An Analytical Approach for Predicting QoS of Web Services Choreographies

Alfredo Goldman, Yanik Ngoko, Dejan Milojicic

An Analytical Approach for Predicting QoS of Web Services Choreographies*

Alfredo Goldman, Yanik Ngoko
Institute of Mathematic and Statistic
University of São Paulo
São Paulo, Brasil
{gold, yanik}@ime.usp.br

Dejan Milojicic
Hewlett Packard Laboratories
Palo Alto, CA USA
dejan.milojicic@hp.com

ABSTRACT

Given a Web Services Composition, we deal with the prediction of the mean service response time that can be expected from a user request that is serviced. This challenge is a key issue in the design of middleware, managing Web Services Composition. We focus on complex services composition that can be described as BPMN choreographies of services. Our main contribution is a mathematical programming based approach for the prediction of the response time of Web Services Compositions. This new approach occurs through the automatic generation of a linear program whose number of variables and constraints is polynomial in the number of elements used to represent the Service Composition. The equations of the linear program are based on well known aggregation rules for service composition and a new modeling that we introduced for handling communication within Web Services.

Categories and Subject Descriptors

H.4 [**Information Systems applications**]: *Workflow management*; B.8 [**Performance and reliability**]: *Performance Analysis and Design Aids*

General Terms

Theory, Performance, Algorithms

Keywords

QoS prediction, Web Services Composition, BPMN

*This research was funded by HP Brasil under the Baile Project and from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement number 257178 (project CHOReOS-Large Scale Choreographies for the Future Internet). Yanik Ngoko is supported by the FAPESP foundation of the State of São Paulo.

MGC2012, December 3, 2012, Montreal, Quebec, Canada.

1. INTRODUCTION

In this paper, we are interested in estimating the Quality of Services (QoS) of Web Services Composition (WSC). As stated in many other previous work [Cardoso et al. 2004, Zeng et al. 2004] such a task is primordial in the design of efficient middleware for service composition management. Our actual work is part of a middleware project [1] where QoS estimation must serve for optimal service selection and online planning during the execution [Issarny et al. 2011]. The service composition that we are interested in can be described as BPMN choreography of services, viewed here as multiple BPMN orchestration processes that communicate [Weske 2007]. On these compositions, we focus on the design of a fast and automatic approach for the estimation of the Service Response Time (SRT) that can be expected from a user request.

The SRT estimation of WSC has been investigated in previous work [Cardoso et al. 2004, Zheng et al. 2010, Zeng et al. 2004, Xiangpeng et al. 2007, Goldman and Ngoko 2012]. A common aspect in these studies is that the QoS of a composition is computed based on a set of aggregation rules that state how to infer the QoS of the composition from the one of its service constituents, considered known. We will assume in this work that this assumption holds.

Despite the abundance of work on QoS prediction for services composition, we believe that few match our needs. This is made clear when considering WSC with a business process viewpoint. In this vision, WSC belong to two classes: composition that corresponds to a single enterprise business process and those corresponding to multiple communicating enterprise processes [Weske 2007]. While most work on QoS prediction deal with the first case, it is the multi-processes case of interest in this paper.

There are at least two key differences between the single and multi-process cases. The first one is that the multi-processes introduce at least two communication levels: service communication within a process (for e.g communication in a LAN) and communication between processes (for e.g inter-LAN communication). Most of work on QoS prediction ignore the communication cost. While this assumption can hold when having a single enterprise process (with a fast enterprise network), it is not a reasonable assumption when considering multiple processes and therefore networks. A second difference between these two types of processes appears when considering their BPMN graph representation (see Figure 1). While in the first case, we might have structured graphs [Polyvyanyy et al. 2011] where a Web

[1] www.choreus.eu

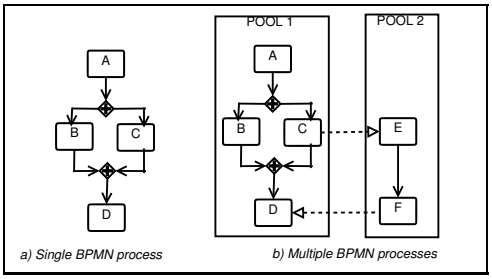

Figure 1: Single and multiple BPMN processes for composition of services. A, B, C, D, E and F correspond to Web Services operations. The notations are based on the BPMN collaboration diagram.

