ECAI 2010
H. Coelho et al. (Eds.)
IOS Press, 2010
© 2010 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-60750-606-5-647

Predicting Responsiveness of BDI Agent

Huiliang Zhang and **Zhiqi Shen** and **Shell Ying Huang** and **Chunyan Miao**[1]

Abstract. A performance mark of a BDI agent is how fast it can react to and process incoming event sequences. To the best of our knowledge, few papers have been published about predicting an agent's average response time for an event sequence before the agent is applied in a real project. In this paper, we first introduce a simulation method. In simulation, a sequence of events with attributes, such as priorities and amounts of time needed to process the events, is input to the agent at designed insertion time. The events are processed by the agent according to the attributes. The statistics of processing time can be recorded. Then we make some theoretical analysis to estimate the average response time when an agent processes a sequence of events based on probability and queueing theory. Comparison experiments show that the results from analysis are quite matching with the results from simulation experiments. The analysis suggests a way to quickly estimate the performance of an agent if the attributes of the incoming event sequence are known in advance. The predicted average response time can help construct efficient BDI agents for various environments.

1 INTRODUCTION

The BDI (belief-desire-intention) model is well understood as an agent architecture to support goal oriented behaviour in intelligent agents. It provides a folk psychological way by simulating the human way of making decisions. The mental attitudes of belief, desire and intention represent the information, motivational, and deliberative states of the agent respectively (see [2] and [9]). The processing of an event inside a BDI agent is in a sequential way: first, *detect* action is executed, then *deliberate* action and finally *execute* action. This popular approach has been used in many agent systems (see [4], [6], [9] and [12]).

How to rationally allocate computation resources to the three actions (*detect*, *deliberate* and *execute*) in a BDI agent is a big problem when designing the agent. A defective mechanism for BDI agent will induce low responsiveness and waste of resources in the agent (see [8] and [11]). So it will be helpful if the responsiveness of the agent with a designed resource allocation mechanism could be known before the agent is applied in a real project. Currently few researches have been performed about predicting the responsiveness of an agent in different situations.

Simulation is a way to predict the performance of a BDI agent before it is implemented in real application (see [13]). A sequence of events with attributes, like priorities and amounts of time which are needed in the three processes (*detect*, *deliberate* and *execute*) of the agent, will be input to the agent at insertion time of the events. After receiving an event, the agent spends time on the event according to the attributes. The time from the moment when the event occurs to the moment when the processing of the event is completed can be recorded. Finally, the average response time for a sequence of events can be calculated. For different environments, event sequences with different attribute settings could be constructed. How the event attributes are designed depends on the specific environmnet. The responsiveness of the agent in different environments can be gotten through simulation experiments with the respective event sequences.

In this paper, we will first introduce how our simulation experiments are designed. Then we show a theoretical analysis of the processing time when a BDI agent is processing a sequence of events. The analysis is performed in a way like queueing network (see [3] and [7]). Based on some prior known attributes of the network and incoming events, queueing theory enables mathematical analysis of several performance measures including the average waiting time in the queues of a system, the number of tasks waiting or receiving service, and the probability of encountering the system in certain states. An application of queueing network to reduce waiting times in hospitals can be seen in [5]. Our analysis is performed based on the prior known average attributes of the incoming events, which represent the characteristics of the specific environment where the agent is working. Then the average waiting time for detection, deliberations (A deliberation is defined as the process to deliberate about a goal and generate an action plan), and intentions (An intention is defined as the process to execute the action plan) in the queues will be gotten through the analysis. This provides an alter-native and quick way to estimate the responsiveness of agents without performing simulation experiments.

The rest of this paper is organized as follows. We first introduce the queueing networks for BDI agents and simulation experiment design in the next section. Then theoretical analysis of average waiting time in BDI agents is shown in section 3. In section 4, comparisons of the results from analysis and simulation are demonstrated and explained. A way to design BDI agents with best performance is discussed in section 5. Finally a conclusion is given.

2 PROBLEM DEFINITION

In this section, a BDI agent is illustrated in the form of queueing networks, and then the design of event sequences for the networks is explained, finally, we introduce the performance mark for the agents, which will be analyzed in following section.

