Reputation-aware Learning for SLA negotiation*

Mohamed Lamine Lamali[1], Dominique Barth[2], and Johanne Cohen[2]

[1] Alcatel-Lucent Bell Labs, Route de Villejust, 91620 Nozay, France
mohamed_lamine.lamali@alcatel-lucent.com
[2] Lab. PRiSM, UMR8144, Université de Versailles
45, av. des Etas-Unis, 78035 Versailles Cedex, France
{dominique.barth, johanne.cohen}@prism.uvsq.fr

Abstract. Assuring Quality of Service (QoS) over multiple Network Service Providers (NSPs) requires to negotiate QoS contracts as Service Level Agreements (SLAs) between NSPs. The goal of an NSP is to maximize its revenues by selling as much as possible SLAs. However, provisioning too much SLAs might increase the risk of violations of the committed QoS thus impacting the NSP's reputation. In order to determine the appropriate provisioning strategies, we propose to extend existing solutions based on Reinforcement Learning with reputation-awareness so that NSPs maximize their revenues.

1 Introduction

Internet applications (*e.g.* Gaming, videoconferencing, etc.) are more and more demanding toward network resources. Some of them might require Quality of Service (QoS) guarantees in order to enhance the end-user experience.

This suggests the Network Service Providers (NSPs) of the Internet to assure QoS and receive compensations accordingly. To support inter-NSP QoS for every service, a Service Level Agreements (SLA) has to be committed between a customer NSP (called *customers* in this paper) and one of its neighbor NSP so as to build up an SLA chain to the destination. An SLA specifies possible QoS guarantees (bandwidth, delay, etc.) and charging conditions (*e.g.* price for a duration). For a given customer request of service, NSPs compete on their SLAs and their prices. The customer will choose an SLA among its neighbor NSPs according to its own utility (sensitivity to the proposed QoS, the price and reputation). In this competition, the provisioning strategy of an NSP is crucial as it conditions the potential SLA violations. SLA violations play the main role in the definition of the NSP reputation.

In this paper, we do not address the reputation propagation mechanisms among customers. We aim to propose a model for NSPs to adapt their SLA provisioning strategy to meet customers' sensitivity to NSPs' reputation. We

* This work has been partially supported by the ETICS-project, a project funded by the European Commission through the 7th ICT-Framework Program. Grant agreement no.: FP7-248567 Contract Number: INFSO-ICT-248567.

opt for a Reinforcement Learning approach as proposed by the authors of [5] and extend their model to take into account the NSP reputation.

Section 2 describes the competition among NSPs and the SLA negotiation problem. Section 3 describes the modeling of this problem. Section 4 presents the Reinforcement Learning algorithm and its using to tackle the problem of SLA negotiation. In sec. 5, we presents the results of simulation performed on an example of inter-NSP network.

2 The SLA negotiation problem and NSPs' reputation

2.1 Competition among NSPs

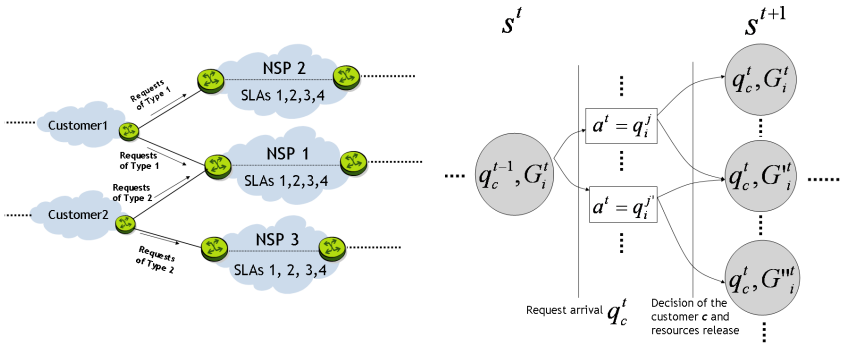
The context of this paper is illustrated by fig. 1(a). There is a set of customers (Customer 1 and Customer 2 on the figure) requesting services with QoS guarantees to their neighbor NSPs (NSP 1, NSP 2 and NSP 3 on the figure).