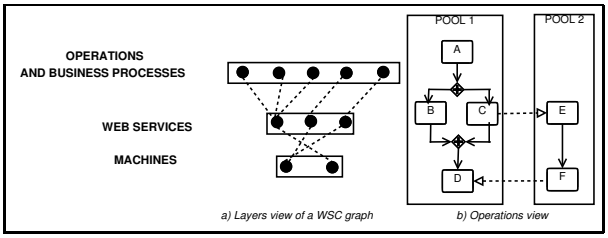

Figure 2: Views of a Web Service Composition

Service operation has only one predecessor, in the second, the graph is by nature unstructured because of precedences induced by inter-processes communication. This observation is of importance because as claimed in [Zheng et al. 2010], most work on QoS predictions are based on structured graphs [Cardoso et al. 2004, Goldman and Ngoko 2012].

In this paper, we consider the prediction of the SRT for WSC corresponding to multiple communicating business processes. As stated previously, it is common to base the QoS computation on a set of aggregation rules. We proceed in the same way here. As aggregation rules, we will use the ones proposed in single process prediction [Cardoso et al. 2004, Goldman and Ngoko 2012], together with new ones that we derive from an interpretation of inter-process communication. Our main contribution is a polynomial time algorithm (in the number of services) that generates from aggregation rules a linear program whose solution is the SRT for the service composition to evaluate. This proposal suggests two steps for SRT prediction: the linear model generation followed by its resolution, using a linear solver.

Our contribution differs from previous work on QoS prediction for services composition on two points. The first is that we consider a modeling of multi-process composition based on BPMN notation for the description of collaboration process [Weske 2007]. The second is that we take into account communication and propose new semantics inspired by work done in distributed and parallel computation [Banikazemi et al. 1999]. This is a major difference when considering other work on QoS prediction with linear programming [Zeng et al. 2004].

The remainder of this paper is organized as follows. In Section 2, we propose a model for WSC. In Section 3 we define our semantics for SRT prediction and in the Section 4, we explain how to estimate the SRT. We conclude in Section 5.

2. MODEL

In this work, we represent a WSC as a Hierarchical Service Graph (HSG) [Goldman and Ngoko 2012]. Such graphs consider three abstraction layers for any services composition: business processes, Web Services and machines layers.

The business processes layer that we will also refer as the **operations graph** describes concurrent business processes. We specify this graph using the BPMN notations for collaboration process. As a result, an operation graph as illustrated by Figure 2b) is made of operations (e.g $A - D$ in Figure 2b), interconnected within a business process or pool by BPMN connectors (e.g AND split connector in Figure 2b) and communicating between pools with messages exchanges (e.g between C and E).

In a general case, operations can belong to multiple Web Services existing in multiple instances on various machines [Goldman and Ngoko 2012]. For the purpose of simplification, we will assume that any operation belongs to a unique Web Service and business process. Moreover, any Web Services exist in one instance on a single machine.

Our approach for QoS prediction is deeply related to the structure of the operation layer. In a formal manner, we define an operations graph by a tuple $G_o = (P, O, C, E_{ocp}, E_{oc}, E_{oo})$ where: $P = \{P_1, \ldots, P_h\}$ is the set of pools, O the set of WSs operations and C the set of BPMN connectors (we considered the AND, XOR and OR connectors). The connectivity of the operations graph is given by the sets E_{oc}, E_{ocp} and E_{oo}. Since, G_o captures a business process, it is an directed graph. Here, E_{oc} describes precedence constraints between operations and connectors, E_{ocp}, states for each operation and connector its pool and E_{oo} describes messages connections between operations of separated pools. We will make various restriction on graph connectivity. For introducing them, let us first consider the following definitions.

2.1 Some definitions

Definition 1 **Pool operations graph:** Given an operations graph $G_o = (P, O, C, E_{ocp}, E_{oc}, E_{oo})$, we define the pool operations graph for the business process P_i as $G_o[P_i] = (O \cup C, E_{oc}[P_i])$ where $E_{oc}[P_i]$ is the subset of E_{oc} containing all elements of the pool P_i.

Definition 2 **Scope:** Let us consider a pool operations graph $G_o[P_i] = (O \cup C, E_{oc}[P_i])$ and an element $x \in O \cup C$. **a)** *x is in the scope of a split connector $y \in C$* if x and y are in the same pool and there exist a **path** $\mu = (y, y_1)$, $(y_1, y_2) \ldots (y_l, x)$ from y to x in which the number of split connectors having the type of y (x excluded) is greater than the number of join connectors of the same type. **b)** In the case where the path $\mu = (y, y_1)$, $(y_1, y_2) \ldots (y_l, x)$ is the shortest possible one from any split connector whose x is in the scope, we say that *x is closely in the scope of y.*

In Figure 3b), the operations A, B, C, D are in the scope of the AND split connector and, B and C are closely in its scope.

