_{\mathcal{V}}$$

$$\|\langle\gamma\rangle\varphi\|^{\mathcal{T}}_{\mathcal{V}} = \{s \mid \exists t \in \mathcal{S}.\ (s,t) \in R_\gamma \wedge t \in \|\varphi\|^{\mathcal{T}}_{\mathcal{V}}\}$$

$$\|\mu Z.\varphi\|^{\mathcal{T}}_{\mathcal{V}} = \bigcap\{S \subseteq \mathcal{S} \mid S \supseteq \|\varphi\|^{\mathcal{T}}_{\mathcal{V}[Z:=S]}\}$$

From $\mathcal{CL}$ to $\mathcal{C}\mu$

$$
\begin{array}{lrcl}
(1) & f^{\mathcal{T}}(O(\&_{i=1}^{n} a_i)) & = & \langle\{a_1,\ldots,a_n\}\rangle(\wedge_{i=1}^{n} O_{a_i}) \\
(2) & f^{\mathcal{T}}(\mathcal{C}_O \oplus \mathcal{C}_O) & = & f^{\mathcal{T}}(\mathcal{C}_O) \wedge f^{\mathcal{T}}(\mathcal{C}_O) \\
(3) & f^{\mathcal{T}}(P(\&_{i=1}^{n} a_i)) & = & \langle\{a_1,\ldots,a_n\}\rangle(\wedge_{i=1}^{n} \neg\mathcal{F}_{a_i}) \\
(4) & f^{\mathcal{T}}(\mathcal{C}_P \oplus \mathcal{C}_P) & = & f^{\mathcal{T}}(\mathcal{C}_P) \wedge f^{\mathcal{T}}(\mathcal{C}_P) \\
(5) & f^{\mathcal{T}}(F(\&_{i=1}^{n} a_i)) & = & [\{a_1,\ldots,a_n\}](\wedge_{i=1}^{n} \mathcal{F}_{a_i}) \\
(6) & f^{\mathcal{T}}(F(\delta) \vee [\beta]F(\delta)) & = & f^{\mathcal{T}}(F(\delta)) \vee f^{\mathcal{T}}([\beta]F(\delta)) \\
(7) & f^{\mathcal{T}}(\mathcal{C}_1 \wedge \mathcal{C}_2) & = & f^{\mathcal{T}}(\mathcal{C}_1) \wedge f^{\mathcal{T}}(\mathcal{C}_2) \\
(8) & f^{\mathcal{T}}(\bigcirc\mathcal{C}) & = & [\mathbf{any}]f^{\mathcal{T}}(\mathcal{C}) \\
(9) & f^{\mathcal{T}}(\mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) & = & \mu Z.f^{\mathcal{T}}(\mathcal{C}_2) \vee (f^{\mathcal{T}}(\mathcal{C}_1) \wedge [\mathbf{any}]Z \wedge \langle\mathbf{any}\rangle\top) \\
(10) & f^{\mathcal{T}}([\&_{i=1}^{n} a_i]\mathcal{C}) & = & [\{a_1,\ldots,a_n\}]f^{\mathcal{T}}(\mathcal{C}) \\
(11) & f^{\mathcal{T}}([(\&_{i=1}^{n} a_i)\alpha]\mathcal{C}) & = & [\{a_1,\ldots,a_n\}]f^{\mathcal{T}}([\alpha]\mathcal{C}) \\
(12) & f^{\mathcal{T}}([\alpha+\beta]\mathcal{C}) & = & f^{\mathcal{T}}([\alpha]\mathcal{C}) \wedge f^{\mathcal{T}}([\beta]\mathcal{C}) \\
(13) & f^{\mathcal{T}}([\varphi?]\mathcal{C}) & = & f^{\mathcal{T}}(\varphi) \Rightarrow f^{\mathcal{T}}(\mathcal{C})
\end{array}
$$

- Obligation

$$f^{\mathcal{T}}(O(\&_{i=1}^{n} a_i)) = \langle\{a_1, \ldots, a_n\}\rangle(\wedge_{i=1}^{n} O_{a_i})$$

$$f^{\mathcal{T}}(O(a\&b)) = \langle\{a, b\}\rangle(O_a \wedge O_b)$$

- Prohibition

$$f^{\mathcal{T}}(F(\&_{i=1}^{n} a_i)) = [\{a_1, \ldots, a_n\}](\wedge_{i=1}^{n} \mathcal{F}_{a_i})$$

