An MCMC Procedure for calibrating a VoD Workload Model

Shubhabrata Roy, Thomas Begin, Paulo Gonçalves

Inria, ENS Lyon, UCB Lyon 1, CNRS, UMR 5668, 46 Allée d'Italie, 69364 Lyon Cedex 07, France
{shubhabrata.roy, thomas.begin, paulo.goncalves}@ens-lyon.fr

Résumé – Nous proposons une procédure d'estimation basée sur MCMC pour calibrer un modèle markovien permettant de reproduire la volatilité de la charge dans un système de Vidéo à la Demande. Nous évaluons numériquement les performances de cette procédure d'identification en termes de biais et de variance et nous comparons ses résultats avec ceux obtenus par une méthode ad-hoc d'identification. Les résultats montrent la supériorité de la méthode MCMC qui offre ainsi un outil de calibrage efficace et indispensable à l'exploitation du modèle.

Abstract – We propose an MCMC based estimation procedure thought for an existing Markovian model that can reproduce the workload volatility occurring in real-life VoD systems. We assess the accuracy of the proposed procedure in terms of bias and variance through several numerical experiments, and we compare its outcome with an existing ad-hoc method. Results show that the MCMC procedure clearly outperforms the other approach, and hence provides an efficient calibration tool which is of utmost importance for the usability of any model.

1 Introduction

In an era of data-intensive applications with pay-as-you-go execution in a cloud environment, efficient resource management has become prime interest for both Cloud Providers and Cloud Users. The choice of resource deployment can be dynamically tuned, thanks to the Software Defined Networks (SDN), which enables resource virtualization to meet the Service Level Agreement (SLA) of individual cloud applications. Of course, the presence of a time-varying demand hinders the design of such resource provisioning schemes. Adequate analytical models can then be helpful for testing and validating new management policies since they can offer a sound approach to capture and get a better insight into the underlying mechanisms of the applications.

In this paper we consider a Video on Demand (VoD) system as a relevant example of a time varying application. Since VoD has stringent streaming rate requirements, each VoD provider needs to reserve a sufficient amount of server outgoing bandwidth to sustain continuous media delivery.

Also, some videos may become popular very quickly (viral), and thus yield a flood of user requests (buzz) on the servers. Gonçalves *et al.* have proposed a Markovian model to reproduce such workload volatility [2]. Although the model has been shown to be theoretically able to correctly reproduce such dynamicity, it involves 7 parameters and assumes the knowledge of unobservable variables (an intermediary process that is impossible to observe in practice). These latter two points are major obstacles to calibrate the model according to workload traces, and need to be circumvented in an efficient identification procedure. To this end, we propose a Markov Chain Monte Carlo (MCMC) based procedure that provides estimation for each of the model parameters.

In Section 2, we provide a brief reminder regarding the Markovian model. Section 3 describes the MCMC procedure as applied in our context. Numerical results are discussed in Section 4.

2 Brief Model Description

Several works (e.g. [1], [2]) have developped epidemic inspired models to represent the way information spreads among the viewers (gossip-like phenomenon) in a VoD system. In [2], VoD users are categorized in three different classes as it is the case of the classical SIR model (Susceptible - Infectious - Recovered). We now provide a short reminder on their model. Class S refers to the people who are not currently watching a video. I pertains to the people who are currently watching a video. Note that I naturally represents the system workload, and may be expressed as the total bandwidth requested at that moment. When users complete watching their video, they move from I to R before ultimately leaving the system. While the evolution of I in time can be easily obtained through server traces, this does not hold for R (since corresponding users already left the VoD system). R is said to be an unobservable variable of the model.

Based on this categorisation, [2] derives a Markovian model whose state description is the current values of (i, r) where $i = I(t) = i$ and $r = R(t)$. Figure 1 (left part) depicts the possible future states and the corresponding transition rates. Arrivals in class I (corresponding to departures from S) occur following a quasi-Poisson process whose rate depends on the actual value of (i, r) so as to reproduce the gossip effect. The rate of the Poisson process is expressed as follows : $\beta(i + r) + l$, and thus involves two parameters. β is the rate of information dissemination carried out by any user in I or R. l corresponds to the rate of viewers that enter into I, spontaneously. All individuals stay in class I and then in R during times that are exponentially distributed, after which they leave the system, definitely. The sojourn time in class I with mean γ^{-1}, corresponds to the watching time of a video, whereas the mean duration μ^{-1} in class R is the period during which, in average, a former viewer keeps propagating the information. Finally, in order to take into account the presence of buzz, β can take two possible values. Its actual value is determined through a simple hidden Markov chain with two states (as shown on the right side of Figure 1). The first state corresponds to the "normal" state and $\beta = \beta_1$. The second state represents the buzz regime and β is then increased to $\beta_2 \gg \beta_1$. The transitions between the normal and the buzz states are parameterized by a_1 and a_2. For more details about this model, please refer to [2].