Definition 3 **Sending and receiving operations :** Given two operations o_u and o_v, if $(o_u, o_q) \in E_{oo}$, then we will say that the operation o_u sends in its execution a message to o_q. We will also qualify o_u in this case as a **sending** operation and o_q as a **receiving** one. With these elements, we for-

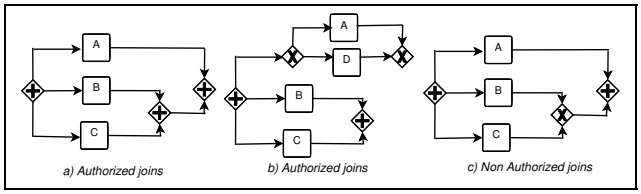

Figure 3: Example of authorized an non-authorized joins

mulate in the next section the structural restriction that we make on the connectivity.

2.2 Structural restrictions

H_1: *Pool operations graphs are obtained by composing the restricted set of generic structures of Figure 4* . Here,

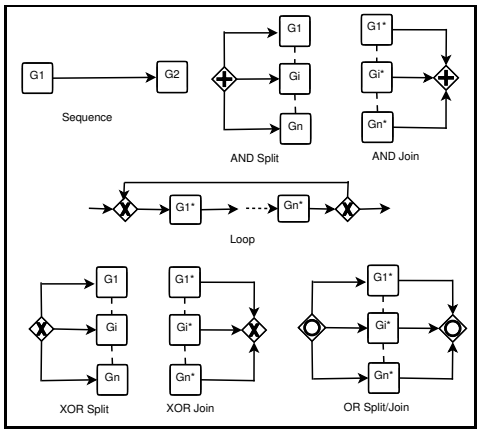

Figure 4: Set of generic subgraph patterns

each G_i refers either to a subgraph built from a composition of these generic structures, or to a single operation. $G_1*, \ldots G_n*$ refer to subgraphs having a unique node without predecessors and a unique one without successors. Finally, let us remark that in the proposed patterns, for each subgraph having a split OR connector, there must be a corresponding join OR connector.

H_2: *Within each pool, there is a unique node without predecessors. This node will be called the* ***initial pool node*** *while nodes without successors will be called* ***terminal nodes****.*

H_3: *All elements that are in a branch leading to a same XOR (resp. AND) join connector must all be in the scope of a same XOR (resp. AND) split connector.* Thus, we can say that there is no violation of restriction ***H_3*** on the graphs of Figures 3a) and 3b), whereas in 3c), there is a violation since B and C lead to an XOR join connector although they are executed within an AND split.

H_4: *Any receiving operation must be linked to exactly one sending operation.* This leads us to consider two categories of communication: one-to-one communication, in the case where for a sending operation there is a unique receiver and, one-to-many communication, where a sender has multiple receivers.

H_5: *If a couple (o_u, o_v) belongs to E_{oo} then o_u and o_v belong to different pools*

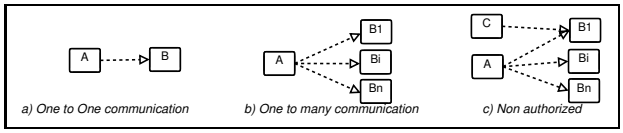

Figure 5: Type of communication

H_6: *A receiving or sending operation must not be in the scope of an XOR, or an OR connector.*

H_7: *In the set of pools $P = \{P_1, \ldots, P_h\}$, there is a subset $P' \subseteq P$ of $h-1$ pools such that the initial node of any graph $G_o[P_i], P_i \in P'$ is a receiving operation. For the unique pool $P^* = P \setminus P'$, the initial node cannot be a receiving operation.*

With the rule H_7, we assume that the execution of a request will always start in the pool P^* and will reach the other pools by message exchanges. The motivation behind the rule H_6 is to avoid deadlock situations where a receiving operation can infinitely wait for a non executed sending operation.

We described the HSG model that we will use for WSC representation. In the next section, we will propose a semantic model for SRT computation on such graphs.

3. EXECUTION SEMANTICS

Our semantical proposal consists of two set of aggregation rules based on the connectivity of the operations graph. The first set states how to aggregate SRT on a pool operations graph. The second takes into account communication between pools.

3.1 Aggregation rules for a pool

For computing the SRT that can be expected from a pool operations graph, we adopt the aggregation rules defined in [Goldman and Ngoko 2012, Cardoso et al. 2004]. Here, each operation $x \in O$ is associated with a mean SRT that we will denote $T(x)$. The SRT led by connectors is set as null.

