Certified Management of Financial Contracts

Patrick Bahr

DIKU
`paba@diku.dk`

joint work with
Jost Berthold & Martin Elsman

10th December, 2014

Introduction

What are financial contracts?

- stipulate future transactions between different parties
- have time constraints
- may depend on stock prices, exchange rates etc.

Introduction

What are financial contracts?

- stipulate future transactions between different parties
- have time constraints
- may depend on stock prices, exchange rates etc.

Example (Foreign Exchange Option)

At any time within the next 90 days, party X may decide to buy USD 100 from party Y, for a fixed rate r of Danish Kroner.

Introduction

What are financial contracts?

- stipulate future transactions between different parties
- have time constraints
- may depend on stock prices, exchange rates etc.

Example (Foreign Exchange Option)

At any time within the next 90 days, party X may decide to buy USD 100 from party Y, for a fixed rate r of Danish Kroner.

Goals

- Express such contracts in a formal language
- Symbolic manipulation and analysis of such contracts.

Introduction

What are financial contracts?

- stipulate future transactions between different parties
- have time constraints
- may depend on stock prices, exchange rates etc.

Example (Foreign Exchange Option)

At any time within the next 90 days, party X may decide to buy USD 100 from party Y, for a fixed rate r of Danish Kroner.

Goals

- Express such contracts in a formal language
- Symbolic manipulation and analysis of such contracts.
- Formal verification

Contract Language Goals in Detail

- **Compositionality**.
  Contracts are time-relative $\Rightarrow$ facilitates compositionality
- **Multi-party**.
  Specify obligations and opportunities for multiple parties,
  (which opens up the possibility for specifying portfolios)
- **Contract management**.
  Contracts can be managed and symbolically evolved;
  a contract gradually reduces to the empty contract.
- **Contract utilities (symbolic)**.
  Contracts can be analysed in a variety of ways
- **Contract pricing (numerical, staged)**.
  Code for payoff can be generated from contracts
  (input to a stochastic pricing engine)

Example

Contract in natural language

- At any time within the next 90 days,
- party X may decide to
- buy USD 100 from party Y,
- for a fixed rate r of Danish Kroner.

Example

Contract in natural language

- At any time within the next 90 days,
- party X may decide to
- buy USD 100 from party Y,
- for a fixed rate *r* of Danish Kroner.

Translation into contract language

$$\begin{array}{l} \textbf{if } \mathit{obs}(X \text{ exercises option}) \textbf{ within } 90 \\ \textbf{then } 100 \times (\mathit{USD}(Y \to X) \,\&\, r \times \mathit{DKK}(X \to Y)) \\ \textbf{else } \emptyset \end{array}$$

Contributions

- Denotational semantics based on cash-flows
- Reduction semantics (sound and complete)
- Correctness proofs for common contract analyses and transformations
- Formalised in the Coq theorem prover
- Certified implementation via code extraction

An Overview of the Contract Language

$\emptyset$	empty contract with no obligations
$a(p_1 \to p_2)$	p_1 has to transfer one unit of a to p_2
$c_1 \,\&\, c_2$	conjunction of c_1 and c_2
$e \times c$	multiply all obligations in c by e
$d \uparrow c$	shift c into the future by d days
$\textbf{let}\ x = e\ \textbf{in}\ c$	observe today's value of e at any time (via x)

An Overview of the Contract Language

$\emptyset$	empty contract with no obligations
$a(p_1 \to p_2)$	p_1 has to transfer one unit of a to p_2
$c_1 \,\&\, c_2$	conjunction of c_1 and c_2
$e \times c$	multiply all obligations in c by e
$d \uparrow c$	shift c into the future by d days
let $x = e$ **in** c	observe today's value of e at any time (via x)

if e **within** d **then** c_1 **else** c_2

- behave like c_1 as soon as e becomes true
- if e does not become true within d days behave like c_2