2.1 BDI agents configuration

There are 3 basic computational components in a BDI agent, the belief manager which *detects* the changes in the environment to generate new beliefs, the intention generator which *deliberates* on the beliefs to generate intentions, and the intention executor which schedules execution of the intentions. The processing inside the BDI

[1] Nanyang Technological University, Singapore, email: {PG04043187, ZQShen, ASSYHuang, ASCYMiao}@ntu.edu.sg

agents can be seen as an open queueing network. Events enter the agent and are processed by the agent as shown in Figure 1.

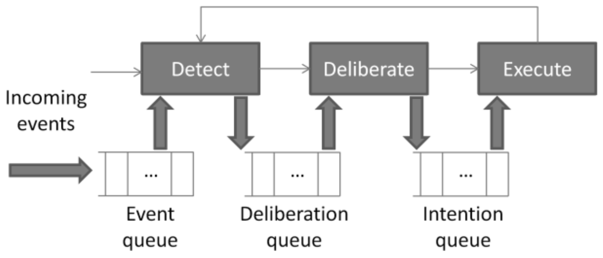

Figure 1. BDI agents in form of queueing network

In Figure 1, a BDI agent's computation is an iteration of three actions: *detect*, *deliberate* and *execute*. An incoming event is inserted in the *event queue* when it is the time for the event to occur. In the agent, when it is the turn for *detect* action, the events are fetched and organized as deliberations in the *deliberation queue*. When it is the turn for *deliberate* action, the agent will deliberate about the deliberations in the *deliberation queue* and save the output intentions in the *intention queue*. Finally, the intentions in the *intention queue* will be executed when it is the turn for *execute* action. The lengths of all the queues in Figure 1 are infinite.

The architecture shown in Figure 1 is a basic model behind many BDI agent systems (see [6], [9] and [12]). In real systems, some details may be different but the basic processing is the same.

2.2 Simulation experiment design

In the simulation, we assume that the smallest time unit is 1 and the system clock starts from 0. A sequence of incoming events is designed. The smallest interval between two events is 1 tick of a simulated clock. The attributes of an event include:

- *Priority* : importance of the event.
- *Insertion time* : the time when the event occurs.
- S_1 : detection time. Service time needed to detect and organize the event to generate a deliberation. We assume it equals to the smallest time unit of 1.
- S_2 : deliberation time. Service time needed to complete the deliberation.
- S_3 : execution time. Service time needed to execute the intention.

The *priority* value of an event is used to decide the importance of the intention which is generated according to the event. The *priority* value has no effect on *detect* or *deliberate* actions because the agent cannot decide the importance of an event before the deliberation about the event is accomplished. The generated intention is assigned with the *priority* value. An intention with bigger *priority* value should be executed earlier. In this paper, the priorities of events are uniformly distributed in [1, P_{max}], where P_{max} is the maximum priority value.

In the experiments, the operations of BDI agents are simulated. A BDI agent must allocate its computation resources to the three actions rationally. In the following experiments and analysis, we will adopt the "fixed time" allocation mechanism as used in Touringmachines (see [1]). Each action is assigned with fixed computation time. When the time for one action is used up, the next action will start. A little improvement is that a task (deliberation / intention) can be suspended and resumed in next cycle if the current remaining time is not sufficient for completing the task. In Touringmachines, the remaining time will be wasted if it is not big enough to complete a task.

Then, for a BDI agent, we define the following variables:

- t_1: time allocated for *detect* action.
- t_2: time allocated for *deliberate* action.
- t_3: time allocated for *execute* action.
- $t = t_1+t_2+t_3$: total time in a cycle.
- $R_2 = t_2/t$: the ratio of deliberate time in a cycle.
- $R_3 = t_3/t$: the ratio of execute time in a cycle.

During the running of a simulation experiment, a sequence of events is input to the BDI agent at the insertion times and processed according to the attributes of the events. The time to process each event will be recorded. Thus the average responsiveness of the agent to the event sequence can be gotten.

