A synopsis of the paper:

Utility-based Optimal Service Selection for Business Processes in Service Oriented Architectures

Vinod K. Dubey and Daniel A. Menascé

2010 IEEE International Conference on Web Services
Talk Agenda

• Introduction

• Definition of the problem

• QoS composition, utility and cost functions

• Service selection algorithms

• Experimental results

• Contribution and conclusion
Introduction

• SOA enables service composition with functionally equivalent, loosely coupled and reusable services

• Services have different QoS and costs

• Need to optimally select services to optimize utility function - "Optimal Service Selection" problem

• Paper researches this problem when the utility f() has many QoS metrics

• Efficient heuristics solution is presented which provides near-optimal solution at low cost - 100 vs. 1M
Definition of the Problem

Maximize \( U(E[R(z)], A(z), X(z)) \)
subject to
\[
\begin{align*}
E[R(z)] & \leq R_{\text{max}} \\
A_{\text{min}} & \leq A(z) \leq 1 \\
X(z) & \geq X_{\text{min}} \\
C(z) & \leq C_{\text{max}}
\end{align*}
\]
\( z \in Z \)

- \( Z \) is the set of all possible service provider selections of the business activities of business process \( B \).
- \( z \in Z \) is a service selection of \( N \) service providers that support the execution of the activities of \( B \).
- \( E[R(z)], A(z), X(z), \) and \( C(z) \) are, respectively, the values of the average end-to-end execution time \( R(z) \), the availability, the throughput, and the cost of executing \( B \) for service provider allocation \( z \).
- \( U(E[R(z)], A(z), X(z)) \) is a dimensionless global utility function in terms of \( E[R(z)], A(z), \) and \( X(z) \). The value of utility is in the range \([0, 1]\).
- \( R_{\text{max}}, A_{\text{min}}, X_{\text{min}}, \) and \( C_{\text{max}} \) are, respectively, the constraints on the average execution time, availability, throughput, and cost.
- \( S_i = \{s_{i1}, \ldots, s_{ik}\} \) is the set of service providers that can be used to implement business activity \( a_i \).
QoS composition, utility and cost functions

- Response time is a random variable - given by a distribution
- Availability and throughput values are deterministic
- Decrease(Response) \rightarrow Increase(Cost)
- Increase(Availability/Throughput) \rightarrow Increase(Cost)

\[ E[\max_{i=1}^{n} R_i] = \int_{0}^{\infty} x \left[ \prod_{i=1}^{n} P_i(x) \right] \sum_{i=1}^{n} \frac{p_i(x)}{P_i(x)} \, dx \]
QoS composition - BPEL Constructs

```
<sequence>
  <invoke a1>
  <switch>
    <case q1>
      <flow>
        <invoke a2>
        <sequence>
          <invoke a3>
          <invoke a4>
        </sequence>
      </flow>
      <case q2=(1-q1)>
        <invoke a5>
      </case>
    </switch>
  <invoke a6>
</sequence>
```
Algorithm 1 Availability Computation of a BPEL process

1: function $A$(node $i$)
2: if label($i$) = leaf node then
3: return $A_i$;
4: else
5: if label($i$) = sequence then
6: return $\prod_{k \in \text{children}(i)} A(k)$;
7: else if label($i$) = switch then
8: return $\sum_{k \in \text{children}(i)} q_k \times A(k)$;
9: else if label($i$) = flow then
10: return $\prod_{k \in \text{children}(i)} A(k)$;
11: end if
12: end if

Algorithm 2 Throughput Computation of a BPEL process

1: function $X$(node $i$)
2: if label($i$) = leaf node then
3: return $X_i$;
4: else
5: if label($i$) = sequence then
6: return $\min_{k \in \text{children}(i)} X(k)$;
7: else if label($i$) = switch then
8: return $\sum_{k \in \text{children}(i)} q_k \times X(k)$;
9: else if label($i$) = flow then
10: return $\min_{k \in \text{children}(i)} X(k)$;
11: end if
12: end if
QoS composition, utility and cost functions

\[ U_i(v(z)) = K_i \frac{e^{\alpha_i(\beta_i - v(z))}}{1 + e^{\alpha_i(\beta_i - v(z))}} \]

\[ U_g(z) = \left( \prod_{i=1}^{3} (U_i(z))^{w_i} \right)^{\frac{1}{\sum_j w_j}} \]

\( v(z) \) is the value of the end-to-end QoS metric — average execution time, availability, and throughput.

\( K_i \) is a normalization factor equal to \( (1 + \varepsilon^{\alpha_i\beta_i}) / \varepsilon^{\alpha_i\beta_i} \) for execution time, \( 1 \) for throughput, and \( 1 + \frac{\varepsilon^{\alpha_i(\beta_i-1)}}{1 + \varepsilon^{\alpha_i(\beta_i-1)}} \)

\( w_j \) are importance weights assigned to each of the three metrics. Weight number is in the \((0,1)\) range and the sum of all weights is one. Other utility function can be used.
• Does not require to generate the entire solution space - only a subset of the solution space where each point represents a feasible solution - one that satisfies the cost and execution time constraints.

• Analyzes partial selection and looks for constraint violation

• If any sub-selection exceeds QoS, all selections having that sub-selection is discarded from further consideration

• Very expensive computation especially for large number of business activities
Service Selection Algorithm - HCB Method

• Hill Climbing Based (HCB) heuristic algorithm to deal with the complexity of Jensen based method
• Neighborhood of points for the current point
• Move to the “best” point in the neighborhood
• Several random restarts are required to avoid the problem of local optima.
Experimental Evaluation

• Compare HBC and JOSeS
  - Effectiveness of HBC
  - Number of points examined
  - Computation time needed
  - Wide ranging parameters - topology complexity, tightness of QoS and number of SPs per activity

• Total cost = $C(r) + C(a) + C(x)$
  - $C(r) = 1/E[R]$ and so on for $C(a)$ and $C(x)$
  - Cost range is similar for all 3 metrics
Experimental Evaluation Setup

- Number of SP/activity - between 2 and 7

- Fifty BPEL processes randomly created – 6 to 9 activities with sequence, flow and switch

- Constraints varying - 10%, 20%, 30%, 40%

- 3 complexity groups of business processes

- Weights for r, a, x are 0.34, 0.33, 0.33 respectively
Experimental Evaluation Result

Figure 1. Average $U_h/U_o$ (%) vs. $nspa$ for four constraint strengths.
Figure 2. Average number of points examined $N_h$ and $N_o$ vs. $n_{spa}$ for four constraint strengths
Experimental Evaluation Result

Figure 3. Average computation time $T_h$ and $T_o$ vs. $nspa$ for four constraint strengths
Figure 4. Average $U_h/U_o$ (%) vs. $n_{spa}$ for simple, medium, and complex business processes
Figure 5. Average number of points examined \( N_h \) and \( N_o \) vs. \( n_{spa} \) for simple, medium, and complex business processes
Experimental Evaluation Result

Figure 6. Average $N_h$ vs. $nspa$
Contributions and Conclusion

• Provides efficient optimal service selection algorithm
• Provides efficient heuristics algorithm
• Stochastic QoS metrics – response time
• Non-linear utility functions allowed
• No fixed utility function for SP – any combination of QoS metrics allowed
• Detail algorithms provided for various activities