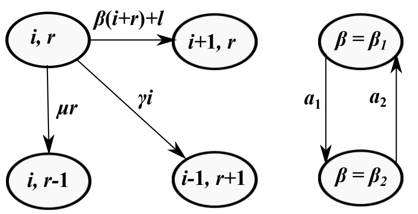

FIGURE 1 – Possible transitions issuing from the state (i, r) and the hidden Markov model for β.

3 Model Calibration using MCMC

An MCMC method is an algorithm used to generate samples from a *target* distribution of interest. MCMC methods are commonly used to estimate parameters of a given model when missing data needs to be inferred. Typically, the target distributions coincide with the posterior distributions of the parameters to be estimated. If I is the observable data and we want to estimate the model parameters $\mathbf{\Theta}$, the posterior distribution of $\mathbf{\Theta}$ derives from the Bayes rule :

$$p(\mathbf{\Theta} \mid I) \propto p(\mathbf{\Theta}) \cdot p(I \mid \mathbf{\Theta}). \tag{1}$$

Here, $p(\mathbf{\Theta})$ is the pdf of the prior distribution of $\mathbf{\Theta}$ and $p(I \mid \mathbf{\Theta})$ is the likelihood of $\mathbf{\Theta}$. As in general, $p(\mathbf{\Theta})$ is unknown, a standard practice [4][6] in MCMC algorithms is to choose adequate conjugate priors that multiplied with the likelihood yield computationally convenient posterior distributions.

There are several algorithms in MCMC family, the Metropolis and the Gibbs algorithms being certainly the most widely used in practice. A Metropolis algorithm is used when it is difficult to draw samples of $\mathbf{\Theta}$ from the posterior $p(\mathbf{\Theta} \mid I)$. It then replaces the posterior with an instrumental distribution $q(\mathbf{\Theta})$ from which it is easier to draw the samples. To closely match to the actual posterior distribution, at each step, the new sample is accepted (otherwise the previous draw is kept) with a probability given by the Metropolis ratio :

$$\alpha = min\left\{1, \frac{p(\mathbf{\Theta}^{(k)}|I)}{p(\mathbf{\Theta}^{(k-1)}|I)} \cdot \frac{q(\mathbf{\Theta}^{(k-1)})}{q(\mathbf{\Theta}^{(k)})}\right\}. \tag{2}$$

As for the Gibbs sampler, it is mainly used when the $\mathbf{\Theta}$ parameter of the model is multi-dimensional. The Gibbs sampler iteratively and sequentially generates each component of $\mathbf{\Theta}$ from its conditional posterior law, assuming the other components are fixed. Whenever these conditional posteriors are hard to sample, instrumental laws can be used, leading to the so-called Metropolis within Gibbs sampler. We now apply theses algorithms to fit our model to a workload trace.

In our case, $(I(t), t \in [0, T])$ is the only observation we can access to calibrate the proposed model. From it, we readily identify the instants $\{t_{a_n}\}_{n=1,\dots n_1}$ and $\{t_{p_n}\}_{n=1\dots n_2}$ at which individuals enter and leave the state I, respectively. As the exponential parameter γ of the watching time only depends on the sojourn time in I, it can then straightforwardly be estimated with a maximum likelihood procedures and reads [5]

$$\hat{\gamma}_{\text{MLE}} = n_2 \cdot (\textstyle\int_0^T I(t)\,\mathrm{d}t)^{-1}. \tag{3}$$

