

# Contract-based Discovery and Adaptation of Web Services

Luca Padovani

Jointly with  
Samuele Carpineti, Giuseppe Castagna, Nils Gesbert,  
Cosimo Laneve

9th International School on Formal Methods for the Design of  
Computer, Communication and Software Systems: Web Services

# Contracts for Web services

## We've got...

- behavioral descriptions of Web services (wsdl, wscl, ws-bpel, put your favorite technology here)
- repositories of Web services descriptions (uddi)

## We'd like to...

- look for Web services with a given behavior
- see if it's safe to replace a service with another one
- if not, see whether we can *adapt* one service to safely replace another

## We're gonna use...

- **contracts** = abstractions of Web services' behavior

# Finding Web services by contract

Compliance = client's satisfaction

$$\rho \dashv \sigma$$

Running a query with *compliance*

$$\mathcal{Q}(\rho) = \{\sigma \mid \rho \dashv \sigma\}$$

Running a query with *duality*  $\rho^\perp$  and *subcontract*  $\sigma \preceq \tau$

$$\mathcal{Q}(\rho) = \{\sigma \mid \rho^\perp \preceq \sigma\}$$

# The quest for $\preceq$

## Desired properties of $\preceq$

- **reduction** of nondeterminism ( $a \oplus b \preceq a$ )
- **extension** of functionalities ( $a \preceq a + b$ )
- some **permutation** of messages ( $a.c \preceq c.a$ )

## The problem

- *reduction* alone is **too strict**
- *extension* is **unsafe**
- *extension;reduction* is **not transitive**
- *permutation* is **not allowed**

## The solution

- use (simple) **orchestrators**

# Summary

- 1 contracts
- 2 simple orchestrators
- 3 simple orchestrators with buffers
- 4 duality
- 5 recursive behaviors
- 6 orchestrator synthesis
- 7 related and ongoing work

# Oh no! More ws-bpe1!

```
<process>
  <sequence>
    <receive operation="Order" variable="Request"/>

    <flow>
      <invoke operation="Deposit" inputVariable="Request" outputVariable="Deposit"/>
      <invoke operation="Charge" inputVariable="Request" outputVariable="Charge"/>
    </flow>

    <switch>
      <case condition="getVariableData(Deposit) == true && getVariableData(Charge) == true">
        <invoke operation="Ship" inputVariable="Request"/>
        <reply operation="Order" value="OK"/>
      </case>

      <case condition="getVariableData(Charge) == true">
        <invoke operation="Refund" inputVariable="Request"/>
        <reply operation="Order" value="NO"/>
      </case>

      <otherwise>
        <reply operation="Order" value="NO"/>
      </otherwise>
    </switch>
  </sequence>
</process>
```

# What's in a contract

## Actions

O Order  
D Deposit  
C Charge  
S Ship  
R Refund

## Traces

{ODCSO, OCDSO, ODCRO, OCDRO, ODCO, OCDO}

## Branching points

- OD... and OC... is an *external choice*
- ...SO, ...RO, and ...O is an *internal choice*

## **Basic theory of contracts**



# Contracts

## Syntax

$\sigma ::=$	<b>contract</b>	$\alpha ::=$	<b>action</b>
$\mathbf{0}$	(null)	$a$	(input)
$\alpha.\sigma$	(action prefix)	$\bar{a}$	(output)
$\sigma + \sigma$	(external choice)		
$\sigma \oplus \sigma$	(internal choice)		

## Example

$0.(\bar{D}.\bar{C}.D.C.(\bar{S}.\bar{0} \oplus \bar{R}.\bar{0} \oplus \bar{0})) + \bar{C}.\bar{D}.D.C.(\bar{S}.\bar{0} \oplus \bar{R}.\bar{0} \oplus \bar{0}))$

# Operational semantics

$$\alpha.\sigma \xrightarrow{\alpha} \sigma \quad \sigma \oplus \tau \rightarrow \sigma \quad \frac{\sigma \xrightarrow{\alpha} \sigma'}{\sigma + \tau \xrightarrow{\alpha} \sigma'} \quad \frac{\sigma \rightarrow \sigma'}{\sigma + \tau \rightarrow \sigma' + \tau}$$

# Compliance, formally

## Systems

$$\rho \parallel \sigma$$

## System transitions

$$\frac{\rho \rightarrow \rho'}{\rho \parallel \sigma \rightarrow \rho' \parallel \sigma}$$

$$\frac{\sigma \rightarrow \sigma'}{\rho \parallel \sigma \rightarrow \rho \parallel \sigma'}$$

$$\frac{\rho \xrightarrow{\bar{\alpha}} \rho' \quad \sigma \xrightarrow{\alpha} \sigma'}{\rho \parallel \sigma \rightarrow \rho' \parallel \sigma'}$$

## Compliance

$$\rho \dashv \sigma \stackrel{\text{def}}{\iff} \rho \parallel \sigma \implies \rho' \parallel \sigma' \dashv \implies \rho' \xrightarrow{e}$$

