Analysing web-orchestrations under stress using uncertainty profiles

Joaquim Gabarro[1], Maria Serna[1] and Alan Stewart[2]

[1] *ALBCOM Research group, Universitat Politècnica de Catalunya, Jordi Girona Salgado, 1-3, Barcelona, 08034, Spain.*
[2] *School of Computer science, The Queen's University of Belfast, Belfast BT7 1NN, Northern Ireland.*
Email: mjserna@ lsi.upc.edu

An orchestration is a multi-threaded computation which invokes a number of remote services. In practice the responsiveness of a web-service fluctuates with demand; during surges in activity service responsiveness may be degraded, perhaps even to the point of failure. An *uncertainty profile* formalises a user's *perception* of the effects of stress on an orchestration of web-services; it describes a strategic situation, modelled by a zero-sum *angel-daemon* game. Stressed web-service scenarios are analysed, using game theory, in a realistic way, lying between over-optimism (services are entirely reliable) and over-pessimism (all services are broken). The "resilience" of an uncertainty profile can be assessed using the *valuation* of its associated zero-sum game. In order to demonstrate the validity of the approach we consider two measures of resilience and a number of different stress models. It is shown how (i) uncertainty profiles can be ordered by risk (as measured by game valuations) and (ii) the structural properties of risk partial orders can be analysed.

Keywords: Web orchestrations, zero-sum games, services, failures, delays, uncertainty profile, partial order, angel-daemon games, game valuation, assessment, Nash equilibria.

1. INTRODUCTION

A service is a computational method that is made available for general use on the Internet. The response behaviour of a set of services which have been made available for general use on an open network varies with respect to demand. Demand for a particular service S can fluctuate – if the cost and quality of service (QoS) of S are attractive then S is likely to acquire additional users. If demand is excessive then S may fail to deliver its QoS (perhaps, in an extreme situation, S may fail). The aim of this paper is to propose a way to analyse the behaviour of service-based computations in a realistic way that lies between over-optimism (all services satisfy their Service Level Agreements) and over-pessimism (all services are degraded, perhaps even broken).

Risk management is a well-establish discipline in finance (see [1, 2]) which has also been applied to analyse software design [3], information security [4], web services [5, 6] and system failures [7]. The economist Frank Knight has made a distinction between *risk* and *uncertainty* [1] as illustrated by the following quotation taken from Chapter 11 of [8]:

> "*Risk* refers to something that can be measured by mathematical probabilities. In contrast, *uncertainty* refers to something that cannot be measured (using probabilitites) because there are no objective standards to express these probabilities."

Knight gives the following example of a *risky but not uncertain* situation in Part III, Chapter VII of [1]:

> "Thus, in the example given by von Mangoldt, *the bursting of bottles does not introduce an uncertainty or hazard into the business of producing champagne; since in the operations of any producer a practically constant and known proportion of the bottles burst, it does not especially matter even whether the proportion is large or small.* The loss becomes a fixed cost in the industry and is passed on to the consumer, like the outlays for labor or materials or any other."

In contrast an example of an *uncertain situation* is given by John Maynard Keynes in Chapter 12 of [9]:

> "The outstanding fact is the extreme precariousness of the basis of knowledge on which our estimates of prospective yield have to be made....If we speak frankly, we have to admit

> that our basis of knowledge for estimating the yield ten years hence of a railway, a copper mine, a textile factory, the goodwill of a patent medicine, an Atlantic liner, a building in the City of London amounts to little and sometimes to nothing; or even five years hence."

It is debatable if probability is an appropriate technique to use to model service reliability realistically when unpredictable surges in demand can occur. In this paper game theory (see Section 13.6 in [10]) is used to analyse the uncertain behaviour of service-based computations by measuring the robustness and the response delays of user orchestrations when stressed by surges in demand.

Informally, both the delay and the degree of functionality of a *stressed* service-based computation can be analysed using a two-player zero-sum game. Consider an orchestration O which utilises a set S of web-services. In the simplest situation one of the players (the daemon) models over-demand stress whereas the other player (the angel) models the beneficial effects of elasticity. A game proceeds with both players making "moves" in order to stress or remove stress from a set of web-services. For example, the daemon and angel may select the services $s \in S$ and $s' \in S$, respectively, as their moves. This game situation models the evaluation of O in an environment where service s is severely stressed by overdemand whereas service s' employs extra servers to improve performance. In a game the two players have different goals; the daemon tries to damage the user's application as much as possible whereas the angel tries to ensure the least harmful outcome. Both players have bounds placed on the number of services that they can influence. Thus stress is modelled as fluid patches which move over the service space. Each player's scope for stressing services is specified using an *uncertainty profile* which provides a macroscopic and qualitative description of an orchestration under stress without recourse to probability. Occasionally configurations of angel and daemon moves arise in which neither player has an incentive to move (Pure Nash Equilibria). In other games there are randomized rather than pure "fixed points". A game is assessed with respect to a metric specified in its uncertainty profile (perhaps measuring delay or the degree of functionality). An application may be assessed for uncertainty by "playing" a number of games, each with a different uncertainty profile, in order to assess the effects of either (i) different network stress patterns or (ii) varying amounts of redundancy within an orchestration.

An orchestration is defined as "the sequence and conditions in which one web service invokes other web-services in order to realize some useful function" [11]. An orchestration defines control from a *one party perspective* [12]. User-defined orchestrations may include dynamic mechanisms for altering behaviour depending on the responsiveness of underlying web-services. The robustness of an orchestration to underlying service failures (or delays) is referred to as its resilience. Orchestrations of web-services can be specified using the language Orc [13] in which services are called upon to perform sub-computations. An Orc expression specifies how a set of service calls are orchestrated to realise some overall goal. An orchestration may consist of a number of parallel threads each of which returns (publishes) a result.

The *resilience* of an orchestration can be measured in different ways.

- strong assumption: severe stress causes services to fail and become non-operational. Robustness is measured by counting the number of operational threads that remain functioning within a stressed user orchestration.
- weak assumption: stress causes a service's responsiveness to be altered (usually degraded). Orchestration performance is measured by assessing the delay between an orchestration's activation and its completion. Several different delay *stress models* are considered. The basic delay model has four forms of service responsiveness: normal (unstressed service behaviour), service stress due to over-demand, service improvement due to elasticity and service behaviour resulting from the interaction between over-demand and elasticity. We also consider an incremental stress model where delays are modified by angelic and demonic perturbations.

Although a set of profiles may be assessed for risk by obtaining associated angel daemon game valuations it has been shown that computing the value of a angel-daemon game is EXP-complete [14]. In order to avoid the difficulties of directly computing Nash equilibria we show how a set of uncertainty profiles forms a partial order which is compatible with underlying game valuations.

Two classes of monotonic (w.r.t. assessment) mappings between uncertainty profiles are defined. *Angelic up* functions are guaranteed to improve the utility measurement whereas *daemonic down* functions have the opposite effect. Particular instances of up and down functions are used to compare different uncertainty profiles.

In this paper we develop a new unified framework which can be used to assess both the robustness and performance of an orchestration using game theory. Our work extends and generalises previous work for orchestrations which only utilised a robustness measure - see [15] for the initial definition of angel-daemon games, [16] for some elementary monotonicity properties of risk profiles and [17] for the monotonicity properties of uncertainty profiles. In this paper the following significant innovations have been introduced:

- the moves of both players can now coincide (this gives rise to a richer class of more realistic

orchestration games in which a service can be severely stressed by overdemand while at the same time employing elasticity to improve performance);
- a set of new metrics for assessing orchestration delay has been devised;
- new theoretical work on ordering uncertainty profiles offers a means of comparing different stress scenarios without having to calculate game valuations.

The paper is structured as follows. In Section 2 orchestrations, the language Orc, zero-sum games and uncertainty profiles are introduced. In Section 3 two resilience measures for stressed orchestrations are described, namely (i) the number of functioning threads within an orchestration and (ii) the evaluation delay. In Section 4 some examples of assessing resilience are given and some basic properties of uncertainty profiles are derived. In Section 5 up and down functions are used to compare uncertainty profiles involving different orchestrations. The same approach is used in Section 6 to compare uncertainty profiles with the same orchestration but with varying amounts of stress. In Section 7 a review of related work is given. Finally in Section 8 some conclusions are drawn.

2. ORCHESTRATION GAMES

An overview of (i) the orchestration language Orc [13, 18] and (ii) some relevant concepts from game theory are given below.

2.1. Orc

An orchestration is a user-defined program that utilises web services. Typical examples of services might be: an eigensolver, a search engine or a database [19]. A *service* accepts an argument and *publishes* a result value[3]. For example, a call to a search engine, $\mathit{find}(s)$, may publish the set of sites which *currently* offer service s. A service call can publish *at most one response*. A service s may fail to respond (i.e. it is silent) when it is called in an unreliable environment. Service calls may induce *side effects*. The orchestration language Orc [13, 18] contains a number of inbuilt services: 0 is always silent whereas $1(x)$ always publishes its argument x. The service $\mathit{RTimer}(t)$ returns a signal after t units of time – *RTimer* is often used to program time-outs. $\mathit{if}(b)$ publishes a signal if b is true and remains silent otherwise. More generally, an Orc expression may compose a number of service calls into a complex computation. The simplest kind of Orc expression is a service call. The Orc expressions P and Q can be combined using the following operators:

- Sequence $P > x > Q(x)$: Orchestration P is evaluated: for each output v, published by P, an instance $Q(v)$ is invoked. If P publishes the stream of values, $v_1, v_2, \ldots v_n$, then orchestration $P > x > Q(x)$ publishes some interleaving of the set $\{Q(v_1), Q(v_2), \ldots, Q(v_n)\}$. The abbreviation $P \gg Q$ is used in situations where Q does not depend on x.
- Parallelism $P \mid Q$: The independent orchestrations P and Q are executed in parallel; $P \mid Q$ publishes *some* interleaving of the values published by P and Q.
- Pruning $P(x) < x < Q$: Orchestrations P and Q are evaluated in parallel; P may become blocked by a dependency on x. The first result published by Q is bound to x, the remainder of Q's evaluation is terminated and evaluation of the blocked residual of P is resumed.

EXAMPLE 1. Consider the following orchestrations which distribute digital newspapers by email.

$$\mathit{Two\_Each} \;=\; (\mathit{CNN} \mid \mathit{BBC}) > x > (\mathit{EmailAlice}(x) \mid \mathit{EmailBob}(x))$$

Two_Each delivers digital newspapers from both *CNN* and *BBC* to Alice and Bob. However, the orchestration

$$\mathit{One\_Each} \;=\; ((\mathit{CNN} > x > \mathit{EmailAlice}(x)) \mid (\mathit{BBC} > y > \mathit{EmailBob}(y)))$$

only delivers the *CNN* paper to Alice and the *BBC* paper to Bob. These two orchestrations are instances of classical parallel workflow patterns (see [20]) $\mathit{SEQ\_of\_PAR} = (P \mid Q) \gg (R \mid S)$ and $\mathit{PAR\_of\_SEQ} = (P \gg Q) \mid (R \gg S)$. Finally

$$\mathit{Just\_One} \;=\; (\mathit{EmailAlice}(x) \mid \mathit{EmailBob}(x)) < x < (\mathit{CNN} \mid \mathit{BBC})$$

delivers the same journal to Alice and Bob.

All parametrised sites (services) are assumed to have well-defined behaviours:

DEFINITION 2.1 (non-blocking service). *A parametrised service s is* non-blocking *if $s(v_1, \ldots, v_n)$ must publish a result for all well-defined arguments $v_1, \ldots, v_n$; otherwise s is* potentially blocking.

For example, $\mathit{Rtimer}(t)$ is non-blocking while $\mathit{if}(b)$ is potentially blocking[4]. Hereafter we assume that all services used in orchestrations are non-blocking.

> Parameter independent blocking. An orchestration has *parameter independent blocking if all its underlying parametrised services are non-blocking. Such an orchestration can utilise the null service* 0.

[3] The words "publishes" and "returns" are used interchangeably.

[4] It is possible to define a non-blocking variant form of $\mathit{if}(b)$ – when $\mathit{if}(b)$ fails to publish the variant site returns an error signal.

2.2. Zero-sum games

Zero-sum games, Nash equilibria and the *value of a game* (as defined by von Neumann and Morgenstern) are introduced below. Zero-sum games are a particular instance of two player strategic games. Games are described using conventional notation – see [21].

Definition 2.2 (zero-sum game). *A* zero-sum game *is a strategic game, described by a tuple* $\Gamma = \langle A_1, A_2, u\rangle$*, with two players. For* $i \in \{1,2\}$*,* A_i *is the finite set of actions that can be selected by player* i*. The mapping* u *from* $A_1 \times A_2$ *to the rational numbers is the* utility *(or* pay-off*) of player* 1*. Player* 2 *has utility* $-u$*.*

It is assumed that both players are rational and wish to maximize their personal utility (for games with a cost function c player 1 will wish to minimise c). In game theory it is straightforward to move from a utility situation to a cost situation (and vice-versa).

The set of combined actions $A = A_1 \times A_2$ is called the *set of strategy profiles*. Given a strategic game Γ, player i can "make a move" by selecting an action $a_i \in A_i$ (a_i is called a strategy). If each player i selects a strategy a_i *independently* then the joint *strategy profile* is $a = (a_1, a_2)$. Players 1 and 2 have utilities $u(a)$ and $-u(a)$, respectively. The game is zero-sum because $u(a) - u(a) = 0$ (when player 1 benefits from a move then player 2 incurs an equi-sized penalty).

A player's choice of action can be defined probabilistically. Mixed strategies for players 1 and 2 are probability distributions $\alpha : A_1 \to [0,1]$ and $\beta : A_2 \to [0,1]$, respectively. A *mixed strategy profile* is a tuple (α, β) such that

$$u(\alpha,\beta) = \sum_{(a_1,a_2)\in A_1\times A_2} \alpha(a_1)u(a_1,a_2)\beta(a_2)$$

Let Δ_1 and Δ_2 denote the set of mixed strategies for players 1 and 2, respectively. A (pure) strategy profile (a_1, a_2) is a special case of a mixed strategy profile (α, β) where $\alpha(a_1) = 1$ and $\beta(a_2) = 1$.

Definition 2.3. *A mixed strategy profile* (α, β) *is a* Nash equilibrium *if for any* $\alpha' \in \Delta_1$ *we have* $u(\alpha,\beta) \geq u(\alpha',\beta_2)$ *and for any* $\beta' \in \Delta_2$ *we have* $u(\alpha,\beta) \leq u(\alpha,\beta')$*.*

A pure Nash equilibrium, pne, is a Nash equilibrium (a_1, a_2) where a_1 and a_2 a pure strategies. It is well known that any strategic game has a mixed Nash equilibrium [22]. For zero sum games the following stronger result holds:

Theorem 2.1 ([23]). *Let* $\Gamma = (A_1, A_2, u)$ *be a zero sum game. Then:*

- *The value* $\max_{\alpha'\in\Delta_1} \min_{\beta'\in\Delta_2} u(\alpha',\beta')$ *coincides with* $\min_{\beta'\in\Delta_2} \max_{\alpha'\in\Delta_1} u(\alpha',\beta')$*, and*
- (α, β) *is a Nash equilibrium iff*

$$u(\alpha,\beta) = \max_{\alpha'\in\Delta_1} \min_{\beta'\in\Delta_2} u(\alpha',\beta') \quad (1)$$

$$= \min_{\beta'\in\Delta_2} \max_{\alpha'\in\Delta_1} u(\alpha',\beta') \quad (2)$$

Consequently:

Definition 2.4. *The value* $\nu(\Gamma)$ *of a zero sum game with a utility is* $\nu(\Gamma) = \max_{\alpha'\in\Delta_1} \min_{\beta'\in\Delta_2} u(\alpha',\beta')$*.*

The value of a zero sum game with a cost function is:

$$\nu(\Gamma) = \min_{\alpha'\in\Delta_1} \max_{\beta'\in\Delta_2} c(\alpha',\beta') \quad (3)$$

$$= \max_{\beta'\in\Delta_2} \min_{\alpha'\in\Delta_1} c(\alpha',\beta') \quad (4)$$

Nash equilibria can also be characterised as follows (see Proposition 116.2 in [21]):

Proposition 2.1. *A mixed strategy profile* (α, β) *is a mixed Nash equilibrium iff:*

- *for any* $a_1 \in A_1$ *such that* $\alpha(a_1) > 0$ *it follows that* $u(a_1,\beta) = \max_{\alpha'} u(\alpha',\beta)$*, and*
- *for any* $a_2 \in A_2$ *such that* $\beta(a_2) > 0$ *it follows that* $u(\alpha,a_2) = \min_{\beta'} u(\alpha,\beta')$*.*

Example 2. Consider the zero sum game Γ where player 1 can choose one of two actions, top (`t`) or bottom (`b`), and player 2 can choose one of two actions, left (`l`) or right (`r`). The utility u of the game is:

		Player 2 `l`	Player 2 `r`
Player 1	`t`	$u(\mathtt{t},\mathtt{l}) = 0$	1
	`b`	1	0

Γ has no pure Nash equilibrium because, for any pure strategy profile, one player will always want to change strategy. If $(\mathtt{t},\mathtt{l})$ is the current strategy then player 1 will change `t` to `b` (to increase the utility by 1). This move is schematised as $(\mathtt{t},\mathtt{l}) \xrightarrow{u=1} (\mathtt{b},\mathtt{l})$. But profile $(\mathtt{b},\mathtt{l})$ is not stable because player 2 will wish to change `l` to `r` (thereby increasing his utility by 1). The evolution in strategies is schematised below:

$$(\mathtt{t},\mathtt{l}) \xrightarrow{u=1} (\mathtt{b},\mathtt{l}) \xrightarrow{u=0} (\mathtt{b},\mathtt{r}) \xrightarrow{u=1} (\mathtt{t},\mathtt{r}) \xrightarrow{u=0} (\mathtt{t},\mathtt{l}) \xrightarrow{u=1} \cdots$$

Proposition 2.1 can be used to compute a mixed strategy profile (α, β) for this game. Let $\alpha(\mathtt{t}) = p$ where $p > 0$ and $\alpha(\mathtt{b}) = 1 - p$. Then $u(\mathtt{t},\beta) = u(\mathtt{b},\beta)$ and so $p = 1/2$. Likewise if $\beta(\mathtt{l}) = q > 0$ then $\beta(\mathtt{r}) = 1 - q$. In a similar way $u(\alpha,\mathtt{l}) = u(\alpha,\mathtt{r})$ and so $q = 1/2$. Consequently, $\nu(\Gamma) = u(\alpha,\beta) = 1/2$.

2.3. Uncertainty profiles

An *a priori* (global and macroscopic) measure of the behaviour of an orchestration E in a stressful

environment is modelled by an [17] *uncertainty profile* $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$. The stressed environment is described by the parameters $\mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}$ and u:

- $\mathcal{A}$ and $\mathcal{D}$ denote sets of sites (services) which may be effected by stress. The orchestrator has the perception that when an angelic site in $\mathcal{A}$ fails this is unlikely to have a serious impact on the functionality or performance of his web-application. In contrast the orchestrator has the perception that when a daemonic site in $\mathcal{D}$ fails this may well have catastrophic implications for the application. Sites whose behaviour is unknown can occur in both $\mathcal{A}$ and $\mathcal{D}$.
- The spread of network stress is modelled bounding the number of sites under stress[5]. Parameters $b_{\mathcal{A}}$ and $b_{\mathcal{D}}$ specify the number of sites to suffer angelic stress and daemonic stress, respectively.
- The effects of stress are measured by a utility function u (or by a cost function c).

Let $\alpha(E)$ denote the set of sites called by orchestration E, $\alpha_+(E)$ denote $\alpha(E) \setminus \{0\}$ and $\#s$ denote the cardinality of a set of sites s. An uncertainty profiles is defined as:

DEFINITION 2.5 (Uncertainty profile). *An* uncertainty profile *for an orchestration E is a tuple $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ where $\mathcal{A} \cup \mathcal{D} \subseteq \alpha_+(E)$, $b_{\mathcal{A}} \leq \#\mathcal{A}$, $b_{\mathcal{D}} \leq \#\mathcal{D}$ and $u(a, d) \geq 0$ is a utility function defined over $a \subseteq \alpha_+(E)$, $d \subseteq \alpha_+(E)$.*

Given an uncertainty profile $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ a strategic situation arises when E is subjected to combined angel and daemon actions [15].

