Supporting Compensations with WS–Agreement*

Carlos Müller, Pablo Fernandez, Octavio Martín-Díaz, Antonio M. Gutierrez, Manuel Resinas, and Antonio Ruiz-Cortés

University of Sevilla

Abstract. During the last years the use of service level agreements (SLA) is rising uncontrollably to describe the rights and obligations of parties involved in service provisioning (typically the service consumer and the service provider); amongst other information, SLA could define guarantees associated with the idea of service level objectives (SLOs) that normally represent key performance indicators of either the consumer or the provider. In case the guarantee is under or over fulfilled SLAs could also define some compensations (i.e. penalties or rewards). In such a context, there have been important steps towards the automation of the analysis of SLAs. One of these steps is a characterization model of SLAs with compensations proposed by the authors in a previous work; and another step is the standardisation effort in the SLAs notation made by WS–Agreement. However, real-world SLAs includes complex concepts that must be considered, namely: (i) SLA terms that specify compensations without an explicit SLO; and (ii) a limit for the compensations. In this paper we extend our prior characterization model considering these complex concepts. Specifically, (i) we provide up to five real-world scenarios whose SLAs incorporate aforementioned new concepts; (ii) we extend our model for compensable guarantees considering terms without an explicit SLO; and (iii) we provide a novel WS–Agreement-based syntax to model SLAs with compensations considering these concepts. These contributions aim to establish a foundation to elaborate tools that could provide an automated support to the modelling and analysis of SLAs with compensations.

1 Introduction

In the recent years the use of service level agreements (SLAs) is in continuous rising to describe the parties rights and obligations to support a reliable service consumption. Specifically, SLAs are composed by different terms that typically define guarantees associated with a certain service level objective (SLOs) and they should be enforced by one party (the guarantor) to another party (the beneficiary); in most cases the former correspond to the service provider, and the latter to the service consumer. For instance, Amazon as provider of their Web

* This work was partially supported by the Spanish and the Andalusian R&D&I programmes (grants P12-TIC-1867, TIN2015-70560-R).

Services (AWS), guarantees an availability $>= 99.95\%$ as general SLO. Additionally, real-world SLAs usually include a set of compensations that represent the consequences of underfulfilling (penalties) or overfulfilling (rewards) the SLOs. For instance, Amazon is penalised if the availability drops below 99.95%. In a previous work [3] we coined the concept of Compensable SLAs referring to such SLAs that include at least a compensation action, either a penalty or a reward.

In such a context, there have been important steps towards the automation of the analysis of SLAs. One of these steps is a characterization model of SLAs with compensations proposed in [3]; and another step is the standardisation effort of WS–Agreement to provide a notation for the SLAs. However, real-world SLAs are thoroughly described and therefore they include many complex concepts that must be considered in the SLA models and notations, namely: (i) guarantees defined by several regional government of Canada and Spain specify compensations without an explicit SLO; and (ii) a limit for the compensations that companies as Amazon establish to compensate consumers of the Elastic Cloud service (AWS EC2) up to a limit of the 10% of the EC2 monthly bill[1].

In this paper we perform a decisive step towards automating the analysis of Compensable SLAs by modelling them considering the aforementioned complex concepts as follows. First, we incorporate to our previous characterization model the definition of *Optimal Thresholds* that can be inferred depending on whether the SLA was specified by the guarantor or the beneficiary. These thresholds are used to validate compensations when the guarantees do not include an explicit SLO. And second, we propose a novel and user friendly notation to specify compensations within the *iAgree syntax*, a WS–Agreement-based [1] language [8, 9]. The advantage of extending iAgree is to leverage its existing analysis tooling support that already incorporates the notion of consistence and validity of SLAs without compensations [8].

This paper is structured as follows: section 2 summarises the characterisation model provided for Compensations in [3]. Such a characterisation model is extended with the Optimal Threshold definition in section 3. In section 4 we propose our iAgree syntax extension to model compensations. Our modelling proposal is validated in five real-world scenarios in section 5. In section 6 we analyse the literature to identify related approaches dealing with compensations. Finally, in section 7 we outline some conclusions and future work.

2 Compensations Model in a Nutshell

In this section we summarised the compensations model we proposed in [3] comprising: (i) the conceptualization of the Compensation Function to express consistently penalties and rewards and (ii) a model for Compensable Guarantees that associate SLOs with Compensation Functions.