# Compliance, examples

$$\bar{a}.e \dashv ? a \quad \text{😊}$$

$$\bar{a}.e \oplus \bar{b}.e \dashv ? a + b \quad \text{😊}$$

$$e \dashv ? \sigma \quad \text{😊}$$

$$\bar{a}.e \oplus \bar{b}.e \dashv ? a \oplus b \quad \text{😞}$$

$$\bar{a}.e \dashv ? a \oplus \mathbf{0} \quad \text{😞}$$

$$\mathbf{0} \dashv ? \sigma \quad \text{😞}$$

# Subcontract, formally

## Set-theoretic interpretation of contracts

$$\llbracket \sigma \rrbracket^s \stackrel{\text{def}}{=} \{ \rho \mid \rho \dashv \sigma \}$$

## Subcontract

$$\sigma \sqsubseteq \tau \stackrel{\text{def}}{\iff} \llbracket \sigma \rrbracket^s \subseteq \llbracket \tau \rrbracket^s$$

$$\approx = \sqsubseteq \cap \sqsupseteq$$

## Subcontract, (counter)examples

$$\bar{a}.e \quad a + b \not\sqsubseteq a \oplus b$$

$$\bar{a}.e \quad a \not\sqsubseteq \mathbf{0}$$

$$e + a \quad \mathbf{0} \not\sqsubseteq \bar{a}$$

## Reduction of nondeterminism

$$\sigma \oplus \tau \sqsubseteq \sigma$$

# Properties of strong subcontract

Internal choice = intersection

$$\llbracket \sigma \oplus \tau \rrbracket^s = \llbracket \sigma \rrbracket^s \cap \llbracket \tau \rrbracket^s$$

External choice  $\neq$  union

- there are clients in  $\llbracket a + b \rrbracket^s$  that are not in  $\llbracket a \rrbracket^s \cup \llbracket b \rrbracket^s$ :

$$\bar{a}.e \oplus \bar{b}.e \in \llbracket a + b \rrbracket^s \quad \bar{a}.e \oplus \bar{b}.e \notin \llbracket a \rrbracket^s \quad \bar{a}.e \oplus \bar{b}.e \notin \llbracket b \rrbracket^s$$

- sometimes  $+$  is  $\oplus$  in disguise:

$$\alpha.\sigma + \alpha.\tau \simeq \alpha.(\sigma \oplus \tau)$$

- interferences:

$$\bar{a}.e + \bar{b} \in \llbracket a \rrbracket^s \quad \bar{a}.e + \bar{b} \notin \llbracket a + b \rrbracket^s$$

# Properties of strong subcontract

## Proposition

$\sqsubseteq$  is a precongurence

$$\sigma \sqsubseteq \tau \quad \Rightarrow \quad \begin{cases} \alpha.\sigma \sqsubseteq \alpha.\tau \\ \sigma \oplus \sigma' \sqsubseteq \tau \oplus \sigma' \\ \sigma + \sigma' \sqsubseteq \tau + \sigma' \end{cases}$$

- + nice axiomatization
- + can be used for safe replacement of **parts** of services



## Strong subcontract: axioms

$$\begin{aligned} \text{(e1)} \quad & \sigma + \sigma = \sigma \\ \text{(e2)} \quad & \sigma + \tau = \tau + \sigma \\ \text{(e3)} \quad & \sigma + (\sigma' + \sigma'') = (\sigma + \sigma') + \sigma'' \\ \text{(e4)} \quad & \sigma + \mathbf{0} = \sigma \end{aligned}$$

$$\begin{aligned} \text{(i1)} \quad & \sigma \oplus \sigma = \sigma \\ \text{(i2)} \quad & \sigma \oplus \tau = \tau \oplus \sigma \\ \text{(i3)} \quad & \sigma \oplus (\sigma' \oplus \sigma'') = (\sigma \oplus \sigma') \oplus \sigma'' \end{aligned}$$

$$\begin{aligned} \text{(d1)} \quad & \sigma + (\sigma' \oplus \sigma'') = (\sigma + \sigma') \oplus (\sigma + \sigma'') \\ \text{(d2)} \quad & \sigma \oplus (\sigma' + \sigma'') = (\sigma \oplus \sigma') + (\sigma \oplus \sigma'') \\ \text{(d3)} \quad & \alpha.\sigma + \alpha.\tau = \alpha.(\sigma \oplus \tau) \\ \text{(d4)} \quad & \alpha.\sigma \oplus \alpha.\tau = \alpha.(\sigma \oplus \tau) \end{aligned}$$

$$\text{(red)} \quad \sigma \oplus \tau \leq \sigma$$

# Orchestrators

## Limitations of $\sqsubseteq$

$\sqsubseteq$  does not support extensions...