DEFINITION 2.6 (angel-daemon game). *The uncertainty profile $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ has an associated zero-sum* angel-daemon game $\Gamma(\mathcal{U}) = \langle A_{\mathfrak{a}}, A_{\mathfrak{d}}, u\rangle$*: players $\mathfrak{a}$ (angel) and $\mathfrak{d}$ (daemon) have the following sets of actions,*

- *player $\mathfrak{a}$ selects $b_{\mathcal{A}}$ distinct stressed services from $\mathcal{A}$. The angel's set of actions is*

$$A_{\mathfrak{a}} = \{a \subseteq \mathcal{A} \mid \#a = b_{\mathcal{A}}\}$$

- *player $\mathfrak{d}$ selects $b_{\mathcal{D}}$ distinct stressed services from $\mathcal{D}$. The daemon's set of actions is*

$$A_{\mathfrak{d}} = \{d \subseteq \mathcal{D} \mid \#d = b_{\mathcal{D}}\}$$

- *Services in $\alpha_+(E)\setminus(a\cup d)$ are unaffected by stress.*

Given a strategy profile $s = (a, d)$ the associated utility is $u(a, d)$.

In orchestration games $\mathfrak{a}$ and $\mathfrak{d}$ model "ambiguity or uncertainty" (called *Web Incerta Spiriti* by [8]) in the restless web [17].

[5]Setting a priori bounds for the number of failures in distributed systems is conventional – see Chapter 6 in [24].

DEFINITION 2.7 (assessment). *The assessment $\nu(\mathcal{U})$ of an uncertainty profile $\mathcal{U}$ is defined to be the value of its associated angel-daemon $\Gamma(\mathcal{U})$ (i.e. – see equations (1) and (2) or (3) and (4)).*

In the remainder of the paper utilities are used to measure the robustness of orchestrations to service failure while cost functions are used to measure how stress effects performance.

3. MEASURING RESILIENCE

We wish to measure the *a priori* resilience of an orchestration in various stress scenarios. Two resilience measures are used:

- An *extreme form* of analysis concerns the robustness of orchestrations to service failures. From an observational point of view a broken service is identical to site 0. The effect of broken services on the evaluation of an orchestration E can be assessed by counting the number of functioning threads within E. This corresponds to counting the number of results, $\mathsf{out}(E)$, published by E.
- A more *moderate form* of analysis measures the *delay* caused by stress on the evaluation of an orchestration. Over-demand causes service performance to degrade (increased delays). In order to counteract stress a service may employ elasticity to bring extra servers into operation – in such a situation the performance of a service under stress could even conceivably improve. The delay associated with evaluating E in a stressed environment is denoted by $\delta(E)$.

The measures above provide complementary viewpoints of orchestration resilience in unreliable environments.

3.1. Robustness to failure

Web environments are unreliable. Sites evolve and a user has little (or no) control over the execution environment. When a complex orchestration E is evaluated it may be *unrealistic* to assume that all underlying services will be operational.

> Reliability assumption. Service performance is variable and services can fail. A broken service does not respond when called. A working service always conforms to its specification.

Let $\mathcal{F}$ denote the set of broken sites in a particular environment. The effect of evaluating an orchestration E in such an environment is found by replacing all occurrences of services s, $s \in \mathcal{F}$, by 0. Let $\mathsf{fail}_{\mathcal{F}}(E)$ denote the effect of evaluating E in an environment where services in $\mathcal{F}$ are broken. A measure of the resilience of E in this environment is $\mathsf{out}(\mathsf{fail}_{\mathcal{F}}(E))$ (i.e. the number of outputs, out, published by $\mathsf{fail}_{\mathcal{F}}(E)$) where $0 \leq \mathsf{out}(\mathsf{fail}_{\mathcal{F}}(E)) \leq \mathsf{out}(E)$.

Example 3. Consider the orchestration

$$Robust = \big((Worker_1(x) | Worker_2(x)) < x < (Data | ProxyData)\big)$$

where *Data* and *ProxyData* are data publishing services with the same functionality and $Worker_1$ and $Worker_2$ are worker services. *Robust* is resilient to the failure of a single data service. For $s \in \{Data, ProxyData\}$:

$$\mathsf{out}(\mathsf{fail}_{\{s\}}(Robust)) = \mathsf{out}(\mathsf{fail}_{\{\}}(Robust))$$

However, *Robust* is not resilient to the failure of a worker service.

The notion of parameter independent blocking is used to analyse orchestrations with broken services:

Lemma 3.1. *If orchestration E has parameter independent blocking and $\mathcal{F} \subseteq \alpha_+(E)$ then $\mathsf{fail}_{\mathcal{F}}(E)$ has parameter independent blocking.*

Consequently:

Lemma 3.2. *Given a well formed expression E and a failure set $\mathcal{F} \subseteq \alpha_+(E)$, the values $\mathsf{out}(E)$ and $\mathsf{out}(\mathsf{fail}_{\mathcal{F}}(E))$ can be computed in polynomial time with respect to the length of the expression E.*

Proof. By definition $\mathsf{out}(0) = 0$, $\mathsf{out}(1) = 1$. For a non-blocking service s it follows that

$$\mathsf{out}(s(v_1, \ldots, v_k)) = \begin{cases} 1 & \text{if all parameters are defined} \\ 0 & \text{otherwise} \end{cases}$$

Let E_1, E_2 be parameter independent blocking, well formed Orc expressions. Then

$$\mathsf{out}(E_1 | E_2) = \mathsf{out}(E_1) + \mathsf{out}(E_2)$$
$$\mathsf{out}(E_1 > z > E_2(z)) = \mathsf{out}(E_1) * \mathsf{out}(E_2(z))$$

Finally

$$\mathsf{out}(E_1(z) < z < E_2) = \begin{cases} \mathsf{out}(E_1(z)) & \text{if } \mathsf{out}(E_2) > 0 \\ \mathsf{out}(E_1(\bot)) & \text{otherwise} \end{cases}$$

Here $E(\bot)$ denotes the behaviour of E when z is undefined; $E(\bot)$ is found by replacing all service calls with a z-dependency by 0. Therefore if E has parameter independent blocking then $\mathsf{out}(E)$ can be computed in polynomial time with respect to the length of the expression E. □

Example 4. Reconsider the digital newspaper example. If no failures arise then $\mathsf{out}(Two\_Each) = 4$ and $\mathsf{out}(One\_Each) = 2$. If $\mathcal{F} = \{CNN\}$ then $\mathsf{out}(\mathsf{fail}_{\mathcal{F}}(Two\_Each)) = 2$. If $\mathcal{F} = \{CNN, EmailAlice\}$ then $\mathsf{out}(\mathsf{fail}_{\mathcal{F}}(Two\_Each)) = 1$.

From Lemma 3.2 it follows that

Lemma 3.3 (monotonicity). *Given a parameter independent blocking, well formed expression E and failure sets $\mathcal{F}, \mathcal{F}'$ in $\alpha_+(E)$ where $\mathcal{F} \subseteq \mathcal{F}'$, it follows that $\mathsf{out}(\mathsf{fail}_{\mathcal{F}'}(E)) \leq \mathsf{out}(\mathsf{fail}_{\mathcal{F}}(E))$.*

3.2. Performance (delay costs)

Services may provide "performance guarantees" using service level agreements (SLAs). Given a notion of universal time it is possible to provide response time bounds for services. Service s has a maximal delay $\delta(s)$, (or $\delta(s, x)$ if the delay is input dependent) if it publishes no later than time $\delta(s)$ after being called, $\delta(s) \geq 0$. It is possible to give guarantees about the performance of an orchestration defined over a set of services provided that each underlying service has a response SLA [25].

> Reliability assumption. Given an orchestration E we assume that every site $s \in \alpha(E)$ has a non-negative finite response delay $0 \leq \delta(s) < \infty$. Delay $\delta(s)$ may be perturbed (by a finite amount) when s is placed under stress.

Orchestrations also have performance delays. An evaluation of an orchestration E may publish multiple results; $\mu(E)$ and $\delta(E)$ denote the minimum and maximum delays necessary for publication of *at least one* evaluation result of E and *all* evaluation results of E, respectively. Delays can be lifted from sites to orchestrations in various ways, depending on the underlying computation model. In the following we assume that evaluations utilise the maximum degree of parallelism.

3.2.1. Call-and-Delay environments

In [26] an operational and a traced-based semantics of Orc are presented. These definitions include all of the *transitions* that arise in an orchestration evaluation. For example if expressions P and Q have traces $(t, !m)$ and $(t', !m')$, $t < t'$, respectively, where t and t' are times and m and m' are publication events then the trace of $P|Q$ is $(t, !m)(t', !m')$. However, all that is needed to assess the performance of an orchestration O is the time of its final publication (ie we are only interested in the value t' above) [6]. Below call-and-delay environments are used to calculate *directly* the times of the (first and) last publications of an orchestration. For parameterless expressions E_1 and E_2 and service $s(x)$ we have

$$\delta(E_1 \mid E_2) = max\{\delta(E_1), \delta(E_2)\}$$
$$\delta(E_1 \gg E_2) = \delta(E_1) + \delta(E_2)$$
$$\delta(s(x) < x < E_1) = \mu(E_1) + \delta(s(x))$$

Assessing publication time is more complex when parameterised expressions arise and, in particular, when involved terms appear within prunning expressions. During an orchestration evaluation it is possible that multiple (i) invocations of sub-expressions and (ii) instantiations of variables occur. For example, in an evaluation of $(s \mid s') > x > E_1$ the variable x is instantiated twice and the expression E is called

[6] For pruning it is also necessary to define the time at which the first publication occurs.

twice. In the following call-and-delay environments are used to record the times of variable and sub-expression instantiations, thus enabling the direct calculation of the delay associated with an orchestration evaluation. Our approach adopts time intervals [27, 28] for the special case of Orc.

DEFINITION 3.1 (call-and-delay environment). *Given a sub-orchestration $E(x_1, \ldots, x_n)$, a* call-and-delay environment *for E is a pair $\mathbb{C} = (\mathbb{D}, \mathbb{T})$ where*

$$\mathbb{D} = \{[x_i, \mu_i, \delta_i] \mid 1 \leq i \leq n\},\ \mathbb{T} = [t_1, t_2]$$

Here $\mu_i \leq \delta_i$, for $1 \leq i \leq n$, and $t_1 \leq t_2$.

A call-and-delay environment associates each variable x with a tuple $[x, \mu, \delta]$ where μ and δ are, respectively, the starting and finishing times for the stream values used to instantiate x. In a complementary way t_1 and t_2 denote, respectively, the times of the first and last calls to E.

A call-and-delay environment $\mathbb{C}$ is used to calculate the delays $\mu_{\mathbb{C}}(E)$ and $\delta_{\mathbb{C}}(E)$ of orchestration E. In the following we assume that each thread has its own local delay environment and that the time taken to spawn a thread is negligible.

In order to analyse the expression $E(x_1, \ldots, x_n)$ parameters having the same dependencies are grouped together; in this way E is written as $E(X, Y, Z)$ where X etc. denotes a set of parameters with the same dependencies. $\mathbb{D}[X, Y]$ and $\mathbb{D}[Y, Z]$ etc. denoting different delay environments.

Delays are defined by structural induction over Orc expressions:

Site call $E = s(x_1, \ldots, x_n)$. Then

$$\mu_{\mathbb{C}}(s(x_1, \ldots, x_n)) = \max\{\mu_1, \ldots, \mu_n, t_1\} + \delta(s)$$
$$\delta_{\mathbb{C}}(s(x_1, \ldots, x_n)) = \max\{\delta_1, \ldots, \delta_n, t_2\} + \delta(s)$$

This models a situation where site s is first called at time t_1, all variables must be instantiated before evaluation of s can begin and execution of s takes time $\delta(s)$.

In the case of a parameterless site s:

$$\mu_{\mathbb{C}}(s) = t_1 + \delta(s) \qquad \delta_{\mathbb{C}}(s) = t_2 + \delta(s)$$

Parallel Composition.

$$E(X, Y, Z) = E_1(X, Y) \mid E_2(Y, Z)$$

The delay associated with E is defined using the call-and-delay environments for E_1 and E_2:

$$\mathbb{C}_1 = (\mathbb{D}[X, Y], \mathbb{T}), \quad \mathbb{C}_2 = (\mathbb{D}[Y, Z], \mathbb{T})$$

Then:

$$\mu_{\mathbb{C}}(E(X, Y, Z)) = \max\{\mu_{\mathbb{C}_1}(E_1(X, Y)), \mu_{\mathbb{C}_2}(E_2(Y, Z))\}$$
$$\delta_{\mathbb{C}}(E(X, Y, Z)) = \max\{\delta_{\mathbb{C}_1}(E_1(X, Y)), \delta_{\mathbb{C}_2}(E_2(Y, Z))\}$$

Sequential Composition.

$$E(X, Y, Z) = E_1(X, Y) > u > E_2(u, Y, Z)$$

Here E_1 has the call-and-delay environment

$$\mathbb{C}_1 = (\mathbb{D}[X, Y], \mathbb{T})$$

The values $\mu = \mu_{\mathbb{C}_1}(E_1(X, Y))$ and $\delta = \delta_{\mathbb{C}_1}(E_1(X, Y))$ denote the first and last times at which u is instantiated. An environment for E_2 is:

$$\mathbb{C}_2 = (\mathbb{D}[Y, Z] \cup \{[u, \mu, \delta]\}, \mathbb{T}_2)$$

where $\mathbb{T}_2 = [\mu, \delta]$. The delays associated with E are:

$$\mu_{\mathbb{C}}(E(X, Y, Z)) = \mu_{\mathbb{C}_2}(E_2(u, Y, Z))$$
$$\delta_{\mathbb{C}}(E(X, Y, Z)) = \delta_{\mathbb{C}_2}(E_2(u, Y, Z))$$

Pruning.

$$E(X, Y, Z) = E_1(X, Y, u) < u < E_2(Y, Z)$$

Here the delay of E_2 is calculated using the environment

$$\mathbb{C}_2 = (\mathbb{D}[Y, Z], \mathbb{T})$$

Variable u is only instantiated once - in order to model this part of the evaluation we use the call-and-delay environment

$$\mathbb{C}_1 = (\mathbb{D}[X, Y] \cup \{[u, \mu, \mu]\}, \mathbb{T})$$

where $\mu = \mu_{\mathbb{C}_2}(E_2(Y, Z))$. The delays associated with E are:

$$\mu_{\mathbb{C}}(E(X, Y, Z)) = \mu_{\mathbb{C}_1}(E_1(X, Y, u))$$
$$\delta_{\mathbb{C}}(E(X, Y, Z)) = \delta_{\mathbb{C}_1}(E_1(X, Y, u))$$

The initial orchestration E has no variables and is only called once; consequently $\mathbb{C} = (\emptyset, [0, 0])$. In this setting $\delta(E) = \delta_{\mathbb{C}}(E)$ and $\mu(E) = \mu_{\mathbb{C}}(E)$.

EXAMPLE 5. Consider the orchestration

$$\mathit{Two\_Each} = \mathit{Newspapers} > x > \mathit{Emails}(x)$$

from Example 1 where

$$\mathit{Newspapers} = (\mathit{CNN} \mid \mathit{BBC})$$
$$\mathit{Emails}(x) = (\mathit{EmailAlice}(x) \mid \mathit{EmailBob}(x))$$

The delays associated with *Newspapers* are calculated using the call-and-delay environment $\mathbb{C}_1 = (\emptyset, [0, 0])$:

$$\mu_{\mathbb{C}_1}(\mathit{Newspapers}) = \mu_N = \min\{\delta_{CNN}, \delta_{BBC}\}$$

$$\delta_{\mathbb{C}_1}(\mathit{Newspapers}) = \delta_N = \max\{\delta_{CNN}, \delta_{BBC}\}$$

The delays associated with *Emails* are calculated using the call-and-delay environment $\mathbb{C}_2 = (\{[x, \mu_N, \delta_N]\}, [\mu_N, \delta_N])$. Thus,

$$\mu(\mathit{Two\_Each}) = \mu_N + \mu_E,\ \delta(\mathit{Two\_Each}) = \delta_N + \delta_E$$

where

$$\mu_E = \min\{\delta(EmailAlice), \delta(EmailBob)\}$$
$$\delta_E = \max\{\delta(EmailAlice), \delta(EmailBob)\}$$

Consider the delays associated with the orchestration

$$Just\_One = Emails(x) < x < Newspapers$$

The delay of *Emails* is calculated using the environment $(\{[x, \mu_N, \mu_N]\}, [0, 0])$ and so

$$\mu(Just\_One) = \mu_N + \mu_E,\ \delta(Just\_One) = \mu_N + \delta_E$$

A more complete example of calculating delays is given in Appendix A. Orchestration delays are monotonic in the sense that:

Lemma 3.4. *If $\delta'(s) \geq \delta(s)$ for all $s \in \alpha(E)$ then $\delta'(E) \geq \delta(E)$ and $\mu'(E) \geq \mu(E)$.*

3.2.2. Stress Models

The behaviour of an orchestration E under stress can be modelled by using a modified delay function, $\delta(s)$, for each site s, $s \in \alpha(E)$. Stress *delay* can be modelled in different ways. Initially we consider a model with three sources of stress: *over-demand* which causes service degradation; *elasticity* which deploys servers to meet demand and so maintain (or even improve) performance; and stress resulting from the combined effects of *over-demand* and *elasticity*. Specific delays are defined for each of these cases:

Definition 3.2 (stress model). *A* stress model *for a service s is a tuple $(\delta(s), \delta_o(s), \delta_e(s), \delta_{o+e}(s))$ modelling the performance delays associated with the stress level of service s:*

- *$\delta(s)$ is the delay when s is unstressed;*
- *$\delta_o(s)$ is the delay associated with* over-demand*;*
- *$\delta_e(s)$ is the delay associated with* elasticity*;*
- *$\delta_{o+e}(s)$ is the delay when* over-demand *and* elasticity *interact.*

We assume the following constraints:

$$\delta_e(s) < \delta(s) < \delta_o(s),\ \ \delta_e(s) < \delta_{o+e}(s) < \delta_o(s)$$

A stress model *for an orchestration E is a set $\mathcal{S}$ of underlying service stress models:*

$$\mathcal{S} = \{(\delta(s), \delta_o(s), \delta_e(s), \delta_{o+e}(s)) \mid s \in \alpha(E)\}$$

The delay associated with an evaluation of E, acting under stress model $\mathcal{S}$, is denoted by $\mathsf{delay}_{\mathcal{S}}(E)$ (or $\mathsf{delay}(E)$ when $\mathcal{S}$ is clear form the context).

Example 6. Consider the orchestration

$$RobustData = Worker(x) < x < (Data \mid ProxyData)$$

under a uniform stress model $\mathcal{S}$:

$$\delta(Worker) = \delta(Data) = \delta(ProxyData) = 0.2$$
$$\delta_o(Worker) = \delta_o(Data) = \delta_o(ProxyData) = 50$$
$$\delta_e(Worker) = \delta_e(Data) = \delta_e(ProxyData) = 0.1$$
$$\delta_{o+e}(Worker) = \delta_{o+e}(Data) = \delta_{o+e}(ProxyData) = 5$$

In an unstressed network:

$$\mathsf{delay}(RobustData) = \delta(Worker) + \min\{\delta(Data), \delta(ProxyData)\} = 0.4$$

When the services *Data* and *ProxyData* are both severely stressed by over-demand then

$$\mathsf{delay}(RobustData) = \delta(Worker) + \min\{\delta_o(Data), \delta_o(ProxyData)\} = 50.2$$

If *Data* employs elasticity to combat over-demand stress then

$$\mathsf{delay}(RobustData) = \delta(Worker) + \min\{\delta_{o+e}(Data), \delta_o(ProxyData)\} = 5.2$$

Stress models are monotonic in the sense that:

Lemma 3.5. *Let S and S' be two stress models for an orchestration E:*

$$\mathcal{S} = \{(\delta(s), \delta_o(s), \delta_e(s), \delta_{o+e}(s)) \mid e \in \alpha(E)\}$$
$$\mathcal{S}' = \{(\delta'(s), \delta'_o(s), \delta'_e(s), \delta'_{o+e}(s)) \mid e \in \alpha(E)\}$$

If for all $s \in \alpha(E)$ it follows that

$$\delta(s) \leq \delta'(s),\ \delta_o(s) \leq \delta'_o(s),$$
$$\delta_e(s) \leq \delta'_e(s),\ \delta_{o+e}(s) \leq \delta'_{o+e}(s)$$

then $\mathsf{delay}_{\mathcal{S}}(E) \leq \mathsf{delay}_{\mathcal{S}'}(E)$.