---

[1] http://aws.amazon.com/es/ec2/sla/

Compensation Functions

Compensation Functions (CF) are defined over services properties in the context of a guarantee satisfied by a guarantor to a beneficiary. Specifically, they associate two types of compensations depending on the subject and recipient of the compensations: on the one hand, a penalty represents a compensation from the guarantor to the beneficiary and, on the other hand a reward represents a compensation from the beneficiary to the guarantor. Following we formalize CFs by means of a set of supporting core definitions:

Definition 1 (Service Property Values). *The set SP_{sp} denotes the set of all possible values of a service property sp ($SP_{sp} = \{v_1, ..., v_n\}$).*

Definition 2 (Utility Function). *An Utility Function for a certain service property sp, denoted by UF_{sp}, is a function from SP to $\mathbb{R}$ that associates a utility to each of the values; i.e. it defines which service properties values SP_{sp} are more interesting for a given party.*

Definition 3 (Utility Precedence). *Let v_1 and v_2 be values of the set SP_{sp} of a service property sp, and UF_{sp} a utility function defined on the same service property; a precedence relation called utility precedence is defined on SP_{sp} by UF_{sp}. Thus, we denote that v_1 is less interesting than v_2 by $v_1 \prec v_2$.*

Definition 4 (Compensation Function). *A compensation function for a given service property sp, denoted by CF_{sp}, is a function from SP to $\mathbb{R}$ that associates a compensation to each of the values.*

Definition 5 (Compensation Regions). *A compensation function for a given service property sp CF_{sp} defines up to three compensation regions:*
$Penalized(CF_{sp}) = \{v_i \in SP_{sp} \cdot CF_{sp}(v_i) > 0\}$, $Neutral(CF_{sp}) = \{v_i \in SP_{sp} \cdot CF_{sp}(v_i) = 0\}$, and $Rewarded(CF_{sp}) = \{v_i \in SP_{sp} \cdot CF_{sp}(v_i) < 0\}$.

Figure 1 shows a typical compensation function with the regions.

The Validity of CFs is formalized as a property related with the consistence in terms of utility and the saturability of compensations.

Property 1 ($Consistent_{CF}$). A compensation function CF_{sp} is said to be consistent if the compensation for a less interesting value of service property is less or equal than the compensation for a more interesting value according with the utility precedence defined by the Utility function of the beneficiary.
$Consistent_{CF}(CF_{sp}) \iff \forall v_1, v_2 \in SP \cdot v_1 \preceq v_2 \Rightarrow CF_{sp}(v_1) \geq CF_{sp}(v_2)$

Property 2 (Saturated). A compensation function CF_{sp} is said to be saturated if there exist two values (v_{min} and v_{max}) for the service property, that delimit the higher compensation, either penalty or reward. $Saturated(CF_{sp}) \iff \forall v_i \in SP, \exists v_{max}, v_{min} \in SP \cdot CF_{sp}(v_i) \leq CF_{sp}(v_{max}) \wedge CF_{sp}(v_i) \geq CF_{sp}(v_{min})$

Property 3 ($Valid_{CF}$). A compensation function CF_{sp} is said to be valid if it is consistent and saturated. $Valid_{CF}(CF_{sp}) \iff Consistent_{CF}(CF_{sp}) \wedge Saturated(CF_{sp})$

Compensable SLA

A key element of SLAs are the guarantee terms [1] that are typically defined over a service level objective. Based on this conceptualization, we coin the concept of Compensable Guarantees (CG) to those which include a CF and subsequently, Compensable SLAs represent a type of SLA that includes at least one CG.

Definition 6 (Service Level Objective). *An SLO_{sp} is a valid*[2] *assertion defined over a service property sp.*

Definition 7 (Fulfillment Regions). *The assertion defined by an SLO_{sp} determines two regions over the values of the service properties they dealt with: $Fulfilled(SLO_{sp}) = \{v_i \in SP_{sp} \cdot SLO_{sp}\}$, and $Unfulfilled(SLO_{sp}) = \{v_i \in SP_{sp} \cdot \neg SLO_{sp}\}$. This regions are delimited by the threshold Th_{SLO}.*

Definition 8 (Compensable Guarantee). *A compensable guarantee CG_{sp} is a two-tuple of the form (SLO_{sp}, CF_{sp}) in which SLO_{sp} is a service level objective and CF_{sp} is a compensation function that are defined over the same service property sp. $CG_{sp} =< CF_{sp}, SLO_{sp} >$*

Figure 1 depicts the relationships between the fulfillment regions delimited by the SLO (cf. Th_{SLO}) and the compensation regions defined by the CF.