$$a \not\sqsubseteq a + b$$

... because extra actions may cause interferences

$$\bar{a}.e + \bar{b} \dashv a \qquad \bar{a}.e + \bar{b} \not\sqsubseteq a + b$$

# Why not all failures are equal

## Failure due to client nondeterminism

$$\bar{a}.e \oplus \bar{b}.e \parallel a \rightarrow \bar{b}.e \parallel a$$

## Failure due to service nondeterminism

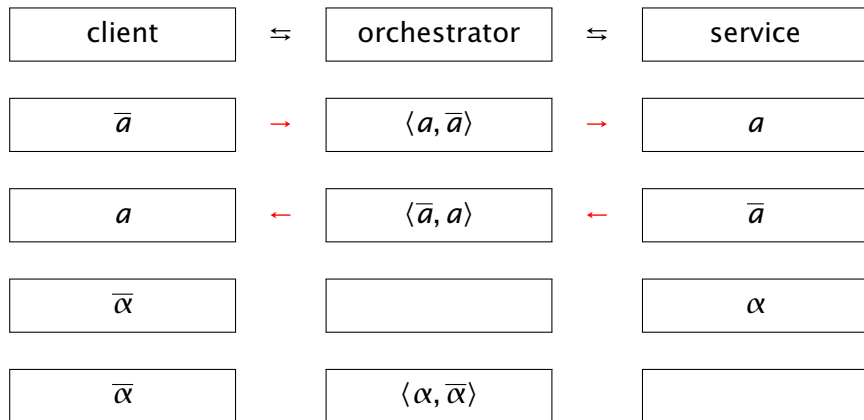
$$\bar{a}.e \parallel a \oplus b \rightarrow \bar{a}.e \parallel a$$

## Failure due to “system” nondeterminism

$$\bar{a}.e + \bar{b}.c.e \parallel a + b.\bar{d} \rightarrow c.e \parallel \bar{d}$$

$$\bar{a}.e + \bar{b}.c.e \parallel a + b.\bar{d} \rightarrow e \parallel \mathbf{0}$$

# Idea: orchestrated interaction



# Synchronous orchestrators

$f ::=$	<b>orchestrator</b>
$\mathbf{0}$	(null)
$\mu.f$	(action prefix)
$f \vee f$	(disjunction)
$\mu ::=$	<b>orchestration action</b>
$\langle a, \bar{a} \rangle$	(input/output)
$\langle \bar{a}, a \rangle$	(output/input)

# Orchestrator semantics

## Orchestrator transitions

$$\mu.f \xrightarrow{\mu} f \quad \frac{f \xrightarrow{\mu} f'}{f \vee g \xrightarrow{\mu} f'}$$

## Trace semantics for orchestrators

$$\llbracket f \rrbracket \stackrel{\text{def}}{=} \{ \mu_1 \cdots \mu_n \mid \exists g : f \xrightarrow{\mu_1} \cdots \xrightarrow{\mu_n} g \}$$

# Orchestrated compliance, formally

## Orchestrated systems

$$\rho \parallel_f \sigma$$

## Orchestrated system transitions

$$\frac{\rho \rightarrow \rho'}{\rho \parallel_f \sigma \rightarrow \rho' \parallel_f \sigma} \quad \frac{\sigma \rightarrow \sigma'}{\rho \parallel_f \sigma \rightarrow \rho \parallel_f \sigma'}$$

$$\frac{\rho \xrightarrow{\bar{\alpha}} \rho' \quad f \xrightarrow{\langle \alpha, \bar{\alpha} \rangle} f' \quad \sigma \xrightarrow{\alpha} \sigma'}{\rho \parallel_f \sigma \rightarrow \rho' \parallel_{f'} \sigma'}$$

## Weak compliance

$$f : \rho \dashv\vdash \sigma \stackrel{\text{def}}{\iff} \rho \parallel_f \sigma \implies \rho' \parallel_{f'} \sigma' \dashv\vdash \text{implies } \rho' \xrightarrow{e}$$



## Weak compliance, examples

$$\langle a, \bar{a} \rangle : \quad \bar{a}.e \dashv\vdash ? a + b \quad \text{😊}$$

$$\langle a, \bar{a} \rangle \vee \langle b, \bar{b} \rangle : \quad \bar{a}.e \dashv\vdash ? a + b \quad \text{😊}$$

$$\langle a, \bar{a} \rangle \vee \langle b, \bar{b} \rangle : \quad \bar{a}.e \oplus \bar{b}.e \dashv\vdash ? a + b \quad \text{😊}$$

$$\mathbf{0} : \quad e + a \dashv\vdash ? \bar{a} \quad \text{😊}$$

$$f : \quad \bar{a}.e \oplus \bar{b}.e \dashv\vdash ? a \oplus b \quad \text{😞}$$

# Weak subcontract, formally

## Set-theoretic interpretation of contracts

$$\llbracket \sigma \rrbracket^w \stackrel{\text{def}}{=} \{ \rho \mid \exists f : f : \rho \dashv\vdash \sigma \}$$

## Weak subcontract

$$\sigma \preceq \tau \stackrel{\text{def}}{\iff} \llbracket \sigma \rrbracket^s \subseteq \llbracket \tau \rrbracket^w$$

## A few doubts...

- is  $\preceq$  the subcontract relation we're looking for?
- is  $\preceq$  a preorder?