Alternative models of stress are considered in §4.

4. ORCHESTRATIONS UNDER STRESS

The behaviour of orchestrations under *mobile* network stress (i.e. demand for services fluctuates) is modelled and analysed below using game theory. Assumptions about the underlying web environment are specified by uncertainty profiles. The associated games are assessed using game valuations. Example orchestrations and uncertainty profiles are used to illustrate how the resilience measures out and delay can be used to quantify the effects of stress. In addition some properties about the assessment of uncertainty profiles are established.

4.1. Robustness uncertainty profiles

A *robustness profile* is a special kind of uncertainty profile where the utility out counts the number of outputs published by an orchestration.

DEFINITION 4.1 (robustness profile). *A* robustness profile *is a uncertainty profile* $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, \mathsf{out}\rangle$ *with utility* $\mathsf{out}(a,d) = \mathsf{out}(\mathsf{fail}_{a\cup d}(E))$*, for failure sets* $a \subseteq \mathcal{A}$ *and* $b \subseteq \mathcal{B}$

Using out as a utility gives rise to assessments that are non-intuitive to interpret because the assessment range varies with the number of threads in an orchestration. Alternatively, assessments in the range 0–1 can be derived by using a normalised version of out

$$\mathsf{ratio\_out}_{\mathcal{F}}(E) = \begin{cases} \frac{1}{\mathsf{out}(E)}\mathsf{out}(\mathsf{fail}_{\mathcal{F}}(E)) & \text{if } \mathsf{out}(E) > 0 \\ 0 & \text{otherwise} \end{cases}$$

which satisfies $0 \leq \mathsf{ratio\_out}_{\mathcal{F}}(E) \leq 1$. For any failures pair $\mathcal{F} \subseteq \mathcal{F}' \subseteq \alpha_+(E)$ it follows that $\mathsf{ratio\_out}(\mathsf{fail}_{\mathcal{F}'}(E)) \leq \mathsf{ratio\_out}(\mathsf{fail}_{\mathcal{F}}(E))$.

LEMMA 4.1. *Given uncertainty profiles* $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, \mathsf{out}\rangle$ *and* $\mathcal{U}' = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, \mathsf{ratio\_out}\rangle$ *then* $\nu(\mathcal{U}') = \nu(\mathcal{U})/\mathsf{out}(E)$.

The term *resilience factor* is used to refer to normalised assessments.

EXAMPLE 7. Reconsider the workflow pattern $SEQ\_of\_PAR = (P \mid Q) \gg (R \mid S)$ from Example 1. The profile

$$\mathcal{U}_1 = \langle SEQ\_of\_PAR, \{P,Q\}, \{P,Q\}, 1, 1, \mathsf{out}\rangle$$

models a scenario in which an orchestrator is uncertain about the consequences of either P or Q failing but in which he has complete confidence in the operation of both R and S (i.e. $\mathcal{A} = \mathcal{D} = \{P,Q\}$). The resulting game $\Gamma(\mathcal{U}_1)$ is

$\mathfrak{a}$ \ $\mathfrak{d}$	$\{P\}$	$\{Q\}$
$\{P\}$	2	0
$\{Q\}$	0	2

$\Gamma(\mathcal{U}_1)$ has no pure Nash equilibrium – the inherent instability of the game is illustrated by the following schema:

$$(\{Q\},\{Q\}) \xrightarrow{\mathfrak{d}} (\{Q\},\{P\}) \xrightarrow{\mathfrak{a}} (\{P\},\{P\}) \xrightarrow{\mathfrak{d}} (\{P\},\{Q\}) \xrightarrow{\mathfrak{a}} (\{Q\},\{Q\}) \xrightarrow{\mathfrak{d}} \cdots$$

There is a mixed Nash equilibrium with $\alpha = \beta = (1/2, 1/2)$ and game assessment $\nu(\mathcal{U}_1) = 1$ (resilience factor = 0.25).

Given that SEQ_of_PAR is vulnerable to stress it may be desirable to add some failure recovery capacity to the system. Suppose that calls to the vulnerable services P and Q are replaced by the more robust calls:

$$1(x) < x < (P \mid Mirror\_P)$$
$$1(z) < y < (Q \mid Mirror\_Q)$$

The modified orchestration

$$Robust\_SEQ\_of\_PAR = \big(1(x) < x < (P \mid Mirror\_P) \mid 1(z) < y < (Q \mid Mirror\_Q)\big) \gg (R \mid S)$$

now incorporates the back-up services $Mirror\_P$ and $Mirror\_R$ to provide recovery mechanisms in case either P or R fail. Suppose that the orchestrator is unsure of the consequences of failure of either the original or the mirror services. This situation is modelled by the profile

$$\mathcal{U}_2 = \langle Robust\_SEQ\_of\_PAR, \mathcal{F}, \mathcal{F}, 1, 1, \mathsf{out}\rangle$$

where $\mathcal{F} = \{P, Q, Mirror\_P, Mirror\_Q\}$. The game $\Gamma(\mathcal{U}_2)$ is schematised as:

	$\{P\}$	$\{Q\}$	$\{Mirror\_P\}$	$Mirror\_Q$
$\{P\}$	4	4	2	4
$\{Q\}$	4	4	4	2
$\{Mirror\_P\}$	2	4	4	4
$\{Mirror\_Q\}$	4	2	4	4

There is no pure Nash equilibria. Consider a mixed equilibria $\alpha = (\alpha_1, \ldots, \alpha_4)$, $\beta = (\beta_1, \ldots, \beta_4)$ where $0 < \alpha_i$ and $0 < \beta_i$ for $1 \leq i \leq 4$. It follows that

$$\mathsf{out}(\{P\}, \beta) = 4(\alpha_1 + \alpha_2 + \alpha_4) + 2\alpha_3 = 4 - 2\alpha_3$$

and similarly $\mathsf{out}(\{Q\}, \beta) = 4 - 2\alpha_4$. By Proposition 2.1, $\mathsf{out}(\{P\}, \beta) = \mathsf{out}(\{Q\}, \beta)$ and $\alpha_3 = \alpha_4$. By similar arguments $\alpha_i = \alpha_j$ and $\beta_i = \beta_j$ for $i \neq j$. Therefore $\beta_i = \alpha_i = 1/4$ for $1 \leq i \leq 4$ and $\nu(\mathcal{U}_2) = 7/2$ (resilience factor 0.875). It can be seen from this example that uncertainty profiles and game assessments can be used to *quantify* the benefits of adding redundancy to orchestrations.

Uncertainty profiles and $\mathfrak{a}/\mathfrak{d}$-games offer a means of modelling and analysing a wide variety of risky scenarios. Some assessment bounds are given below:

LEMMA 4.2. *The assessment of the profile* $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, \mathsf{out}\rangle$ *satisfies* $0 \leq \nu(\mathcal{U}) \leq \mathsf{out}(E)$.

Proof. Let (α, β) a Nash equilibrium such that $u(\alpha,\beta) = \nu(\mathcal{U})$. For any (a,d) it follows that $\mathsf{out}(\mathsf{fail}_{a\cup d}(E)) \leq \mathsf{out}(E)$. By Lemma 3.3:

$$\begin{aligned} u(\alpha,\beta) &= \sum_{(a,d)\in A_{\mathfrak{a}}\times A_{\mathfrak{d}}} \alpha(a)\beta(d)\mathsf{out}(\mathsf{fail}_{a\cup d})(E) \\ &\leq \sum_{(a,d)\in A_{\mathfrak{a}}\times A_{\mathfrak{d}}} \alpha(a)\beta(d)\mathsf{out}(E) \\ &= \mathsf{out}(E) \sum_{(a,d)\in A_{\mathfrak{a}}\times A_{\mathfrak{d}}} \alpha(a)\beta(d) \;=\; \mathsf{out}(E) \end{aligned}$$

□

Let E and F be expressions. Then

$$\nu(\langle E, \mathcal{A}, \mathcal{D}, 0, 0, \mathsf{out}\rangle) = \mathsf{out}(E)$$
$$\nu(\langle E, \mathcal{A}, \mathcal{D}, \#\mathcal{A}, \#\mathcal{D}, \mathsf{out}\rangle) = \mathsf{out}(\mathsf{fail}_{\mathcal{A}\cup\mathcal{D}}(E))$$
$$\nu(\langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, 0, \mathsf{out}\rangle) = \max_{a\in A_{\mathfrak{a}}} \mathsf{out}(\mathsf{fail}_a(E))$$
$$\nu(\langle E, \mathcal{A}, \mathcal{D}, 0, b_{\mathcal{D}}, \mathsf{out}\rangle) = \min_{d\in A_{\mathfrak{d}}} \mathsf{out}(\mathsf{fail}_d(E))$$

For some restricted cases assessment satisfies:

Lemma 4.3. *Consider E and F where $\alpha_+(E) \cap \alpha_+(F) = \emptyset$ and $\mathcal{A} \subseteq \alpha_+(E)$, $\mathcal{D} \subseteq \alpha_+(F)$. Then*

$$\nu(\langle E \mid F, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, \mathit{out}\rangle) = \max_{a\in A_{\mathfrak{a}}} \mathit{out}(\mathit{fail}_a(E)) + \min_{d\in A_{\mathfrak{d}}} \mathit{out}(\mathit{fail}_d(F))$$
$$\nu(\langle E \gg F, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, \mathit{out}\rangle) = \max_{a\in A_{\mathfrak{a}}} \mathit{out}(\mathit{fail}_a(E)) * \min_{d\in A_{\mathfrak{d}}} \mathit{out}(\mathit{fail}_d(F))$$

Finally we consider an analysis of the Gabrmn [7] information system (adapted from [29]).

Example 8. The Gabrmn IT system is used to manage a number of databases containing clinical data for brain tumour research. The data contains magnetic resonance spectra for both mice and people. New data is constantly being generated; the system must generate backups and provide sufficient storage capacity. The Gabrmn system comprises a number of independents servers.

- Relevant clinical information is stored in a number of *Databases* (eg *MySql* and *Oracle*).
- A number of clinical applications (eg *IDL*, *LcModel* and *SpectraClassifier*) are stored on a designated server, *Apps*.
- A master server (*Proxy*) maintains overall system behaviour.
- Email capability is a fundamental component of the system. The mail service is provided using two servers, *Mail* and *Mirror*. The *Mirror* server is activated when the *Mail* server is down. We approximate the mail system by the Orc expression

$$1(x) < x < (\mathit{Mail} \mid \mathit{Mirror})$$

By using a *coarse grained approach* the components of the system are treated as fundamental units in an Orc expression (i.e. components take the place of sites):

$$\mathit{IT\_System} = \mathit{Proxy} \gg \big((1(x) < x < (\mathit{Mail} \mid \mathit{Mirror})) \mid \mathit{Apps} \mid \mathit{Backups} \mid \mathit{Databases}\big)$$

The number of outputs (out) of *IT_System* provides a measure of the "well-being" of the system (the maximum number of outputs of the system is 4). The system manager should be able to determine the behaviour of *IT_System* under different levels of stress. The system manager may use tentative *heuristic rules* (see [30]) to assign system components to either $\mathcal{A}$ or $\mathcal{D}$ (or both), or to determine the values $b_{\mathcal{A}}$ and $b_{\mathcal{D}}$:

- When there is a *realistic external threat* which may cause the failure of a subsystem S and when *recovery may be difficult* (ie time consuming or expensive or both) then S is assigned to $\mathcal{D}$.
- When $\mathcal{D}$ comprises a number of components then the magnitude of the global threat may be tuned by using a number of different values for $b_{\mathcal{D}}$. Note that when $\#\mathcal{D} = b_{\mathcal{D}}$ we have a version of "Murphy's Law": if anything can go wrong, it will.
- If the threat to a component S is *unrealistic* or S's recovery capability is high then it may be assigned to $\mathcal{A}$. In the following we assume that *Proxy* has a very high recovery capability.
- If $\mathcal{A}$ contains both critical and non-critical subsystems then $b_{\mathcal{A}}$ is chosen in such a way that some critical subsystem will be chosen by the angel in the corresponding game.
- When the threat to a subsystem S is unknown then $S \in \mathcal{A} \cap \mathcal{D}$.
- When there is no threat to S then $S \notin \mathcal{A} \cup \mathcal{D}$.
- In general the values $\#\mathcal{A} - b_{\mathcal{A}}$ and $\#\mathcal{D} - b_{\mathcal{D}}$ quantify the assessor's degree of optimism.

To analyse the system's behaviour in Example 8 a number of scenarios are proposed and an uncertainty profile is constructed for each situation.

First Scenario. The system is poorly maintained and so $\mathcal{A} \cup \mathcal{D} = \alpha(\mathit{IT\_System})$. The systems *Apps*, *Databases*, and *Mail* are considered to be very vulnerable to stress: $\mathcal{D} = \{\mathit{Apps}, \mathit{Databases}, \mathit{Mail}\}$. Assume that it is unlikely that more than one simultaneous serious failure will occur – ie $f_{\mathcal{D}} = 1$. Failure of any of the subsystems $\{\mathit{Backups}, \mathit{Proxy}, \mathit{Mirror}\}$ only gives rise to minor problems – ie $\mathcal{A} = \{\mathit{Backups}, \mathit{Proxy}, \mathit{Mirror}\}$. Again simultaneous failure of components is considered unlikely (ie $f_{\mathcal{A}} = 1$). Let $\mathcal{U}$ denote the corresponding uncertainty profile; the corresponding $\mathfrak{a}$-$\mathfrak{d}$ game $\Gamma(\mathcal{U})$ is:

	$\{\mathit{Apps}\}$	$\{\mathit{Databases}\}$	$\{\mathit{Mail}\}$
$\{\mathit{Backups}\}$	2	2	3
$\{\mathit{Proxy}\}$	0	0	0
$\{\mathit{Mirror}\}$	3	3	3

The game has PNE $(\{\mathit{Mirror}\}, \mathit{Apps}\})$, $(\{\mathit{Mirror}\}, \{\mathit{Databases}\})$ and $(\{\mathit{Mirror}\}, \mathit{Mail}\})$ and assessment 3. The resilience factor $\nu(\mathcal{U})/\mathsf{out}(\mathit{IT\_System})$ is 0.75).

Second Scenario. Now consider a situation where the mail system can fail and the consequences of such a failure are uncertain (ie $\{\mathit{Mail}, \mathit{Mirror}\} \in \mathcal{A} \cap \mathcal{D}$). The system *Apps* is considered to have a high recovery capability, whereas the threat level to *Databases* is

[7] *Aplicacions Biomèdiques de la Ressonància Magnètica Nuclear (Gabrmn),of the Autonomous University of Barcelona (UAB) – see http://gabrmn.uab.es.*

considered to be high. The other subsystems are assumed to be robust to stress. Thus:

$$\mathcal{A} = \{Apps, Mail, Mirror\}$$
$$\mathcal{D} = \{Databases, Mail, Mirror\}$$

As before $b_{\mathcal{A}} = b_{\mathcal{D}} = 1$. The $\mathfrak{a}$-$\mathfrak{d}$ game is

	$\{Databases\}$	$\{Mail\}$	$\{Mirror\}$
$\{Apps\}$	2	3	3
$\{Mail\}$	3	4	3
$\{Mirror\}$	3	3	4

The game has PNE $(Mail, Databases)$ and $(Mirror, Databases)$ with assessment 3.

Third scenario. Suppose that the level of stress applied to Scenario Two is increased so that $b_{\mathcal{A}} = 2$. Then we have the game:

	$\{Databases\}$	$\{Mail\}$	$\{Mirror\}$
$\{Apps, Mail\}$	2	3	2
$\{Apps, Mirror\}$	2	2	3
$\{Mail, Mirror\}$	2	3	4

The game has PNE $\{(a, \{Databases\}) \mid a \in A_{\mathfrak{a}}\}$ with valuation 2 (resilience factor 0.5). As expected increased stress causes functionality to decrease.

4.2. Delay uncertainty profiles

A delay profile is a special kind of uncertainty profile where the cost function delay measures responsiveness. delay is parametrised to account for player actions:

DEFINITION 4.2 (delay profile). *A* delay profile *is an uncertainty profile* $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, \mathsf{delay}_{\mathcal{S}} \rangle$ *where* $\mathcal{S}$ *is a stress model for* E *and* $\mathsf{delay}_{\mathcal{S}}(s)$ *is:*

$$\mathsf{delay}_{\mathcal{S}}(s)[a, d] = \begin{cases} \delta(s) & \text{if } s \notin a \cup d. \\ \delta_o(s) & \text{if } s \in d \setminus a. \\ \delta_e(s) & \text{if } s \in a \setminus d. \\ \delta_{o+e}(s) & \text{if } s \in a \cap d. \end{cases}$$

$\mathsf{delay}_{\mathcal{S}}$ *can be lifted from sites to* Orc *expressions using call-and-delay environments.*

LEMMA 4.4. *For an orchestration* E *and stress model* $\mathcal{S}$ *the function* $\mathsf{delay}_{\mathcal{S}}$ *satisfies:*

- $\mathsf{delay}_{\mathcal{S}}(E)[\emptyset, \emptyset] = \delta(E)$,
- *for any pair* (a, d), $a \subseteq \alpha(E)$, $d \subseteq \alpha(E)$ *we have*

$$\mathsf{delay}_{\mathcal{S}}(E)[\alpha(E), \emptyset] \leq \mathsf{delay}_{\mathcal{S}}(E)[a, d] \leq \mathsf{delay}_{\mathcal{S}}(E)[\emptyset, \alpha(E)]).$$

- *the assessment of* $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, \mathsf{delay}_{\mathcal{S}} \rangle$ *satisfies*

$$\mathsf{delay}_{\mathcal{S}}(E)[\alpha(E), \emptyset] \leq \nu(\mathcal{U}) \leq \mathsf{delay}_{\mathcal{S}}(E)[\emptyset, \alpha(E)]).$$

When appropriate $\mathsf{delay}_{\mathcal{S}}(E)[a, d]$ is abbreviated to $\mathsf{delay}_{\mathcal{S}}(a, d)$ or $\mathsf{delay}(a, d)$. In a delay game player $\mathfrak{a}$ wishes to minimize delay whereas player $\mathfrak{d}$ has the opposite intent.

EXAMPLE 9. Consider the (abbreviated) workflow *RobustData* from Example 6.

$$RD = W(x) < x < (D \mid P)$$

The profile $\mathcal{U} = \langle RD, \alpha(RD), \alpha(RD), 1, 1, \mathsf{delay}_{\mathcal{S}} \rangle$ has over-demand and elasticity in balance. If both players stress site W then the responsiveness of RD is

$$\mathsf{delay}_{\mathcal{S}}(RD)[\{W\}, \{W\}] = \delta_{o+e}(W) + \min\{\delta(D), \delta(P)\} = 5.2$$

The complete game $\Gamma(\mathcal{U})$ is

		$\mathfrak{d}$		
		$\{W\}$	$\{D\}$	$\{P\}$
$\mathfrak{a}$	$\{W\}$	5.2	0.3	0.3
	$\{D\}$	50.1	0.4	0.3
	$\{P\}$	50.1	0.3	0.4

with a PNE $(\{W\}, \{W\})$ and an assessment $\nu(\mathcal{U}) = 5.2$. The assessment quantifies (in seconds) the expected delay associated with orchestration RD in this stressed environment. In the unbalanced profile $\mathcal{U}' = \langle RD, \alpha(RD), \alpha(RD), 1, 2, \mathsf{delay}_{\mathcal{S}} \rangle$ over-demand exceeds the capacity of the system to employ elasticity. The resulting game is

		$\mathfrak{d}$		
		$\{W, D\}$	$\{W, P\}$	$\{D, P\}$
$\mathfrak{a}$	$\{W\}$	5.2	5.2	50.1
	$\{D\}$	50.2	50.1	5.2
	$\{P\}$	50.1	50.2	5.2

There is no PNE. An approximation to the value of this game is 27.66 (see the Appendix B for details). The example illustrates that uncertainty profiles and game theory can be used to quantify the effects of increasing network stress on orchestration responses (delays).