An Overview of the Contract Language

$\emptyset$	empty contract with no obligations
$a(p_1 \to p_2)$	p_1 has to transfer one unit of a to p_2
$c_1 \,\&\, c_2$	conjunction of c_1 and c_2
$e \times c$	multiply all obligations in c by e
$d \uparrow c$	shift c into the future by d days
let $x = e$ **in** c	observe today's value of e at any time (via x)

if e **within** d **then** c_1 **else** c_2

- behave like c_1 as soon as e becomes true
- if e does not become true within d days behave like c_2

Expression Language

Real-valued and Boolean-valued expressions, extended by

$obs(l, d)$	observe the value of l at time d
$acc(f, d, e)$	accumulation over the last d days

Example: Asian Option

$$
\begin{aligned}
90 \uparrow\ & \textbf{if}\ \mathit{obs}(X \text{ exercises option})\ \textbf{within}\ 0 \\
& \textbf{then}\ 100 \times (\mathit{USD}(Y \to X) \,\&\, (\mathit{rate} \times \mathit{DKK}(X \to Y))) \\
& \textbf{else}\ \emptyset
\end{aligned}
$$

where

$$\mathit{rate} = \frac{1}{30} \cdot \mathit{acc}(\lambda r.r + \mathit{obs}(\mathit{FX}\ \ \mathit{USD}/\mathit{DKK}), 30, 0)$$

Denotational Semantics

The semantics of a contract is given by the cash-flow it stipulates.

$$\mathcal{C}\left[\!\left[\cdot\right]\!\right]_{\cdot} : \mathsf{Contr} \qquad\qquad \rightarrow \mathsf{CashFlow}$$

Denotational Semantics

The semantics of a contract is given by the cash-flow it stipulates.

$$\mathcal{C}\left[\!\left[\cdot\right]\!\right]_{\cdot} : \text{Contr} \qquad\qquad \rightarrow \text{CashFlow}$$

$$\begin{aligned} \text{CashFlow} &= \mathbb{N} \rightarrow \text{Transactions} \\ \text{Transactions} &= \text{Party} \times \text{Party} \times \text{Asset} \rightarrow \mathbb{R} \end{aligned}$$

Denotational Semantics

The semantics of a contract is given by the cash-flow it stipulates.

$$\mathcal{C}[\![\cdot]\!]_{\cdot} : \mathsf{Contr} \times \mathsf{Env} \rightarrow \mathsf{CashFlow}$$
$$\mathsf{Env} = \mathsf{Label} \times \mathbb{Z} \rightarrow \mathbb{B} \cup \mathbb{R}$$

$$\mathsf{CashFlow} = \mathbb{N} \rightarrow \mathsf{Transactions}$$
$$\mathsf{Transactions} = \mathsf{Party} \times \mathsf{Party} \times \mathsf{Asset} \rightarrow \mathbb{R}$$

Denotational Semantics

The semantics of a contract is given by the cash-flow it stipulates.

$$\mathcal{C}[\![\cdot]\!]_{\cdot} : \mathsf{Contr} \times \mathsf{Env} \to \mathsf{CashFlow}$$

$$\mathsf{Env} = \mathsf{Label}_{\alpha} \times \mathbb{Z} \to \alpha$$

$$\mathsf{CashFlow} = \mathbb{N} \to \mathsf{Transactions}$$

$$\mathsf{Transactions} = \mathsf{Party} \times \mathsf{Party} \times \mathsf{Asset} \to \mathbb{R}$$

Contract Analyses

Examples

- contract dependencies
- contract causality
- contract horizon

Contract Analyses

Examples

- contract depender
- contract causality
- contract horizon

$obs(FX\ USD/DKK, 1) \times DKK(X \to Y)$

Contract Analyses

Examples

- contract dependencies
- contract causality
- contract horizon

Semantics vs. Syntax

- these analyses have precise semantic definition
- they cannot be effectively computed
- we provide sound approximations, e.g. type system

Contract Causality

Refined Types

- $e : \mathrm{Expr}^{t}_{\alpha}$ value of e available at time t (or later)
- $c : \mathrm{Contr}^{t}$ no obligations strictly before t