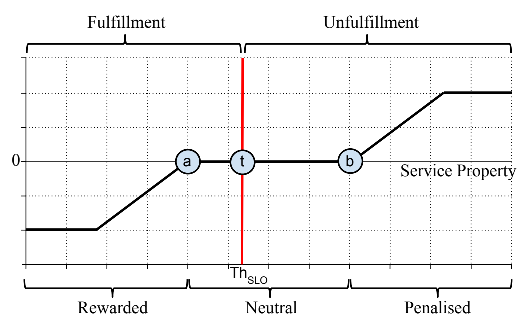


Fig. 1. Generic CF with CG showing compensation and fulfillment regions.

The Validity of CGs is formalized as a property related with the consistency between its SLO and its CF.

Property 4 (Consistent). A compensable Guarantee CG_{sp} is said to be consistent if there is at least one fulfilled and neutral value (that would be TH_{SLO}) and the fulfillment regions are coherent with compensation regions: the fulfilled values are either neutral or rewarded and, complementary, the unfulfilled values are either neutral or penalized.

Property 5 (Valid). Let CG_{sp} a compensable guarantee, it is said to be valid if it is consistent and it contains a valid CF.
$Valid_{CG}(CG_{sp}) \iff Valid_{CF}(CG_{sp}.CF) \wedge Consistent_{CG}(CG_{sp})$

[2] A formal validity criteria for SLOs is presented in [8].

3 Extending the Compensable Guarantees Model

Some compensable guarantees of public SLAs do not explicitly express a particular SLO and the guarantee is only expressed in terms of compensations (c.f. GNWT and IT_AHS scenarios in the following section 5). In these cases two optimal[3] thresholds can be inferred to delimit the fulfillment regions.

Definition 9 (Optimal Thresholds). *Let CG_{sp} a compensable guarantee without defining an explicit SLO SLO_{sp}, two optimal thresholds can be inferred: (i) the optimal threshold for the guarantor (Th_{Gtor}) that corresponds with the less compensable service property value for the beneficiary that does not involve a penalty (e.g. value b in Figure 1) and (ii) the optimal threshold for the beneficiary (Th_{Ben}) that corresponds with the more compensable service property value for the beneficiary that does not involve a reward (e.g. value a in Figure 1).*

$$Th_{Gtor} = v_j | \forall v_i, v_j, v_k \in SP \cdot \quad v_i \preceq v_j \prec v_k \wedge \\ CF_{sp}(v_i) \leq 0 \wedge CF_{sp}(v_j) = 0 \wedge CF_{sp}(v_k) > 0$$

$$Th_{Ben} = v_j | \forall v_i, v_j, v_k \in SP \cdot \quad v_i \prec v_j \preceq v_k \wedge \\ CF_{sp}(v_i) < 0 \wedge CF_{sp}(v_j) = 0 \wedge CF_{sp}(v_k) \geq 0$$

These optimal thresholds are required to check the compensable guarantee validity when they do not define explicitly the SLO. Moreover, the optimal thresholds are also useful for checking if an SLO of a compensable guarantee has been optimally defined in term of compensations. i.e., an SLO is optimally defined if the SLO and the optimal threshold delimit the same fulfillment regions.

4 Extending iAgree with Compensations

The main reason for extending iAgree to model compensations is the possibility of updating its tooling support. Currently, iAgree is supported by IDEAS[4], a service based application with provides, between others, analysis and edition facilities.

Figure 2 includes the proposed iAgree syntax for specifying compensations. Specifically, it comprises: (i) the interval (e.g. `monthly`, `weekly`, etc) in which a compensation is applied; (ii) an optional compensation limit that is an expression to cap the amount to compensate; and (iii) the compensation itself, that is applied when a compensation condition is fulfilled.

[3] Note that these thresholds are optimal in terms of compensations, but not necessary in terms of cost or any other SLO-related aspect.