Bounds for a number of boundary case delay profiles are:

LEMMA 4.5.

$$\nu(\langle E, \mathcal{A}, \mathcal{D}, 0, 0, \mathsf{delay}_{\mathcal{S}} \rangle) = \delta(E)$$
$$\nu(\langle E, \mathcal{A}, \mathcal{D}, \#\mathcal{A}, \#\mathcal{D}, \mathsf{delay}_{\mathcal{S}} \rangle) = \mathsf{delay}_{\mathcal{S}}(\mathcal{A}, \mathcal{D})$$
$$\nu(\langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, 0, \mathsf{delay}_{\mathcal{S}} \rangle) = \min_{a \in A_{\mathfrak{a}}} \mathsf{delay}_{\mathcal{S}}(a, \emptyset)$$
$$\nu(\langle E, \mathcal{A}, \mathcal{D}, 0, b_{\mathcal{D}}, \mathsf{delay}_{\mathcal{S}} \rangle) = \max_{d \in A_{\mathfrak{d}}} \mathsf{delay}_{\mathcal{S}}(\emptyset, d)$$

Lemma 4.6. *Given E and F where $\alpha(E)\cap\alpha(F)=\emptyset$, $\mathcal{A}\subseteq\alpha(E)$ and $\mathcal{D}\subseteq\alpha(F)$ then*

$$\begin{aligned}\nu(\langle E \mid F,\mathcal{A},\mathcal{D},b_{\mathcal{A}},b_{\mathcal{D}},\mathsf{delay}_{\mathcal{S}}\rangle)&\\ =\max\{\nu(\langle E,\mathcal{A},\emptyset,b_{\mathcal{A}},0,\mathsf{delay}_{\mathcal{S}}\rangle),&\\ \nu(\langle F,\emptyset,\mathcal{D},0,b_{\mathcal{D}},\mathsf{delay}_{\mathcal{S}}\rangle)\}&\\ \nu(\langle E\gg F,\mathcal{A},\mathcal{D},b_{\mathcal{A}},b_{\mathcal{D}},\mathsf{delay}_{\mathcal{S}}\rangle)=&\\ =\nu(\langle E,\mathcal{A},\emptyset,b_{\mathcal{A}},0,\mathsf{delay}_{\mathcal{S}}\rangle)&\\ +\nu(\langle F,\emptyset,\mathcal{D},0,b_{\mathcal{D}},\mathsf{delay}_{\mathcal{S}}\rangle)&\end{aligned}$$

4.3. Delay games with incremental stress perturbations

In the previous delay games the angel player controlled elasticity while the daemon player controlled over-demand. We now consider other control possibilities.

Suppose that over-demand and elasticity affect the base performance level of a service s in a prespecified way; over-demand increases the response delay by a fixed amount, $\delta_{\mathcal{D}}(s)$ where $\delta_{\mathcal{D}}(s)\geq 0$, and elasticity decreases the delay by a fixed amount $\delta_{\mathcal{A}}(s)$, $\delta(s)\geq\delta_{\mathcal{A}}(s)\geq 0$. Consequently, the base level delay can be increased or decreased $\mathsf{delay}(s)=\delta(s)\pm\cdots$ by prespecified delay variations. The net effect of combining both forms of delay simultaneously is found by additively combining the delay modifiers.

Definition 4.3 (incremental stress model). *An* incremental stress model *for an orchestration E is a tuple $\mathcal{S}=\{(\delta(s),\delta_{\mathcal{A}}(s),\delta_{\mathcal{D}}(s))\mid s\in\alpha(E)\}$ which specifies how sites' response delays vary with stress.*

For each site s, $\delta_{\mathcal{A}}(s)$ and $\delta_{\mathcal{D}}(s)$ denote the potential delay variations that can be applied by the players. Types can be used to model situations where the angel and daemon players can control either over-demand or elasticity. A *stress type* is a pair $t=(t_{\mathcal{A}},t_{\mathcal{D}})$ where $t_{\mathcal{A}}\in\{-1,1\}$ and $t_{\mathcal{D}}\in\{-1,1\}$. Type 1 is used to denote a delay increase whereas type -1 is used to denote a reduced delay.

Definition 4.4 (incremental delay profile). *The* incremental delay profile

$$\mathcal{U}=\langle E,\mathcal{A},\mathcal{D},b_{\mathcal{A}},b_{\mathcal{D}},\mathsf{delay}_{S,t}\rangle.$$

specifies the stress conditions for orchestration E using the incremental stress model

$$\mathcal{S}=\{(\delta(s),\delta_{\mathcal{A}}(s),\delta_{\mathcal{D}}(s))\mid s\in\alpha(E)\}$$

and the stress type $t=(t_{\mathcal{A}},t_{\mathcal{D}})$.

$\Gamma(\mathcal{U})$ is a class of type-dependent games – constraints are necessary to ensure that all site delays remain positive. In stress type $(-1,1)$ the angel controls elasticity and the daemon controls over-demand. To ensure that all delays are positive it is necessary that $\delta(s)\geq\delta_{\mathcal{A}}(s)$. In stress type $(1,-1)$ the angel controls over-demand while the daemon controls elasticity; the constraint $\delta(s)\geq\delta_{\mathcal{D}}(s)$ ensures that delays must remain positive. In stress type $(-1,-1)$ both players control elasticity and it is necessary that $\delta(s)\geq\delta_{\mathcal{A}}(s)+\delta_{\mathcal{D}}(s)$ For type $(1,1)$ both players control over-demand and no constraints are necessary. A summary of the situation is:

type	mnemonic	abbrv	$\mathfrak{a}$	$\mathfrak{d}$
$(-1,+1)$	*standard*	s	provides most help	applies most damage
$(-1,-1)$	*pangloss*	p	provides most help	provides least help
$(+1,+1)$	*faust*	f	applies least damage	applies most damage
$(+1,-1)$	*machiavelli*	m	applies least damage	provides least help

Stress pair $(t_{\mathcal{A}},t_{\mathcal{D}})$ can be abbreviated to the initial letter of its type mnemonic; $t=(-1,1)$ is shortened to $t=s$.

In a delay framework $\mathcal{S}$ with stress type t the definition of site delays is parametrised:

$$\mathsf{delay}_{\mathcal{S},t}(s)[a,d]=\begin{cases}\delta(s) & s\notin a\cup d\\ \delta(s)+t_{\mathcal{A}}\delta_{\mathcal{A}}(s) & s\in a\setminus d\\ \delta(s)+t_{\mathcal{D}}\delta_{\mathcal{A}}(s) & s\in d\setminus a\\ \delta(s)+t_{\mathcal{A}}\delta_{\mathcal{A}}(s)+t_{\mathcal{D}}\delta_{\mathcal{D}}(s) & s\in a\cap d\end{cases}$$

Delays can be lifted from sites to orchestrations. A bound for incremental delay games is:

Lemma 4.7. *The game associated with the profile $\mathcal{U}=\langle E,\mathcal{A},\mathcal{D},b_{\mathcal{A}},b_{\mathcal{D}},\mathsf{delay}_{\mathcal{S},t}\rangle$ with stress model*

$$\mathcal{S}=\{(\delta(e),\delta_{\mathcal{A}}(e),\delta_{\mathcal{D}}(e))\mid e\in\alpha(E)\}$$

has assessment bounds

$$\begin{aligned}\mathsf{delay}_{\mathcal{S},p}(E)[\alpha(E),\alpha(E)]&\\ \leq\nu(\mathcal{U})\leq\mathsf{delay}_{\mathcal{S},f}(E)[\alpha(E),\alpha(E)]&\end{aligned}$$

The types pangloss and faust are used in the bound.

4.4. Uniform delay uncertainty profiles

In a *uniform stress model* all sites have identical delays.

Definition 4.5 (uniform stress model). *The* uniform stress model $\mathcal{S}=\langle\delta,\delta_{\mathcal{A}},\delta_{\mathcal{D}}\rangle$ *has delays $\delta(e)=\delta$, $\delta_{\mathcal{A}}(e)=\delta_{\mathcal{A}}$ and $\delta_{\mathcal{D}}(e)=\delta_{\mathcal{D}}$ for all sites $e\in\alpha_+(E)$.*

$\mathsf{delay}_{\mathcal{S},t}$ is abbreviated to $\mathsf{delay}_{\mathcal{S}}$ or delay_t or even just delay when the context allows.

Example 10. Consider the orchestration

$$eNews = CNN > x > (EmailAlice(x) \mid EmailBob(x))$$

in a uniform stress model $\mathcal{S} = \langle \delta, \delta_{\mathcal{A}}, \delta_{\mathcal{D}} \rangle$. Suppose that the angel has the stress set $a = \{CNN, EmailAlice\}$ and the daemon has an empty stress set ($d = \emptyset$). Then

$$\mathsf{delay}_{\mathcal{S},m}(a,d) = 2\delta + 2\delta_{\mathcal{A}}$$

$$\mathsf{delay}_{\mathcal{S},s}(a,d) = 2\delta - 2\delta_{\mathcal{A}}$$

A uniform delay profile may have type dependent assessments:

type	assessment
standard	$\nu(\mathcal{U}_s) = 2\delta - \delta_A + \delta_D$
pangloss	$\nu(\mathcal{U}_p) = 2\delta - \delta_A - \delta_D$
faust	$\nu(\mathcal{U}_f) = 2\delta + \delta_{\mathcal{D}}$
machiavelli	if $\delta_{\mathcal{A}} \geq \delta_{\mathcal{D}}$ $\nu(\mathcal{U}_m) = 2\delta - \frac{1}{2}\delta_{\mathcal{D}}$ otherwise $\nu(\mathcal{U}_m) = 2\delta - \delta_{\mathcal{D}} + \frac{1}{2}\delta_{\mathcal{A}}$.

FIGURE 1. Analysis of the orchestration $RD = W(x) < x <(D \mid P)$ under the uncertainty profile $\mathcal{U}_t = \langle RD, \{W,D,P\}, \{W,D,P\}, 1, 1, \mathsf{delay}_{\mathcal{S},t} \rangle$, with a uniform stress type $\mathcal{S} = \{\delta, \delta_{\mathcal{A}}, \delta_{\mathcal{D}}\}$ and a type-dependent delay function. For each type t the assessment $\nu(\mathcal{U}_t)$ of the corresponding game $\Gamma(\mathcal{U}_t)$ is shown.

EXAMPLE 11. Consider again the abbreviated *RobustData* workflow:

$$RD = W(x) < x <(D \mid P)$$

An analysis of the workflow when both players apply stress to underlying services can be carried out using the profile

$$\mathcal{U}_t = \langle RD, \{W,D,P\}, \{W,D,P\}, 1, 1, \mathsf{delay}_{\mathcal{S},t} \rangle,$$

where the stress type $\mathcal{S} = \{\delta, \delta_{\mathcal{A}}, \delta_{\mathcal{D}}\}$ is uniform. The resulting analysis is type-dependent.

Type standard. Type $s = (t_{\mathcal{A}}, t_{\mathcal{D}}) = (-1, +1)$. Let $v_S = 2\delta - \delta_A + \delta_D$. The game $\Gamma(\mathcal{U}_s)$ correspond to

$\mathfrak{a}$ \ $\mathfrak{d}$	$\{W\}$	$\{D\}$	$\{P\}$
$\{W\}$	v_S	$2\delta - \delta_{\mathcal{A}}$	$2\delta - \delta_{\mathcal{A}}$
$\{D\}$	v_S	$\min\{2\delta, v_S\}$	$2\delta - \delta_{\mathcal{A}}$
$\{P\}$	v_S	$2\delta - \delta_A$	$\min\{2\delta, v_S\}$

This case models a classic situation in which over-demand and elasticity interact and influence performance. The strategies $(\{W\}, \{W\})$, $(\{D\}, \{W\})$ and $(\{P\}, \{W\})$ are all PNE and $\nu(\mathcal{U}_s) = v_S = 2\delta - \delta_A + \delta_D$.

Type pangloss. Type $p = (t_{\mathcal{A}}, t_{\mathcal{D}}) = (-1, -1)$ let $v_P = 2\delta - \delta_{\mathcal{A}} - \delta_{\mathcal{D}}$ and $\delta_M = \max\{\delta_{\mathcal{A}}, \delta_{\mathcal{D}}\}$ The game $\Gamma(\mathcal{U}_p)$ corresponds to

$\mathfrak{a}$ \ $\mathfrak{d}$	$\{W\}$	$\{D\}$	$\{P\}$
$\{W\}$	v_P	v_P	v_P
$\{D\}$	v_P	v_P	$2\delta - \delta_M$
$\{P\}$	v_P	$2\delta - \delta_M$	v_P

This case provides an analysis of the beneficial effects of elasticity on orchestration delay. In the model application of elasticity is split between the angel and the daemon so as to achieve an "averaging" effect. The strategies $(\{W\}, \{W\})$, $(\{W\}, \{P\})$ and $\{W\}, \{D\})$ are all PNE and $\nu(\mathcal{U}_p) = v_p = 2\delta - \delta_A - \delta_D$.

Type faust. Type $f = (t_{\mathcal{A}}, t_{\mathcal{D}}) = (+1, +1)$. Let $v_F = 2\delta + \delta_{\mathcal{D}}$ and $\delta_m = \min\{\delta_{\mathcal{A}}\delta_{\mathcal{D}}\}$. The game $\Gamma(\mathcal{U}_f)$ is

$\mathfrak{a}$ \ $\mathfrak{d}$	$\{W\}$	$\{D\}$	$\{P\}$
$\{W\}$	$v_F + \delta_{\mathcal{A}}$	$2\delta + \delta_{\mathcal{A}}$	$2\delta + \delta_{\mathcal{A}}$
$\{D\}$	v_F	2δ	$2\delta + \delta_m$
$\{P\}$	v_F	$2\delta + \delta_m$	2δ

This case provides an analysis of the detrimental effects of over-demand on orchestration delay. Application of over-demand is split between the angel and the daemon so as to achieve an "averaging" effect. The failure games used earlier to assess orchestration robustness have similarities with this category (albeit using a utility rather than a cost function). The pure Nash equilibria of this type are $(\{D\}, \{W\})$, $(\{P\}, \{W\})$ and $\nu(\mathcal{U}_f) = v_F = 2\delta + \delta_{\mathcal{D}}$.

Type machiavelli. Type $m = (t_{\mathcal{A}}, t_{\mathcal{D}}) = (+1, -1)$. Let $v_M = 2\delta - \delta_{\mathcal{D}}$ and $\delta_0 = \min\{0, \delta_{\mathcal{A}} - \delta_{\mathcal{D}}\}$. The game $\Gamma(\mathcal{U}_m)$ is

$\mathfrak{a}$ \ $\mathfrak{d}$	$\{W\}$	$\{D\}$	$\{P\}$
$\{W\}$	$v_M + \delta_{\mathcal{A}}$	$v_M + \delta_{\mathcal{A}}$	$v_M + \delta_{\mathcal{A}}$
$\{D\}$	v_M	$2\delta + \delta_0$	v_M
$\{P\}$	v_M	v_M	$2\delta + \delta_0$

This case provides an alternative model of the interplay between over-demand and elasticity. Here the daemon is used to limit the beneficial effects of elasticity while the angel restricts the damage caused by over-demand. Case analysis is carried out below:

Case $\delta_{\mathcal{A}} \geq \delta_{\mathcal{D}}$. Here $\delta_0 = \min\{0, \delta_{\mathcal{A}} - \delta_{\mathcal{D}}\} = 0$ and it is easy to show that there is no PNE. We show that $\alpha = \beta = (0, 1/2, 1/2)$ is a mixed Nash equilibrium. Recall that Proposition 2.1 (for costs) means that (α, β) is a mixed equilibrium if the constraint $\mathsf{delay}_{\mathcal{S},\mathsf{m}}(\{W\}, \beta) \geq \mathsf{delay}_{\mathcal{S},\mathsf{m}}(\{D\}, \beta)$ holds. When $\mathfrak{a}$ chooses a pure strategy the delays are:

$$\mathsf{delay}_{\mathcal{S},\mathsf{m}}(\{W\}, \beta) = v_M + \delta_{\mathcal{A}}$$

$$\mathsf{delay}_{\mathcal{S},\mathsf{m}}(\{D\}, \beta) = \mathsf{delay}_{\mathcal{S},\mathsf{m}}(\{P\}, \beta) = v_M + \frac{1}{2}\delta_{\mathcal{D}}$$

As $\delta_{\mathcal{A}} > \delta_{\mathcal{D}}$, $\mathsf{delay}_{\mathcal{S},\mathsf{m}}(\{W\},\beta) \geq \mathsf{delay}_{\mathcal{S},\mathsf{m}}(\{D\},\beta)$.

When $\mathfrak{d}$ chooses a pure strategy the delays are.

$$\mathsf{delay}_{\mathcal{S},\mathsf{m}}(\alpha,\{W\}) = v_M$$
$$\mathsf{delay}_{\mathcal{S},\mathsf{m}}(\alpha,\{D\}) = \mathsf{delay}_{\mathcal{S},\mathsf{m}}(\alpha,\{P\}) = v_M + \frac{1}{2}\delta_{\mathcal{D}},$$

and $\mathsf{delay}_{\mathcal{S},\mathsf{m}}(\alpha,\{W\}) \leq \mathsf{delay}_{\mathcal{S},\mathsf{m}}(\alpha,\{D\})$ showing that $\alpha = \beta = (0, 1/2, 1/2)$ is a Nash Equilibrium. Therefore, when $\delta_{\mathcal{A}} \geq \delta_{\mathcal{D}}$ the value of the game is $\nu(\mathcal{U}_m) = 2\delta - \frac{1}{2}\delta_{\mathcal{D}}$.

Case $\delta_{\mathcal{A}} < \delta_{\mathcal{D}}$. Here $\delta_0 = \min\{0, \delta_{\mathcal{A}} - \delta_{\mathcal{D}}\} = \delta_{\mathcal{A}} - \delta_{\mathcal{D}} > 0$ and $2\delta + \delta_0 = v_M + \delta_A$. Using Proposition 2.1 it is straightforward to show that $\alpha = \beta = (0, 1/2, 1/2)$ is a mixed Nash equilibrium. Thus the value of the game is $\nu(\mathcal{U}_m) = v_M + \frac{1}{2}\delta_{\mathcal{A}} = 2\delta - \delta_{\mathcal{D}} + \frac{1}{2}\delta_{\mathcal{A}}$.

As expected the elasticity game (pangloss) has the least delay, the over-demand game (faust) the greatest delay while the mixed situation games, with both over-demand and elasticity in balance, (standard and machiavelli) have intermediate delays. The intent of this example is to show the flexibility of uncertainty profiles and game theory in modelling a variety of commonplace situations in service-based computing.

5. COMPARING UNCERTAINTY PROFILES

The value of an uncertainty profile may be used to generate a complete ordering on the set of uncertainty profiles. Unfortunately, the calculation of game valuations can be computationally expensive. In [14] it is shown that computing the value of a robustness profile is EXP-complete. We are interested in deriving conditions for ordering profiles in a way that is consistent with a game assessment ordering. To do this we initially consider a family of transformations over mixed strategies.

5.1. Up and Down functions

Game assessments provide a basis for ordering uncertainty profiles according to their resilience – the bottom element in an ordering is the least resilient profile. We consider a class of transformations between two uncertainty profiles in which either the angelic or daemonic choices are identical.

DEFINITION 5.1 (up and down functions). *Let*

$$\mathcal{U} = \langle E, \mathcal{A}, \mathcal{B}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$$
$$\mathcal{U}' = \langle E', \mathcal{A}', \mathcal{B}, b'_{\mathcal{A}}, b_{\mathcal{D}}, u'\rangle$$
$$\mathcal{U}'' = \langle E'', \mathcal{A}, \mathcal{B}', b_{\mathcal{A}}, b'_{\mathcal{D}}, u''\rangle$$

Consider the pairs of profiles $(\mathcal{U},\mathcal{U}')$ *and* $(\mathcal{U},\mathcal{U}'')$ *which have the same daemonic and angelic choices, respectively. Let* $\Delta_{\mathfrak{a}}$ *be the set of mixed angelic strategies in* $\Gamma(\mathcal{U})$ *and* $\Gamma(\mathcal{U}'')$ *and* $\Delta'_{\mathfrak{a}}$ *be the set of mixed strategies for* $\mathfrak{a}$ *in* $\Gamma(\mathcal{U}')$. *Analogously, let* $\Delta_{\mathfrak{d}}$ *be the set of mixed strategies for* $\mathfrak{d}$ *in* $\Gamma(\mathcal{U})$ *and* $\Gamma(\mathcal{U}')$ *and let* $\Delta'_{\mathfrak{d}}$ *be the set of mixed strategy for* $\mathfrak{d}$ *in* $\Gamma(\mathcal{U}'')$.

