Global and Local QoS Constraints Guarantee in Web Service Selection

Danilo Ardagna

Barbara Pernici

Politecnico di Milano
Italy

Motivations

- In SOA, complex applications can be composed as business processes invoking a variety of available Web services
- Develop applications by specifying component Web services only through their required functional characteristics and non-functional constraints
- Select Web services during process execution from a registry of available services, satisfying the specified requirements

WS Selection is an Optimization Problem

- Local optimization: run time selection of the best candidate service which supports the execution of the running high level activity
- Global optimization: identification of the set of candidate services which satisfy the end user preferences for the whole application
- Quality of Service (QoS) constraints at local and global level

Outline

- Formal definitions
- MAIS Framework
- Mixed Integer Programming Model Formulation for the WSC problem
- Experimental results
- Conclusions and Future Work

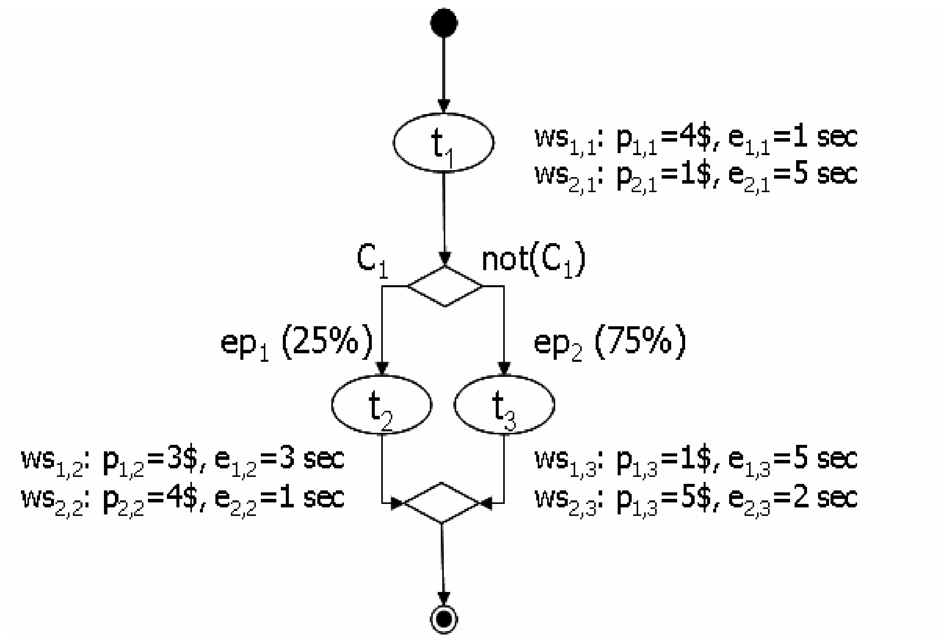
Process Model

- The composite service is specified by using BPEL4WS and is characterized by a single initial and end task
- Annotations specify branches probability of executions, maximum number of iterations of cycles, and local and global constraints
- Each Web service implements a single operation
- Cycles are unfolded according to their maximum number of iterations

Formal Definitions

- Let $ws_{i,j}$ be the i-th Web service candidate for the execution of task t_j
- Execution Path (ep_l): a set of tasks $\{t_1, t_2, \ldots, t_n\}$, such that t_1 is the initial task, t_n is the final task and no t_i, t_j belong to alternative branches
- Sub path (sp_m^l): a sequence of tasks $[t_1, t_2, \ldots, t_n]$, $t_i \in ep_l \ \forall i$, from the begin to the end task which does not contain any parallel sequence
- Execution Plan (epl_k^l): a set of ordered couples (t_j, $ws_{\bar{\imath},j}$), indicating that task t_j included in epl_k^l is executed by a given WS $ws_{\bar{\imath},j}$
- Global Plan: a set of ordered couples (t_j, $ws_{\bar{\imath},j}$), which associates every task t_j to a given WS $ws_{\bar{\imath},j}$ and satisfies local and global constraints for all execution paths