$$f^{\mathcal{T}}(F(a)) = [\{a\}]\mathcal{F}_a$$

- Permission

$$f^{\mathcal{T}}(P(\&_{i=1}^{n} a_i)) = \langle\{a_1, \ldots, a_n\}\rangle(\wedge_{i=1}^{n} \neg\mathcal{F}_{a_i})$$

$$f^{\mathcal{T}}(P(a)) = \langle a\rangle(\neg\mathcal{F}_a)$$

- Obligation

$$f^{\mathcal{T}}(O(\&_{i=1}^{n} a_i)) = \langle\{a_1, \ldots, a_n\}\rangle(\wedge_{i=1}^{n} O_{a_i})$$

$$f^{\mathcal{T}}(O(a\&b)) = \langle\{a, b\}\rangle(O_a \wedge O_b)$$

- Prohibition

$$f^{\mathcal{T}}(F(\&_{i=1}^{n} a_i)) = [\{a_1, \ldots, a_n\}](\wedge_{i=1}^{n} \mathcal{F}_{a_i})$$

$$f^{\mathcal{T}}(F(a)) = [\{a\}]\mathcal{F}_a$$

- Permission

$$f^{\mathcal{T}}(P(\&_{i=1}^{n} a_i)) = \langle\{a_1, \ldots, a_n\}\rangle(\wedge_{i=1}^{n} \neg\mathcal{F}_{a_i})$$

$$f^{\mathcal{T}}(P(a)) = \langle a\rangle(\neg\mathcal{F}_a)$$

- Obligation

$$f^{\mathcal{T}}(O(\&_{i=1}^{n} a_i)) = \langle\{a_1, \ldots, a_n\}\rangle(\wedge_{i=1}^{n} O_{a_i})$$

$$f^{\mathcal{T}}(O(a\&b)) = \langle\{a, b\}\rangle(O_a \wedge O_b)$$

- Prohibition

$$f^{\mathcal{T}}(F(\&_{i=1}^{n} a_i)) = [\{a_1, \ldots, a_n\}](\wedge_{i=1}^{n} \mathcal{F}_{a_i})$$

$$f^{\mathcal{T}}(F(a)) = [\{a\}]\mathcal{F}_a$$

- Permission

$$f^{\mathcal{T}}(P(\&_{i=1}^{n} a_i)) = \langle\{a_1, \ldots, a_n\}\rangle(\wedge_{i=1}^{n} \neg\mathcal{F}_{a_i})$$

$$f^{\mathcal{T}}(P(a)) = \langle a\rangle(\neg\mathcal{F}_a)$$

Outline

Ross's paradox

1. It is obligatory that one mails the letter
2. It is obligatory that one mails the letter or one destroys the letter

In Standard Deontic Logic (SDL) these are expressed as:

1. $O(p)$
2. $O(p \vee q)$

Problem: in SDL one can infer that $O(p) \Rightarrow O(p \vee q)$

Avoided in $\mathcal{CL}$ –Proof Sketch:

- $f^{\mathcal{T}}(O(a)) = \langle a \rangle O_a$
- $O(a+b) \equiv O(a) \oplus O(b) \stackrel{f^{\mathcal{T}}}{=} \langle a \rangle O_a \wedge \langle b \rangle O_b$
- $\langle a \rangle O_a \not\Rightarrow \langle a \rangle O_a \wedge \langle b \rangle O_b$

Ross's paradox

1. It is obligatory that one mails the letter
2. It is obligatory that one mails the letter or one destroys the letter

In Standard Deontic Logic (SDL) these are expressed as:

1. $O(p)$
2. $O(p \vee q)$

Problem: in SDL one can infer that $O(p) \Rightarrow O(p \vee q)$

Avoided in $\mathcal{CL}$ –Proof Sketch:

- $f^{\mathcal{T}}(O(a)) = \langle a \rangle O_a$
- $O(a+b) \equiv O(a) \oplus O(b) \stackrel{f^{\mathcal{T}}}{=} \langle a \rangle O_a \wedge \langle b \rangle O_b$
- $\langle a \rangle O_a \not\Rightarrow \langle a \rangle O_a \wedge \langle b \rangle O_b$

Properties of the contract language

Theorem

The following paradoxes are avoided in $\mathcal{CL}$:

- *Ross's paradox*
- *The Free Choice Permission paradox*
- *Sartre's dilemma*
- *The Good Samaritan paradox*
- *Chisholm's paradox*
- *The Gentle Murderer paradox*

Properties of the contract language (II)

Theorem

The following hold in $\mathcal{CL}$:

- $P(\alpha) \equiv \neg F(\alpha)$
- $O(\alpha) \Rightarrow P(\alpha)$
- $P(a) \not\Rightarrow P(a\&b)$
- $F(a) \not\Rightarrow F(a\&b)$
- $F(a\&b) \not\Rightarrow F(a)$
- $P(a\&b) \not\Rightarrow P(a)$

Outline

Model Checking Contracts

1. Model the conventional contract written in English using the formal language $\mathcal{CL}$;
2. Translate syntactically the $\mathcal{CL}$ specification into the extended μ-calculus $\mathcal{C}\mu$;
3. Obtain a Kripke-like model (a labelled transition system, LTS) of the $\mathcal{C}\mu$ formulae;
4. Translate the LTS into the input language of NuSMV;
5. Perform model checking using NuSMV;
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. Finally, repair the original contract by adding a corresponding clause, if applicable.

Case Study
Contract in English

1. The **Client** shall not 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. The **Provider** may, at its sole discretion, without notice or giving any reason or incurring any liability for doing so: Suspend Internet Services immediately if **Client** is in breach of Clause 1;

$$
\begin{aligned}
\phi &= \text{the Internet traffic is high}\\
fi &= \text{client supplies false information to Client Relations Department}\\
h &= \text{client increases Internet traffic to } high \text{ level}\\
p &= \text{client pays [price]}\\
d &= \text{client delays payment}\\
n &= \text{client notifies by e-mail}\\
l &= \text{client lowers the Internet traffic}\\
sfD &= \text{client sends the Personal Data Form to the Client Relations Department}\\
o &= \text{provider activates the Internet Service (it becomes operative)}\\
s &= \text{provider suspends service}
\end{aligned}
$$

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. $\mathcal{CL}$ into $\mathcal{C}\mu$ (not showing the outer $\Box$)
   1. $[fi]\mathcal{F}_{fi} \wedge [fi]\langle s\rangle P_s$
   2. $[h](\phi \Rightarrow (\langle p\rangle O_p \wedge \langle\{d,n\}\rangle(O_d \wedge O_n)))$
   3. $[\{d,n\}](\langle l\rangle O_l \wedge [l](\mu Z.\langle\{p,p\}\rangle O_p \vee ([\mathbf{any}]Z \wedge \langle\mathbf{any}\rangle\top)))$
   4. $[\{d,n\}][\bar{l}](\mu Z.\langle\{p,p,p\}\rangle O_p \vee ([\mathbf{any}]Z \wedge \langle\mathbf{any}\rangle\top))$
   5. $[o]\langle sfD\rangle O_{sfD}$
2. From $\mathcal{C}\mu$ to input language in NuSMV (using *direct specification*)
3. Model checking
   1. Prove model satisfies the original clauses (represented in LTL)
   2. Property about client obligations: "*After the Internet is high and the client pays then the client is not obliged to pay again immediately*"
      - FALSE – Modify the contract
   3. Property about payment: "*If the Internet is high and the client delays and notifies, and afterwards lowers the Internet traffic, the client does not pay twice until the Internet traffic is high again*"
      - FALSE – Modify the contract

Case Study
Model Checking

1. $\mathcal{CL}$ into $\mathcal{C}\mu$ (not showing the outer $\Box$)
   1. $[fi]\mathcal{F}_{fi} \wedge [fi]\langle s\rangle P_s$
   2. $[h](\phi \Rightarrow (\langle p\rangle O_p \wedge \langle\{d,n\}\rangle(O_d \wedge O_n)))$
   3. $[\{d,n\}](\langle l\rangle O_l \wedge [l](\mu Z.\langle\{p,p\}\rangle O_p \vee ([\mathbf{any}]Z \wedge \langle\mathbf{any}\rangle\top)))$
   4. $[\{d,n\}][\bar{l}](\mu Z.\langle\{p,p,p\}\rangle O_p \vee ([\mathbf{any}]Z \wedge \langle\mathbf{any}\rangle\top))$
   5. $[o]\langle sfD\rangle O_{sfD}$
2. From $\mathcal{C}\mu$ to input language in NuSMV (using *direct specification*)
3. Model checking
   1. Prove model satisfies the original clauses (represented in LTL)
   2. Property about client obligations: "*After the Internet is high and the client pays then the client is not obliged to pay again immediately*"
      - FALSE – Modify the contract
   3. Property about payment: "*If the Internet is high and the client delays and notifies, and afterwards lowers the Internet traffic, the client does not pay twice until the Internet traffic is high again*"
      - FALSE – Modify the contract