- *An* angelic up function f *from* $\mathcal{U}$ *to* $\mathcal{U}'$ *denoted by* $\mathcal{U} \xrightarrow{\mathfrak{a}:f} \mathcal{U}'$ *is any function* $f : \Delta_{\mathfrak{a}} \to \Delta'_{\mathfrak{a}}$ *satisfying* $u(\alpha,\beta) \leq u'(f(\alpha),\beta)$ *for any* $(\alpha,\beta) \in \Delta_{\mathfrak{a}} \times \Delta_{\mathfrak{d}}$.
- *A* daemonic down function f *from* $\mathcal{U}$ *to* $\mathcal{U}''$ *denoted by* $\mathcal{U} \xrightarrow{\mathfrak{d}:f} \mathcal{U}''$ *is any function* $f : \Delta_{\mathfrak{d}} \to \Delta'_{\mathfrak{d}}$ *satisfying* $u(\alpha,\beta) \geq u''(\alpha, f(\beta))$ *for any* $(\alpha,\beta) \in \Delta_{\mathfrak{a}} \times \Delta_{\mathfrak{d}}$.

For profiles with *cost* functions:

- An up function f verifies $c(\alpha,\beta) \geq c'(f(\alpha),\beta)$.
- A down function f verifies $c(\alpha,\beta) \leq c''(\alpha, f(\beta))$.

EXAMPLE 12. Consider the orchestrations

$$E = (T \mid M \mid B) \gg (L \mid R)$$
$$E' = (T \mid B) \gg (L \mid R)$$

and the associated incremental delay profiles

$$\mathcal{U} = \langle E, \{T, M, B\}, \{L, R\}, 1, 1, \mathsf{delay}_{\mathcal{S},f}\rangle,$$
$$\mathcal{U}' = \langle E', \{T, B\}, \{L, R\}, 1, 1, \mathsf{delay}_{\mathcal{S}',f}\rangle,$$

where $\mathcal{S}$ and $\mathcal{S}'$ are incremental stress models of type *faust*. In $\mathcal{S}$ stressed delays always exceed unstressed delays: $\delta(s) + \delta_{\mathcal{A}}(s) > \delta(s')$ and $\delta(s) + \delta_{\mathcal{D}}(s) > \delta(s')$, $\forall$ sites s, s'. In $\mathcal{S}'$ the delays associated with site T are:

$$\delta'(T) = \min\{\delta(T), \delta(M)\}$$
$$\delta'_{\mathcal{A}}(T) = \min\{\delta_{\mathcal{A}}(T), \delta_{\mathcal{A}}(M)\}$$
$$\delta'_{\mathcal{D}}(T) = \delta_{\mathcal{D}}(T)$$

All other delays in $\mathcal{S}'$ are in same as their counterparts in $\mathcal{S}$. Even though $\mathcal{U}$ and $\mathcal{U}'$ have different orchestrations and stress models it is still possible to construct an angelic up function from mixed strategies in $\Gamma(\mathcal{U})$ to mixed strategies in $\Gamma(\mathcal{U}')$. Given two mixed strategies $\alpha \in \Delta_{\mathfrak{a}}$ (in $\Gamma(\mathcal{U})$) and $\alpha' \in \Delta'_{\mathfrak{a}}$ (in $\Gamma(\mathcal{U}')$) the function $f(\alpha) = \alpha'$ where

$$\alpha'(\{T\}) = \alpha(\{T\}) + \alpha(\{M\}) \quad \text{and} \quad \alpha'(\{B\}) = \alpha(\{B\})$$

is an angelic up function if $\mathsf{delay}_{\mathcal{S},f}(\alpha,\beta) \geq \mathsf{delay}_{\mathcal{S}',f}(f(\alpha),\beta)$. Now

$$\mathsf{delay}_{\mathcal{S},f}(\{T\},\{L\}) = \delta(T) + \delta_{\mathcal{A}}(T) + \delta(L) + \delta_{\mathcal{D}}(L)$$
$$\mathsf{delay}_{\mathcal{S}',f}(\{T\},\{L\}) = \min\{\delta(T),\delta(M)\} + \min\{\delta_{\mathcal{A}}(T),\delta_{\mathcal{A}}(M)\} + \delta(L) + \delta_{\mathcal{D}}(L)$$

and so

$$\mathsf{delay}_{\mathcal{S}',f}(\{T\},\{L\}) \leq \mathsf{delay}_{\mathcal{S},f}(\{T\},\{L\})$$
$$\mathsf{delay}_{\mathcal{S}',f}(\{T\},\{L\}) \leq \mathsf{delay}_{\mathcal{S},f}(\{M\},\{L\})$$

Abbreviating $\mathsf{delay}_{\mathcal{S},f}$ to delay and $\mathsf{delay}_{\mathcal{S}',f}$ to delay' we have:

$$\begin{aligned}
&\mathsf{delay}(\alpha,\beta) = \\
&\quad \alpha(\{T\})\beta(\{L\})\mathsf{delay}(\{T\},\{L\}) \\
&\quad + \alpha(\{M\})\beta(\{L\})\mathsf{delay}(\{M\},\{L\}) + \cdots \\
&\quad + \alpha(\{B\})\beta(\{L\})\mathsf{delay}(\{B\},\{L\}) \\
&\quad + \alpha(\{B\})\beta(\{R\})\mathsf{delay}(\{B\},\{R\}) \\
&\geq (\alpha(\{T\}) + \alpha(\{M\}))\beta(\{L\})\mathsf{delay}'(\{T\},\{L\}) + \cdots \\
&\quad + \alpha(\{B\})\beta(\{L\})\mathsf{delay}'(\{B\},\{L\}) \\
&\quad + \alpha(\{B\})\beta(\{R\})\mathsf{delay}'(\{B\},\{R\}) \\
&= \mathsf{delay}'(\alpha',\beta).
\end{aligned}$$

Up and down functions are monotonic in the sense that:

LEMMA 5.1. *If there is an angelic* up *function from* $\mathcal{U}$ *to* $\mathcal{U}'$ *then* $\nu(\mathcal{U}) \leq \nu(\mathcal{U}')$. *If there is a daemonic* down *function from* $\mathcal{U}''$ *to* $\mathcal{U}$ *then* $\nu(\mathcal{U}) \leq \nu(\mathcal{U}'')$.

Proof. Consider the profiles $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{B}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$, $\mathcal{U}' = \langle E', \mathcal{A}', \mathcal{B}, b'_{\mathcal{A}}, b_{\mathcal{D}}, u'\rangle$ and $\mathcal{U}'' = \langle E'', \mathcal{A}, \mathcal{B}', b_{\mathcal{A}}, b'_{\mathcal{D}}, u''\rangle$. Given an up function up from $\mathcal{U}$ and $\mathcal{U}'$ it follows that $u(\alpha,\beta) \leq u'(\mathsf{up}(\alpha),\beta)$ for all $\alpha \in \Delta_{\mathfrak{a}}$ and $\beta \in \Delta_{\mathfrak{d}}$. For any $\alpha \in \Delta_{\mathfrak{a}}$ choose $\beta^*(\alpha) \in \Delta_{\mathfrak{d}}$ so that $u'(\mathsf{up}(\alpha),\beta^*(\alpha)) = \min_\beta u'(\mathsf{up}(\alpha),\beta)$. This is the case for any $\alpha \in \Delta_{\mathfrak{a}}$ and so

$$\min_\beta u'(\mathsf{up}(\alpha),\beta) \leq \max_{\alpha'} \min_\beta u'(\alpha',\beta) = \nu(\mathcal{U}')$$

Therefore $u(\alpha,\beta^*(\alpha)) \leq u'(\mathsf{up}(\alpha),\beta^*(\alpha)) \leq \nu(\Gamma')$ for any α. In particular $\max_\alpha u(\alpha,\beta^*(\alpha)) \leq \nu(\mathcal{U}')$. For any $\alpha \in \Delta_{\mathfrak{a}}$ we have $\beta^*(\alpha) \in \Delta_{\mathfrak{d}}$, and so $\min_\beta u(\alpha,\beta) \leq u(\alpha,\beta^*(\alpha))$. Therefore $\nu(\mathcal{U}) = \max_\alpha \min_\beta u(\alpha,\beta) \leq \max_\alpha u(\alpha,\beta^*(\alpha)) \leq \nu(\mathcal{U}')$.
Given a down function down from $\mathcal{U}''$ to $\mathcal{U}$ it follows that $u''(\alpha,\beta') \geq u(\alpha,\mathsf{down}(\beta'))$ for any α, β'. Using similar arguments (but replacing the angel by the daemon and reversing the roles of min and max) we have $\nu(\mathcal{U}'') \geq \nu(\mathcal{U})$. □

A simple example of an angelic up function arises when two profiles differ only in their utility functions:

LEMMA 5.2. *Consider* $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ *and* $\mathcal{U}' = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u'\rangle$ *such that for all* $(a,d) \in \mathcal{P}(\mathcal{A}) \times \mathcal{P}(\mathcal{D})$ *it holds* $u(a,d) \leq u'(a,d)$, *then* $\nu(\mathcal{U}) \leq \nu(\mathcal{U}')$.

Proof. Let $A_{\mathfrak{a}} = \{a \subseteq \mathcal{A} \mid \#a = b_{\mathcal{A}}\}$ and $A_{\mathfrak{d}} = \{d \subseteq \mathcal{D} \mid \#d = b_{\mathcal{D}}\}$. Then $\Gamma(\mathcal{U}) = \langle A_{\mathfrak{a}}, A_{\mathfrak{d}}, u\rangle$, $\Gamma(\mathcal{U}') = \langle A_{\mathfrak{a}}, A_{\mathfrak{d}}, u'\rangle$ and both games have the same mixed strategies $\Delta_{\mathfrak{a}}$ and $\Delta_{\mathfrak{d}}$. The identity function $\mathsf{id} : \Delta_{\mathfrak{a}} \to \Delta_{\mathfrak{a}}$ is an up function $\mathcal{U} \xrightarrow{\mathfrak{a}:\mathsf{id}} \mathcal{U}'$ because

$$\begin{aligned}
u(\alpha,\beta) &= \sum_{(a,d)\in A_{\mathfrak{a}}\times A_{\mathfrak{d}}} \alpha(a)\beta(d)u(a,d) \\
&\leq \sum_{(a,d)\in A_{\mathfrak{a}}\times A_{\mathfrak{d}}} \alpha(a)\beta(d)u'(a,d) \\
&= u'(\mathsf{id}(\alpha),\beta)
\end{aligned}$$

Therefore, by Lemma 5.1, $\nu(\mathcal{U}) \leq \nu(\mathcal{U}')$. □

This lemma can be used to compare profiles having different types.

LEMMA 5.3. *Given an orchestration* E *and a stress model* $\mathcal{S} = \{(\delta(s),\delta_{\mathcal{A}}(s),\delta_{\mathcal{D}}(s)) \mid s \in \alpha(E)\}$ *then, for any* (a,d), *it is the case that*

$$\begin{aligned}
\mathit{delay}_{\mathcal{S},p}(a,d) &\leq \mathit{delay}_{\mathcal{S},s}(a,d) \leq \mathit{delay}_{\mathcal{S},f}(a,d) \\
\mathit{delay}_{\mathcal{S},p}(a,d) &\leq \mathit{delay}_{\mathcal{S},m}(a,d) \leq \mathit{delay}_{\mathcal{S},f}(a,d)
\end{aligned}$$

Proof. It suffices to prove that for any site s and (a,d):

$$\begin{aligned}
&\mathsf{delay}_{\mathcal{S},p}(s)[a,d] \\
&\quad \leq \mathsf{delay}_{\mathcal{S},s}(s)[a,d] \leq \mathsf{delay}_{\mathcal{S},f}(s)[a,d] \\
&\mathsf{delay}_{\mathcal{S},p}(s)[a,d] \\
&\quad \leq \mathsf{delay}_{\mathcal{S},m}(s)[a,d] \leq \mathsf{delay}_{\mathcal{S},f}(s)[a,d]
\end{aligned}$$

There are four cases to consider.
Case 1: $e \notin a \cup d$:
$\mathsf{delay}_t(e)[a,d] = \delta(e)$ for each $t \in \{s,p,f,m\}$.
Case 2: $e \in a \cap d$:
By definition:

$$\begin{aligned}
\mathsf{delay}_{\mathcal{S},p}(s)[a,d] &= \delta(s) - \delta_{\mathcal{A}}(s) - \delta_{\mathcal{D}}(s) \\
\mathsf{delay}_{\mathcal{S},s}(s)[a,d] &= \delta(s) - \delta_{\mathcal{A}}(s) + \delta_{\mathcal{D}}(s) \\
\mathsf{delay}_{\mathcal{S},m}(s)[a,d] &= \delta(s) + \delta_{\mathcal{A}}(s) - \delta_{\mathcal{D}}(s) \\
\mathsf{delay}_{\mathcal{S},f}(s)[a,d] &= \delta(s) + \delta_{\mathcal{A}}(s) + \delta_{\mathcal{D}}(s)
\end{aligned}$$

Consequently the required inequalities hold.
Case 3: $e \in a \setminus d$ and Case 4 $e \in d \setminus a$: These cases follow by similar case analysis. Lifting delays from sites to strategy profiles gives the required result. □

From Lemmas 5.3 and 5.2 we have:

THEOREM 5.1. *The family of delay profiles* $\mathcal{U}_t = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, \mathit{delay}_{\mathcal{S},t}\rangle$, *where* $t \in \{s,p,f,m\}$, *satisfy* $\nu(\mathcal{U}_p) \leq \nu(\mathcal{U}_s) \leq \nu(\mathcal{U}_f)$ *and* $\nu(\mathcal{U}_p) \leq \nu(\mathcal{U}_m) \leq \nu(\mathcal{U}_f)$.

Two different uncertainty profiles can be compared using *up* and *down* functions and an intermediate profile:

THEOREM 5.2. *If* $\mathcal{U} \xrightarrow{\mathfrak{a}:f} \mathcal{U}' \xleftarrow{\mathfrak{d}:f'} \mathcal{U}''$ *then* $\nu(\mathcal{U}) \leq \nu(\mathcal{U}'')$.

5.2. Useful up functions

Consider two uncertainty profiles over orchestration E and utility u: $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ and $\mathcal{U}' =$

$\langle E, \mathcal{A}', \mathcal{D}', b'_{\mathcal{A}}, b'_{\mathcal{D}}, u\rangle$. Assume that $\mathcal{A} \subseteq \mathcal{A}'$. By Definition 5.1 an (angelic) up function $f : \Delta_{\mathfrak{a}} \to \Delta'_{\mathfrak{a}}$ has the property that $u(\alpha, \beta) \leq u(f(\alpha), \beta)$. We develop two different instances of up functions (which have different bound conditions).

5.2.1. Joint map
If $\mathcal{A} \subseteq \mathcal{A}'$ and $b_{\mathcal{A}} \leq b'_{\mathcal{A}}$ then the sets of angelic actions for $\mathcal{U}$ and $\mathcal{U}'$ are, respectively: $A_{\mathfrak{a}} = \{a \subseteq \mathcal{A} \mid \#a = b_{\mathcal{A}}\}$, $A'_{\mathfrak{a}} = \{a' \subseteq \mathcal{A}' \mid \#a = b'_{\mathcal{A}}\}$.

Definition 5.2 (joint strategy). *Given $\alpha \in \Delta_{\mathfrak{a}}$ define $\mathsf{joint}(\alpha) \in \Delta'_{\mathfrak{a}}$, for any $a \in \Delta_{\mathfrak{a}}$, as:*

$$\mathsf{joint}(\alpha)(a') = \frac{1}{\binom{\#\mathcal{A}' - b_{\mathcal{A}}}{b'_{\mathcal{A}} - b_{\mathcal{A}}}} \sum_{\{a \in A_{\mathfrak{a}} \mid a \subseteq a'\}} \alpha(a)$$

The set a can be extended to a' by the addition of $b'_{\mathcal{A}} - b_{\mathcal{A}}$ services from $\mathcal{A}' \setminus a$. There are $\binom{\#\mathcal{A}' - b_{\mathcal{A}}}{b'_{\mathcal{A}} - b_{\mathcal{A}}}$ different ways to do this. The probability of a particular a' is found by adding the probabilities of all a, $a \subseteq a'$.

Example 13. Let $\mathcal{A} = \{1, 2, 3, 4\}$ and $\mathcal{A}' = \mathcal{A} \cup \{5, 6, 7\}$ with $b_{\mathcal{A}} = 2$ and $b'_{\mathcal{A}} = 3$. Then $\binom{\#\mathcal{A}' - b_{\mathcal{A}}}{b'_{\mathcal{A}} - b_{\mathcal{A}}} = 5$. The sets of available actions are:

$$A_{\mathfrak{a}} = \{\{1,2\},\{1,3\},\{1,4\},\{2,3\},\{2,4\},\{3,4\}\}$$
$$A'_{\mathfrak{a}} = \{\{1,2,3\},\{1,2,4\},\{1,2,5\},\ldots,\{5,6,7\}\}$$

Consider the set $\{1,2,3\}$ in $A'_{\mathfrak{a}}$. The set $\{a \in A_{\mathfrak{a}} \mid a \subseteq \{1,2,3\}\}$ is $\{\{1,2\},\{1,3\},\{2,3\}\}$ and so

$$\mathsf{joint}(\alpha)(\{1,2,3\}) = 1/5(\alpha(\{1,2\}) + \alpha(\{1,3\}) + \alpha(\{2,3\}))$$

Similarly

$$\mathsf{joint}(\alpha)(\{1,2,5\}) = \frac{1}{5}\alpha(\{1,2\})$$
$$\mathsf{joint}(\alpha)(\{5,6,7\}) = 0$$

joint is an angelic up function from $\mathcal{U}$ to $\mathcal{U}'$ when the daemonic choice sets are the same in $\mathcal{U}$ and $\mathcal{U}'$.

Lemma 5.4. *Let $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ and $\mathcal{U}' = \langle E, \mathcal{A}', \mathcal{D}, b'_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ be two uncertainty profiles in which $\mathcal{A} \subseteq \mathcal{A}'$ and $b_{\mathcal{A}} \leq b'_{\mathcal{A}}$. For any $\alpha \in \Delta_{\mathfrak{a}}$, $\mathsf{joint}(\alpha) \in \Delta'_{\mathfrak{a}}$ and $\beta \in \Delta_{\mathfrak{d}}$, $u(\alpha, \beta) \leq u(\mathsf{joint}(\alpha), \beta))$.*

Proof. Let $\#\mathcal{A}' = n$, $b'_{\mathcal{A}} = p'$, $b_{\mathcal{A}} = p$ with $p \leq p'$. Any $a \in A'_{\mathfrak{a}}$ contains p' services. Consider a $a \in A_{\mathfrak{a}}$ where $a \subseteq a'$. Strategy a' can be realised from a by adding $p' - p$ services from the set $\mathcal{A}' \setminus a$. As this set contains $n - p$ elements there are $\binom{n-p}{p'-p}$ ways to choose d. Since $u(a, d) \leq u(a', d)$ we have

$$u(a, d) \leq \frac{1}{\binom{n-p}{p'-p}} \sum_{\{a' \in A'_{\mathfrak{a}} \mid a \subseteq a'\}} u(a', d)$$

To prove that $\mathsf{joint}(\alpha) \in \Delta'_{\mathfrak{a}}$ consider the construction of $\mathsf{joint}(\alpha)(a')$. By definition $\mathsf{joint}(\alpha)(a') \geq 0$ The summation $\sum_{a \in A_{\mathfrak{a}}} \sum_{\{a' \in A'_{\mathfrak{a}} \mid a \subseteq a'\}}$ can be rewritten as $\sum_{a' \in A'_{\mathfrak{a}}} \sum_{\{a \in A_{\mathfrak{a}} \mid a \subseteq a'\}}$ (in both cases there is a sum over the set of pairs $\{(a', a) \in A'_{\mathfrak{a}} \times A_{\mathfrak{a}} \mid a \subseteq a'\}$). Thus,

$$\sum_{a' \in A'_{\mathfrak{a}}} \mathsf{joint}(\alpha)(a') = \sum_{a' \in A'_{\mathfrak{a}}} \Big(\frac{1}{\binom{n-p}{p'-p}} \sum_{\{a \in A_{\mathfrak{a}} \mid a \subseteq a'\}} \alpha(a) \Big)$$
$$= \frac{1}{\binom{n-p}{p'-p}} \sum_{\{(a', a) \in A'_{\mathfrak{a}} \times A_{\mathfrak{a}} \mid a \subseteq a'\}} \alpha(a)$$
$$= \frac{1}{\binom{n-p}{p'-p}} \sum_{a \in A_{\mathfrak{a}}} \sum_{\{a' \in A'_{\mathfrak{a}} \mid a \subseteq a'\}} \alpha(a)$$
$$= \sum_{a \in A_{\mathfrak{a}}} \alpha(a) \Big(\frac{1}{\binom{n-p}{p'-p}} \sum_{\{a' \in A'_{\mathfrak{a}} \mid a \subseteq a'\}} 1 \Big) = \sum_{a \in A_{\mathfrak{a}}} \alpha(a) = 1$$

The inequality between utilities is derived as follows:

$$u(\alpha, \beta) = \sum_{a \in A_{\mathfrak{a}}} \sum_{d \in A_{\mathfrak{d}}} \alpha(a) u(a, d) \beta(d)$$
$$\leq \sum_{a} \sum_{d} \alpha(a) \left(\frac{1}{\binom{n-p}{p'-p}} \sum_{\{a' \in A'_{\mathfrak{a}} \mid a \subseteq a'\}} u(a', d) \right) \beta(d)$$
$$= \sum_{a'} \sum_{d} \left(\frac{1}{\binom{n-p}{p'-p}} \sum_{\{a \in A'_{\mathfrak{a}} \mid a \subseteq a'\}} \alpha(a) \right) u(a', d) \beta(d)$$
$$= \sum_{a', d} \mathsf{joint}(\alpha)(a') u(a', d) \beta(d) = u(\mathsf{joint}(\alpha), \beta)$$

□

Therefore if $\mathcal{A} \subseteq \mathcal{A}'$ and $b_{\mathcal{A}} \leq b'_{\mathcal{A}}$ then joint is an up function from $\mathcal{U}$ to $\mathcal{U}'$,

$$\Gamma(\mathcal{U}) \xrightarrow{\mathfrak{a}:\mathsf{joint}} \Gamma(\mathcal{U}')$$

and so $\nu(\mathcal{U}) \leq \nu(\mathcal{U}')$.