An Example

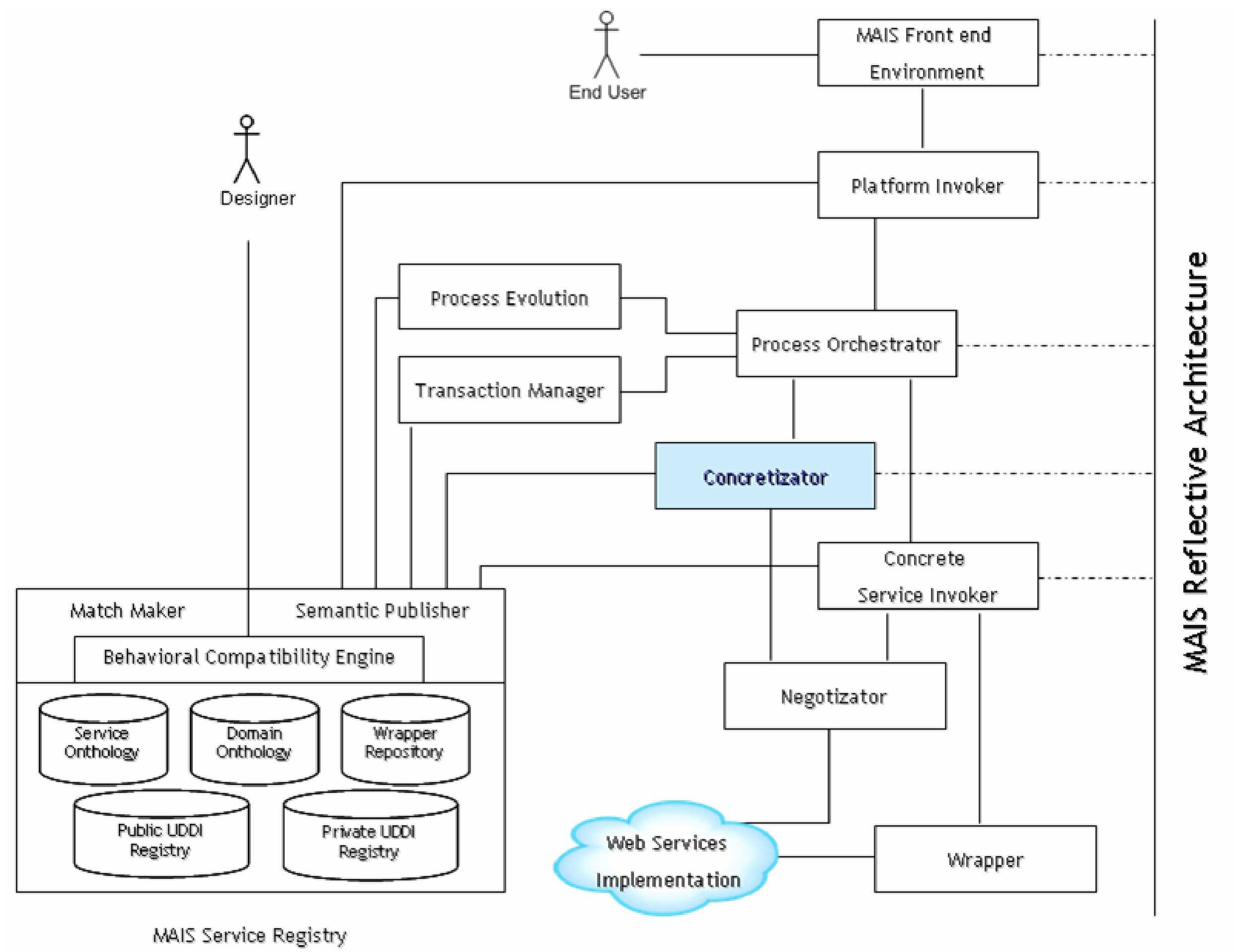


- Global constraints:
  - Price≤6$
  - Execution time≤7 sec.
- Solution: $ws_{2,1}$, $ws_{2,2}$ and $ws_{2,3}$ (no global constraints violation in all cases)