Contract Causality

Refined Types

- $e : \text{Expr}_\alpha^t$ value of e available at time t (or later)
- $c : \text{Contr}^t$ no obligations strictly before t

Typing Rules

$$\frac{t_1, t_2 \in \mathbb{Z} \quad l \in \text{Label}_\alpha \quad t_1 \leq t_2}{\Gamma \vdash obs(l, t_1) : \text{Expr}_\alpha^{t_2}} \qquad \frac{p_1, p_2 \in \text{Party} \quad a \in \text{Asset}}{\vdash a(p_1 \to p_2) : \text{Contr}^0}$$

Contract Causality

Refined Types

- $e : \mathsf{Expr}_{\alpha}^{t}$ value of e available at time t (or later)
- $c : \mathsf{Contr}^{t}$ no obligations strictly before t

Typing Rules

$$\frac{t_1, t_2 \in \mathbb{Z} \quad l \in \mathsf{Label}_{\alpha} \quad t_1 \leq t_2}{\Gamma \vdash obs(l, t_1) : \mathsf{Expr}_{\alpha}^{t_2}} \qquad \frac{p_1, p_2 \in \mathsf{Party} \quad a \in \mathsf{Asset}}{\vdash a(p_1 \to p_2) : \mathsf{Contr}^{0}}$$

$$\frac{\vdash e : \mathsf{Expr}_{\mathbb{R}}^{t} \quad \vdash c : \mathsf{Contr}^{t}}{\vdash e \times c : \mathsf{Contr}^{t}} \qquad \frac{d \in \mathbb{N} \quad \vdash c : \mathsf{Contr}^{t}}{\vdash d \uparrow c : \mathsf{Contr}^{t+d}}$$

Contract Causality

Refined Types

- $e : \mathrm{Expr}_\alpha^t$ value of e available at time t (or later)
- $c : \mathrm{Contr}^t$ no obligations strictly before t

Typing Rules

$$\frac{t_1, t_2 \in \mathbb{Z} \quad l \in \mathrm{Label}_\alpha \quad t_1 \leq t_2}{\Gamma \vdash obs(l, t_1) : \mathrm{Expr}_\alpha^{t_2}}$$

$$\frac{p_1, p_2 \in \mathrm{Party} \quad a \in \mathrm{Asset}}{\vdash a(p_1 \to p_2) : \mathrm{Contr}^0}$$

$$\frac{\vdash e : \mathrm{Expr}_\mathbb{R}^t \quad \vdash c : \mathrm{Contr}^t}{\vdash e \times c : \mathrm{Contr}^t}$$

$$\frac{d \in \mathbb{N} \quad \vdash c : \mathrm{Contr}^t}{\vdash d \uparrow c : \mathrm{Contr}^{t+d}}$$

$$\vdots$$

Contract Transformations

Contract equivalences

When can we replace a sub-contract with another one, without changing the semantics of the contract?

Reduction semantics

What does the contract look like after n days have passed?

Contract Specialisation

What does the contract look like after we learned the actual value of some observables?

Contract Equivalences

$$e_1 \times (e_2 \times c) \simeq (e_1 \cdot e_2) \times c \qquad d \uparrow \emptyset \simeq \emptyset$$

$$d_1 \uparrow (d_2 \uparrow c) \simeq (d_1 + d_2) \uparrow c \qquad r \times \emptyset \simeq \emptyset$$

$$d \uparrow (c_1 \,\&\, c_2) \simeq (d \uparrow c_1) \,\&\, (d \uparrow c_2) \qquad 0 \times c \simeq \emptyset$$

$$e \times (c_1 \,\&\, c_2) \simeq (e \times c_1) \,\&\, (e \times c_2) \qquad c \,\&\, \emptyset \simeq c$$

$$d \uparrow (e \times c) \simeq (d \Uparrow e) \times (d \uparrow c) \qquad c_1 \,\&\, c_2 \simeq c_2 \,\&\, c_1$$

$$\begin{aligned} & d \uparrow \textbf{if } b \textbf{ within } e \textbf{ then } c_1 \textbf{ else } c_2 \simeq \\ & \quad \textbf{if } d \Uparrow b \textbf{ within } e \textbf{ then } d \uparrow c_1 \textbf{ else } d \uparrow c_2 \end{aligned}$$

$$(e_1 \times a(p_1 \to p_2)) \,\&\, (e_2 \times a(p_1 \to p_2)) \simeq (e_1 + e_2) \times a(p_1 \to p_2)$$