Sequence	AND Split	AND Join
$T(G_1)+T(G_2)$	$\max\{T(G_1),\ldots,T(G_1)\}$	$\max\{T(G_1^*),\ldots,T(G_1^*)\}$
XOR Split	**XOR Join**	**LOOP**
$\sum_{i=1}^{n} p_i.T(G_i)$	$\sum_{i=1}^{n} p_i.T(G_i^*)$	$\sum_{i=1}^{n_l} p_{li}.\sum T(G_i^*)$
OR Split/Join	$p_{or1}.T(G_1)+p_{or2}.T(G_2)+p_{or\|\|}.\max\{T(G_1),T(G_2)\}$	

Table 1: SRT aggregation on subgraphs patterns

On the generic subgraphs of Figure 4, we present in Table 1 the rules that we will use. Here, $T(G_i)$ denotes the SRT that can be expected from a subgraph G_i. On a generic XOR split graph, a request can be routed towards one subgraph $G_1, \ldots, G_n$. We suppose that there is a probability p_i for each of these possibilities. For a loop subgraph, we assume that there is a maximal number of loops n_l and a probability p_{li} to loop i times. Finally, for an OR split/join , we assume for simplification that the request can only be routed to the subgraph G_1, G_2 or simultaneously, to both. Each routing occurrence, has a known probability p_{or1}, p_{or2} and $p_{or||}$.

3.2 Communication semantics

Given a sending or receiving operation A, we consider that its execution comprises two parts: an **operational part**

of cost $T(A)$ and, a **communication part** where message exchanges occur. On sending operations, we assume that the operational part is always executed before, while we have the inverse on receiving operations.

For modeling the costs of the communication part, we adopt the one port model without recovering [Banikazemi et al. 1999]. Assuming that an operation A sends a message to B, this model proposes a decomposition of the communication in: a **sending** phase, where the machine on which A is engaged in the sending of information on communication links, a **routing** phase ($T(A,B)$), where information go from the machine of A to the one of B and, a **receiving** phase where the machine on which B is run, gathers the sent information. The time duration of these phases is respectively denoted by $T_{Send}(A,B)$, $T(A,B)$ and $T_{Recv}(A,B)$.

In the time evaluation, we will interpret one-to-many communication as series of one-to-one communication. On Figure 5b for example, the total time required for A to send information on the link will be $\sum_{i=1}^{n} T_{Send}(A,B_i)$. More, the reception cannot be completed for any operation B_i before $\min_{1\leq u\leq n}\{T_{Send}(A,B_u)+T(A,B_u)+T_{Recv}(A,B_u)\}$ time units.

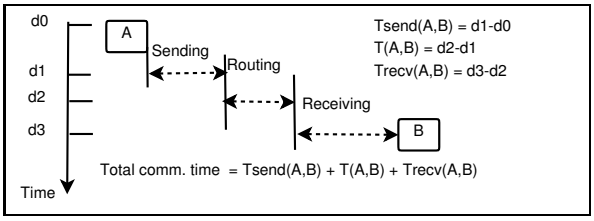

Figure 6: Exchange of information between A and B, starting at the date d_0 and ending at d_3

At this point, we presented a modeling of our service composition and requests execution semantics. In the next, we will present our approach for QoS prediction.

4. SRT PREDICTION

We propose automatically generating linear program (LP) from SRT prediction for any service composition. Its solution gives the SRT of the considered service composition.

4.1 Algorithm for LP generation

For generating LP, with each element $x \in O \cup C$, we associate a real variable t_x. The LP objective function targets the minimization of the objective value denoted by $Z(LP)$. The idea in the LP generation is to define a set of constraints on each variable t_x and $Z(LP)$ (according to the aggregation rules) such that if LP is executed, then $Z(LP)$ will contain the mean SRT.

The constraints definitions for any node x are based on two concepts: the reachability subgraph $RG(x)$ and the reachability probability $p_a(x)$. We define $p_a(x)$ as the probability for a request started in the initial pool node where x is located to reach it. In Figure 7 for instance, $p_a(C)$ is the probability for a request started on g_1 to reach C. By definition, $p_a(C)=1$. We define $RG(x)$ as the subgraph of G_o that comprises all paths leading to x, starting from the initial pool node. In Figure 7 for instance, $RG(D)$ is given by the subgraph, in the dashed box, containing g_1, A, g_2, D.

It is important to notice that a subgraph $RG(x)$ might not fullfill the structural restrictions defined for operation graphs. $RG(D)$ for instance has a split connector with only one successor. In such situation, we consider in $RG(x)$ that this connector is a virtual operation, with a null SRT value.