[4] Available at http://www.isa.us.es/IDEAS/

```
Agreement ident version ...
AgreementTerms ...
  Guarantee Terms
    guaranteeId: Provider/Consumer guarantees SLO; //id, obligated, SLO
      with interval penalty/reward          //interval, compensation type
      [upTo compensationLimit;]                      //compensation limit
        of compensation if compCondition;     //compensation, condition
        ...;
      end
```

Fig. 2. Schema of compensable guarantee terms in *iAgree* syntax

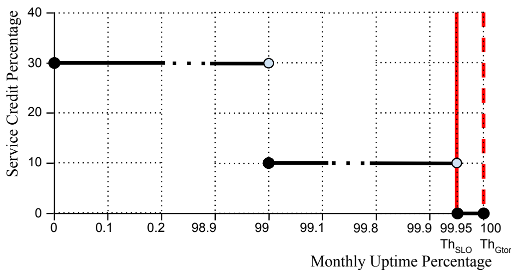
5 Modelling Compensable SLAs of Real-World Scenarios

In this section we introduce each scenario, we present an example of compensation identified in its SLA, and we model it with our proposed iAgree extension. Due to the space limit, we do not include all the presented examples in iAgree, but all of them are publicly available in an IDEAS demo workspace[5]. In addition, some scenarios (c.f. AWS EC2 and GNWT) also include a figure showing the guarantee term and the corresponding CF (as a black line) along with the Th_{SLO} (as a solid vertical line) derived from the SLO; in case there is no explicit SLO, we have inferred an optimal threshold (Th_{Gtor} or Th_{Ben}) depending on whether the SLA was specified by the guarantor (AWS EC2 scenario) or the beneficiary (GNWT scenario); these optimal thresholds are depicted as discontinuous lines. While dark points in the figures denote the inclusion of the service property value in the interval, gray points mean the value exclusion.

AWS EC2 SLA. Amazon Web Services (AWS) is a service catalogue that has boosted the idea of cloud computing in the industry; amongst them, the Elastic Computing Cloud (EC2) represents a widely used Infrastructure as a Service.

Amazon has explicitly published an SLA for EC2[6] that is based on the idea of Monthly Uptime Percentage (MUP); this element characterizes a guarantee over the availability of the virtual resources requested. Specifically, the consequences of failing a certain MUP is defined by Amazon in two levels: in case the MUP drops below 99.95% and in case the MUP drops below 99%. Figure 3(a) depicts the CF of this scenario that is defined as a percent of discount in the next billing cycle a.k.a Service Credit Percentage (SCP). The figure shows that the SLO threshold does not coincide with the optimal threshold of the guarantor in term of compensations. As denoted in the iAgree document of Figure 3(b), Amazon compensates consumers up to a limit of the 10% of the EC2 monthly bill.

[5] https://labs.isa.us.es:8181/IDEAS/demo/Compensations
[6] Available at http://aws.amazon.com/es/ec2/sla/



(a) CG of AWS EC2 (graph).

```
Agreement AmazonAWS version ...
AgreementTerms ...
 Guarantee Terms
  AWS: Provider guarantees
                   MUP >= 99.95;
    onlyIf RegionUnits > 2;
    with monthly penalty
    upTo SCP = 0.1*monthlyBill;
     of SCP = 10 if MUP >= 99
                AND MUP < 99.95;
     of SCP = 30 if MUP < 99;
    end
```

(b) CG of AWS EC2 (iAgree).

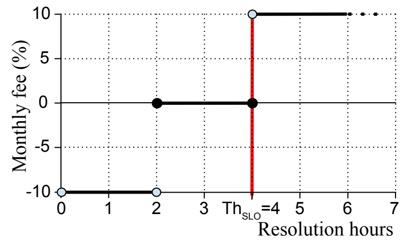
Fig. 3. Compensable Guarantee of AWS EC2 in graph and iAgree.

Telecomm SLA. The regional Government of Andalusia in Spain outsources the installation and management of telecommunication networks. An SLA[7] is specified by the government including some penalties for the services provider.

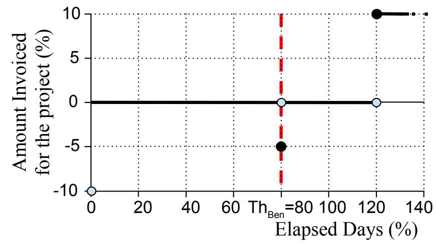
The term (ARG-1) selected from the agreement specifies an SLO demanding that 90% of interventions must be solved. However, some penalties apply for a range of values that fulfill such a demand. Specifically, if the service provider solve more than 95% of interventions no penalties apply, but some bill penalties apply from 90% to 95% of interventions solved. This situation implies a definition error [3]. In addition, compensations are capped at the 20% of the monthly bill.