The MAIS Framework

- Candidate Web services are retrieved from the MAIS registry
- Quality values are parameters of the WSC problem
- Web services selection and execution are interleaved (optimization and re-optimization)

MAIS Architecture



Notation

Symbol	Description
$e_{i,j}$	$ws_{i,j}$ execution time
$a_{i,j}$	$\mathrm{ws}_{\mathrm{i,j}}$ availability
$p_{i,j}$	price for $ws_{i,j}$ execution
$r_{i,j}$	$ws_{i,j}$ reputation
WS_j	set of indexes corresponding to candidate Web services of task t_j
$\mathcal{A}$	set of tasks included in the composite service specification
$\mathcal{A}_l$	set of tasks included in the execution path ep_l
L	number of execution path arising from the composite service specification

Problem Formulation

- The WSC problem is multi-objective and a simple additive weighting technique is used to evaluate the overall value of QoS from multiple quality dimensions
- Decision variables:
  - $y_{i,j}$ equals 1 if the Web service $ws_{i,j}$ executes task t_j, 0 otherwise
  - x_j start time of task t_j
  - e_j task t_j duration

Objective Function

- Plan score:

$$score(ep_l) = w_1 \frac{\max Q_1^l - exeTime_l}{\max Q_1^l - \min Q_1^l} + w_2 \frac{avail_l - \min Q_2^l}{\max Q_2^l - \min Q_2^l} +$$
$$+ w_3 \frac{\max Q_3^l - price_l}{\max Q_3^l - \min Q_3^l} + w_4 \frac{rep_l - \min Q_4^l}{\max Q_4^l - \min Q_4^l}$$

- Problem objective:

$$\max \sum_{l=1}^{L} freq_l score(ep_l)$$

Problem Constraints

- Assignment constraints:

$$\sum_{i \in WS_j} y_{i,j} = 1; \quad \forall j \in \mathcal{A}$$

- Task duration constraints:

$$\sum_{i \in WS_j} e_{i,j} y_{i,j} = e_j; \quad \forall j \in \mathcal{A}$$

$$x_k - (e_j + x_j) \geq 0; \quad \forall t_j \rightarrow t_k$$

Problem Constraints

- Local constraints:

$$\sum_{i \in WS_{j_1}} p_{i,j_1} y_{i,j_1} \leq \overline{p}_{j_1}$$

- Execution time constraints:

$$\sum_{j \in sp_m^l} e_j \leq exeTime_l; \quad \forall sp_m^l \in ep_l$$

$$exeTime_l \leq E; \quad \forall l$$

- Availability constraints:

$$avail_l = \prod_{j \in \mathcal{A}_l} \prod_{i \in WS_j} a_{i,j}^{y_{i,j}};$$

$$avail_l \geq A; \quad \forall l$$

- Price constraints:

$$price_l = \sum_{j \in \mathcal{A}_l} \sum_{i \in WS_j} p_{i,j} y_{i,j}$$

$$price_l \leq B; \quad \forall l$$

- Reputation constraints:

$$rep_l = \frac{1}{|\mathcal{A}_l|} \sum_{j \in \mathcal{A}_l} \sum_{i \in WS_j} r_{i,j} y_{i,j}$$