The LP generation is based on the two reachability concepts introduced below. More precisely, let us suppose that from aggregation rules, we have the mean SRT $T(RG(x))$ for a request to traverse $RG(x)$. The LP generation is done in two main stages. The first stage objective is to generate constraints for setting in variables t_x, the value $p_a(x).T(RG(x))$. Based on these *partial cumulated* SRT, the second stage defines constraints on $Z(LP)$ such as to cumulate the values t_x on terminal nodes in order to return the mean SRT.

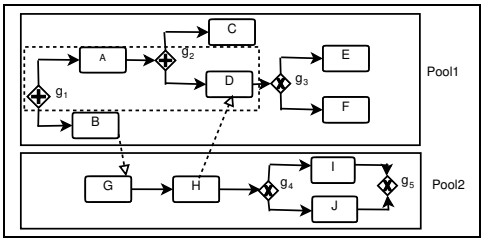

Figure 7: Example of operations graph. In dashed box, we have $RG(D)$.

For setting the values $p_a(x).T(RG(x))$ in t_x, we consider again two-sub stages depending on the fact that an element of $RG(x)$ is or is not involved in communication between pools. Therefore, the general steps of the LP generation process are given in Figure 8. These steps will be discussed in the next sections.

1. **Computation of reachability probabilities;**
2. **Generation of intra-pools constraints;**
3. **Generation of inter-pools constraints;**
4. **Generation of the constraints defining the global SRT.**

Figure 8: Stages for LP generation

4.2 Computation of reachability probabilities

We will compute reachability probabilities on each arc $(u,v) \in E_{oc}$. As input, we assume that on each arc (u,v), we have the conditional (or relative) probability $p_r[(u,v)]$ for a request to be routed on v, knowing that it reached the connector whose v is closely in the scope. At first glance, this input assumption may seem different from classical ones used in QoS prediction. However, this is not the case. Indeed, let us consider an arc (u,v), such that v is strictly in the scope of an XOR split connector c. If the i^{th} branch of c is the one leading to v and has a probability p_i to be executed, then by definition, $p_r[(u,v)] = p_i$. In the same way, if v is closely in the scope of an AND split connector, we have $p_r[(u,v)] = 1$. Finally, in the case where for an arc (u,v), v is not in the scope of a connector, we assume that $p_r[(u,v)] = 1$. More globally, we can easily relate the conditional probabilities $p_r[.]$ to the classical ones p_i and p_{ori} (see Section 3.1) used in aggregation rules.

The reachabilitiy probabilities are derived from the conditional ones. We will distinguish between three cases. In the first case, u does not have any predecessor, then we set

$p_a[(u,v)] = p_r[(u,v)]$. In the second, u is not a join connector and has a predecessor u'. We define:

$$p_a[(u,v)] = \begin{cases} p_a[(u',u)] \text{ if } u \in O; \\ p_r[(u,v)].p_a[(u',u)] \text{ otherwise.} \end{cases}$$

In Figure 7, this gives: $p_a[(g_2,E)]=p_r[(g_1,A)].p_r[(g_2,D)].p_r[(g_2,E)]$ and $p_a[(g_4,I)]=p_r[(G,H)].p_r[(g_4,I)]$. Finally if u is a join connector, with the predecessors $u'_1, \dots u'_m$ then, we have:

$$p_a[(u,v)] = \begin{cases} p_a[(u'_1,u)] \text{ if } u \text{ is an AND Join connector ;} \\ \sum_{i=1}^{m} p_a[(u'_i,u)] \text{ otherwise.} \end{cases}$$

In the definition of reachability for the arcs (u,v), we ignored cases where a request can be routed to multiple branches (with OR connectors). For capturing this, we assume that for each OR join connector v, $p_{or||}(v)$ defines the probability for a request to be routed simultaneously on both branches leading to v. Again, one can notice that this input also exists in classical aggregation rules.

