Towards QoS Prediction Based on Composition Structure Analysis and Probabilistic Models

D. Ivanović[1], M. Carro[1,2], P. Kaowichakorn[3]

[1] *IMDEA Software Institute*, Madrid, Spain
[2] *Universidad Politécnica de Madrid*, Spain
[3] *Exxon Mobile*, Thailand

ICSOC 2014, Paris
6 Nov 2014

Introduction

- Quality of Service (QoS) critical for real-world usability

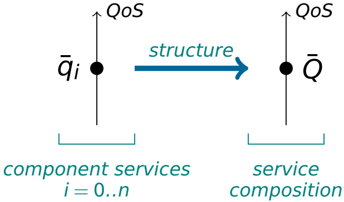
  (*performance*, *cost*, *user experience*...)

  - design-time analysis ⇒ evolving high quality services
  - run-time prediction ⇒ proactive adaptation
  - simulation modeling ⇒ configuration, SLA offering

- QoS analysis for service compositions: *uncertainty*
  - component services: 3rd party components
  - limited information on implementation
  - many actors ⇒ many factors ⇒ uncertain data / QoS

Motivation: Uncertainty

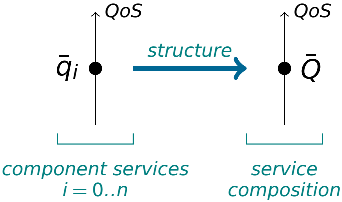
Averages



- how representative?
- spread?
- distribution shape?

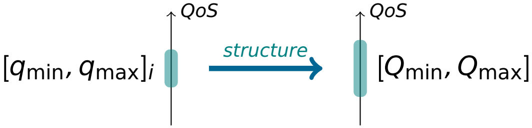
Motivation: Uncertainty

Averages



- how representative?
- spread?
- distribution shape?

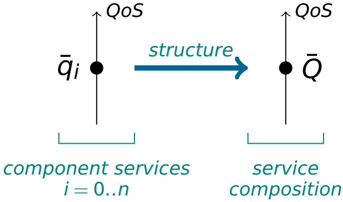
Intervals



- completeness?
- level of confidence?
- granularity?

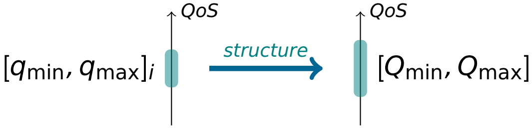
Motivation: Uncertainty

Averages



- how representative?
- spread?
- distribution shape?

Intervals

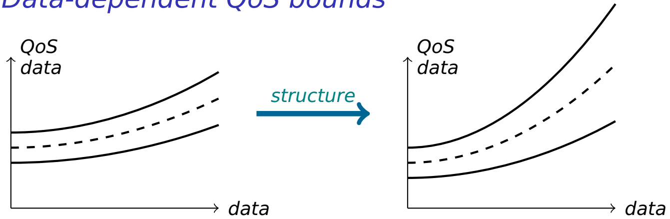


- completeness?
- level of confidence?
- granularity?

Both cases: effects of data?

Motivation: Uncertainty

Data-dependent QoS bounds



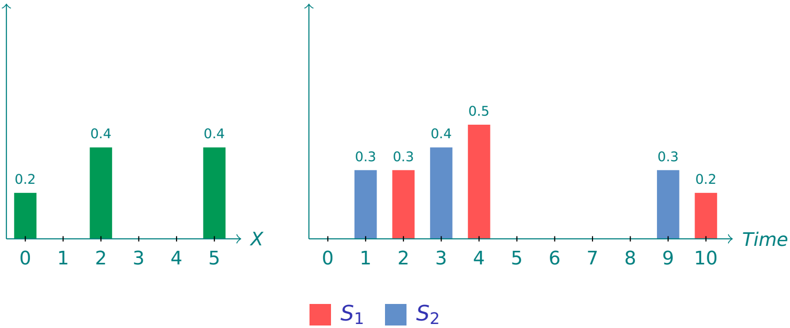
- safe bounds: data + cost
- functions of input data (size)
- complex control

- interval for every data point
- loss of precision?
- data uncertainty?