A **QoS request** of a customer c is a 3-uple q_c s.t. $\mathrm{q}_c = (l_c, d_c, b_c)$ of threshold values (packet-loss, delay and bandwidth respectively) chosen in a finite and discrete set, as for instance the QoS classes defined in [8]. When a customer sends such a request, it sends it simultaneously to all its neighbor NSPs. In fig. 1(a), Customer 1 sends QoS requests to NSP 1 and NSP 2. Customer 2 sends its QoS requests to NSP 2 and NSP 3. When receiving a request, each NSP chooses one SLA in its list of SLAs to make an offer to the customer which sent the request. An **SLA** $\mathbf{q}_i^j$ of an NSP i, where j is the SLA index, is a 3-uple (l_i^j, d_i^j, b_i^j). After receiving all the offers, the customer selects an SLA according to its utility function. Each NSP has a limited capacity. The higher is the part of used capacity, the more the NSP risks to violate its SLAs (*i.e.*, do not comply the committed QoS parameters). The reputation of an NSP is defined as the ratio of the number of its violated SLAs over the number of all its accepted SLAs.We assume that customers share the reputation of NSPs and that:

1. Each NSP has a set of predefined SLAs, and when receiving a request it chooses one of these SLAs to make an offer to the customer,
2. The NSPs know neither the utility function of the customers nor the SLAs proposed by the other NSPs,
3. The customers are sensitive to the price and the QoS parameters of an SLA, and also to the reputation of an NSP.

In such a context, the goal of an NSP is to maximize its revenues by selling its SLAs to the customers. The SLA proposed by an NSP must comply with the customer's QoS request and must not be too expensive in order to avoid customer's rejection. On the other hand, if an NSP sells too many SLAs (and thus uses much of its capacity), the probability of violation of its SLAs increases, therefore its reputation decreases and its next offers will be rejected because of its low reputation. Thus, each NSP must make a trade-off between selling its SLAs and keeping its used capacity low (and its reputation high).

This paper aims to propose an algorithmic solution based on Reinforcement Learning in order to maximize the NSP's revenues by learning the best trade-off



(a) Example of inter-NSP network (b) Example of a part of an MDP

Fig. 1. Inter-NSP network and MDP

between sold SLAs and used capacity. Thus we do not address the issue of next hops and end-to-end computation.

2.2 Related Work

In [5] the authors propose two distributed algorithms based on Reinforcement Learning to compute an SLA chain insuring end-to-end QoS guarantees. The model of the customer acceptance presented in [5] is based on the price and the QoS but does not take into account the NSP reputation. The relation between the reputation and the provisioning strategies is also not considered. The authors of [3] propose a model where NSPs develop an economic alliance allowing them to share their knowledge and improve their revenues. This model takes into consideration the NSPs' capacity but not the reputation. The authors of [9] propose a path selection algorithm an provide a game theoretic analysis and equilibrium policies in the case of two NSPs.

The authors of [7] provide an analytical study of reputation based systems. Some recent works [1, 2] propose management risk and ranking mechanisms to assure QoS in Grid Systems.

The goal of this work is a bit different as it proposes a model of competition taking into account the relation between capacity and reputation, and also the customers' sensitivity to the NSPs' reputation. This works also aims to provide mechanisms that allows NSPs to:

- Infer the customers' sensitivity to the reputation,
- Optimize their long-term revenues by finding a trade-off between selling SLAs and reputation,
- Answer in real-time to customers' requests,
- Take their decisions independently without cooperation (and thus avoid disclosing confidential information).

3 Modeling inter-NSP competition

This section describes the decision system of an NSP and how it can be modeled a Markov Decision Process (MDP). Reinforcement Learning Theory is based on MDP proprieties.

3.1 Decision system of an NSP

The remaining capacity of an NSP i is modeled as a *capacity grade* denoted G_i. It evolves according to the customers' reservations. To avoid the use of all possible values of the remaining capacity, the granularity of the capacity grade should be coarser in order to avoid combinatorial explosion of the number of states. Table 1 provides an example of possible values of G_i with corresponding levels of remaining capacity and probabilities of SLA violation. We consider a discretized time model where each step might not have the same duration but coincides with the decision taken at a request arrival. We assume that an NSP does not receive several requests at the same decision epoch t. The customers send simultaneously their requests to all their neighbor NSPs.

Capacity grade G_i	Remaining capacity	SLA violation probability
0	0% of total capacity	$f_0 = 1$
1	between 0% and 5% of total capacity	$f_1 = 0.95$
2	between 5% and 10% of total capacity	$f_2 = 0.8$
3	between 10% and 20% of total capacity	$f_3 = 0.6$
4	between 20% and 40% of total capacity	$f_4 = 0.2$
5	more than 40% of total capacity	$f_5 = 0.01$

Table 1. Relation between capacity grade, remaining capacity and SLA violation.