4.3 Generation of intra-pool constraints

In intra-pool constraints, we ignore the communication between operations. These constraints aim at setting in each variable t_x the value $p_a(x).T(RG(x))$. Here, constraints are generated based on arcs of the operation graph. For each $(u,v) \in E_{oc}$ we will generate an equation Eq as follows:
$\boldsymbol{C_1}$: [u does not have a predecessor]: $Eq \longleftarrow t_u \geq T(u)$;
$\boldsymbol{C_2}$: $[u,v \in O] : Eq \longleftarrow t_v \geq t_u + p_a[(u,v)].T(v)$;
$[u \in O \cup C, v \in C]$
$\boldsymbol{C_3}$: If v is an XOR join that closes a loop (there is $(v,s) \in E_{oc}$ and a path $(s,y_1),\dots,(y_n,u)$ of E_{oc}), then $Eq \longleftarrow t_v \geq \sum_{i=1}^{nl[v]} p_{li}(t_v - t_s) + t_s$; Here, $nl[v]$ is the maximal index of looping stage;
$\boldsymbol{C_4}$: If v is a split connector then $Eq \longleftarrow t_v \geq t_u$;
$[u \in C, v \in O \cup C]$
$\boldsymbol{C_5}$: If u is a join connector then $Eq \longleftarrow t_v \geq t_u$;
$\boldsymbol{C_6}$: If u is a split then
$Eq \longleftarrow t_v \geq p_r[(u,v)].t_u + p_a[(u,v)].T(v)$;
[v is a join connector whose predecessors are $u_1,\dots,u_n$]
$\boldsymbol{C_7}$: If v is an AND JOIN $Eq \longleftarrow t_v \geq t_{u_i} \forall i \in \{1,\dots,n\}$;
$\boldsymbol{C_8}$: If v is an XOR JOIN but does not close a loop $Eq \longleftarrow t_v \geq \sum_{i=1}^{n} t_{u_i}$;
$\boldsymbol{C_9}$: If v is a OR JOIN we generate $Eq \longleftarrow t_v \geq \sum_{i=1}^{2} t_{u_i} + p_{or||}(v).tm(u_1,u_2)$ and $Eq \longleftarrow tm(u_1,u_2) \geq \frac{t_{u_i}}{p_r[(u_i,v)]}, i = 1,\dots 2$;

The application of our constraints on the graph of Figure 7, will give the equations listed in Figure 9.

1. $t_{g_1} \geq 0$	7. $t_{g_3} \geq t_D$
2. $t_A \geq t_{g_1} + T(A)$	8. $t_E \geq p_a[(g_3,E)](t_{g_3} + T(E))$
3. $t_B \geq t_{g_1} + T(B)$	9. $t_F \geq p_a[(g_3,F)](t_{g_3} + T(F))$
4. $t_{g_2} \geq T(A)$	10. $t_H \geq t_G + T(H)$
5. $t_C \geq t_{g_2} + T(C)$	11. $t_I \geq p_a[(g_4,I)](t_{g_4} + T(I))$
6. $t_D \geq t_{g_2} + T(D)$	12. $t_J \geq p_a[(g_4,J)](t_{g_4} + T(J))$
13. $t_G \geq T(G)$	14. $t_{g_5} \geq t_I + t_J$

Figure 9: Intra-pools constraints on the example 7

4.4 Generation of inter-pools constraints

The approximation of $T(RG(x))$ proposed in intra-pools constraints might be incorrect since we neglect here communication. These values will be corrected with the generation of inter-pools constraints. For this, let us recall that we have two type of communication between operations: one-to-one communication and one-to-many communication. For each type of communication, we generate the constraints as follows.

One-to-one communication: we have an arc $(u,v) \in E_{oo}$. Since u is an operation, then we know that from intra-pool constraints, there is a generate equation with the syntax $t_u \geq rhs(u)$. Here $rhs(u)$ is the right hand side of the equation. For handling messages, we will add the equations:
$\boldsymbol{C_{10}}$: $Eq \longleftarrow t_u \geq rhs(u) + T_{Send}(u,v)$;
$\boldsymbol{C_{11}}$: $Eq \longleftarrow t_v \geq t_u + T(u,v) + T_{Recv}(u,v) + T(v)$;

The idea in these equations is that an arc (u,v) states that u will end its execution after the termination of all its predecessors (after the time $rhs(u)$) added to message sending cost. More, the receiving on v can start only if the message is available and its predecessors have been all executed. In these constraints, probability values do not appear because we only admit message exchanges done by operations on which a user request will be routed with an absolute probability of 1.