5.2.2. Split map
Consider again the profiles $\mathcal{U}$ and $\mathcal{U}'$ where $\mathcal{A} \subseteq \mathcal{A}'$ but now where $b'_{\mathcal{A}} \leq b_{\mathcal{A}}$. We construct a mapping split [17] which is an up function:

Definition 5.3 (split the strategy). *Given profiles $\mathcal{U}$ and $\mathcal{U}'$ (from above) where $\mathcal{A} \subseteq \mathcal{A}'$ and $b'_{\mathcal{A}} \leq b_{\mathcal{A}}$ the mapping $\mathsf{split}(\alpha)$ is defined, for any $\alpha \in \Delta_{\mathfrak{a}}$, as:*

$$\mathsf{split}(\alpha)(a') = \frac{1}{\binom{b_{\mathcal{A}}}{b'_{\mathcal{A}}}} \sum_{\{a \in A_{\mathfrak{a}} \mid a' \subseteq a\}} \alpha(a)$$

Note that $\#\{a' \mid a' \subseteq a\} = \binom{b_{\mathcal{A}}}{b'_{\mathcal{A}}}$.

Example 14. Consider the situation where $\mathcal{A} = \{1, 2, 3, 4\}$ and $\mathcal{A}' = \mathcal{A} \cup \{5, 6, 7\}$ with $b'_{\mathcal{A}} = 2$ and $b_{\mathcal{A}} = 3$. The sets of angelic actions in $\mathcal{U}$ and $\mathcal{U}'$ are:

$$A_{\mathfrak{a}} = \{\{1,2,3\},\{1,2,4\},\{1,3,4\},\{2,3,4\}\}$$
$$A'_{\mathfrak{a}} = \{\{1,2\},\{1,3\},\{1,4\},\{1,5\},$$
$$\{1,6\},\{1,7\},\{2,3\},\ldots\}$$

Then $\mathsf{split}(\alpha)(\{1,2\}) = 1/3(\alpha(\{1,2,3\}) + \alpha(\{1,2,4\}))$ and $\mathsf{split}(\alpha)(\{1,5\}) = 0$. □

LEMMA 5.5. *Let $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ and $\mathcal{U}' = \langle E, \mathcal{A}', \mathcal{D}, b'_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ be two uncertainty profiles with $\mathcal{A} \subseteq \mathcal{A}'$, $b'_{\mathcal{A}} \leq b_{\mathcal{A}}$. For any $\alpha \in \Delta_{\mathfrak{a}}$, $\mathit{split}(\alpha) \in \Delta'_{\mathfrak{a}}$, $\alpha \in \Delta_{\mathfrak{a}}$ and $\beta \in \Delta_{\mathfrak{d}}$ it is the case that $u(\alpha, \beta) \leq u(\mathit{split}(\alpha), \beta)$.*

Proof. The first part of the proof is similar to the first part of the proof given in Lemma 5.4. Let $b_{\mathcal{A}} = p$ and $b'_{\mathcal{A}} = p'$. Given $a \in A_{\mathfrak{a}}$, there are $\#\{a' \mid a' \subseteq a\} = \binom{p}{p'}$ ways to choose a' such that $a' \subseteq a$. For any $a' \in A'_{\mathfrak{a}}$, $a' \subseteq a$, and $d \in A_{\mathfrak{d}}$ it holds (by hypothesis) that $u(a, d) \leq u(a', d)$. Applying this inequality to each a' where $a' \subseteq a$ we get

$$u_{\mathfrak{a}}(a, d) \leq \frac{1}{\binom{p}{p'}} \sum_{\{a' \mid a' \subseteq a\}} u(a', d).$$

To prove that $\mathsf{split}(\alpha)$ is a mixed strategy the term $\sum_{a' \in A'_{\mathfrak{a}}} \sum_{\{a \in A_{\mathfrak{a}} | a' \subseteq a\}}$ is rewritten as $\sum_{a \in A_{\mathfrak{a}}} \sum_{\{a' \in A'_{\mathfrak{a}} | a' \subseteq a\}}$. The inequality between utilities is derived as follows:

$$\begin{aligned}
u(\alpha, \beta) &= \sum_{a \in A_{\mathfrak{a}}} \sum_{d \in A_{\mathfrak{d}}} \alpha(a) u(a, d) \beta(d) \\
&\leq \sum_{a \in A_{\mathfrak{a}}} \sum_{d \in A_{\mathfrak{d}}} \frac{1}{\binom{p}{p'}} \sum_{\{a' \in A'_{\mathfrak{a}} \mid a' \subseteq a\}} \alpha(a) u(a', d) \beta(d) \\
&= \sum_{a' \in A'_{\mathfrak{a}}} \sum_{d \in A_{\mathfrak{d}}} \left(\frac{1}{\binom{p}{p'}} \sum_{\{a \in A_{\mathfrak{a}} \mid a' \subseteq a\}} \alpha(a) \right) u(a', d) \beta(d) \\
&= \sum_{a' \in A'_{\mathfrak{a}}} \sum_{d \in A_{\mathfrak{d}}} \mathsf{split}(\alpha)(a') u(a', d) \beta(d) \\
&= u(\mathsf{split}(\alpha), \beta)
\end{aligned}$$

□

Thus split is an up function from $\mathcal{U}$ to $\mathcal{U}'$

$$\Gamma(\mathcal{U}) \xrightarrow{\mathfrak{a}:\mathsf{split}} \Gamma(\mathcal{U}')$$

and so $\nu(\mathcal{U}') \leq \nu(\mathcal{U})$.

Similar results hold for uncertainty profiles over cost functions with the proviso that the directions of the inequalities are reversed to give $\nu(\mathcal{U}) \geq \nu(\mathcal{U}')$.

5.3. Down functions for $\mathfrak{d}$

Now we give examples of (demonic) down functions from $\mathcal{U}'$ to $\mathcal{U}$.

5.3.1. Joint map
Let $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ and $\mathcal{U}' = \langle E, \mathcal{A}, \mathcal{D}', b_{\mathcal{A}}, b'_{\mathcal{D}}, u\rangle$ such that $\mathcal{D}' \subseteq \mathcal{D}$, $b'_{\mathcal{D}} \leq b_{\mathcal{D}}$. The joint mapping given in Definition 5.2 can be adapted to give a daemon down mapping $\mathsf{joint}(\beta') : A_{\mathfrak{d}} \to \mathbb{R}$:

$$\mathsf{joint}(\beta')(d) = \frac{1}{\binom{\#\mathcal{D} - b'_{\mathcal{D}}}{b_{\mathcal{D}} - b'_{\mathcal{D}}}} \sum_{\{d' \in A'_{\mathfrak{d}} | d' \subseteq d\}} \beta'(d')$$

LEMMA 5.6. *Let $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ and $\mathcal{U}' = \langle E, \mathcal{A}, \mathcal{D}', b_{\mathcal{A}}, b'_{\mathcal{D}}, u\rangle$ be two uncertainty profiles such that $\mathcal{D}' \subseteq \mathcal{D}$ and $b'_{\mathcal{D}} \leq b_{\mathcal{D}}$. Then, for any $\beta' \in \Delta'_{\mathfrak{d}}$, $\mathit{joint}(\beta') \in \Delta_{\mathfrak{d}}$, $\alpha \in \Delta_{\mathfrak{a}}$ and $\beta' \in \Delta'_{\mathfrak{d}}$ it follows that $u(\alpha, \beta') \geq u(\alpha, \mathit{joint}(\beta'))$.*

Proof. By averaging over all such d, $d' \subseteq d$, we get

$$u(a, d') \geq \frac{1}{\binom{n-q'}{q-q'}} \sum_{\{d \in A_{\mathfrak{d}} | d' \subseteq d\}} u(a, d)$$

The inequality between utilities is derived as follows:

$$\begin{aligned}
u(\alpha, \beta') &= \sum_{a \in A_{\mathfrak{a}}} \sum_{d' \in A'_{\mathfrak{d}}} \alpha(a) u(a, d') \beta(d') \\
&\geq \sum_{a, d'} \alpha(a) \left(\frac{1}{\binom{n-q'}{q-q'}} \sum_{\{d \in A_{\mathfrak{d}} | d' \subseteq d\}} u(a, d) \right) \beta(d') \\
&= \sum_{a, d} \alpha(a) u(a, d) \mathsf{joint}(\beta')(d) = u(\alpha, \mathsf{joint}(\beta'))
\end{aligned}$$

□

Thus joint is a down function from $\mathcal{U}'$ to $\mathcal{U}$

$$\mathcal{U} \xleftarrow{\mathfrak{d}:\mathsf{joint}} \mathcal{U}'$$

and so $\nu(\mathcal{U}) \leq \nu(\mathcal{U}')$.

5.3.2. Split map
Consider $\mathcal{U}$ and $\mathcal{U}'$ again where $\mathcal{D}' \subseteq \mathcal{D}$ and $b_{\mathcal{D}} \leq b'_{\mathcal{D}}$. The definition of split is adapted to give a function $\mathsf{split} : \mathcal{A}'_{\mathfrak{d}} \to \mathbb{R}$:

$$\mathsf{split}(\beta')(d) = \frac{1}{\binom{b'_{\mathcal{D}}}{b_{\mathcal{D}}}} \sum_{\{d' \in A'_{\mathfrak{d}} | d \subseteq d'\}} \beta'(d')$$

Using similar techniques we obtain:

LEMMA 5.7. *Let $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ and $\mathcal{U}' = \langle E, \mathcal{A}, \mathcal{D}', b_{\mathcal{A}}, b'_{\mathcal{D}}, u\rangle$ be two uncertainty profiles such that $\mathcal{D}' \subseteq \mathcal{D}$, $b_{\mathcal{D}} \leq b'_{\mathcal{D}}$. Then, for any $\beta' \in \Delta'_{\mathfrak{d}}$, $\mathit{split}(\beta') \in \Delta_{\mathfrak{d}}$ and, for any $\alpha \in \Delta_{\mathfrak{a}}$ and $\beta' \in \Delta'_{\mathfrak{d}}$, $u(\alpha, \beta') \geq u(\alpha, \mathit{split}(\beta'))$.*

Again the mapping split is a down function from $\mathcal{U}'$ to $\mathcal{U}$ and

$$\mathcal{U} \xleftarrow{\mathfrak{d}:\mathsf{split}} \mathcal{U}'$$

and the assessments verify $\nu(\mathcal{U}) \leq \nu(\mathcal{U}')$.

6. MONOTONIC PROPERTIES OF RESILIENCE MEASURES

The goal of this section is to show how uncertainty profiles can be ordered according to risk. Partial orders over pairs of stressed sets (denoting the control options for the angel and daemon players) are derived for all stress types; these partial orders are monotonic with respect to the delay function. We keep the

same names for types and for the associated preorders. Subsequently preorders over a fixed orchestration are defined for each type of incremental delay profile. The profile orderings are shown to compatible with game assessment. Similarly, an ordering for robustness profiles is derived which is again monotonic with respect to game assessment.

Definition 6.1. *Consider the following partial orders on $\mathcal{P}(\alpha(E)) \times \mathcal{P}(\alpha(E))$. For $a, d, a', d' \subseteq \alpha(E)$:*

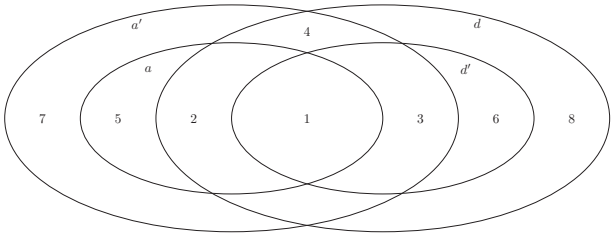
- *Standard:* $(a, d) \sqsubseteq_s (a', d')$ *iff* $a \subseteq a'$, $d' \subseteq d$.
- *Pangloss:* $(a, d) \sqsubseteq_p (a', d')$ *iff* $a \subseteq a'$, $d \subseteq d'$.
- *Faust:* $(a, d) \sqsubseteq_f (a', d')$ *iff* $a' \subseteq a$, $d' \subseteq d$.
- *Machiavelli:* $(a, d) \sqsubseteq_m (a', d')$ *iff* $a' \subseteq a$, $d \subseteq d'$.

The orderings are related by:

$$(a, d) \sqsubseteq_s (a', d') \text{ iff } (a', d') \sqsubseteq_m (a, d)$$
$$(a, d) \sqsubseteq_p (a', d') \text{ iff } (a', d') \sqsubseteq_f (a, d)$$

Lemma 6.1. *Let E be an* Orc *expression and $\mathcal{S}$ be an incremental stress model for E. For any type $t \in \{s, p, f, m\}$ and any $a, d, a', d' \subseteq \alpha(E)$ it follows that $(a, d) \sqsubseteq_t (a', d')$ implies $\mathsf{delay}_{\mathcal{S},t}(a, d) \geq \mathsf{delay}_{\mathcal{S},t}(a', d')$.*

Proof. Consider the standard stress type, $t = s$, and an arbitrary site e, $e \in \alpha_+(E)$. Assume that $(a, d) \sqsubseteq_s (a', d')$. Case analysis is used to show that, for any $e \in \alpha_+(E)$, $\mathsf{delay}_{\mathcal{S},s}(e)[a, d] \geq \mathsf{delay}_{\mathcal{S},s}(e)[a', d']$. In the standard partial order $(a, d) \sqsubseteq_s (a', d')$ implies that $a \subseteq a'$ and $d' \subseteq d$. There 8 separate cases to consider:



The condition $\mathsf{delay}_{\mathcal{S},s}(e)[a, d] \geq \mathsf{delay}_{\mathcal{S},s}(e)[a', d']$ is shown to hold in each case:
Case 1: $e \in a \cap d'$. Then $e \in a \cap d$ and $e \in a' \cap d'$; therefore:

$$\begin{aligned} \mathsf{delay}_{\mathcal{S},s}(e)[a, d] &= \delta(e) - \delta_{\mathcal{A}}(e) + \delta_{\mathcal{D}}(e) \\ &= \mathsf{delay}_{\mathcal{S},s}(e)[a', d'] \end{aligned}$$

Case 2: $e \in a$ and $e \in d \setminus d'$. The inequality holds because $a' \subseteq a$, $\delta_{\mathcal{D}} \geq 0$ and so:

$$\mathsf{delay}_{\mathcal{S},s}(e)[a, d] = \delta(e) - \delta_{\mathcal{A}}(e) + \delta_{\mathcal{D}}(e)$$
$$\mathsf{delay}_{\mathcal{S},s}(e)[a', d'] = \delta(e) - \delta_{\mathcal{A}}(e)$$

Case 3: $e \in a' \setminus a$ and $e \in d'$. As $e \in d$ we have:

$$\mathsf{delay}_{\mathcal{S},s}(e)[a, d] = \delta(e) + \delta_{\mathcal{D}}(e)$$
$$\mathsf{delay}_{\mathcal{S},s}(e)[a', d'] = \delta(e) - \delta_{\mathcal{A}}(e) + \delta_{\mathcal{D}}(e)$$

and so the required inequality holds.
Case 4: $e \in a' \setminus a$ and $e \in d \setminus d'$. Here:

$$\mathsf{delay}_{\mathcal{S},s}(e)[a, d] = \delta(e) + \delta_{\mathcal{D}}(e)$$
$$\mathsf{delay}_{\mathcal{S},s}(e)[a', d'] = \delta(e) - \delta_{\mathcal{A}}(e)$$

and the required inequality holds.
Case 5: $e \in a \setminus d$. In both cases the delay is

$$\mathsf{delay}_{\mathcal{S},s}(e)[a, d] = \delta(e) - \delta_{\mathcal{A}}(e) = \mathsf{delay}_{\mathcal{S},s}(e)[a', d']$$

Case 6: $e \in d' \setminus a'$ and $e \notin a'$. In both cases the delay is

$$\mathsf{delay}_{\mathcal{S},s}(e)[a, d] = \delta(e) + \delta_{\mathcal{D}}(e) = \mathsf{delay}_{\mathcal{S},s}(e)[a', d']$$

Case 7: $e \in a' \setminus a$ and $e \notin d$.

$$\mathsf{delay}_{s}(e)[a, d] = \delta(e)$$
$$\mathsf{delay}_{s}(e)[a', d'] = \delta(e) - \delta_{\mathcal{A}}(e)$$

Case 8: $e \in d \setminus d'$, $e \notin a'$.

$$\mathsf{delay}_{\mathcal{S},s}(e)[a, d] = \delta(e) + \delta_{\mathcal{D}}(e)$$
$$\mathsf{delay}_{\mathcal{S},s}(e)[a', d'] = \delta(e)$$

Similar proofs for $t \in \{p, f, m\}$ can be done using the matching partial orders. □

It is also possible to construct a partial order for pairs of stress sets which is monotonic with respect to the utility out. Such a partial order should contain the relations $u(\alpha_+(E), \alpha_+(E)) \leq u(a, d) \leq u(\emptyset, \emptyset)$. The *faust* partial order $\sqsubseteq_f$ has the required properties:

Lemma 6.2. *Let E be an* Orc *expression. For all stress sets $a, d, a', d' \subseteq \alpha_+(E)$ it holds that $(a, d) \sqsubseteq_f (a', d')$ implies $\mathsf{out}(a, d) \leq \mathsf{out}(a', d')$.*

Proof. When $(a, d) \sqsubseteq_f (a', d')$ the corresponding failure sets are $\mathcal{F} = a \cup d$ and $\mathcal{F}' = a' \cup d'$. Using the properties of the *faust* partial order it follows that $\mathcal{F}' \subseteq \mathcal{F}$. Therefore, by Lemma 3.3, it follows that $\mathsf{out}(\mathsf{fail}_{\mathcal{F}}(E)) \leq \mathsf{out}(\mathsf{fail}_{\mathcal{F}'}(E))$. □

Now we extend the notion of a partial order from stress sets to uncertainty profiles (over a fixed orchestration and resilience measure).