In contrast to γ though, all other parameters of the model rely on the unobserved time series $(R(t), t \in [0, T])$, or more precisely on the unknown departure instants from state R, that we note $\{t_{s_n}\}_{n=1,\dots,n_3}$. With this incomplete dataset, a maximum likelihood estimate of the form of (3) is precluded to estimate the propagation parameter μ. Instead we resort to a Metropolis-Hastings within Gibbs procedure to estimate simultaneously and iteratively $\widehat{\mathbf{t_s}}$ and $\hat{\mu}$, assuming at each iteration step k, known values for all the other parameters. This step is described below in Algorithm 1, which defines the outer loop of our complete estimation procedure. Now, regarding the current estimates of the remaining parameters $(\hat{\beta}_1, \hat{\beta}_2, \hat{l}, \hat{a}_1, \hat{a}_2)$ at step k, they also need to be updated according to the ongoing values of $\hat{\mu}^{(k)}$ and $\widehat{\mathbf{t_s}}^{(k)}$. To do so, we derive another MCMC estimator based on the posteriors expressed for each of these parameters taken individually (and conditioned to all others). This part is instantiated with Algorithm 2 as a inner loop of Algorithm 1.

3.1 Outer loop : estimation of $\hat{\mu}$ and $\widehat{\mathbf{t_s}}$

We start deriving the likelihood function of the sought parameters $\mathbf{\Theta} = (\mu,\ \beta_1,\ \beta_2,\ l,\ a_1,\ a_2)$:

$$\begin{aligned} p(\mathbf{t_a}, \mathbf{t_p}, \mathbf{t_s} \mid \mathbf{\Theta}) \quad \propto \quad & \textstyle\prod_{j=1}^{n_1} [((1-\pi)\beta_1 + \pi\beta_2)(I(t_{a_j}^-) + R(t_{a_j}^-)) + l] \\ & \times \textstyle\prod_{j=1}^{n_2} \hat{\gamma}_{\text{MLE}} I(t_{p_j}^-) \prod_{j=1}^{n_3} \mu R(t_{s_j}^-) \times \\ & e^{-\int_0^T [((1-\pi)\beta_1 + \pi\beta_2)(I(t)+R(t)) + l + \hat{\gamma}_{\text{MLE}} I(t) + \mu R(t)]\mathrm{d}t} \end{aligned} \tag{4}$$

where t^- stands for the time just before t and where we set for convenience $\pi = a_1/(a_1 + a_2)$, the probability that $\beta = \beta_2$. Eq. (4) plays a central role as it will be directly the target distribution of $\mathbf{t_s}$ and, combined with appropriate conjugate priors, it will yield the posterior of μ.

Given a current set of values for $\hat{\beta}_1$, $\hat{\beta}_2$, $\hat{l}$, $\hat{a}_1$, $\hat{a}_2$, we update the values of $\hat{\mu}$ and $\widehat{\mathbf{t_s}}$ as follows. First, we use a Gamma distribution parameterized by (λ_μ, ν_μ) as the prior distribution for μ. This latter multiplied by the likelihood of Eq. (4), leads to the posterior distribution of $\hat{\mu}$:

$$p(\hat{\mu}|\mathbf{t_a}, \mathbf{t_p}, \widehat{\mathbf{t_s}}) \propto \Gamma\Big(\lambda_\mu + n_3 - 1, \nu_\mu + \int_0^T \widehat{R(t)}\mathrm{d}t\Big), \tag{5}$$

from which we draw an updated value for $\hat{\mu}$. Note that the posterior distribution for $\hat{\mu}$ does not depend directly on $\hat{\beta_1}, \hat{\beta_2}, \hat{l}, \hat{a_1}, \hat{a_2}$. Second, we update $\widehat{\mathbf{t_s}}$ by modifying randomly one of its component. However, the acceptance of this new $\widehat{\mathbf{t_s}}$ is not systematic and depends on the outcome of the Metropolis ratio of Eq. (2). Then, considering the updated time series $\widehat{\mathbf{t_s}}$, we refresh the current values of $\hat{\beta_1}, \hat{\beta_2}, \hat{l}, \hat{a_1}, \hat{a_2}$ applying the inner loop described in Section 3.2. We iterate these three steps until $\hat{\mu}$ converges to a stable estimate. Algorithm 1 summarises the details of this outer loop.