Unifying analysis

Interpret composition structure in a *probabilistic domain*

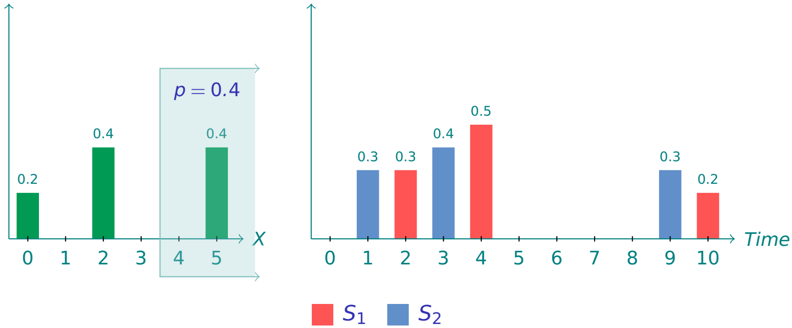
if $X > 3$ **then** **call** S_1 **else** **call** S_2



Unifying analysis

Interpret composition structure in a *probabilistic domain*

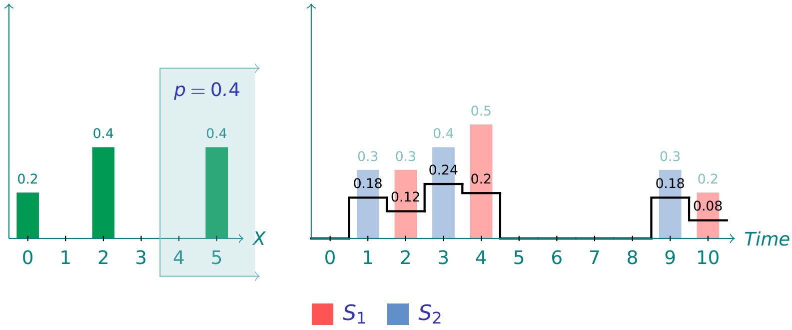
if $X > 3$ **then** **call** S_1 **else** **call** S_2



Unifying analysis

Interpret composition structure in a *probabilistic domain*

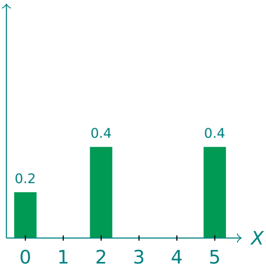
if $X > 3$ **then** **call** S_1 **else** **call** S_2