An MDP is formalized as $\{\mathcal{S}, \mathcal{A}, P(.,.,.), \mathcal{R}(.,.,.)\}$ where $\mathcal{S}$ is the set of states, $\mathcal{A}$ the set of actions, the transition function $P(s, a, s')$ denotes the probability to move from a state s to a state s' when choosing an action a, and $\mathcal{R}(s, a, s')$ the reward obtained when choosing action a at a state s and moving to a state s'. According to our model, an MDP is defined for each NSP as follows:

- Each state $s \in \mathcal{S}$ is a pair (q_c, G_i) where q_c is a request of the customer c and G_i is a capacity grade. The state $s^t = (q_c^{t-1}, G_i^t)$ denotes the state at decision epoch t, q_c^{t-1} is a request treated at epoch $t-1$, and G_i^t is the remaining capacity at epoch t. The state changes when a request is treated.
- The set of actions $\mathcal{A}$ corresponds to the set of SLAs of the NSP. At a decision epoch t an SLA is chosen, corresponding to action a^t, and proposed as an offer to the customer request. The chosen SLA must respect the QoS requirements of the request. Hence, $\mathcal{A}_t \subseteq \mathcal{A}$.
- The transition function $P(s, a, s') = \Pr(s^{t+1} = s' \mid s^t = s, a_t = a)$. As the last QoS request and the capacity state are included in each state, $P(s, a, s')$

is conditional upon q_c^t and G_i^{t+1}. The capacity grade G_i^{t+1} is itself conditional upon the release of network resources and the acceptance or refusal of the customer which issued q_c^t.
- If the chosen SLA $a^t = \mathrm{q}_i^j$ is accepted by the customer, then the NSP reward $\mathcal{R}(s, a, s') = p_i^j$, the price of the SLA q_i^j. Otherwise $\mathcal{R}(s, a, s') = 0$. The reward obtained at decision epoch t is denoted $r^t = \mathcal{R}(s^t, a^t, s^{t+1})$.

3.2 Customer's utility

As explained in sec. 2.1, the customers are sensitive to the QoS and the price of an SLA, but also to the NSPs' reputation. The utility of each customer increases according the QoS of the proposed SLA and to the reputation of the offering NSP, but it decreases according to the price of the SLA. Thus the utility function of a customer c associated to an SLA $\mathrm{q}_i^j = (l_i^j, d_i^j, b_i^j)$ proposed by an NSP i is defined as:

$$U_c^t(\mathrm{q}_i^j) = \frac{\rho_c ||\mathrm{q}_i^j||_c + \eta_c \mathrm{rep}^{t-1}(i)}{\beta_c p_i^j} \tag{1}$$

where:

- $||\mathrm{q}_i^j||_c$ is a QoS measure of q_i^j and is equal to $\frac{l_c}{l_i^j} + \frac{d_c}{d_i^j} + \frac{b_i^j}{b_c}$,
- $\rho_c \in \Re_+$ is the *QoS rate* of the customer; *i.e.*, the weight of QoS measure in the customer's utility,
- $\beta_c \in \Re_+$ is the weight of the price of an SLA in the customer's utility,
- $\mathrm{rep}^{t-1}(i)$ is the reputation of NSP i at decision epoch $t-1$, it is defined in sec. 3.3. The value η_c is the weight of reputation in the customer's utility.

3.3 Provisioning strategies and reputation

Remaining capacity and violations. As illustrated by Table 1, we consider that there is a strong relation between the level of available capacity and the probability of SLA violations (or trouble in general). In real networks, the level of remaining capacity is usually quite high to avoid troubles but allowing more resources to be provisioned might be in the interest of NSPs which can thus delay their investments and earn more revenues. Moreover, whatever the level of provisioned capacity is, the risk of trouble exists. The authors of [6] provide a tree of trouble causes in networks and their occurences.

Violations and Reputation. In our model, the reputation evolves according to the customers' experience. The SLA proposed by an NSP i can be violated with some probability denoted f_k. The customers take into account their own experience (how many times the SLAs proposed by some NSP failed) to determine the NSPs' reputation. The reputation of an NSP i is at decision epoch t is defined as:

$$\mathrm{rep}^t(i) = 1 - \frac{\#fail(i)}{\#select(i)} \tag{2}$$

where $\#fail(i)$ is the number of times the offers of i were selected by a customer and was violated, and $\#select(i)$ is the number of times the offers of i were selected by a customer.