1. $\mathcal{CL}$ into $\mathcal{C}\mu$ (not showing the outer $\Box$)
   1. $[fi]\mathcal{F}_{fi} \wedge [fi]\langle s\rangle P_s$
   2. $[h](\phi \Rightarrow (\langle p\rangle O_p \wedge \langle\{d,n\}\rangle(O_d \wedge O_n)))$
   3. $[\{d,n\}](\langle l\rangle O_l \wedge [l](\mu Z.\langle\{p,p\}\rangle O_p \vee ([\mathbf{any}]Z \wedge \langle\mathbf{any}\rangle\top)))$
   4. $[\{d,n\}][\bar{l}](\mu Z.\langle\{p,p,p\}\rangle O_p \vee ([\mathbf{any}]Z \wedge \langle\mathbf{any}\rangle\top))$
   5. $[o]\langle sfD\rangle O_{sfD}$
2. From $\mathcal{C}\mu$ to input language in NuSMV (using *direct specification*)
3. Model checking
   1. Prove model satisfies the original clauses (represented in LTL)
   2. Property about client obligations: "*After the Internet is high and the client pays then the client is not obliged to pay again immediately*"
      - FALSE – Modify the contract
   3. Property about payment: "*If the Internet is high and the client delays and notifies, and afterwards lowers the Internet traffic, the client does not pay twice until the Internet traffic is high again*"
      - FALSE – Modify the contract

Outline

Main Features of $\mathcal{CL}$

We have seen:

- A formal specification language for contracts with semantics based on a variant of μ-calculus
- The language
  - Is specially tailored for specifying contracts
  - Adopts the ought-to-do approach
  - Allows the representation of conditional obligations, permissions and prohibitions
  - Allows the representation of nested CTDs and CTPs
  - Is proved to avoid many of the principal deontic paradoxes
  - Enjoys some nice desirable properties
  - Combines the logic approach with the automata-like approach
- Initial ideas on how to model check contracts

Limitations / Questions

- Action algebra
  - Differentiate between $\neg a$ and $\overline{a}$
  - Study better the use of $\overline{a}$ (only on CTDs?)
  - Restricted non-determinism –introduce priorities
- Ought-to-do vs ought-do-be?
- Next operator not good for refinement
- Restrictions on prohibitions (needs better study)
- No notion of time
  - Timed $\mu-$calculus, TCTL, ...?
  - Time associated with actions or formulae?
  - Durations, time stamps, beginning and end, dates, ...?

Further Work

- Add time
- Semantics
  - $\mu-$calculus vs CTL vs ...
- Develop a theory of contracts
  - "Normative" automata
- Model checking
  - Automate the process
  - Model synthesis
- Further theoretical investigations of the underlying actions
- Contract-as-types (Curry-Howard isomorphism) (?!)

Links and Papers

- Nordunet3 project "Contract-Oriented Software Development for Internet Services" (http://folk.uio.no/gerardo/nordunet3/index.shtml)

- C. Prisacariu and G. Schneider. A formal language for electronic contracts. In FMOODS'07, vol. 4468 of LNCS, pages 174-189, June 2007
- C. Prisacariu and G. Schneider. Model Checking Contracts –a case study. In ATVA'07, to appear in LNCS, October 2007

- FLACOS'07 – First Workshop on Formal Languages and Analysis of Contract-Oriented Software (http://www.ifi.uio.no/flacos07/)
  - Oslo, 9-10 October 2007

Thank you!

Rewriting Rules for Obligations

(1)	$O(a) \wedge O(b)$	$\rightsquigarrow$	$O(a\&b)$
(2)	$O(a) \wedge O(a\&b)$	$\rightsquigarrow$	$O(a\&b)$
(3)	$O(a) \wedge (O(a) \oplus O(b))$	$\rightsquigarrow$	$O(a)$
(4)	$O(a) \wedge O(a)$	$\rightsquigarrow$	$O(a)$
(5)	$O(a) \oplus O(a)$	$\rightsquigarrow$	$O(a)$
(6)	$O(c) \wedge (O(a) \oplus O(b))$	$\rightsquigarrow$	$(O(c) \wedge O(a)) \oplus (O(c) \wedge O(b))$
(7)	$(\oplus_i O(a_i)) \wedge (\oplus_j O(b_j))$	$\rightsquigarrow$	$\oplus_{i,j}(O(a_i) \wedge O(b_j)) \quad a_i \neq b_j$