Definition 6.2. *The uncertainty profiles $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ and $\mathcal{U}' = \langle E, \mathcal{A}', \mathcal{D}', b'_{\mathcal{A}}, b'_{\mathcal{D}}, u\rangle$, can be partially ordered, for each stress type $t \in \{s, p, f, m\}$, as follows:*

- *$\mathcal{U} \sqsubseteq_s \mathcal{U}'$ if $\mathcal{A} \subseteq \mathcal{A}'$, $\mathcal{D}' \subseteq \mathcal{D}$, $b_{\mathcal{A}} \leq b'_{\mathcal{A}}$ and $b'_{\mathcal{D}} \leq b_{\mathcal{D}}$.*
- *$\mathcal{U} \sqsubseteq_p \mathcal{U}'$ if $\mathcal{A} \subseteq \mathcal{A}'$, $\mathcal{D}' \subseteq \mathcal{D}$, $b_{\mathcal{A}} \leq b'_{\mathcal{A}}$, $b_{\mathcal{A}} \leq b'_{\mathcal{A}}$ and $b_{\mathcal{D}} \leq b'_{\mathcal{D}}$.*
- *$\mathcal{U} \sqsubseteq_f \mathcal{U}'$ if $\mathcal{A} \subseteq \mathcal{A}'$, $\mathcal{D}' \subseteq \mathcal{D}$, $b'_{\mathcal{A}} \leq b_{\mathcal{A}}$ and $b'_{\mathcal{D}} \leq b_{\mathcal{D}}$.*
- *$\mathcal{U} \sqsubseteq_m \mathcal{U}'$ if $\mathcal{A} \subseteq \mathcal{A}'$, $\mathcal{D}' \subseteq \mathcal{D}$, $b'_{\mathcal{A}} \leq b_{\mathcal{A}}$ and $b_{\mathcal{D}} \leq b'_{\mathcal{D}}$.*

Informally when $\mathcal{U} \sqsubseteq \mathcal{U}'$ then $\mathcal{U}'$ is less risky than $\mathcal{U}$ – $\mathcal{U}'$ is said to be an (optimistic) *refinement* of $\mathcal{U}$.

THEOREM 6.1. *Let* $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, \mathsf{delay}_{\mathcal{S},t}\rangle$ *and* $\mathcal{U}' = \langle E, \mathcal{A}', \mathcal{D}', b'_{\mathcal{A}}, b'_{\mathcal{D}}, \mathsf{delay}_{\mathcal{S},t}\rangle$ *be uncertainty profiles over an incremental stress model* $\mathcal{S}$ *and a stress type* t, $t \in \{s, p, f, m\}$. *If* $\mathcal{U} \sqsubseteq_t \mathcal{U}'$ *then* $\nu(\mathcal{U}) \geq \nu(\mathcal{U}')$.

Proof. We use the proof technique from Theorem 5.2 and define an intermediate uncertainty profile $\mathcal{U}'' = \langle E, \mathcal{A}', \mathcal{D}, b'_{\mathcal{A}}, b_{\mathcal{D}}, \mathsf{delay}_{S,t}\rangle$. For each $t \in \{s, p, f, m\}$ define an up function f from $\mathcal{U}$ to $\mathcal{U}''$ and a down function f' from $\mathcal{U}'$ to $\mathcal{U}''$. The situation is schematised as:

$$\mathcal{U} \xrightarrow{\mathfrak{a}:f} \mathcal{U}'' \xleftarrow{\mathfrak{d}:f'} \mathcal{U}'$$

Since the delay function is a cost then Lemma 5.1 and Theorem 5.2 give $\nu(\mathcal{U}) \geq \nu(\mathcal{U}'') \geq \nu(\mathcal{U}')$. Consider the partial orders associated with different types of stress:
Case 1: standard ($t = s$). $\mathcal{U} \sqsubseteq_s \mathcal{U}'$ implies that $b_{\mathcal{A}} \leq b'_{\mathcal{A}}$ and $b'_{\mathcal{D}} \leq b_{\mathcal{D}}$. As $b_{\mathcal{A}} \leq b'_{\mathcal{A}}$ the function $\mathsf{joint} : \Delta_{\mathfrak{a}} \to \Delta'_{\mathfrak{a}}$ (Definition 5.2) is an angelic up function (Lemma 5.4). As $b'_{\mathcal{D}} \leq b_{\mathcal{D}}$ the function $\mathsf{joint} : \Delta'_{\mathfrak{d}} \to \Delta_{\mathfrak{d}}$ is a daemonic down function (Lemma 5.6). The situation is summarized by:

$$\mathcal{U} \xrightarrow{\mathfrak{a}:\mathsf{joint}} \mathcal{U}'' \xleftarrow{\mathfrak{d}:\mathsf{joint}} \mathcal{U}'.$$

Case 2: pangloss ($t = p$). In this case Lemmas 5.4 and 5.7 are used to generate the schema above.
Case 3: faust ($t = f$). In this case Lemmas 5.5 and 5.6 give the schema above.
Case 4: machiavelli ($t = m$). In this case Lemmas 5.5 and 5.7 give the schema above. □

In a similar way it can be shown that robustness profiles, ordered by $\sqsubseteq_f$ (faust), are monotonic with respect to assessment.

THEOREM 6.2. *Let* $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, \mathsf{out}\rangle$ *and* $\mathcal{U}' = \langle E, \mathcal{A}', \mathcal{D}', b'_{\mathcal{A}}, b'_{\mathcal{D}}, \mathsf{out}\rangle$ *be robustness profiles. If* $\mathcal{U} \sqsubseteq_f \mathcal{U}'$ *then* $\nu(\mathcal{U}) \leq \nu(\mathcal{U}')$.

Some example situations are used to illustrate Theorems 6.1, 6.2. The first example deals with robustness profiles (with the partial order $\sqsubseteq_f$).

EXAMPLE 15. For $k \geq 1$ consider the following orchestration

$$E = A \mid B \mid ((C_1 \mid C_2 \mid \cdots \mid C_k) \ll (D \mid F))$$

Orchestration E is analysed for robustness (using the utility out) using two different profiles. First consider the profile $\mathcal{U} = \langle E, \{A\}, \{B, D\}, 1, 1, \mathsf{out}\rangle$. In $\Gamma(\mathcal{U})$, the utility for $\mathsf{out}(\{A\}, \{B\})$ is

$$\mathsf{out}((0 \mid 0 \mid (C_1 \mid C_2 \mid \cdots \mid C_k) \ll (D \mid F))) = k$$

The game $\Gamma(\mathcal{U})$ is shown fully below:

$\mathfrak{a}$ \ $\mathfrak{d}$	$\{B\}$	$\{D\}$
$\{A\}$	k	$k+1$

$\Gamma(\mathcal{U})$

$\mathfrak{a}$ \ $\mathfrak{d}$	$\{B\}$	$\{D\}$
$\{A\}$	k	$k+1$
$\{F\}$	$k+1$	2

$\Gamma(\mathcal{U}')$

In $\Gamma(\mathcal{U})$ player $\mathfrak{d}$ tries to minimize out and so the profile $(\{A\}, \{B\})$ is a PNE with $\nu(\mathcal{U}) = k$.
Now consider the profile

$$\mathcal{U}' = \langle E, \{A, F\}, \{B, D\}, 1, 1, \mathsf{out}\rangle$$

Since $U \sqsubseteq_d \mathcal{U}'$ the inequality $\Gamma(\mathcal{U}) \leq \Gamma(\mathcal{U}')$ should hold. The game $\Gamma(\mathcal{U}')$ (shown above) has no PNE; Proposition 2.1 is used to compute the game value $\nu(\mathcal{U}')$. Consider a mixed Nash equilibrium (α, β) where $\alpha = (p, 1-p)$ and $\beta = (q, 1-q)$ with $0 < p, q < 1$. As $p > 0$ and $1 - p > 0$ it follows that $\mathsf{out}(\{A\}, \beta) = \mathsf{out}(\{F\}, \beta) = \mathsf{out}(\alpha, \beta)$. As $\mathsf{out}(\{A\}, \beta) = k + 1 - q$ and $\mathsf{out}(\{F\}, \beta) = qk + 2 - q$ we obtain $q = (k-1)/k$. Therefore $\mathsf{out}(\{A\}, \beta) = \mathsf{out}(\{F\}, \beta) = k + 1 - (k-1)/k$.

As $q > 0$ and $1 - q > 0$ it also follows that $\mathsf{out}(\alpha, \{B\}) = \mathsf{out}(\alpha, \{D\}) = \mathsf{out}(\alpha, \beta)$. As $\mathsf{out}(\alpha, \{B\}) = k + 1 - p$ and $\mathsf{out}(\alpha, \beta) = pk - p + 2$ we get $p = (k-1)/k$. Thus $\Gamma(\mathcal{U}') = \mathsf{out}(\alpha, \beta) = k + 1 - (k-1)/k$ $(> k)$ (as predicted).

The same stressed orchestration E can also be analysed for delay using uniform uncertainty profiles (we assume that $0 < \delta, \delta_{\mathcal{A}}, \delta_{\mathcal{D}} < \infty$). Let

$$\mathcal{U} = \langle E, \{A\}, \{B, D\}, 1, 1, \mathsf{delay}_f\rangle$$
$$\mathcal{U}' = \langle E, \{A, F\}, \{B, D\}, 1, 1, \mathsf{delay}_f\rangle$$

be the two profiles to be used for analysis. Assume that th profiles are ordered using $\sqsubseteq_f$. As $\mathcal{U} \sqsubseteq_f \mathcal{U}'$ the cost delay_f of the first profile should be less than or equal to that of the second profile. We have

$\mathfrak{a}$ \ $\mathfrak{d}$	$\{B\}$	$\{D\}$
$\{A\}$	2δ	2δ

$\Gamma(\mathcal{U})$

$\mathfrak{a}$ \ $\mathfrak{d}$	$\{B\}$	$\{D\}$
$\{A\}$	2δ	2δ
$\{F\}$	2δ	$2\delta + \min\{\delta_{\mathcal{A}}, \delta_{\mathcal{D}}\}$

$\Gamma(\mathcal{U}')$

Clearly $\nu(\mathcal{U}) = \nu(\mathcal{U}') = 2\delta$ (which is consistent with Theorem 6.1).

7. RELATED WORK

An overview of related work encompassing the areas of orchestration, fault tolerance, risk analysis, uncertainty and quality of service (QoS) is given below. We focus on those concerning Web services and/or Game theory.

The language Orc has been used to describe workflows [31] over internal and remote services - see [32]. In [26]

a timed operational semantics of Orc is given; there are similarities between call-and-delay environments (used here to estimate response delays) and the execution sequences and traces defined in [26]. The elements of a trace are pairs (t, m), where t is a response time and m is a publication result. A *handle* specifies the times at which particular values could be potentially returned (including the possibility of perpetual non-response, ω). In [26], the term $\Sigma(S, x)$ denotes the set of handles corresponding to a call to service S with argument x. In order to study delays *directly* we have adopted a time interval approach where every site has a maximal delay $\delta(s)$ and is executed in a call-and-delay environment.

Game theory has been used to study other problems in service-oriented computing. The problem of assigning infrastructures to applications is treated as a strategic game in [33]. Web services negotiation (1-to-1) is modelled using bargaining game theory in [34]. Cooperative game theory has also been used to model (i) cloud provider cooperation [35], (ii) community-based cooperation using autonomous Web services [36], and cooperative packet delivery with uncertainty (cooperative Bayesian games) [37].

Uncertainty in the Semantic Web has been modelled using Bayesian frameworks. A Bayesian framework for probabilistic service composition is given in [38] where plausible reasoning services are developed by using historical QoS data, recorded by brokers; uncertainty is represented using Bayesian networks. Bayesian games have also been used to analyse security risks. In [39] risk is modelled by an unknown probability distribution. A Bayesian game is defined under the assumption that there is some probability distribution describing players' beliefs about the relevant risk parameters. The goal of the paper is to determine equilibrium conditions for a corresponding Bayesian game and compare these with those of a game in which the players have full risk distribution information.

A number of proposals have been made for service evaluation and selection in situations where a high quality of service (QoS) is required. Often QoS parameters have multi-dimensionality and are volatile. In [40, 41] a QoS-driven web-service selection approach is proposed, based on assessments of linear combinations of scaled QoS parameters. A probabilistic approach to QoS in web systems is proposed in [42] using *soft contracts*. A risk-driven approach to the quality of web-services, based on payments, is developed in [6].

The problems of analysing failures and realising fault tolerance in distributed computing [43] is of great significance. Robustness under malicious attack is an active field of research – see for example [44, 45, 46, 47]; there are numerous results for a wide variety of problems involving malicious faults [48, 49, 50, 51]. Game has been applied to the analysis of fault tolerant systems – for example games can be constructed with both malicious and rational players (similar to the approach used in this paper), and the resulting situations analysed [52, 53, 7, 54, 55, 56]. This approach has resulted in the definition of a so-called *price of malice* [7]. Another game-theoretic approach to faulty systems is [57] where an attempt is made to improve system behaviour by imposing a centralized control over a subset of players.

Finally, it would be interesting to relate the unified model with two earlier generalisations of the angel-daemon framework. In [58] strategic behaviour is extended to allow multi-player games. In addition to the angel and the daemon, each site or service can strategically decide their behaviour; the system is modelled as a unit resource allocation game. This model gives rise to a more complex multi-player game in which only some Nash equilibria properties have been analysed. In addition in [59] strategic behaviour was extended to allow Bayesian games; a comparison between uncertainty profiles and probabilistic profiles was also made. It is an open question as to whether the new framework proposed here can be further generalized to study the robustness and performance of multi-player games and allow a comparison with Bayesian games to be made.

8. DISCUSSION

A distinction has been made between business *risk* (as viewed by economists) and the *uncertainty* associated with the outcome of business decisions [1, 8]. This same distinction can also be applied to the analysis of service-based computation; while some details of the expected performance of services may be given (e.g. QoS, SLAs) there remains a fundamental uncertainty about the response behaviour of web-services in stressed environments. In this paper we propose the use of *uncertainty profiles* and game theory as a means of analysing stressed service-based computations.

The performance of an orchestration of web-services fluctuates as parts of the web suffer stress. Stress can have several different forms, from simple over-demand to elasticity and even the possible beneficial effects of brokering. In this paper we have shown that uncertainty profiles can be used to model different types of stress that can arise in practical situations; the types standard, pangloss, faust and machiavelli are used to characterise different aspects of stress.

Superficially there would appear to be a contradiction between having on the one hand an uncertain environment and on the other a very concrete assessment of the same situation. There is however no contradiction: in the process of construcing an uncertainty profile an assessor imposes his own personal beliefs, perceptions and past experiences onto an uncertain environment[8]. In constructing a profile

[8]To quote from J. Maynard Keynes [60], "But in the actual exercise of reason we do not wait on certainty, or deem it irrational to depend on a doubtful argument"

an assessor moves from a more general uncertain environment to a more concrete (and more measurable) world.

One benefit of our approach is that different aspects of an uncertain situation can be analysed by employing different uncertainty profiles, with each profile capturing a different aspect of the uncertainty. Consider a situation in which an assessor constructs a number of different profiles of the form $\mathcal{U} = \langle E, \mathcal{A}, \mathcal{D}, b_{\mathcal{A}}, b_{\mathcal{D}}, u\rangle$ to model an uncertain situation. In the process of constructing these profile variations the assessor can:

- use *heuristic rules* to assign designated services to the sets $\mathcal{A}$ and $\mathcal{D}$ (see Example 8);
- set the values of $b_{\mathcal{A}}$ and $b_{\mathcal{D}}$ to reflect different stress levels (see Example 8);
- use different *game types* to model different stress scenarios (see Section 4.3); and
- utilise profile partial orders and *up* and *down* mappings (see Section 5) in order to navigate through different levels of uncertainty.

Uncertainty profiles provide a *general and powerful framework* which can be *tuned* to focus on particular aspects of an uncertain situation. The general approach adopted here is fundamentally different from other proposed applications of game theory for analysing service behaviour [33, 34, 35, 36, 37] where the decision process is intrinsic to the semantics of the model. In this paper the role of $\mathfrak{a}$-$\mathfrak{d}$ games is much more open in that different games can be devised to assess different aspects of uncertainty. As far as we are aware, our open approach to modelling uncertainty is new.

Angel daemon games have the potential to be tuned to model other characteristics of web-service stress. For example a cost-based scenario is developed below. Suppose that an orchestrator prefers a *cheap* service C with monetary cost m_C to an *expensive* service E with identical functionality and monetary cost m_E, $m_C < m_E$. In situations where the QoS of C is poor and C fails to respond after a time Δ the orchestrator additionally calls on service E to try to improve performance:

$$\begin{aligned} &\mathit{Try\_Cheap\_First} \\ &\quad = 1(x) < x < \big(C \mid (\mathit{Rtimer}(\Delta) \gg E)\big) \end{aligned}$$

Assume that *all services incur a charge at the point of call*.

Orchestration *Try_Cheap_First* can be analysed with respect to delay or monetary cost or by using a hybrid function. For example, given an orchestration O with a *delay* $\delta(O)$ and a *monetary cost* $m(O)$, a hybrid cost function could be defined, using Benjamin Franklin's maxim [61] that *time is money*:

$$\mathsf{time\_money}(O) = \delta(O) + m(O)$$

In order to provide a concrete cost example consider a uniform situation where the unstressed delays of C and E are both δ. Assume that $\Delta > \delta$ so that in favourable environments the service E is not called and the cost of using *Try_Cheap_First* is just m_C (and $\mathsf{time\_money}(\mathit{Try\_Cheap\_First}) = \delta + m_C$).
The behaviour of *Try_Cheap_First* is analysed using a uniform incremental stress model $\mathcal{S} = \langle \delta, \delta_{\mathcal{A}}, \delta_{\mathcal{D}}\rangle$ and a *standard* stress type. Assume that $\delta_{\mathcal{A}} = \frac{1}{4}\delta$, $\delta_{\mathcal{D}} = \frac{3}{4}\delta$. An analysis is carried out using the uncertainty profile $\mathcal{U}$ where

$$\mathcal{A} = \mathcal{D} = \{c, e\}, \quad \text{and} \quad b_{\mathcal{A}} = b_{\mathcal{D}} = 1$$

Consider the delay that arises for strategy $(\{C\}, \{C\})$:

$$\mathsf{delay}(C)[\{C\}, \{C\}] = \Big(1 - \frac{1}{4} + \frac{3}{4}\Big)\delta = \frac{3}{2}\delta$$

In a similar way $\mathsf{delay}(E)[\{C\}, \{C\}] = \Delta + \delta$. Thus

$$\begin{aligned} &\mathsf{delay}(\mathit{Try\_Cheap\_First})[\{C\}, \{C\}] \\ &\qquad = \min\Big\{\frac{3}{2}\delta, \Delta + \delta\Big\} = \frac{3}{2}\delta \end{aligned}$$

A monetary threshold function with a boolean argument B is used below to specify monetary costs:

$$\mathsf{threshold}(B) = \begin{cases} m_C & \text{if } B \\ m_C + m_E & \text{otherwise} \end{cases}$$

For example the monetary cost of the strategy $(\{C\}, \{C\})$ is:

$$m(\{C\}, \{C\}) = \mathsf{threshold}(\Delta > \frac{3}{2})$$

Other delay and monetary cases can be calculated in a similar way. The resultant delay game is:

$\mathfrak{a}$ \ $\mathfrak{d}$	$\{C\}$	$\{E\}$
$\{C\}$	$\frac{3}{2}\delta$	$\frac{3}{4}\delta$
$\{E\}$	$\frac{7}{4}\delta$	δ

This delay game has PNE $(\{C\}, \{C\})$. The corresponding monetary game is:

$\mathfrak{a}$ \ $\mathfrak{d}$	$\{C\}$	$\{E\}$
$\{C\}$	$\mathsf{threshold}(\Delta > \frac{3}{2})$	m_C
$\{E\}$	$\mathsf{threshold}(\Delta > \frac{7}{4})$	m_C

Here $(\{C\}, \{C\})$ is always a PNE while the other strategies maybe PNE, depending on the value of Δ. The hybrid *time is money* game is shown below for the situation where $\frac{3}{2}\delta < \Delta < \frac{7}{4}\delta$:

$\mathfrak{a}$ \ $\mathfrak{d}$	$\{C\}$	$\{E\}$
$\{C\}$	$\frac{3}{2}\delta + m_c$	$\frac{3}{4}\delta + m_c$
$\{E\}$	$\frac{7}{4}\delta + m_c + m_e$	$\delta + m_c$

Again $(\{C\},\{C\})$ is a PNE. This example illustrates that uncertainty profiles have the potential to be used to analyse cost scenarios; we hope to develop these ideas in future work.
Other possible future topics for investigation include:

- testing the $\mathfrak{a}$-$\mathfrak{d}$ approach using a wider range of practical examples;
- developing richer models where both the performance and the price of a service can vary with stress;
- and extending uncertainty profiles to deal with recursive orchestrations. One way to do this might be to develop the theory of repeated games (see 2.3.B in [62]) to the special case of $\mathfrak{a}$-$\mathfrak{d}$ games.