4 The Q-Learning algorithm

We focus on the Q-learning algorithm because of its "model-free" ability (*i.e.*, it does not require a complete definition of function $P(s, a, s')$). Hence, it is particularly adapted to the inter-NSP SLA competition. The Q-Learning algorithm [10] learns optimal **Q-values** of each pair (state, action) at each decision epoch t. A Q-value of a pair (state, action) at decision epoch t is defined as $Q_t(s, a) = \mathbb{E}[R^t | s^t = s, a^t = a]$ where $R^t = \sum_{k=0}^{T} \gamma^k r^{t+k}$. The Q-values are updated at each iteration according to formula (3). These values converge to the expected gain corresponding to the definition above.

$$Q_{t+1}(s, a) \leftarrow (1 - \alpha^t) Q_t(s, a) + \alpha^t (r^t + \gamma \max_{a' \in \mathcal{A}} Q_t(s', a')) \tag{3}$$

Algorithm 1 initializes the Q-value of each pair (state, action) and updates it when observing the reward r_t and using a discount factor γ and a"learning-rate" denoted α (α^t denotes the value of the learning-rate at decision epoch t). This latter also evolves at each decision epoch. A Q-based policy is the way to select an action based on Q-values. We focus on the ϵ-greedy policy because its behavior is adapted to the trade-off exploration/exploitation. The ϵ-greedy selects the action having the highest Q-value with a probability $1 - \epsilon$, and a random action with probability ϵ.

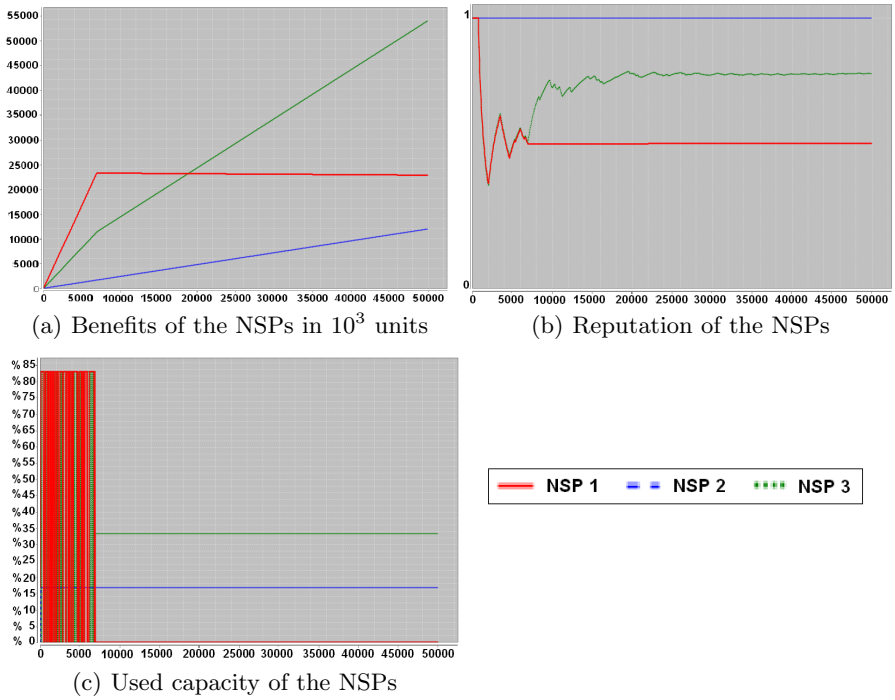
Convergence. The Q-Learning algorithm is proven to converge to optimal Q-values under two assumptions[4]: all pairs (state, action) must be visited infinitely, and $\sum_{t=0}^{\infty} \alpha^t = \infty$ and $\sum_{t=0}^{\infty} (\alpha^t)^2 < \infty$. There is an upper bound according to the update of α. If α is updated using a polynomial function ($\alpha^t = \frac{1}{(t+1)^\omega}$, with $\frac{1}{2} < \omega < 1$) then the convergence time is polynomial in $\frac{1}{1-\gamma}$. If $\omega = 1$ then the convergence time is exponential in $\frac{1}{1-\gamma}$.

Algorithm 1 Q-Learning algorithm

```
Initialization
loop
  At each decision epoch t
    Select an SLA a^t = q_i^j according to ε-greedy policy
    Observe reward r^t and new state s^{t+1}
    Update the Q-values according to formula (3)
end loop
```

5 Experiments

This section relates the results of the simulation performed on the network illustrated by fig. 1(a). Each NSP has a set of 4 SLAs. The first SLA $\mathrm{q}_i^1 = (0.1\%, 30\mathrm{ms}, 250\mathrm{Mbps})$, the second one $\mathrm{q}_i^2 = (0.05\%, 20\mathrm{ms}, 500\mathrm{Mbps})$, the third