# Universal orchestrators

$\sigma \preceq \tau \iff$  for every  $\rho, \rho \dashv \sigma$  implies  $f : \rho \dashv \tau$  for some  $f$

## Universal orchestrator

$f : \sigma \preceq \tau \stackrel{\text{def}}{\iff}$  for every  $\rho, \rho \dashv \sigma$  implies  $f : \rho \dashv \tau$

$f$  is the *universal orchestrator* for  $\sigma \preceq \tau$

### Proposition (existence of universal orchestrator)

$\sigma \preceq \tau$  if and only if  $f : \sigma \preceq \tau$  for some orchestrator  $f$

# Orchestrators as morphisms

Orchestrator application  $f(\sigma)$

$f$	$\sigma$	$f(\sigma)$
$\langle a, \bar{a} \rangle \vee \langle b, \bar{b} \rangle$	$a + b$	$a + b$
$\langle a, \bar{a} \rangle \vee \langle b, \bar{b} \rangle$	$a \oplus b$	$a \oplus b$
$\langle a, \bar{a} \rangle$	$a + b$	$a$
$\langle a, \bar{a} \rangle$	$a \oplus b$	$a \oplus \mathbf{0}$
$\mathbf{0}$	$\sigma$	$\mathbf{0}$

## Theorem

$f : \rho \dashv\vdash \sigma$  if and only if  $\rho \dashv f(\sigma)$

## Corollary

$f : \sigma \leq \tau$  if and only if  $\sigma \sqsubseteq f(\tau)$

## Weak subcontract, examples

Reduction of nondeterminism ( $\leq$  embeds  $\sqsubseteq$ )

$$\begin{aligned} \langle a, \bar{a} \rangle \vee \langle b, \bar{b} \rangle &: a \oplus b \leq a \\ \Downarrow \\ a \oplus b &\sqsubseteq (\langle a, \bar{a} \rangle \vee \langle b, \bar{b} \rangle)(a) = a \end{aligned}$$

Width extension

$$\begin{aligned} \langle a, \bar{a} \rangle &: a \leq a + b \\ \Downarrow \\ a &\sqsubseteq \langle a, \bar{a} \rangle(a + b) = a \end{aligned}$$

Depth extension

$$\begin{aligned} \mathbf{0} &: \mathbf{0} \leq \sigma \\ \Downarrow \\ \mathbf{0} &\sqsubseteq \mathbf{0}(\sigma) = \mathbf{0} \end{aligned}$$

## On transitivity of $\preceq$

Is  $\preceq$  transitive?

$$f : \sigma \preceq \sigma' \quad g : \sigma' \preceq \tau \quad \stackrel{?}{\Rightarrow} \quad h : \sigma \preceq \tau$$

$$\langle a, \bar{a} \rangle \vee \langle b, \bar{b} \rangle . \langle c, \bar{c} \rangle : a \oplus b.c \preceq a$$

$$\langle a, \bar{a} \rangle : a \preceq a + b.d$$

$$??? : a \oplus b.c \preceq a + b.d$$

**Proposition (Orchestrator application is monotone)**

$\sigma \sqsubseteq \tau$  implies  $f(\sigma) \sqsubseteq f(\tau)$

$$\sigma \sqsubseteq f(\sigma') \quad \sigma' \sqsubseteq g(\tau) \quad \Rightarrow \quad \sigma \sqsubseteq f(g(\tau))$$

$\preceq$  is transitive if  $f \circ g$  is an orchestrator

# Orchestrator composition and transitivity of $\preceq$

## Orchestrator composition

$$f \wedge g$$

- $f \wedge g$  permits the traces permitted by  $f$  **and** by  $g$
- $f \wedge g$  forbids the traces forbidden by either  $f$  **or** by  $g$

$$\llbracket f \wedge g \rrbracket \stackrel{\text{def}}{=} \llbracket f \rrbracket \cap \llbracket g \rrbracket$$

### Proposition

$$f(g(\sigma)) \simeq (f \wedge g)(\sigma)$$

# Towards a deduction system for $\preceq$

Problem:  $\preceq$  is not a precongruence w.r.t.  $+$

$$\begin{array}{l} \mathbf{0} \preceq a \\ a.b \preceq a.b \\ a.b + \mathbf{0} \not\preceq a.b + a \simeq a.(b \oplus \mathbf{0}) \end{array}$$

## Proposition (distributivity of orchestration application)

- 1  $f(\sigma) + f(\tau) \simeq f(\sigma + \tau)$
- 2  $f(\sigma) \oplus f(\tau) \simeq f(\sigma \oplus \tau)$