One-to-many communication: we have $(u,v_1),\dots,(u,v_n) \in E_{oo}$. Then, we will generate the following equations:
$\boldsymbol{C_{12}}$: $Eq \longleftarrow t_u \geq rhs(u) + \sum_{i=1}^{n} T_{Send}(u,v)$;
$\boldsymbol{C_{13}}$: $Eq \longleftarrow t_{v_i} \geq rhs(u) + H(n).\mathbb{E}[T_S] + T(u,v_i) + T_{Recv}(u,v_i) + T(v_i)$. Here, $H(n) = \frac{1}{n} + \frac{2}{n} + \dots + 1$ and $\mathbb{E}[T_S] = \frac{1}{n}\sum_{i=1}^{n} T_{Send}(u,v_i)$. In comparison to the one-to-one case, the main changes in these equations is the time required for an operation to start the reception. The challenge in its computation is that we do not have the order in which the sending will occur. We approximate it as follows. u will make a sending to each operation v_i that will lead to a mean sending time of $\bar{T}_{Send}(u,V) = \frac{1}{n}\sum_{i=1}^{n} T_{Send}(u,v_i)$. Since at a moment only one sending is done, any operation v_i can be at any position between 1 and n in the order of sending. If an operation is at the position j, then the reception for it can start after approximately $j.\bar{T}_{Send}(u,V) + T(u,v_i)$ time units. Admitting that the operation has an equiprobability to be at any position $j \in \{1,\dots,n\}$, we have the formula.

From messages constraints, we derive the equations of Figure 10.

15. $t_B \geq t_{g_1} + T(B) + T_{Send}(B,G)$;
16. $t_G \geq t_B + T(B,G) + T_{Recv}(B,G) + T(G)$;
17. $t_H \geq t_G + T(H) + T_{Send}(H,D)$;
18. $t_D \geq t_H + T(H,D) + T_{Recv}(H,D) + T(D)$;

Figure 10: Inter-pools contraints in the example 7

4.5 Constraints of the objective value

The generation of these constraints aims at cumulating the minimal partial costs set in variables t_x by the other constraints. These constraints are generated pool by pool. Let us suppose that for a pool operations graph P_i, we have the terminal nodes $u_1,\dots,u_e$. We will create for each split connector c_i a set S_{c_i} containing the terminal elements that are in its scope. This gives us a global set $S = \{S_{c_1},\dots,S_{c_m}\}$ that we will denote as the **terminal set cover**. We then choose arbitrarily one set S_{c_i} such that the c_i connector does not comprise another connector $c_j \in \{c_1,\dots,c_m\}$ in its scope. If c_i is an AND split, we generate the equation $t_{S_{c_i}} \geq t_{u_j}, \forall u_j \in S_{c_i}$. If c_i is an XOR split, we generate the equation $t_{S_{c_i}} \geq \sum_{u_j \in S_{c_i}} t_{u_j}$. After this generation, we create

a node $h(S_{c_i})$ whose associate time variable is $t_{S_{c_i}}$ and replace all elements of S_{c_i} in the sets of S by $h(S_{c_i})$. We next remove S_{c_i} from S and start again the process.

At the end, the terminal set cover will contain a unique set S_f. We will then set the constraint $t_{P_i} \geq t_{h(S_{c_u})}, \forall h(S_{c_u}) \in S_f$. We recall that P_i refers to the pool on which we are generating constraints. After the generations of objective values for each pool, we generate the constraints $Z(LP) \geq t_{P_i}, \forall P_i$. The objective function of our linear program consists of minimizing $Z(LP)$.

In applying this principle on the example 7, we will consider the terminal set cover $S = \{S_{g_1}, S_{g_2}, S_{g_3}, S_{g_4}\}$ where: $S_{g_1} = \{B, C, E, F\}$, $S_{g_2} = \{C, E, F\}$, $S_{g_3} = \{E, F\}$, $S_{g_4} = \{g_5\}$. The set eliminations will generate the equations of Figure 11. We end here the presentation of our algorithm

19. $t_{S_{g_3}} \geq t_E + t_F$	20. $t_{S_{g_2}} \geq t_C$	21. $t_{S_{g_2}} \geq t_{S_{g_3}}$	
22. $t_{S_{g_1}} \geq t_{S_{g_2}}$	23. $t_{P_1} \geq t_{S_{g_1}}$	24. $t_{S_{g_4}} \geq t_{g_5}$	25. $t_{P_2} \geq t_{S_{g_4}}$
26. $Z(LP) \geq t_{P_1}$	27. $Z(LP) \geq t_{P_2}$	28. $t_{S_{g_1}} \geq t_B$	

Figure 11: Objective function equations 7

for the generation of LP. Since, we are interested in fast predictions it is important to consider its complexity. We have the following result.

THEOREM 4.1. *Given a graph $G_o = (P, O, C, E_{ocp}, E_{oc}, E_{oo})$, the linear program that we will generate has at most $O(n)$ variables and $O(n)$ equations where $n = |O \cup C|$.*

PROOF. Any element of $O \cup C$ and any pool has one variable to which it is related in the linear program. Since the number of pools is lower than n, we have at most $2|O|$ variables. The constraints in the linear program can be related to pools or to the objective function generation. We have 13 types of pool constraints $C_i, i \in \{1, \ldots, 13\}$. We can easily notice that there exists a positive integer α, such that the number of equations issued from is lower to $13.\alpha.n$.