(a) Benefits of the NSPs in 10^3 units

(b) Reputation of the NSPs

(c) Used capacity of the NSPs

Fig. 2. Simulation results

one $q_i^3 = (0.01\%, 10\text{ms}, 1000\text{Mbps})$ and the fourth one $q_i^4 = (0.001\%, 5\text{ms}, 2500\text{Mbps})$, for $i \in \{1, 2, 3\}$. According to a function of the SLA parameters, the price of the first SLA is $p_i^1 = 83$ units, the price of the second one is $p_i^2 = 250$ units, the price of the third one is $p_i^3 = 1000$ and the price of the fourth one is $p_i^4 = 5000$ units. Each NSP has a capacity of 3000Mbps. The first customer sends a request $q_c = (0.05\%, 20\text{ms}, 500\text{Mbps})$ at each decision epoch t. The request duration is 2 steps (if the request is sent at decision epoch t, the resource release occurs at epoch $t+2$). The second customer sends a request $q'_c = (0.01\%, 10\text{ms}, 1000\text{Mbps})$ at each decision epoch t and its durations is also 2 steps. The utility functions of the two customers are the same, with $\rho_c = \rho'_c = 1$, $\eta_c = \eta'_c = 5$ and $\beta_c = \beta'_c = 1$. All the NSPs use the Q-Learning algorithm over their own MDP, as described in sec. 3.1. The violation probabilities are those indicated in Table 1.

Figure 5 shows the results of a simulation of 10^5 steps. Figure 2(a) shows that after a phase of learning of about 7000 steps, NSP 2 and NSP 3 monopolize the market and NSP 1's offers are never selected by the customers. Figure 2(b) shows that the reputations of NSP 1 and NSP 3 decrease during the learning phase, but after this phase NSP 3's reputation increases again and stabilizes at 0.8. The NSP 1's reputation stagnates at 0.55 after NSP 1 is never selected. Figure 2(c) shows that NSP 3's used capacity stabilizes at 33%, thus the remaining capacity stabilizes at 67%, which corresponds to a capacity grade $G_2 = 5$.

It appears that after a learning phase, NSP 2 and NSP 3 are in a dominant situation. NSP 1 uses must of its capacity because it offers its SLAs to both customers, and thus its reputation decreases and its offers are rejected. NSP 3 learns a trade-off between used capacity and SLA violation, and thus it recovered a better reputation when its used and remaining capacity stabilizes. Thus, it appears that if there is a dominant position for an NSP, the Q-Learning algorithm converges to the corresponding strategy and learns what is the amount of capacity which maximizes the NSP's revenues while keeping a good reputation.

6 Conclusion

In this paper, we address the SLA negotiation problem taking into account the SLA violation probabilities and their impact on the NSP reputation. The model presented in this paper takes into account the relation between the provisioned capacity of an NSP and the violation probabilities of its SLA. We propose a solution based on Reinforcement Learning techniques in order to allow NSPs to learn the best trade-off between provisioned capacity and good reputation. The simulations performed show that the proposed algorithm is able to learn this trade-off and maximize the NSPs' revenues.

As a future work, we plan to study other learning algorithms and the impact of the different learning parameters on the convergence. Another perspective is to investigate the possible existing equilibria between the NSPs.

References

1. Axel Keller & al. Quality assurance of grid service provisioning by risk aware managing of resource failures. In *CRiSIS*, 2008.
2. Dominic Battré & al. Assessgrid strategies for provider ranking mechanisms in risk-aware grid systems. In *GECON*, 2008.
3. Dominique Barth, Thierry Mautor, and Daniel Villa Monteiro. Impact of Alliances on End-to-End QoS Satisfaction in an Interdomain Network. In *ICC*, 2009.
4. Eyal Even-Dar and Yishay Mansour. Learning Rates for Q-Learning. *Journal of Machine Learning Research*, 5, 2003.
5. Tristan Groléat and Hélia Pouyllau. Distributed inter-domain SLA negotiation using Reinforcement Learning. In *Integrated Network Management*, 2011.
6. A. Medem, M.-I. Akodjenou, and R. Teixeira. Troubleminer: Mining network trouble tickets. In *Integrated Network Management-Workshops, 2009. IM '09. IFIP/IEEE International Symposium on*, june 2009.
7. Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, 2007.
8. N. Seitz. ITU-T QoS standards for IP-based networks. *Communications Magazine, IEEE*, 41(6), june 2003.
9. Kalika Suksomboon, Panita Pongpaibool, and Chaodit Aswakul. An equilibrium policy for providing end-to-end service level agreements in interdomain network. In *WCNC*, 2008.
10. Christopher J. C. H. Watkins and Peter Dayan. Technical Note Q-Learning. *Machine Learning*, 8, 1992.