## Corollary

$f : \sigma_1 \preceq \tau_1$  and  $f : \sigma_2 \preceq \tau_2$  implies  $f : \sigma_1 + \sigma_2 \preceq \tau_1 + \tau_2$

- precongruence is granted if the orchestrator is oblivious of the particular branch taken



## Deduction system for $\preceq$

(red)	$I(\sigma) : \sigma \oplus \tau \leq \sigma$	(prefix)	$\frac{f : \sigma \leq \tau}{\langle \alpha, \bar{\alpha} \rangle . f : \alpha . \sigma \leq \alpha . \tau}$
(width)	$\frac{\llbracket I(\sigma) \wedge I(\tau) \rrbracket = \{\varepsilon\}}{I(\sigma) : \sigma \leq \sigma + \tau}$	(int)	$\frac{f : \sigma \leq \sigma' \quad f : \tau \leq \tau'}{f : \sigma \oplus \tau \leq \sigma' \oplus \tau'}$
(trans)	$\frac{f : \sigma \leq \sigma' \quad g : \sigma' \leq \sigma''}{f \wedge g : \sigma \leq \sigma''}$	(ext)	$\frac{f : \sigma \leq \sigma' \quad f : \tau \leq \tau'}{f : \sigma + \tau \leq \sigma' + \tau'}$

The deduction system is *sound* and *complete*

$$f : \sigma \preceq \tau \iff f : \sigma \leq \tau$$

- completeness regards orchestrators too

# Interpretations of orchestrators

## As mediators

$$\rho \parallel_f \sigma$$

## As morphisms/behavioral coercions

$$f : \sigma \preceq \tau \iff \sigma \sqsubseteq f(\tau) \qquad f : \tau \rightarrow \sigma$$

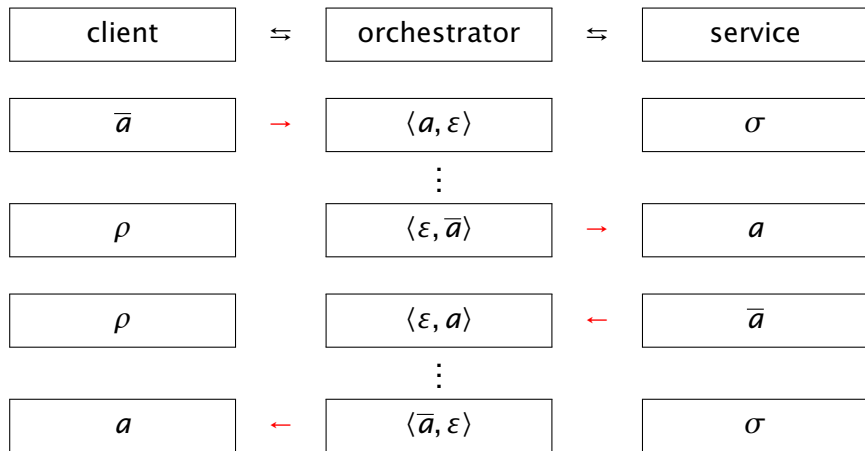
## As assumptions on the environment

$$\langle a, \bar{a} \rangle : a \preceq a + b$$

- it is safe to replace  $a$  with  $a + b$  if nobody ever tries to perform  $\bar{b}$

## Buffered orchestrators

# Buffered orchestrators



# Syntax of buffered orchestrators

$f$	$::=$	<b>orchestrator</b>
	$\mathbf{0}$	(null)
	$  \mu.f$	(action prefix)
	$  f \vee f$	(disjunction)
$\mu$	$::=$	<b>orchestration action</b>
	$\langle a, \bar{a} \rangle$	(input/output)
	$  \langle \bar{a}, a \rangle$	(output/input)
	$  \langle \alpha, \varepsilon \rangle$	(action/-)
	$  \langle \varepsilon, \alpha \rangle$	(-/action)

# Valid $k$ -orchestrators

Not every orchestrator makes sense

orchestrator	valid	rank
$\langle \varepsilon, a \rangle . \langle \bar{a}, \varepsilon \rangle$	😊	$\geq 1$
$\langle a, \varepsilon \rangle . \langle a, \varepsilon \rangle$	😊	$\geq 2$
$\langle \bar{a}, \varepsilon \rangle$	😞	
$\langle \varepsilon, \bar{a} \rangle$	😞	
$\langle a, \varepsilon \rangle . \langle \bar{a}, \varepsilon \rangle$	😞	

Valid  $k$ -orchestrators are...