FUNDING

J. Gabarro and M. Serna were partially supported by "Ministerio de Ciencia e Innovación y el Fondo Europeo de Desarrollo Regional" project TIN-2007-66523 (FORMALISM) and "Generalitat de Catalunya" project 2009SGR 1137 (ALBCOM). Alan Stewart is partially supported by Engineering and Physical Sciences Research Council project EP/I03405X/1 (ECHO).

ACKNOWLEDGEMENTS

The authors are very grateful to the referees for their constructive comments on two earlier drafts of this paper. The authors are also grateful to A. García, M. Clint and P. Kilpatrick for helpful discussions about service-based computation and game theory.

REFERENCES

[1] Knight, F. (1921) *Risk, uncertainty and Profit*. Houghton Mifflin, Boston and New York. Free electronic access in: `http://www.econlib.org/library/Knight/knRUP.html`.

[2] Hull, J. (2009) *Risk Management and Finantial Institutions*, 2 edition. Pearson.

[3] Verdon, D. and McGraw, G. (2004) Risk analysis in software design. *IEEE Security & Privacy*, **2**, 79–84.

[4] Wheeler, E. (2011) *Security Risk Management*. Elsevier, Amsterdam.

[5] Kokash, N. (2007) Risk management for service-oriented systems. In Baresi, L., Fraternali, P., and Houben, G.-J. (eds.), *Web Engineering, 7th International Conference, ICWE 2007, Como, Italy, July 16-20*, LNCS, **4607**, pp. 563–568. Springer-Verlag, Berlin.

[6] Kokash, N. and D'Andrea, V. (2007) Evaluating quality of web services: A risk-driven approach. In Abramowicz, W. (ed.), *Business Information Systems, 10th International Conference, BIS 2007, Poznan, Poland, April 25-27*, LNCS, **4439**, pp. 180–194. Springer-Verlag, Berlin.

[7] Moscibroda, T., Schmid, S., and Wattenhofer, R. (2009) The price of malice: A game theoretic framework for malicious behaviour in distributed systems. *Internet Economics*, **6**, 125 – 155.

[8] Akerlof, G. and Schiller, R. (2009) *Animal Spirits*. Princeton University Press, Princeton and Oxford.

[9] Keynes, J. (1936) *The General Theory of Employment, Interest and Money*. Macmillan and Co. Re-edition, Palgrave-Macmillan for the Royal Economic Society, 2007.

[10] Luce, D. and Raiffa, H. (1989) *Games and Decisions*. Dover.

[11] W3c, web services glossary. `http://www.w3.org/TR/ws-gloss/`.

[12] Peltz, C. (2003) Web services orchestration and choreography. *IEEE Computer*, **36**, 46–52.

[13] Misra, J. and Cook, W. (2007) Computation orchestration: A basis for wide-area computing. *Software and Systems Modeling*, **6**, 83–110.

[14] Gabarro, J., Garcia, A., and Serna, M. (2013) Computational aspects of uncertainty profiles and angel-daemon games. , **?** *Theory of Computing Systems*, in press.

[15] Gabarro, J., Garcia, A., Clint, M., Kilpatrick, P., and Stewart, A. (2008) Bounded site failures: an approach to unreliable grid environments. In Danelutto, M., Fragopoulou, P., and Getov, V. (eds.), *Making Grids Work*, pp. 175–187. Springer-Verlag, Berlin.

[16] Gabarro, J., Garcia, A., Serna, M., Kilpatrick, P., and Stewart, A. (2008) Analysing orchestrations with risk profiles and angel-daemon games. In Gorlatch, S., Fragopoulou, P., and Priol, T. (eds.), *Grid Computing Achievements and Propects*, pp. 121–132. Springer-Verlag, Berlin.

[17] Gabarro, J., Serna, M., and Stewart, A. (2011) Web services and *incerta spiriti*: A game theoretic approach to uncertainty. In Liu, W. (ed.), *Symbolic and Quantitative Approaches to Reasoning with Uncertainty - 11th European Conference, ECSQARU 2011, Belfast, UK, June 29-July 1*, Berlin, LNCS, **6717**, pp. 651–662. Springer-Verlag.

[18] Kitchin, D., Quark, A., Cook, W., and Misra, J. (2009) The Orc programming language. In Lee, D., Lopes, A., and Poetzsch-Heffter, A. (eds.), *Formal Techniques for Distributed Systems, Joint 11th IFIP WG 6.1 International Conference FMOODS 2009 and 29th IFIP WG 6.1 International Conference FORTE 2009, Lisboa, Portugal, June 9-12*, LNCS, **5522**, pp. 1–25. Springer-Verlag, Berlin.

[19] Stewart, A., Gabarro, J., Clint, M., Harmer, T., Kilpatrick, P., and Perrott, R. (2006) Managing grid computations: An orc-based approach. In Guo, M., Yang, L. T., Martino, B. D., Zima, H. P., Dongarra, J., and Tang, F. (eds.), *Parallel and Distributed Processing and Applications, 4th International Symposium, ISPA 2006, Sorrento, Italy, December 4-6*, LNCS, **4330**, pp. 278–291. Springer-Verlag, Berlin.

[20] Bougé, L. (1993) Le mòdele de programmation à parallélisme de données: une perspective sémantique. *Techniques et Sciences Informatiques*, **12**, 541–562.

[21] Osborne, M. and Rubinstein, A. (1994) *A Course on Game Theory*. MIT Press, New York and Oxford.

[22] Nash, J. (1950) Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences of the United States of America*, **36**, 48–49.

[23] von Neumann, J. and Morgenstern, O. (1944) *Theory of Games and Economic Behavior*. Princeton University Press, Princeton and Oxford.

[24] Lynch, N. (1996) *Distributed algorithms*. Morgan Kaufmann, San Francisco, California.

[25] Stewart, A., Gabarro, J., and Keenan, A. (2012). Reasoning about orchestrations of web services using partial correctness. *Formal Aspects of Computing*, in press.

[26] Wehrman, I., Kitchin, D., Cook, W., and Misra, J. (2008) A timed semantics of Orc. *Theoretical Computer Science*, **402**, 234–248.

[27] Allen, J. (1983) Maintaining knowledge about temporal intervals. *Communications of the ACM*, **26**, 832–843.

[28] Römer, K. (2001) Time synchronization in ad hoc networks. *2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2001, October 4-5, Long Beach, CA, USA, Proceedings*, pp. 173–182. ACM.

[29] Garcia, A. (2012) The Complexity of Angel-Daemons and Game Isomorphism. PhD thesis Universitat Politècnica de Catalunya (Barcelona Tech).

[30] Stoneburner, G., Goguen, A., and Feringa, A. (2002) Risk management guide for information technology systems recommendations of the National Institute of Standards and Technology. Technical Report NIST Special publication 800-30.

[31] van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., and Barros, A. P. (2003) Workflow patterns. *Distributed and Parallel Databases*, **14**, 5–51.

[32] Cook, W. R., Patwardhan, S., and Misra, J. (2006) Workflow patterns in Orc. In Ciancarini, P. and Wiklicky, H. (eds.), *Coordination Models and Languages, 8th International Conference, COORDINATION 2006, Bologna, Italy, June 14-16, Proceedings*, LNCS, **4038**, pp. 82–96. Springer, Berlin.

[33] Ardagna, D., Panicucci, B., and Passacantando, M. (2011) A game theoretic formulation of the service provisioning problem in cloud systems. *Proceedings of the 20th international conference on World wide web*, New York, NY, USA WWW '11, pp. 177–186. ACM.

[34] Zheng, X., Martin, P., Powley, W., and Brohman, K. (2010) Applying bargaining game theory to web services negotiation. *2010 IEEE International Conference on Services Computing (SCC)*, Los Alamitos, CA, USA, july, pp. 218 –225. IEEE Computer Society.

[35] Niyato, D., Vasilakos, A. V., and Zhu, K. (2011) Resource and revenue sharing with coalition formation of cloud providers: Game theoretic approach. *11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGrid 2011, Newport Beach, CA, USA, May 23-26*, Los Alamitos, CA, USA, pp. 215–224. IEEE Computer Society.

[36] Liu, A., Li, Q., Huang, L., Ying, S., and Xiao, M. (2012). Coalitional game for community-based autonomous web services cooperation. *IEEE Transactions on Services Computing*, in press.

[37] Akkarajitsakul, K., Hossain, E., and Niyato, D. (2013) Coalition-based cooperative packet delivery under uncertainty: A Dynamic Bayesian Coalitional game. *IEEE Transaction on Mobile Computing*, **12**, 371–385.

[38] Yang, X., Cui, W., Liu, Z., and Ouyang, F. (2008) Study on uncertainty of geospatial semantic web services composition based on broker approach and Bayesian networks. *Proc. SPIE 7143, Geoinformatics 2008 and Joint Conference on GIS and Built Environment: Geo-Simulation and Virtual GIS Environments*. Electronic access.

[39] Johnson, B., Grossklags, J., Christin, N., and Chuang, J. (2010) Uncertainty in interdependent security games. In Alpcan, T., Buttyán, L., and Baras, J. S. (eds.), *Decision and Game Theory for Security - First International Conference, GameSec 2010, Berlin, Germany, November 22-23, 2010*, LNCS, **6442**, pp. 234–244. Springer, Berlin.

[40] Gao, A., Yang, D., Tang, S., and Zhang, M. (2006) QoS-driven web service composition with inter service conflicts. In Zhou, X., Li, J., Shen, H. T., Kitsuregawa, M., and Zhang, Y. (eds.), *Frontiers of WWW Research and Development - APWeb 2006, 8th Asia-Pacific Web Conference, Harbin, China, January 16-18*, LNCS, **3841**, pp. 121–132. Springer, Berlin.

[41] Wang, X., Vitvar, T., Kerrigan, M., and Toma, I. (2006) A QoS-aware selection model for semantic web services. In Dan, A. and Lamersdorf, W. (eds.), *Service-Oriented Computing - ICSOC 2006, 4th International Conference, Chicago, IL, USA, December 4-7*, LNCS, **4294**, pp. 390–401. Springer, Berlin.

[42] Rosario, S., Benveniste, A., and Jard, C. (2010) Flexible probabilistic QoS management of orchestrations. *International Journal of Web Services Research*, **7**, 21–42.

[43] Gärtner, F. C. (1999) Fundamentals of fault-tolerant distributed computing in asynchronous environments. *ACM Computing Surveys*, **31**, 1–26.

[44] Dolev, D. (1982) The Byzantine generals strike again. *Journal of Algorithms*, **3**, 14–30.

[45] Castro, M. and Liskov, B. (1999) Practical Byzantine fault tolerance. In Seltzer, M. I. and Leach, P. J. (eds.), *Proceedings of the Third USENIX Symposium on Operating Systems Design and Implementation (OSDI), New Orleans, Louisiana, USA, February 22-25*, pp. 173–186. USENIX Association.

[46] Lamport, L., Shostak, R. E., and Pease, M. C. (1982) The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, **4**, 382–401.

[47] Pease, M. C., Shostak, R. E., and Lamport, L. (1980) Reaching agreement in the presence of faults. *Journal of the ACM*, **27**, 228–234.

[48] Welch, J. L. and Lynch, N. A. (1988) A new fault-tolerance algorithm for clock synchronization. *Information and Computation*, **77**, 1–36.

[49] Malkhi, D. and Reiter, M. K. (1998) Byzantine quorum systems. *Distributed Computing*, **11**, 203–213.

[50] Koo, C.-Y. (2004) Broadcast in radio networks tolerating Byzantine adversarial behavior. In Chaudhuri, S. and Kutten, S. (eds.), *Proceedings of the Twenty-Third Annual ACM Symposium on Principles of Distributed Computing, PODC 2004, St. John's, Newfoundland, Canada, July 25-28*, pp. 275–282. ACM.

[51] Srikanth, T. K. and Toueg, S. (1987) Simulating authenticated broadcasts to derive simple fault-tolerant algorithms. *Distributed Computing*, **2**, 80–94.

[52] Karakostas, G. and Viglas, A. (2007) Equilibria for networks with malicious users. *Mathematical Programming*, **110**, 591–613.

[53] Roth, A. (2008) The price of malice in linear congestion games. In Papadimitriou, C. H. and Zhang, S. (eds.), *Internet and Network Economics, 4th International Workshop, WINE 2008, Shanghai, China, December 17-20*, LNCS, **5385**, pp. 118–125. Springer, Berlin.

[54] Babaioff, M., Kleinberg, R., and Papadimitriou, C. H. (2009) Congestion games with malicious players. *Games and Economic Behavior*, **67**, 22–35.

[55] Chakrabarty, D., Karande, C., and Sangwan, A. (2009) The effect of malice on the social optimum in linear load balancing games. *CoRR*, **abs/0910.2655**.

[56] Díaz, J., Mitsche, D., Rustagi, N., and Saia, J. (2009) On the power of mediators. In Leonardi, S. (ed.), *Internet and Network Economics, 5th International Workshop, WINE 2009, Rome, Italy, December 14-18*, LNCS, **5929**, pp. 455–462. Springer, Berlin.

[57] Roughgarden, T. (2001) Stackelberg scheduling strategies. *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, Heraklion, Crete, Greece*, pp. 104–113. ACM.

[58] Gabarro, J., Kilpatrick, P., Serna, M., and Stewart, A. (2010) Stressed web environments as strategic games: Risk profiles and weltanschauung. In Wirsing, M., Hofmann, M., and Rauschmayer, A. (eds.), *Trustworthly Global Computing - 5th International Symposium, TGC 2010, Munich, Germany, February 24-26, Revised Selected Papers*, LNCS, **6084**, pp. 189–204. Springer-Verlag, Berlin.

[59] Gabarro, J., Serna, M., and Stewart, A. (2012) Orchestrating unreliable services: strategic and probabilistic approaches to reliability. In Bruni, R. and Sassone, V. (eds.),

Trustworthy Global Computing - 6th International Symposium, TGC 2011, Aachen, Germany, June 9-10, Revised Selected Papers, LNCS, **7173**, pp. 197–211. Springer-Verlag, Berlin.

[60] Keynes, J. (1921) *A Treatise on Probability*. Macmillan and Co. Re-edition, Dover Publications, New-York, 2004.

[61] Franklin, B. (1748) *Advice to a Young Tradesmen, Written by an Old One*. Re-edition, *Franklin: The Autobiography and Other Writings on Politics, Economics and Virtue*, Cambridge University Press, 2004.

[62] Gibbons, R. (1992) *A Primer in Game Theory*. Prentice Hall, London.

[63] Zweigmedia. `http://www.zweigmedia.com/RealWorld/simplex.html`.

APPENDIX A.

A Call-and-delay example: Consider the orchestration *RareEmails*

$$\begin{aligned} &\mathit{RareEmails} \\ &\quad = \big((\mathit{Hello} \mid \mathit{Bye}) > x_1 > \mathit{TwoEmails}(x_1, x_2)\big) \\ &\qquad\qquad < x_2 < (\mathit{CNN} \mid \mathit{BBC}) \end{aligned}$$

which contains the sub-orchestration

$$\mathit{TwoEmails}(x_1, x_2) = (\mathit{EmailAlice}(x_1) \mid \mathit{EmailBob}(x_2))$$

Services *TwoEmails*, *EmailAlice* and *EmailBob* are abbreviated to TE, EA and EB, respectively. The bounds for x_1 are determined by the call-and-delay environment for sequential composition while the bounds for x_2 are given by the call-and-delay environment for pruning. The delay of *RareEmails* is calculated using the environment $\mathbb{C} = (\emptyset, [0, 0])$. Let

$$\begin{aligned} \mu_1 &= \min\{\delta(\mathit{Hello}), \delta(\mathit{Bye})\} \\ \delta_1 &= \max\{\delta(\mathit{Hello}), \delta(\mathit{Bye})\} \\ \mu_2 &= \min\{\delta(\mathit{CNN}), \delta(\mathit{BBC})\} \end{aligned}$$

The call-and-delay environment for TE is $\mathbb{C}_1 = (\mathbb{D}, \mathbb{T})$ with $\mathbb{D} = \{[x_1, \mu_1, \delta_1], [x_2, \mu_2, \mu_2]\}$ and $\mathbb{T} = [\mu_1, \delta_1]$. EA and EB are evaluated using the environments $\mathbb{C}_2 = (\mathbb{D}[\{x_1\}], \mathbb{T})$ and $\mathbb{C}_3 = (\mathbb{D}[\{x_2\}], \mathbb{T})$, respectively. Then

$$\begin{aligned} \mu_{\mathbb{C}_2}(EA(x_1)) &= \mu_1 + \delta(EA) \\ \delta_{\mathbb{C}_2}(EA(x_1)) &= \delta_1 + \delta(EA) \\ \mu_{\mathbb{C}_3}(EB(x_2)) &= \max\{\mu_1, \mu_2\} + \delta(EB) \\ \delta_{\mathbb{C}_3}(EB(x_2)) &= \max\{\delta_1, \mu_2\} + \delta(EB) \\ \mu_{\mathbb{C}_1}(TE(x_1, x_2)) &= \min\{\mu_{\mathbb{C}_2}(EA(x_1)), \mu_{\mathbb{C}_3}(EB(x_2))\} \\ \delta_{\mathbb{C}_1}(TE(x_1, x_2)) &= \max\{\delta_{\mathbb{C}_2}(EA(x_1)), \delta_{\mathbb{C}_3}(EB(x_2))\} \end{aligned}$$

Abbreviating $\mu_{\mathbb{C}}$ to μ and $\delta_{\mathbb{C}}$ to δ we have:

$$\begin{aligned} &\mu(\mathit{RareEmails}) = \mu(TE) \\ &\quad = \min\{\mu_1 + \delta(EA), \max\{\mu_1, \mu_2\} + \delta(EB)\} \\ &\delta(\mathit{RareEmails}) = \delta(TE) \\ &\quad = \max\{\delta_1 + \delta(EA), \max\{\delta_1, \mu_2\} + \delta(EB)\} \end{aligned}$$

APPENDIX B.

Approximation of the value of the game in Example 10: To compute the value of the game in example 10 we reduce this problem to an optimization problem [10]. As $\mathfrak{d}$ tries to increase the cost by randomization the constraint $c(\alpha, d) \leq \nu$ holds for $(\alpha, d) \in \Delta_{\mathfrak{a}} \times A_{\mathfrak{d}}$. Defining $x(a) = 1/\nu \geq 0$ we get $\sum_{a \in A_{\mathfrak{a}}} x(a)c(a, d) \leq 1$. It also holds $\sum_{a \in A_{\mathfrak{a}}} x(a) = 1/\nu$. $\mathfrak{a}$ is interested in minimizing ν (because it is a cost); consequently, $\mathfrak{a}$ wishes to maximize $\sum_{a \in A_{\mathfrak{a}}} x(a)$. Summarizing we have the following optimization problem

> *Optimization problem.* For all $a \in A_{\mathfrak{a}}$, find $x(a) \geq 0$ verifying
>
> $$\sum_{a \in A_{\mathfrak{a}}} x(a)c(a, d) \leq 1 \text{ for } d \in A_{\mathfrak{d}}$$
>
> such that $\sum_{a \in A_{\mathfrak{a}}} x(a)$ is maximised. The value of the game is $\nu = 1/(\sum_{a \in A_{\mathfrak{a}}} x(a))$.

Let ν be the value of the game. Defining

$$x = \alpha(\{W\})/\nu,\ y = \alpha(\{D\})/\nu\ z = \alpha(\{P\})/\nu$$

the constraint $\sum x(a)c(a, \{W, D\}) \leq 1$ is

$$5.2x + 50.2y + 50.1z \leq 1$$

Constraints corresponding to $\{W, P\}$ and $\{D, P\}$ are:

$$5.2x + 50.1y + 5.2z \leq 1$$
$$50.1x + 5.2y + 5.2z \leq 1$$

Maximizing $x + y + z$ with a Simplex method tool [63] we obtain $x + y + z = 0.03615$ and $\nu \approx 27, 66$.