GNWT SLA. The Government of the Northwest Territories (GNWT) of Canada outsources the IT support. They provide an SLA[8] with the desired guarantees and compensations. Two examples of terms whose CFs are included in Figure 4 have been extracted from its SLA, and Figure 5 specifies it using iAgree syntax. Example GNWT-2 depicts specific times for different milestones that take place in the resolution of problems that have made a critical application function unusable or unavailable and no workaround exists (severity 1 code). In this case, a reward for the provider applies if all problems are resolved in less than 2 hours, and a penalty for the provider applies if any of them is resolved in more than 4 hours. This SLA also includes a term relating the scheduled project delivery and the real project delivery that is shown in example GNWT-4. This term includes a reward for the provider if the elapsed days until delivery are less than 20% lesser than planned but also a penalty for the provider if the elapsed days until delivery are exactly 20% greater than planned. In this case the penalty is not correctly defined [3]. Although an SLO is not explicit in this case, the iAgree document of Figure 5 includes the inferred optimal threshold as guarantee.

[7] Available at http://goo.gl/WIke8y
[8] Available at http://www.fin.gov.nt.ca/ocio/sim/sdlc/3/resources/sla.htm



(a) CG of GNWT-2.



(b) CG of GNWT-4.

Fig. 4. Compensable Guarantees of GNWT examples.

```
Agreement GNWT version ...
AgreementTerms ...
  Guarantee Terms ...
    GNWT-2: Provider guarantees resolutionHours <= 4;
      onlyIf SeverityType = 1;
      with monthly reward
        of feeRewardPercentage = 10   if resolutionHours <= 2;
      end
      with monthly penalty
        of feePenaltyPercentage = 10  if resolutionHours > 4;
      end
    ...
    GNWT-4: Provider guarantees elapsedDaysP < 120;
      with monthly reward
        of invoiceRewardP = 5   if elapsedDaysP = 80;
      end
      with monthly penalty
        of invoicePenaltyP = 10 if elapsedDaysP >= 120;
      end
```

Fig. 5. Compensable guarantee terms of GNWT scenario in *iAgree* syntax

IT_AHS. The regional Government of Andalusia in Spain outsources the IT support services for the public health service. An SLA[9] is specified by the regional government including guarantees on the services and defining the applicable penalties. For instance, it is stated that if the availability of the continuous IT improvement service is less than 95%, it is defined a penalty that depends on the hourly price of the demanded human resource profile. In addition, the SLA caps penalties either at 25% of monthly charge, or at 10% of total budget.

Verizon Terremark Cloud Backup Service. Verizon Terremark offers a variety of cloud services being one of them a cloud backup service[10]. An SLA[11] is specified to assure an availability greater than 99.9% under the penalty of a service credit (defined as 1/30 of the cloud backup services monthly fee) plus one

[9] Available at http://ow.ly/4mO6cB
[10] http://ow.ly/4mORhQ
[11] Available at http://ow.ly/4mOFFd

credit more for each 100 extra minutes in which the availability is not restored. However, such a penalty cannot exceed the 50% of the monthly fee.

6 Related Work

As far as we know, there is no proposal to model compensation in SLAs which enables automating its analysis.

The proposal of Leitner et al. in [6] formalizes the problem of finding the optimal set of adaptations, which minimizes the total costs arising from SLA violations and the adaptations to prevent them. In this work, a model for penalty functions is presented; this formalizations has been the starting point of our characterisation model for compensations presented in [3] and summarised in Section 2. In [10] the same authors present an approach for optimally scheduling incoming requests to virtual computing resources in the cloud, so that the sum of payments for resources and loss incurred by SLA violations is minimized. The example relates the penalty with a service property representing the duration of requests to virtual computing resources in the cloud.

Other examples are the following: Buco et al. propose in [7] an SLA management system, called SAM that provides penalties in a Service Level Management process. Grabarnik et al. propose in [4] a model that can be used to reduce total service costs of IT service providers using alternative delivery teams and external service providers. Rana et al. identifies in [12] how SLOs may be impacted by the choice of specific penalty clauses. Paschke et al. [11] model an SLA to automate its management. This SLA defines minimum and maximum thresholds to compensate SLAs underfulfilling or overfulfilling, but this compensation is ad-hoc modelled through event calls.