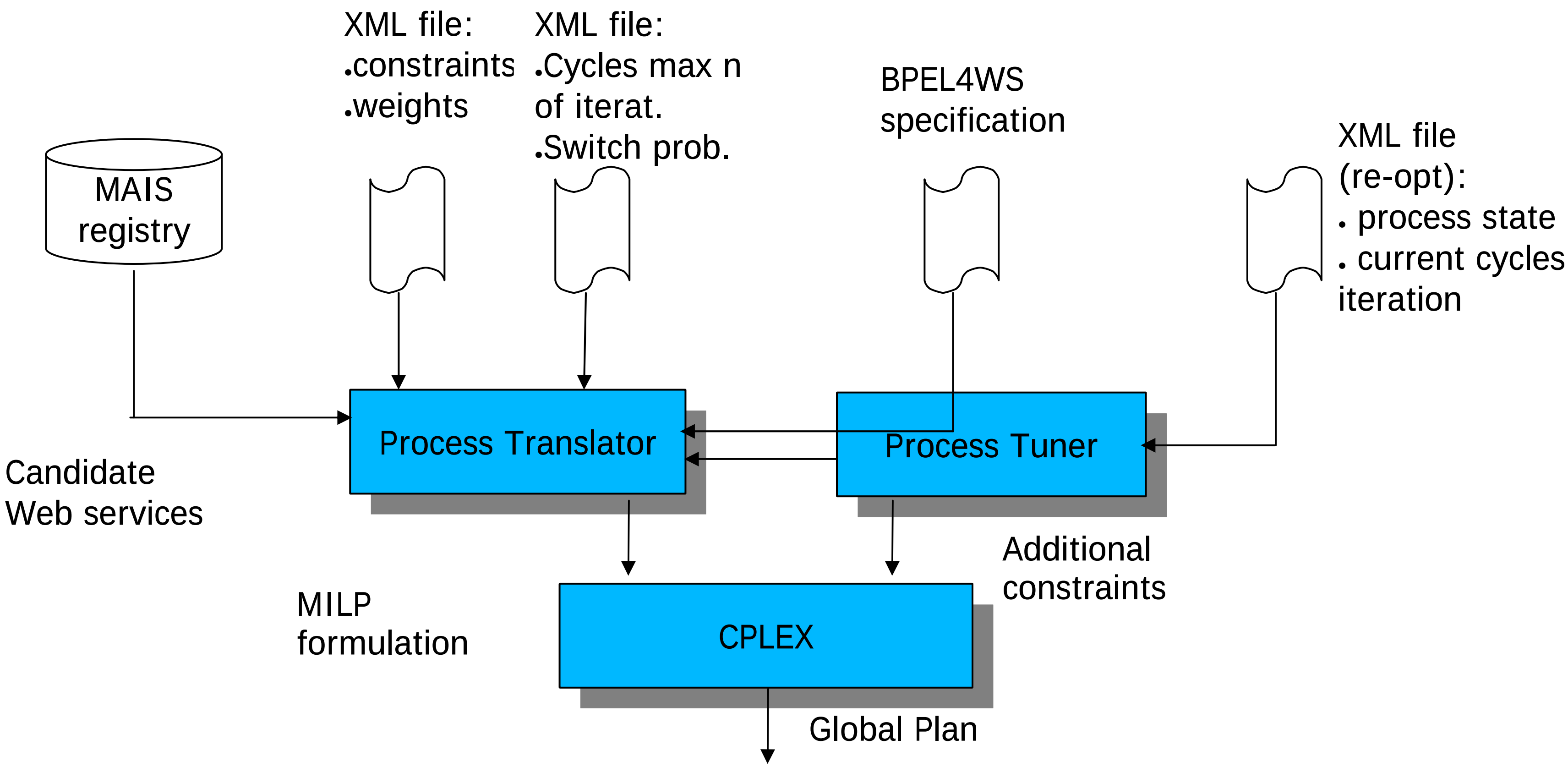
$$rep_l \geq R; \quad \forall l$$

The problem is NP-hard, it is equivalent to a Multiple choice Multiple dimension Knapsack Problem

Re-optimization

- Re-optimization is performed:
  - if the current QoS value q_n differs from the corresponding prediction $\tilde{q}_n$ **(**variability of performance of Web services)
  - if a Web service invocation fails
  - for end user 's context switch
  - after the evaluation of switch conditions
  - periodically with an adaptive time period T_p (new candidate services)
- Re-optimization reduces the number of variables of the problem and can be computed efficiently