- directional
- finite-state
- fair

# Weak $k$ -compliance, formally

## Orchestrated systems

$$\rho \parallel_f \sigma$$

## Orchestrated system transitions

...

$$\frac{\rho \xrightarrow{\bar{\alpha}} \rho' \quad f \xrightarrow{\langle \alpha, \varepsilon \rangle} f'}{\rho \parallel_f \sigma \rightarrow \rho' \parallel_{f'} \sigma'} \quad \frac{f \xrightarrow{\langle \varepsilon, \bar{\alpha} \rangle} f' \quad \sigma \xrightarrow{\alpha} \sigma'}{\rho \parallel_f \sigma \rightarrow \rho \parallel_{f'} \sigma'}$$

## Weak $k$ -compliance

$$f : \rho \dashv\!\! \dashv_k \sigma \stackrel{\text{def}}{\iff} \rho \parallel_f \sigma \implies \rho' \parallel_{f'} \sigma' \dashv\!\! \dashv \text{ implies } \rho' \xrightarrow{e}$$

## Weak $k$ -compliance, an example

$$\langle a, \varepsilon \rangle . \langle b, \varepsilon \rangle . \langle \varepsilon, b \rangle . \langle \varepsilon, a \rangle . \langle \bar{c}, c \rangle : \bar{a} . \bar{b} . c . e \dashv\vdash_1 b . a . \bar{c}$$



## Weak $k$ -subcontract, formally

### Set-theoretic interpretation of contracts

$$\llbracket \sigma \rrbracket_k^w \stackrel{\text{def}}{=} \{ \rho \mid \exists f : f : \rho \dashv\!\! \dashv_k \sigma \}$$

### Weak subcontract

$$\sigma \preceq \tau \stackrel{\text{def}}{\iff} \llbracket \sigma \rrbracket^s \subseteq \llbracket \tau \rrbracket_k^w$$

### Proposition (existence of universal orchestrator)

$\sigma \preceq_k \tau$  if and only if  $f : \sigma \preceq_k \tau$  for some  $k$ -orchestrator  $f$

# Orchestrators as morphisms

## Orchestrator application $f(\sigma)$

$f$	$\sigma$	$f(\sigma)$
$\langle a, \varepsilon \rangle . \langle b, \bar{b} \rangle$	$b$	$a.b$
$\langle \varepsilon, \bar{a} \rangle . \langle b, \bar{b} \rangle$	$a.b$	$b$
$\langle a, \varepsilon \rangle . \langle b, \varepsilon \rangle . \langle \varepsilon, \bar{b} \rangle . \langle \varepsilon, \bar{a} \rangle$	$b.a$	$a.b$

## Theorem

$f : \rho \dashv|_k \sigma$  if and only if  $\rho \dashv f(\sigma)$

## Corollary

$f : \sigma \preceq_k \tau$  if and only if  $\sigma \sqsubseteq f(\tau)$

# Orchestrator composition and transitivity of $\preceq_k$

We've got a problem

$$\sigma \stackrel{\text{def}}{=} \bar{b} + \bar{d}$$

$$f \stackrel{\text{def}}{=} \langle a, \varepsilon \rangle . \langle c, \varepsilon \rangle . (\langle \varepsilon, \bar{a} \rangle . \langle \bar{b}, b \rangle \vee \langle \varepsilon, \bar{c} \rangle . \langle \bar{d}, d \rangle)$$

$$g \stackrel{\text{def}}{=} \langle a, \varepsilon \rangle . \langle \bar{b}, b \rangle \vee \langle c, \varepsilon \rangle . \langle \bar{d}, d \rangle$$

$$f(g(\sigma)) \simeq f(a.\bar{b} + c.\bar{d}) \simeq a.c.(\bar{b} \oplus \bar{d})$$

**No single orchestrator can turn  $\bar{b} + \bar{d}$  into  $a.c.(\bar{b} \oplus \bar{d})$**

Idea

Find an orchestrator  $f \cdot g$  such that  $f(g(\sigma)) \sqsubseteq (f \cdot g)(\sigma)$

# Orchestrator composition, formally

$$\begin{aligned} f \cdot g \stackrel{\text{def}}{=} & \bigvee_{f \xrightarrow{\langle \alpha, \varepsilon \rangle} f'} \langle \alpha, \varepsilon \rangle . (f' \cdot g) \vee \bigvee_{g \xrightarrow{\langle \varepsilon, \bar{\alpha} \rangle} g'} \langle \varepsilon, \bar{\alpha} \rangle . (f \cdot g') \\ & \vee \bigvee_{\langle \varphi, \varphi' \rangle . (f' \cdot g')} \\ & \vee \bigvee_{f \xrightarrow{\langle \varphi, \bar{\alpha} \rangle} f', g \xrightarrow{\langle \alpha, \varphi' \rangle} g', \varphi \varphi' \neq \varepsilon} (f' \cdot g') \\ & \vee \bigvee_{f \xrightarrow{\langle \varepsilon, \bar{\alpha} \rangle} f', g \xrightarrow{\langle \alpha, \varepsilon \rangle} g'} (f' \cdot g') \end{aligned}$$