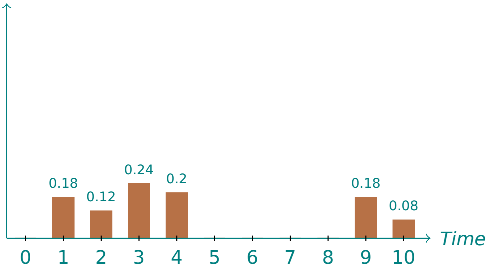


Unifying analysis

Interpret composition structure in a *probabilistic domain*

if $X > 3$ **then** **call** S_1 **else** **call** S_2

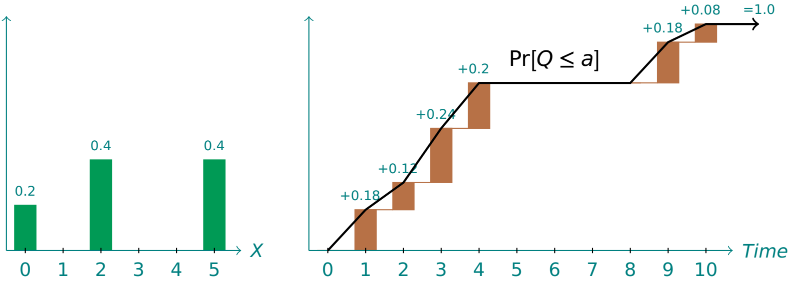




Unifying analysis

Interpret composition structure in a *probabilistic domain*

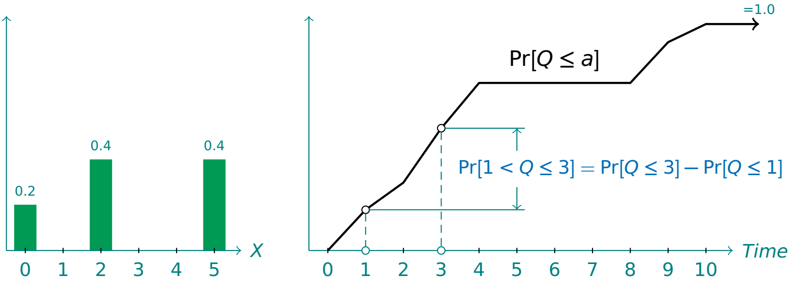
if $X > 3$ **then** **call** S_1 **else** **call** S_2



Unifying analysis

Interpret composition structure in a *probabilistic domain*

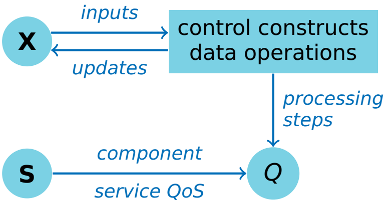
if $X > 3$ **then** **call** S_1 **else** **call** S_2



Proposed Approach

- Unifying analysis: interpreting in a *probabilistic domain*

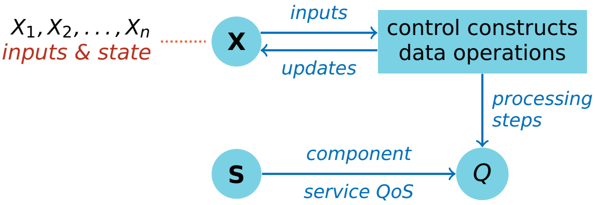
data and QoS $\mapsto$ *discrete random variables*



Proposed Approach

- Unifying analysis: interpreting in a *probabilistic domain*

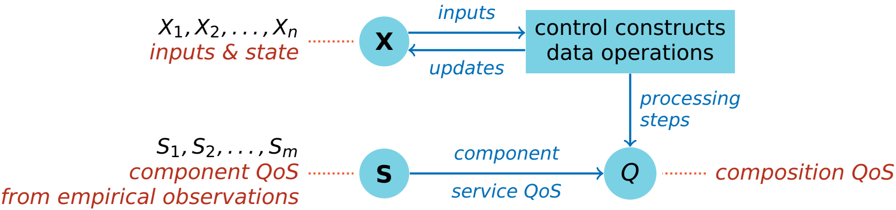
data and QoS $\mapsto$ *discrete random variables*



Proposed Approach

- Unifying analysis: interpreting in a *probabilistic domain*

data and QoS $\mapsto$ *discrete random variables*



Sample Language

- Featuring assignment, arithmetic, boolean conditions, usual control structures, service invocation.
- *Interpretation step*: ρ before $\mapsto$ ρ' after.

Assignments and Arithmetics

$$\boxed{X := E}$$

- X becomes dependent on all variables in E
- Example: $X_1 := X_2 + X_3$

Initial distribution

x_2	ρ_{X_2}
1	0.3
2	0.5
4	0.2

x_3	ρ_{X_3}
0	0.4
1	0.6

Joint distribution

x_1	x_2	x_3	ρ'_{X_1,X_2,X_3}
1	1	0	0.12
2	1	1	0.18
2	2	0	0.20
3	2	1	0.30
4	4	0	0.08
5	4	1	0.12

- Logical condition: $\rho'(x, \mathbf{v}) = \sum\{\rho(u, \mathbf{v}) \mid x = E[u, \mathbf{v}]\}$
  $E[u, \mathbf{v}]$: value of E for $X = u$ and $\mathbf{V} = \mathbf{v}$
- Practical procedure: computing factor tables

Branching

if B **then** C_1 **else** C_2

- Variables in B become *co-dependent*
- Condition $B \to p$ is the probability that that B is true
  - Only some (value, probability) combinations possible when entering C_1 (same for C_2).