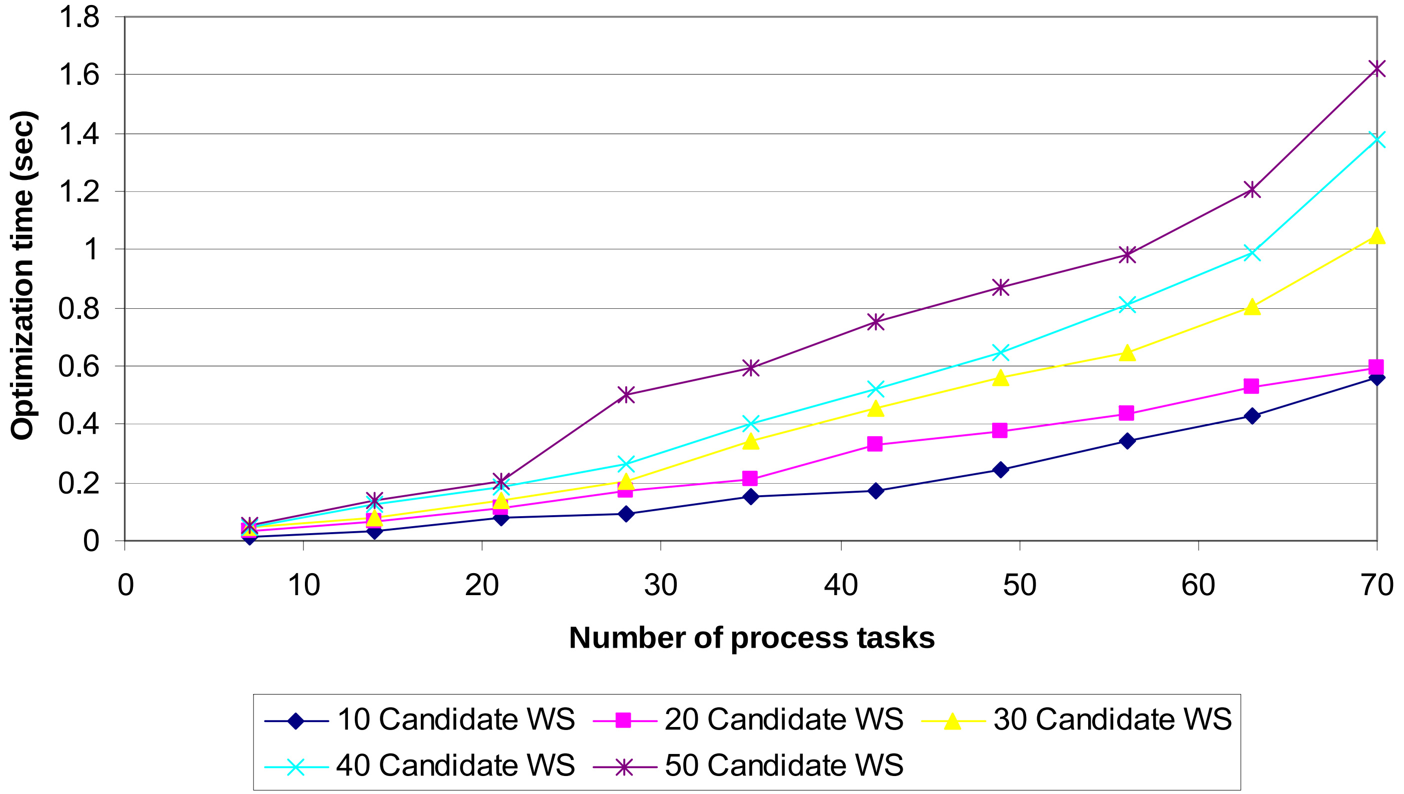
Concretizator Implementation



Optimization Performance

- The effectiveness of our approach has been tested on a wide set of randomly generated instances
- The number of tasks has been varied between 7 and 70. The number of candidate Web services per task has been varied between 10 and 50
- We compared our solutions with the solutions provided by the local optimization
- On average global optimization improves local optimization results by 20-30%

Optimization Performance



Future Work

- Probabilistic execution of cycles included in composite service specifications
- QoS parameters negotiation to further improve a global plan or to identify a feasible solution if it does not initially exist

Thanks! Any questions?