## Theorem

$$f(g(\sigma)) \sqsubseteq (f \cdot g)(\sigma)$$

# Deduction system for $\preceq_k$

**No complete deduction system for  $\preceq_k$  is known**

(swap-inputs)

$$a.b.\sigma = b.a.\sigma$$

(swap-outputs)

$$\bar{a}.\bar{b}.\sigma = \bar{b}.\bar{a}.\sigma$$

(postpone-input)

$$a.\bar{b}.\sigma \leq \bar{b}.a.\sigma$$




# Duality

# Duality

$$\rho^\perp$$

**the  $\preceq$ -smallest service that satisfies  $\rho$**

## Duality, examples

$\rho$	$\rho^\perp$
$a.e$	$\bar{a}$
$a.e \oplus b.e$	$\bar{a} + \bar{b}$
$a.e + b.e$	$\bar{a} \oplus \bar{b}$
$e$	$\mathbf{0}$
$a.\bar{b}.e + a.\bar{c}.e$	$\bar{a}.(b + c)$
$\mathbf{0}$	
$a.e \oplus \mathbf{0}$	
$a.\mathbf{0}$	
$\dots$	

### Definition (viable contract)

$\rho$  is *viable* if there exists  $\sigma$  such that  $\rho \dashv \sigma$



# Duality, formally

$$\rho^\perp \stackrel{\text{def}}{=} \sum_{\rho \Downarrow r, e \notin r} \bigoplus_{\alpha \in r, \text{viable}(\rho(\alpha))} \bar{\alpha} \cdot \rho(\alpha)^\perp$$

## Theorem

Let  $\rho$  be a viable client contract. Then

- 1  $\rho \dashv \rho^\perp$
- 2  $\rho \dashv \sigma$  implies  $\rho^\perp \leq_0 \sigma$

## Recursive behaviors

# Finite syntax for finite behaviors

$\sigma ::=$	<b>contract</b>	$\alpha ::=$	<b>action</b>
	<b>0</b> (null)		$a$ (input)
	$\alpha.\sigma$ (action prefix)		$\bar{a}$ (output)
	$\sigma + \sigma$ (external choice)		
	$\sigma \oplus \sigma$ (internal choice)		

What about recursive behavior?

# Describing recursive behavior

Many different ways...

$$\text{rec } X.a.X \oplus b.\mathbf{0}$$
$$X = a.X \oplus b.\mathbf{0}$$
$$\sigma^*$$

... but it's just syntax!

# Infinite syntax for infinite behaviors

$\sigma ::=$	<b>contract</b>	$\alpha ::=$	<b>action</b>
	$\mathbf{0}$		$a$
	(null)		(input)
	$\alpha.\sigma$		$\bar{a}$
	(action prefix)		(output)
	$\sigma + \sigma$		
	(external choice)		
	$\sigma \oplus \sigma$		
	(internal choice)		

## Definition

The set of contracts is the set of *possibly infinite trees* generated by the grammar above such that

- 1 they have *finitely many different* subtrees
- 2 every *infinite branch* has *infinitely many prefixes*

Every *finite* contracts satisfies these conditions

# Examples

$$\begin{aligned} X &= a.X \\ &= a.a.X \\ &= \dots \end{aligned}$$



$$\begin{aligned} X &= a.X \oplus b.0 \\ &= a.(a.X \oplus b.0) \oplus b.0 \\ &= a.(a.(a.X \oplus b.0) \oplus b.0) \oplus b.0 \\ &= a.(a.(a.(a.X \oplus b.0) \oplus b.0) \oplus b.0) \oplus b.0 \\ &= \dots \end{aligned}$$



$$X = X + X$$



$$X = X \oplus X$$



$$X = X$$



## Recursion: summary

- all the results stated previously still hold (coinduction)
- use your own preferred syntax (but beware of Kleene \*)

Why does it work?

### Proposition

*Let  $D(\sigma) \stackrel{\text{def}}{=} \{\sigma' \mid \sigma \xRightarrow{\varphi} \sigma'\}$ . Then  $D(\sigma)$  is finite for every  $\sigma$*

# Algorithm



# Towards an algorithm for deciding $\preceq_k$

Problem: the orchestrator is not necessarily unique

$$\begin{aligned} \mathbf{0} & : a \oplus b \oplus \mathbf{0} \preceq a + b \\ \langle a, \bar{a} \rangle & : a \oplus b \oplus \mathbf{0} \preceq a + b \\ \langle b, \bar{b} \rangle & : a \oplus b \oplus \mathbf{0} \preceq a + b \\ \langle a, \bar{a} \rangle \vee \langle b, \bar{b} \rangle & : a \oplus b \oplus \mathbf{0} \preceq a + b \end{aligned}$$

$f \preceq g$ :  $g$  is better (more permissive) than  $f$

$$f \preceq g \stackrel{\text{def}}{\iff} \llbracket f \rrbracket \subseteq \llbracket g \rrbracket$$