Algorithm 1

Assume $n_3 \leftarrow n_2$
Set arbitrary initial guess $\hat{\mu}^{(0)} \leftarrow \hat{\gamma}_{\text{MLE}}$
Draw $\Delta\mathbf{t_s}^{(0)} = \{\Delta t_{s_1}{}^{(0)}, \Delta t_{s_2}{}^{(0)}, ...\}$ from exponential distribution with rate $\hat{\mu}^{(0)}$
$\widehat{\mathbf{t_s}}^{(0)} \leftarrow \{t_{p_1} + \Delta t_{s_1}{}^{(0)}, t_{p_2} + \Delta t_{s_2}{}^{(0)}, ...\}$
repeat for k = 1, 2, ...

- Construct $\widehat{R}^{(k)}$ from $\mathbf{t_p}$ and $\widehat{\mathbf{t_s}}^{(k-1)}$
- Estimate $\hat{\beta_1}^{(k)}$, $\hat{\beta_2}^{(k)}$, $\hat{l}^{(k)}$, $\hat{a_1}^{(k)}$ and $\hat{a_2}^{(k)}$ using **Algorithm 2**
- Draw $\hat{\mu}^{(k)}$ according to the posterior distribution described in Eq. (5)
- Generate a new candidate for $\widehat{\mathbf{t_s}}^{(k)}$ by modifying the c^{th} component of $\widehat{\mathbf{t_s}}^{(k-1)}$ with a new value uniformly sampled in $[0, T]$; $c \in [1, n_3]$
- Accept the latter candidate as the new current estimate of $\widehat{\mathbf{t_s}}$ according to the following Metropolis ratio : $\alpha = \min\{1, p(\widehat{\mathbf{t_s}}^{(k+1)}|\hat{\Theta}^{(k)})/p(\widehat{\mathbf{t_s}}^{(k)}|\hat{\Theta}^{(k)})\}$
- Otherwise, $\widehat{\mathbf{t_s}}^{(k)} \leftarrow \widehat{\mathbf{t_s}}^{(k-1)}$

until acceptable convergence

3.2 Inner loop : estimation of $\hat{\beta_1}$, $\hat{\beta_2}$, $\hat{l}$, $\hat{a_1}$, $\hat{a_2}$

We now describe how we can use a Gibbs sampler to estimate β_1, β_2, l, a_1 and a_2 based on the current knowledge of $\widehat{R}$ obtained at each step of the outer loop. To begin with, let us remind that the rate of new viewers grows linearly with $I(t)+R(t)$. More specifically, if we denote by $\mathbf{w}$ the times between two consecutive arrivals, we have :

$$\mathbb{E}(\mathbf{w}|I(t) = i, R(t) = r) = (\beta(i + r) + l)^{-1} \quad (6)$$

If β was constant, we could straightforwardly apply a linear least square regression on the empirical conditional mean of $\mathbf{w}$ to get $\hat{\beta}$ and $\hat{l}$. In our model though, the β parameter alternates between different values (β_1 and β_2) which leads us to design a MCMC procedure to estimate β_1, β_2, l, a_1 and a_2 .

Each $\{w_n\}_{n=1...n_1}$ corresponds to either β_1 or β_2, so we need to consider the w_n's individually through their distribution :

$$w_n \sim \begin{cases} p_1(\mathbf{w}) : \text{exponential law with rate } \beta_1(i + r) + l \\ p_2(\mathbf{w}) : \text{exponential law with rate } \beta_2(i + r) + l. \end{cases} \quad (7)$$

We introduce an intermediate variable $\mathbf{z} = \{z_1, \ldots, z_{n_1}\} \in (0, 1)^{n_1}$ whose elements indicate the current state of the system ($z_n = 0$ when $\beta = \beta_1$ and $z_n = 1$ when $\beta = \beta_2$). With this unobserved variable, we compute the likelihood $p(\mathbf{z},\mathbf{w}|\beta_1, \beta_2, l, \pi)$ (that we do not report here). Considering appropriate conjugate priors for the parameters, namely Normally distributed priors for β_1, β_2, l and Beta distributed prior for π, we get from Eq. (1), the posterior distributions of the parameters. We use a Gibbs sampler to sequentially update the current values of the parameters that we also use to update $\hat{\mathbf{z}}$ (applying a likelihood ratio test built on Eq. (7)). After convergence of the Gibbs sampling, $\hat{a}_1$ and $\hat{a}_2$ are directly obtained from $\hat{\mathbf{z}}$. Algorithm 2 summarises the details of this Gibbs procedure.