Table: Rewriting rules for obligation O

Compositional Rules

$$
\begin{array}{lrcl}
(1) & O(\alpha + \beta) & \equiv & O(\alpha) \oplus O(\beta) \\
(2) & O(a\&b) & \equiv & O(a) \wedge O(b) \\
(3) & O(\alpha\beta) & \equiv & O(\alpha) \wedge [\alpha]O(\beta) \\
(4) & P(\alpha + \beta) & \equiv & P(\alpha) \oplus P(\beta) \\
(5) & P(\alpha\beta) & \equiv & P(\alpha) \wedge \langle\alpha\rangle P(\beta) \\
(6) & F(\alpha\beta) & \equiv & F(\alpha) \vee [\alpha]F(\beta)
\end{array}
$$

Table: Compositional rules

Paradoxes and Practical Oddities

- Deontic paradoxes. A paradox is an apparently true statement that leads to a contradiction, or a situation which is counter-intuitive.
  - *The Gentle Murderer Paradox*
    1. It is obligatory that John does not kill his mother;
    2. If John does kill his mother, then it is obligatory that John kills her gently;
    3. John does kill his mother.

    It could be possible to infer that John is obliged to kill his mother
- Practical oddities. A situation where you can infer two assertions which are contradictory from the intuitive practical point of view, though they might not represent a logical contradiction
  - Assume you have the following norms and facts:
    1. Keep your promise;
    2. If you haven't kept your promise, apologise;
    3. You haven't kept your promise.

    It could be possible to deduce that you are both obliged to keep your promise and to apologise for not keeping it

Paradoxes and Practical Oddities

- Deontic paradoxes. A paradox is an apparently true statement that leads to a contradiction, or a situation which is counter-intuitive.
  - *The Gentle Murderer Paradox*
    1. It is obligatory that John does not kill his mother;
    2. If John does kill his mother, then it is obligatory that John kills her gently;
    3. John does kill his mother.

    It could be possible to infer that John is obliged to kill his mother
- Practical oddities. A situation where you can infer two assertions which are contradictory from the intuitive practical point of view, though they might not represent a logical contradiction
  - Assume you have the following norms and facts:
    1. Keep your promise;
    2. If you haven't kept your promise, apologise;
    3. You haven't kept your promise.

    It could be possible to deduce that you are both obliged to keep your promise and to apologise for not keeping it

1. You may either sleep on the sofa or sleep on the bed.
2. You may sleep on the sofa and you may sleep on the bed.

In SDL this is:

1. $P(p \vee q)$
2. $P(p) \wedge P(q)$

The natural intuition tells that $P(p \vee q) \Rightarrow P(p) \wedge P(q)$. In SDL this would lead to $P(p) \Rightarrow P(p \vee q)$ which is $P(p) \Rightarrow P(p) \wedge P(q)$, so $P(p) \Rightarrow P(q)$. As an example: *If one is permitted something, then one is permitted anything*.

1. It is obligatory I now meet Jones (as promised to Jones).
2. It is obligatory I now do not meet Jones (as promised to Smith).

In SDL this is:

1. $O(p)$
2. $O(\neg p)$

The problem is that in the natural language the two obligations are intuitive and often happen, where the logical formulae are inconsistent when put together (in conjunction) in SDL. (In SDL, $O(p) \Rightarrow \neg O(\neg p)$ and we get a contradiction.)

1. It ought to be the case that Jones helps Smith who has been robbed.
2. It ought to be the case that Smith has been robbed.

And one naturally infers that:

> Jones helps Smith who has been robbed if and only if Jones helps Smith and Smith has been robbed.

In SDL the first two are expressed as:

1. $O(p \wedge q)$
2. $O(q)$

The problem is that in SDL one can derive that $O(p \wedge q) \Rightarrow O(q)$ which is counter intuitive in the natural language, as in the example above.

1. John ought to go to the party.
2. If John goes to the party then he ought to tell them he is coming.
3. If John does not go to the party then he ought not to tell them he is coming.
4. John does not go to the party.

In Standard Deontic Logic (SDL) these are expressed as:

1. $O(p)$
2. $O(p \Rightarrow q)$
3. $\neg p \Rightarrow O(\neg q)$
4. $\neg p$

The problem is that in SDL one can infer $O(q) \wedge O(\neg q)$ which is due to statement (2).

1. It is obligatory that John does not kill his mother.
2. If John does kill his mother, then it is obligatory that John kills her gently.
3. John does kill his mother.

In Standard Deontic Logic (SDL) these are expressed as:

1. $O(\neg p)$
2. $p \Rightarrow O(q)$
3. p

The problem is that when adding a natural inference like $q \Rightarrow p$ then in SDL one can infer that $O(p)$.