Reduction Semantics

$$c \overset{\tau}{\Longrightarrow}_{\rho} c'$$

Reduction Semantics

$$c \overset{\tau}{\Longrightarrow}_{\rho} c'$$

$$\frac{}{a(p_1 \to p_2) \overset{\tau_{a,p_1,p_2}}{\Longrightarrow}_{\rho} \emptyset}$$

Reduction Semantics

$$c \overset{\tau}{\Longrightarrow}_{\rho} c'$$

$$\frac{}{a(p_1 \rightarrow p_2) \overset{\tau_{a,p_1,p_2}}{\Longrightarrow}_{\rho} \emptyset} \qquad \frac{c \overset{\tau}{\Longrightarrow}_{\rho} c' \quad \mathcal{E}\,[\![e]\!]_{\rho} = v}{e \times c \overset{v * \tau}{\Longrightarrow}_{\rho} (-1 \Uparrow e) \times c'}$$

Reduction Semantics

$$c \overset{\tau}{\Longrightarrow}_{\rho} c'$$

$$\frac{}{a(p_1 \to p_2) \overset{\tau_{a,p_1,p_2}}{\Longrightarrow}_{\rho} \emptyset} \qquad \frac{c \overset{\tau}{\Longrightarrow}_{\rho} c' \quad \mathcal{E}\left[\!\left[e \right]\!\right]_{\rho} = v}{e \times c \overset{v * \tau}{\Longrightarrow}_{\rho} (-1 \Uparrow e) \times c'}$$

$$\vdots$$

Reduction Semantics

$$c \xrightarrow{\tau}_{\rho} c'$$

$$\frac{}{a(p_1 \to p_2) \xrightarrow{\tau_{a,p_1,p_2}}_{\rho} \emptyset} \qquad \frac{c \xrightarrow{\tau}_{\rho} c' \quad \mathcal{E}\,[\![e]\!]_{\rho} = v}{e \times c \xrightarrow{v * \tau}_{\rho} (-1 \Uparrow e) \times c'}$$

$$\vdots$$

Theorem (Reduction semantics correctness)

(i) *If* $c \xrightarrow{\tau}_{\rho} c'$, *then*

(a) $\mathcal{C}\,[\![c]\!]_{\rho}\,(0) = \tau$, *and*

(b) $\mathcal{C}\,[\![c]\!]_{\rho}\,(i+1) = \mathcal{C}\,[\![c']\!]_{1 \Uparrow \rho}\,(i)$ *for all* $i \in \mathbb{N}$.

(ii) *If* $\mathcal{C}\,[\![c]\!]_{\rho}\,(0) = \tau$, *then there is a unique* c' *with* $c \xrightarrow{\tau}_{\rho} c'$.

Code Extraction

Coq formalisation

- Denotational & reduction semantics
- Meta-theory of contracts (causality, monotonicity, . . .)
- Definition of contract transformations and analyses
- Correctness proofs

Code Extraction

Coq formalisation

- Denotational & reduction semantics
- Meta-theory of contracts (causality, monotonicity, . . .)
- Definition of contract transformations and analyses
- Correctness proofs

Code Extraction

Coq formalisation

- Denotational & reduction semantics
- Meta-theory of contracts (causality, monotonicity, . . .)
- Definition of contract transformations and analyses
- Correctness proofs

Extraction of executable Haskell code

- efficient Haskell implementation
- embedded domain-specific language for contracts
- contract analyses and contract management

Future Work

- improve code extraction
- further analyses and transformations
  (e.g. scenario generation and “zooming”)
- combine this work with numerical methods