Idea

- synthesize the best orchestrator
- the best orchestrator is unique

## Deciding $\preceq_k$

$$\begin{array}{c} \mathbf{a}_r = \{\langle \varphi, \overline{\varphi}' \rangle \mid \sigma \xRightarrow{\varphi}, \tau \xRightarrow{\varphi'}, \mathbb{B} \vdash_k \langle \varphi, \overline{\varphi}' \rangle\} \\ \mathbf{a} = \{\langle \varphi, \overline{\varphi}' \rangle \in \mathbf{a}_r \mid \mathbb{B} \langle \varphi, \overline{\varphi}' \rangle \vdash_k f_{\langle \varphi, \overline{\varphi}' \rangle} : \sigma(\varphi) \preceq \tau(\varphi')\} \\ \tau \Downarrow \mathbf{s} \Rightarrow (\exists r : \sigma \Downarrow r \wedge r \subseteq \mathbf{a} \circ \mathbf{s}) \vee (\emptyset \bullet \mathbf{a}) \cap \overline{\mathbf{s}} \neq \emptyset \\ \hline \mathbb{B} \vdash_k \bigvee_{\mu \in \mathbf{a}} \mu \cdot f_\mu : \sigma \preceq \tau \end{array}$$

### Theorem

The following properties hold:

- 1 (termination) the algorithm always terminates
- 2 (correctness)  $f : \sigma \preceq_k \tau$  implies that  $f$  has rank  $k$  and  $f : \sigma \preceq_k \tau$
- 3 (completeness)  $f : \sigma \preceq_k \tau$  implies  $g : \sigma \preceq_k \tau$  for some  $g \geq f$

## An example from Wil's lecture (1/3)

$$\sigma \stackrel{\text{def}}{=} \text{order} . (\text{money} + \overline{\text{food}} . \text{money})$$

$$\rho_1 \stackrel{\text{def}}{=} \overline{\text{order}} . \text{food} . \overline{\text{money}} . e$$

$$\rho_1^\perp = \text{order} . \overline{\text{food}} . \text{money}$$

$$f_1 = \langle \text{order}, \overline{\text{order}} \rangle . \langle \overline{\text{food}}, \text{food} \rangle . \langle \text{money}, \overline{\text{money}} \rangle$$

## An example from Wil's lecture (2/3)

$$\sigma \stackrel{\text{def}}{=} \text{order} . (\text{money} + \overline{\text{food}} . \text{money})$$

$$\rho_2 \stackrel{\text{def}}{=} \overline{\text{order}} . (\text{food} . \overline{\text{money}} . \text{e} + \overline{\text{money}} . \text{food} . \text{e})$$

$$\rho_2^\perp = \text{order} . (\overline{\text{food}} . \text{money} \oplus \text{money} . \overline{\text{food}})$$

$$f_2 = \langle \text{order}, \overline{\text{order}} \rangle . \langle \overline{\text{food}}, \text{food} \rangle . \langle \text{money}, \overline{\text{money}} \rangle$$

## An example from Wil's lecture (3/3)

$$\sigma \stackrel{\text{def}}{=} \text{order} . (\text{money} + \overline{\text{food}} . \text{money})$$

$$\rho_3 \stackrel{\text{def}}{=} \overline{\text{order}} . \overline{\text{money}} . \text{food} . e$$

$$\rho_3^\perp = \text{order} . \text{money} . \overline{\text{food}}$$

$$f_3 = \langle \text{order}, \overline{\text{order}} \rangle . \langle \text{money}, \varepsilon \rangle . \langle \overline{\text{food}}, \text{food} \rangle . \langle \varepsilon, \overline{\text{money}} \rangle$$

## Conclusions

# Wrap-up

## Subcontract relation

- tool for *searching* and *reasoning about* services by their **contracts** (= behavioral types)
- $\preceq$  combines **reduction**, **extension**, and **permutation** into a single preorder
- $\preceq$  gives **safe substitution** of services modulo **orchestration**
- $\preceq$  is decidable

## (Simple) orchestrators

- have nice properties (**universality**, **compositionality**)
- can be automatically synthesized

# Contracts vs session types

## What is being typed

- contract = type of a process
- session type = type of a channel
- session type = type of a process projected on a channel

## Testing approach

- session types: subtyping *preserves* correctness
- contracts: compliance *defines* subcontracting



# Essential bibliography

## Contracts with static interfaces and divergence

- C. Laneve, L. Padovani. **The *must* preorder revisited** (CONCUR'07)

## Synthesis of orchestrators

- G. Castagna, N. Gesbert, and L. Padovani. **A theory of contracts for Web services** (POPL'08)
- G. Castagna, N. Gesbert, and L. Padovani. **A theory of contracts for Web Services.** (ACM TOPLAS 2009) *to appear*
- L. Padovani. **Contract-directed synthesis of simple orchestrators** (CONCUR'08)

## Describing name-passing in contracts

- G. Castagna, L. Padovani. **Contracts for Mobile Processes** (CONCUR'09)