Branching

if B **then** C_1 **else** C_2

- Variables in B become *co-dependent*
- Condition $B \rightarrow p$ is the probability that that B is true
  - Only some (value, probability) combinations possible when entering C_1 (same for C_2).
- Example: $B \equiv X_1 > X_2$

x_1	x_2	x_3	$\rho_{X_1,X_2 \mid X_3}$	
1	1	0	0.3	$\neg B$
2	1	1	0.3	B
2	2	0	0.5	$\neg B$
3	2	1	0.5	B
4	4	0	0.2	$\neg B$
5	4	1	0.2	B

B →

x_1	x_2	x_3	$\rho_{X_1,X_2 \mid X_3} \rho_{X_3}$
2	1	1	0.18
3	2	1	0.30
5	4	1	0.12
		Σ :	$0.60 \leftarrow p$

$\neg B$ →

x_1	x_2	x_3	$\rho_{X_1,X_2 \mid X_3} \rho_{X_3}$
1	1	0	0.12
2	2	0	0.20
4	4	0	0.08
		Σ :	$0.40 \leftarrow 1 - p$

- Resulting $\rho' = p \times \rho'_1 + (1 - p) \times \rho'_2$

Loops

while *B* **do** *C*

- Interpretation *unfolds it* into:

```
if B then begin
            C;
            while B do C
          end
      else
          skip
```

- Assuming *termination*

Service Invocations

$$\boxed{\textbf{call } \langle service_i \rangle}$$

- For *monotonic*, *cumulative* QoS metrics ≡

$$Q := Q \oplus S_i$$

- Homogeneous data / QoS treatment.

Implementation Notes

- *Analyzer prototype* implemented in Prolog
  - ease of symbolic manipulation [ASTs, distributions]

Implementation Notes

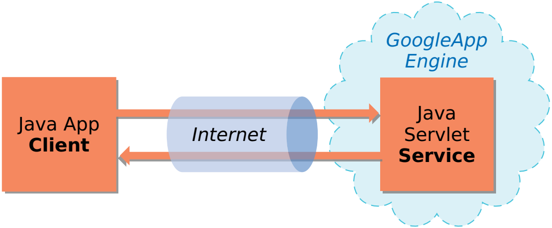
- *Analyzer prototype* implemented in Prolog
  - ease of symbolic manipulation [ASTs, distributions]
- *Inputs*:
  - initial distributions for inputs and state variables [**X**]
  - a description of the composition structure
  - QoS distributions for component services [empirical, **S**]
- *Outputs*:
  - final distribution for composition QoS [Q]
  - final distributions for composition state vars [**X** – *optional*]

Implementation Notes

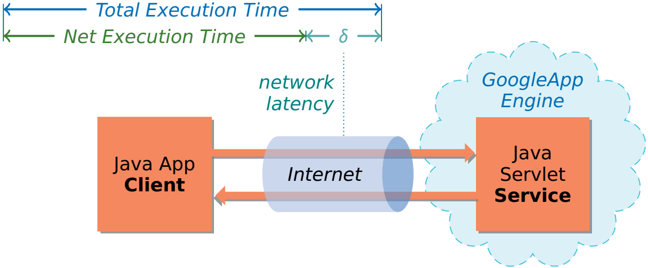
- *Analyzer prototype* implemented in Prolog
  - ease of symbolic manipulation [ASTs, distributions]
- *Inputs*:
  - initial distributions for inputs and state variables [**X**]
  - a description of the composition structure
  - QoS distributions for component services [empirical, **S**]
- *Outputs*:
  - final distribution for composition QoS [Q]
  - final distributions for composition state vars [**X** – *optional*]
- *Example*:

```
?- analyze(if(x>3, call(s1), call(s2)),
           [x=[0-0.2,2-0.4,5-0.4]],
           [s1=[2-0.3,4-0.5,10-0.2],
            s2=[1-0.3,3-0.4,9-0.3]],
          QoS
        ).
QoS = [1-0.18, 2-0.12, 3-0.24, 4-0.2, 9-0.18, 10-0.08]
```