Algorithm 2

Set arbitrary initial values of $\hat{\beta_1}^{(0)}, \hat{\beta_2}^{(0)}, \hat{l}^{(0)}$ and $\hat{\pi}^{(0)}$
Consider current knowledge of $\widehat{R}$ from outer loop
repeat for m = 1, 2, ...

- Generate $\widehat{\mathbf{z}}^{(m)} = \{\hat{z_1}^{(m)}, ...\hat{z_{n_1}}^{(m)}\}$ where $\hat{z_n} = 1$ if $\frac{p_1(w_n)\pi}{p_1(w_n)(1-\pi)} \geq 1$ and $\hat{z_n} = 0$ otherwise
- Draw $\hat{\beta_1}^{(m)}, \hat{\beta_2}^{(m)}, \hat{l}^{(m)}$ and $\hat{\pi}^{(m)}$ according to their conditional posterior distributions

until acceptable convergence

4 Results and Discussion

We validate our estimation procedure, detailed in Section 3, against several synthetic traces corresponding to different sets of parameters. Due to space constraint we exhibit one set of experimental result for the parameter values, reported in Table 1. We conduct 20 independent realizations to produce 20 independent traces (each containing 2^{18} events) from these parameters.

TABLE 1 – Parameter values, used to generate the traces for validating the calibration procedure

$\widehat{\beta_1}$	$\widehat{\beta_2}$	$\widehat{\gamma}$	$\widehat{\mu}$	$\widehat{l}$	$\widehat{a_1}$	$\widehat{a_2}$
$4.7.10^{-4}$	$3.2.10^{-3}$	$1.1.10^{-2}$	$5.0.10^{-4}$	10^{-4}	10^{-7}	$6.6.10^{-2}$

The box plots in Fig. 2 indicate for each estimated parameter (centered and normalized by the corresponding actual value) the sample median (red line), the inter-quartile range (blue box height) along with the extreme samples (whiskers) obtained from traces of 2^{18} points. We also compare the performance our estimation procedure with the ad-hoc procedure described in [5].

Since we estimate γ from maximum likelihood method we do not report it here in our result. Nevertheless we would like to mention that its estimation is the most accurate, both in terms of the bias and variance. β_1 is also reasonably estimated with low bias and variance and performs reasonably better than the estimation of the ad-hoc procedure. Here, the extreme deviations of the estimated values are less than 5% of the true value. Compared to $\hat{\beta_1}$, $\hat{\beta_2}$ behaves less accurately (higher inter-quartile range being around 10%) and does not show significant

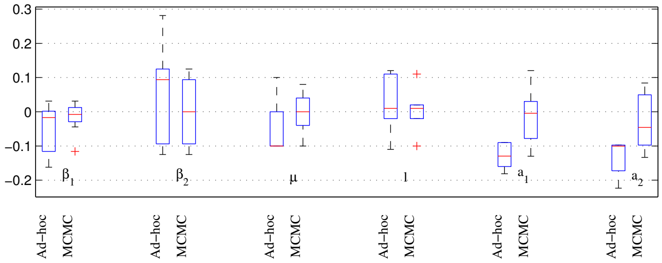

FIGURE 2 – Comparison of the relative precision of the MCMC and the ad-hoc estimators.

improvement over the ad-hoc method, owing to the fact that it is only based on buzz periods which represent a small fraction of the entire trace. However, the estimation is still very low biased. Estimation of μ also improves significantly, with a 5% inter-quartile range and negligible bias. Finally, estimation of the transition parameters a_1 and a_2 also outperform the ad-hoc procedure and shows less bias and 10% inter-quartile range. Our experiments with other sets of parameters also show similar accuracy.

In Fig. 3, we represent the relative estimation error for three key parameters of our model (i.e. β_1, μ and $\pi = a_1/(a_1+a_2)$) as a function of the number of iterations performed by Algorithm 1. In this example, the length of the trace is of 10^4 points. We observe that the convergence of the three parameters is attained after approximately 2.10^4 iterations. Note that we discard the first hundred values for π to ease the readability of the figure.