In business studies, utility function models are also analysed as they are strongly dependent on customer preferences and behaviour. [2] describes a business scenario with cost, customer expectations and reputation variables where reward function follows a non-monotonic behaviour (based on satisfying preferences from different customers). Similarly, Fenghui Ren et al. analyse in [13] how utility function is obtained from customer objective function (i.e., customers timetable preferences affect how transactions distribute through commercial opening hours).

Angelov et al. propose in [5] a formal representation for contracts to detect and solve different kinds of conflicts. Although the proposed contracts representation supports penalties and rewards by means of reparation clauses, they are not validated against utility functions as proposed in the current paper.

7 Conclusions and Future Work

In this paper we contribute with a way of modelling compensable SLAs by extending iAgree, a WS–Agreement-based [1] language [8, 9]. The extended iAgree notation supports: (i) a previously proposed formal model of compensations [3]; (ii) the specification of compensations limits that is a common practice as shown in the five real-world scenarios included in the paper. In addition, we have also considered those cases in which the SLOs are not explicit in the SLA guarantees. Thus, we have defined the optimal thresholds that can be inferred depending on whether the SLA was specified by the guarantor or the beneficiary.

The contribution constitutes an important step towards the automation of analysis of Compensable SLAs because iAgree has a sounded analysis tooling support framework called IDEAS that is publicly available. Moreover, a public IDEAS demo workspace including all compensable SLA examples of the paper modelled with iAgree has been prepared. As future work we will update the analysis operations of IDEAS to validate both, compensation functions and compensable guarantees, in order to detect singular situations and common pitfalls. Such an update will consist on using mathematics libraries and the already used constraints satisfaction problems solvers to check the properties of our compensation model.

References

1. Andrieux et al. Web Services Agreement Specification (WS-Agreement) (v. gfd-r.192), 2011. OGF - Grid Resource Allocation Agreement Protocol WG.
2. Heski Bar-Isaac and Joyee Deb. What is a good reputation? career concerns with heterogeneous audiences. *Journal of Industrial Organization*, 34(0):44 – 50, 2014.
3. Carlos Müller et al. *Towards a Formal Specification of SLAs with Compensations, OTM Conferences: CoopIS, Amantea, Italy*, pages 295–312. 2014.
4. G. Grabarnik et al. Management of service process qos in a service provider - service supplier environment. In *IEEE Int. Conf. on Ent. Comp., E-Commerce, and E-Services. (CEC/EEE).*, pages 543–550, July 2007.
5. Krasimir Angelov et al. A framework for conflict analysis of normative texts written in controlled natural language. *The Journal of Logic and Algebraic Programming*, 82(57):216 – 240, 2013.
6. P. Leitner, W. Hummer, and S. Dustdar. Cost-based optimization of service compositions. *Services Computing, IEEE Transactions on*, 6(2):239–251, April 2013.
7. M.J. Buco et al. Utility computing sla management based upon business objectives. *IBM Systems Journal*, 43(1):159–178, 2004.
8. C. Müller, M. Resinas, and A. Ruiz-Cortés. Automated analysis of conflicts in ws–agreement. *IEEE Transactions on Services Computing (TSC)*, 2013.
9. Carlos Müller. *On the Automated Analysis of WS-Agreement Documents. Applications to the Processes of Creating and Monitoring Agreements.* International dissertation, Universidad de Sevilla, 2013.
10. P. Leitner et al. Cost-Efficient and Application SLA-Aware Client Side Request Scheduling in an Infrastructure-as-a-Service Cloud. In *Cloud Computing (CLOUD), IEEE 5th International Conference on*, pages 213–220, June 2012.
11. Adrian Paschke and Martin Bichler. Knowledge representation concepts for automated SLA management. 23 November 2006.
12. O. F. Rana, M. Warnier, T. B. Quillinan, F. Brazier, and D. Cojocarasu. Managing violations in service level agreements. In *Grid Middleware and Services Chapter Title - Managing Violations in Service Level Agreements*, pages 349–358, 2008.
13. Fenghui Ren and Minjie Zhang. Bilateral single-issue negotiation model considering nonlinear utility and time constraint. *Decision Support Systems*, 60(0), 2014.