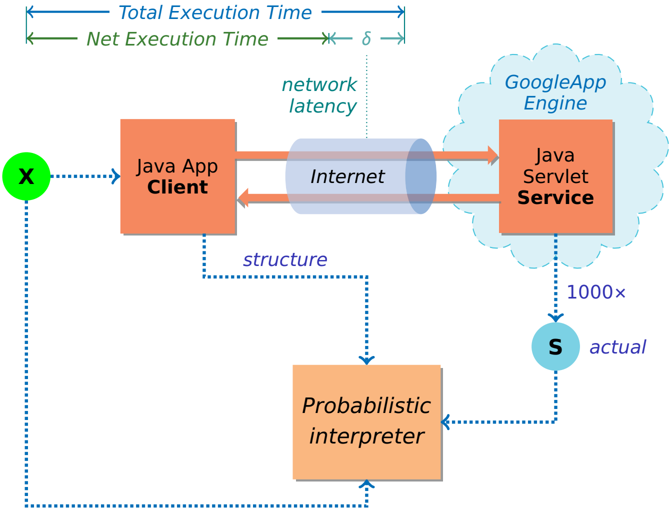
Validation: Execution Time



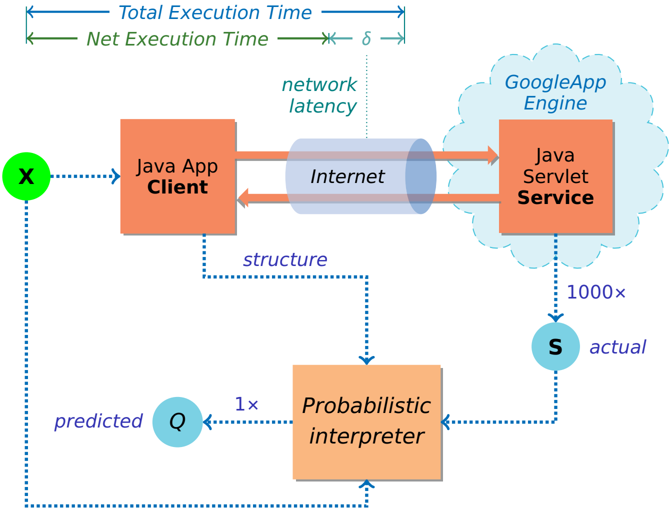
Validation: Execution Time



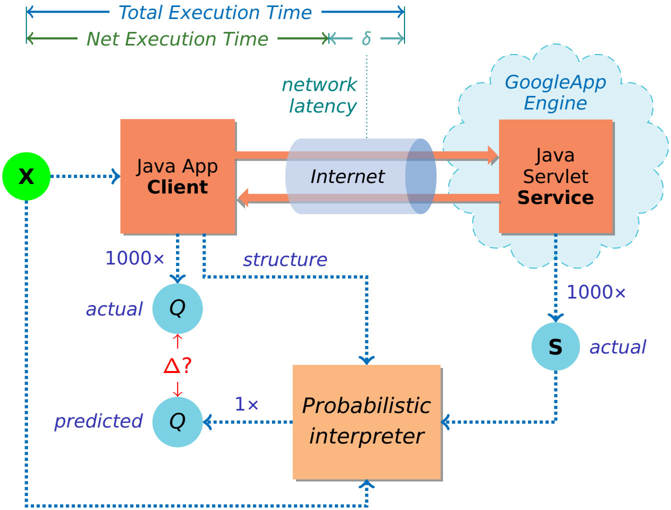
Validation: Execution Time



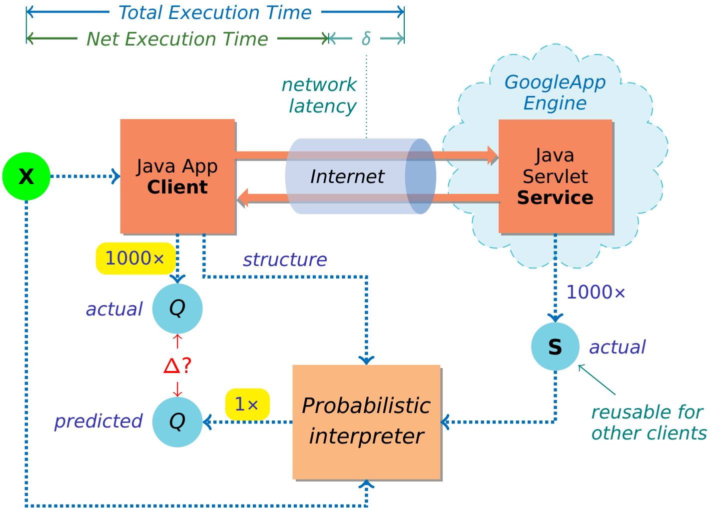
Validation: Execution Time



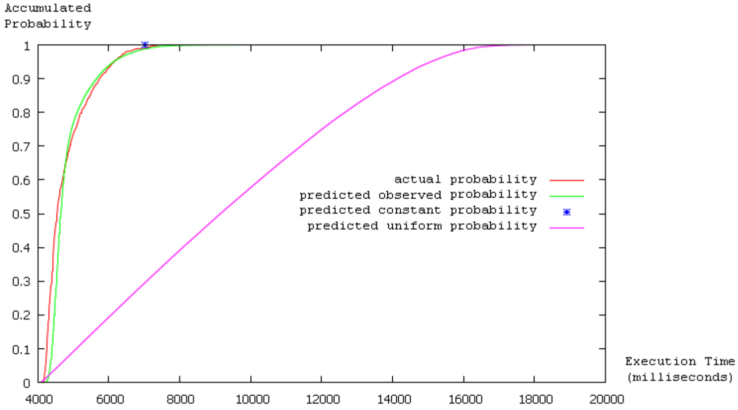
Validation: Execution Time



Validation: Execution Time

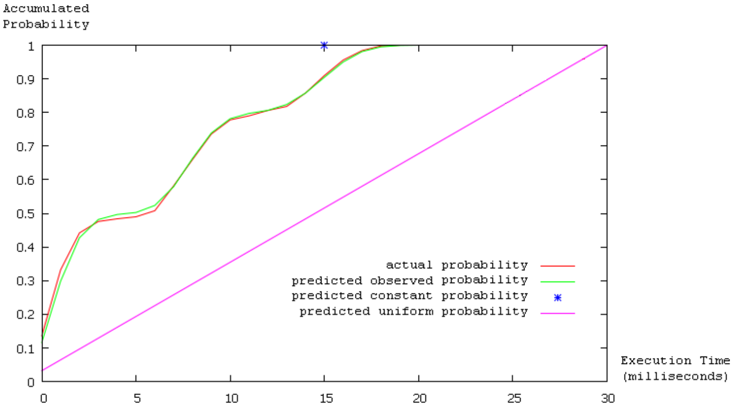


Validation Results: Fitting



- Very small mean square error ≈ 0.07 [total execution time]
  - *order-of-magnitude better* than uniform probability

Validation Results: Multi-Modality



- *Multi-modal distributions*: inflection points
  - difficult to analyze using standard measures of dispersion
  - still, very good fitting

Conclusions

- Modeling uncertainty with probability distributions
  ⇒ *finer grained, more detailed QoS predictions*
- Basic ingredients: structure + empirical data on component QoS → *probabilistic interpretation*
- Future work:
  - reverse: composition QoS → component QoS
  - tradeoffs: complexity ⟷ precision
  - internal representation, algorithms
  - handling dynamic updates for component QoS

Conclusions

- Modeling uncertainty with probability distributions
  $\Rightarrow$ *finer grained, more detailed QoS predictions*
- Basic ingredients: structure + empirical data on component QoS $\rightarrow$ *probabilistic interpretation*
- Future work:
  - reverse: composition QoS $\rightarrow$ component QoS
  - tradeoffs: complexity $\longleftrightarrow$ precision
  - internal representation, algorithms
  - handling dynamic updates for component QoS

Thank you!