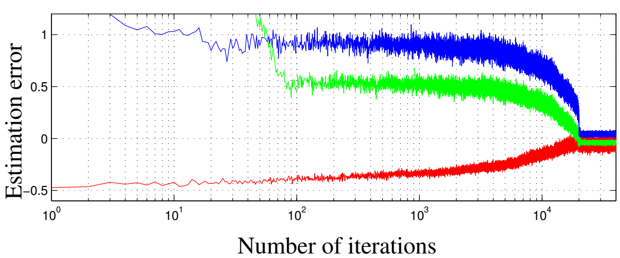

FIGURE 3 – Evolution of relative estimation error *versus* with the number of iterations in a *semilog* plot. The *blue* curve corresponds to $\hat{\mu}$, *green* curve corresponds to $\hat{\beta}_1$ and the *red* curve corresponds to $\hat{\pi}$.

Finally, we use a real-life VoD trace released by [3] to illustrate the ability of our MCMC procedure to correctly calibrate the VoD Markovian model. We apply our estimation procedure on this trace, and from the resulting parameterized model, we generate a synthetic trace. In Fig. 4, we plot both the original and the synthetic traces. Clearly, Fig. 4 shows that, given an adequate calibration, the model succeeds to reproduce different types of buzz with peaks and troughs at many scales similar to those of the real trace. Note that the means and the standard deviations of real and synthetic trace differ by less than 10%.

To better illustrate this adequacy, we consider the steady state distributions of the original trace and of the synthetic trace. Fig. 5(a) shows that the workload distribution resulting from these two traces are very alike, which suggests that the synthe-

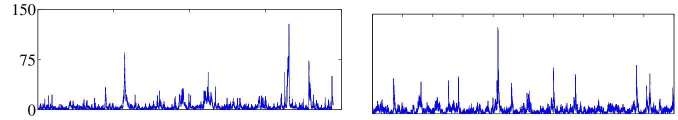

FIGURE 4 – Left plot shows real trace and the right plot shows the corresponding synthetic trace. Horizontal axes represent time (in hours) and vertical axes represent workload (number of active viewers).

tic trace successfully captures the inherent volatility of the original one. Furthermore, we compare the auto-correlation function exhibited by both traces in Fig. 5. Remind that the auto-correlation measures the statistical dependency $R_I(\tau) = \mathbb{E}\{I(t) \times I^*(t+\tau)\}$ between two samples of a (stationary) process I, distant of a time lag τ : the larger $R_I(\tau)$, the smoother the path of I at scale τ. Fig. 5(b) shows that the synthetic trace correctly reproduces the long-term correlative structure of the real trace.

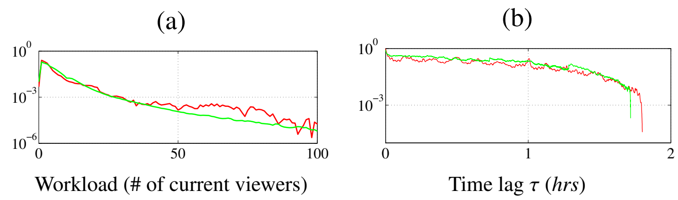

FIGURE 5 – Plot(a) shows the steady-state distributions of the real and synthetic traces while plot(b) shows their empirical autocorrelation functions. In both plots the red-curves correspond to the real trace and the green curves correspond to the synthetic trace.

5 Conclusion

We considered a Markovian model aimed at reproducing the volatility of workload observed in real-life systems such as VoD servers. The model involves several parameters whose estimation is hindered by unobservable data. We derived an original estimator based on a MCMC sampler. Its numerical evaluation on both synthetic and real traces shows accurate results, and demonstrates the procedure's ability at capturing the dynamics of buzz activity.

Références

[1] S. Abdullah and X. Wu. An epidemic model for news spreading on twitter. In *IEEE ICTAI*, 2011.

[2] P. Gonçalves, S. Roy, T. Begin, and P. Loiseau. Dynamic resource management in clouds : A probabilistic approach. *IEICE*, 2012.

[3] GRNET. Video traces from grnet, 2011. `http://vod.grnet.gr/`.

[4] P. D. O'Neil and G.O. Roberts. Bayesian inference for partially observed stochastic epidemics. *J.R Statist. Soc*, 1999.

[5] S. Roy, T. Begin, and P. Gonçalves. A complete framework for modelling and generating workload volatility of a VoD system. In *IEEE IWCMC*, 2013.

[6] J. Browne W. *MCMC estimation in MLwiN*. University of Bristol, 2012.