We have three type of objective function constraints. The first type has the form $t_{S_{c_i}} \geq rhs(S_{c_i})$. For each set S_{c_i}, the number of such equations cannot exceed the outgoing degree of the connector c_i. Thus, we have at most n equations of this form. The second type of objective constraints has the form $t_{P_i} \geq rhs(P_i)$. The number of such equations is bounded by the maximal outgoing degree of a connector in the pool P_i. Thus, the total number of such equations cannot exceed n. Finally, we have at most $|P|$ equations $Z(LP) \geq t_{P_i}$. Since we cannot have more pools than operations and connectors ($|P| \geq n$), we have the proof. □

This result suggests that we can expect a reasonable time in SRT prediction. We did a preliminary evaluation of our approach on 20 examples provided by a discrete time simulator (**ChoreoSim** [2]). Due to space limitation, we can not provide testing details. However, our preliminary results show that our approach can compute SRT with a precision close to the one obtained by a fine simulation.

5. CONCLUSION

In this paper, we introduced a linear programming based approach for SRT prediction on complex services composition that in a business process viewpoint can be considered as choreographies of services. This approach is based on classical aggregation rules for SRT prediction and a new communication model.

For continuing this work, we envision three main issues. The first is to enlarge the set of possible BPMN constructs in the definition of WSC. The second issue is to extend our model to other QoS parameters like the price, reliability or availability of service composition [Cardoso et al. 2004]. Finally, we intend to refine our communication modeling for including more aspects related to Web Services network protocols (SOAP, REST) [Josuttis 2007].

[2]http://ccsl.ime.usp.br/wiki/index.php/ChoreoSim

6. REFERENCES

[Banikazemi et al. 1999] BANIKAZEMI, M., SAMPATHKUMAR, J., PRABHU, S., PANDA, D. K., AND SADAYAPPAN, P. 1999. Communication modeling of heterogeneous networks of workstations for performance characterization of collective operations. In *Proceedings of the Eighth Heterogeneous Computing Workshop*. HCW '99. IEEE Computer Society, Washington, DC, USA, 125–133.

[Cardoso et al. 2004] CARDOSO, J., SHETH, A., MILLER, J., ARNOLD, J., AND KOCHUT, K. 2004. Quality of service for workflows and web service processes. *Web Semantics: Science, Services and Agents on the World Wide Web 1,* 3, 281 – 308.

[Goldman and Ngoko 2012] GOLDMAN, A. AND NGOKO, Y. 2012. On graph reduction for qos prediction of very large web service compositions. In *International Conference on Service Oriented Computing (SCC)*. IEEE Press, Hawai, USA, 258–265.

[Issarny et al. 2011] ISSARNY, V., GEORGANTAS, N., HACHEM, S., ZARRAS, A., VASSILIADIS, P., AUTILI, M., GEROSA, M. A., AND HAMIDA, A. B. 2011. Service-oriented middleware for the future internet: state of the art and research directions. *J. Internet Services and Applications 2,* 1, 23–45.

[Josuttis 2007] JOSUTTIS, N. 2007. *Soa in Practice: The Art of Distributed System Design*. O'Reilly Media, Inc.

[Polyvyanyy et al. 2011] POLYVYANYY, A., GARCÍA-BAÑUELOS, L., FAHLAND, D., AND WESKE, M. 2011. Maximal structuring of acyclic process models. *CoRR abs/1108.2384*.

[Weske 2007] WESKE, M. 2007. *Business Process Management: Concepts, Languages, Architectures*. Springer.

[Xiangpeng et al. 2007] XIANGPENG, Z., CHAO, C., HONGLI, Y., AND ZONGYAN, Q. 2007. A qos view ofweb service choreography. In *Proceedings of the IEEE International Conference on e-Business Engineering*. ICEBE '07. IEEE Computer Society, Washington, DC, USA, 607–611.

[Zeng et al. 2004] ZENG, L., BENATALLAH, B., H.H. NGU, A., DUMAS, M., KALAGNANAM, J., AND CHANG, H. 2004. Qos-aware middleware for web services composition. *IEEE Trans. Softw. Eng. 30,* 5, 311–327.

[Zheng et al. 2010] ZHENG, H., YANG, J., AND ZHAO, W. 2010. Qos analysis and service selection for composite services. In *Services Computing (SCC), 2010 IEEE International Conference on*. Miami, USA, 122 –129.