2.3 Performance mark

The agent performance is evaluated by the average response time (*ART*) for the events in a sequence. *Response time* for an event is defined as the period between the time when the event is inserted and the time when the response to the event is completed. A smaller *ART* indicates that the agent can process the events quicker. Our objective is to estimate the *ART* of the agent in an environment which is specified by a sequence of events.

Given the sequence of events, we can get the following values:

- *N* - number of events in the sequence;
- $\overline{interval}$ - average length of intervals between events;
- $\overline{S_1}$ - average detection time of the events;
- $\overline{S_2}$- average deliberation time of the events;
- $\overline{S_3}$- average execution time of the events.

Then the *ART* can be calculated by:

$$ART = \overline{WT_1} + \overline{S_1} + \overline{WT_2} + \overline{S_2} + \overline{WT_3} + \overline{S_3} \tag{1}$$

where the various symbols denote,
$\overline{WT_1}$: the average waiting time between the moment an event occurs and the moment it gets detected;
$\overline{WT_2}$: the average waiting time between the moment a deliberation is inserted to the *deliberation queue* and the moment it gets serviced;
$\overline{WT_3}$: the average waiting time between the moment an intention is inserted to the *intention queue* and the moment it gets executed.

After the values of the average waiting time are estimated, the *ART* can be calculated by combining them with the values of average service time. We will analyze the values of average waiting time in next section.

3 THEORETICAL ANALYSIS

Because the processing time is decided by the bottleneck of the three actions in an agent, the total time needed to process all the *N* events can be roughly estimated by

$$T = N * \max\left(t * \overline{S_1}/t_1,\ \ t * \overline{S_2}/t_2,\ \ t * \overline{S_3}/t_3,\ \ \overline{interval}\right).$$

In the following, we will analyze the values of average waiting time, $\overline{WT_1}$, $\overline{WT_2}$, and $\overline{WT_3}$.

3.1 Average Waiting Time for Detection

When an event occurs, the probability that the agent is not detecting is calculated by Pro(agent is not detecting) = (t_2+t_3)/t. If an event occurs when the agent is deliberating or executing, the event will not be detected until it is the turn for *detect* action. The event can occur at any time in the period of *deliberate* and *execute* actions. We assume that the chance is uniformly distributed. Thus, the waiting time before the event is detected can be estimated by $\frac{1}{t_2+t_3} * \sum_{i=1}^{t_2+t_3} (t_2 + t_3 - i + 1)$.

The average waiting time for detection of any event is:

$$\begin{aligned} &\overline{WT_1} = Pro(agent\ is\ not\ detecting) * waiting\ time \\ &= \tfrac{t_2+t_3}{t} * \tfrac{1}{t_2+t_3} * \textstyle\sum_{i=1}^{t_2+t_3} (t_2 + t_3 - i + 1) \\ &= \tfrac{(t_2+t_3)*(t_2+t_3+1)}{2*t} \end{aligned} \tag{2}$$

We can see that with a fixed time for a cycle, the agent puts more time on *deliberate* and *execute* actions, and the average waiting time before an event gets detected is longer.

3.2 Average Waiting Time for Deliberation

First, the average number of events which occur in a cycle can be estimated by N_c = $t/\overline{interval}$. The number of events that the agent can deliberate in a cycle is $t_2/\overline{S_2}$. According to whether the agent can complete deliberations about all the new events in a cycle in which the events occur, two cases are considered.

Case 1: $N_c \leq t_2/\overline{S_2}$

Theoretically, the agent can deliberate about all N_c events which occur in a cycle. Then the average waiting time can be estimated by:

$$\begin{aligned} &\overline{WT_2} = (\tfrac{1}{N_c} * \textstyle\sum_{i=1}^{N_c} ((i-1) * \overline{S_2} \\ &= \tfrac{\overline{S_2}}{2} * \left(\tfrac{t}{\overline{interval}} - 1 \right) \end{aligned} \tag{3}$$

It can be seen that with bigger $\overline{interval}$, smaller $\overline{S_2}$, $\overline{WT_2}$ will be smaller. If $t \leq \overline{interval}$, $\overline{WT_2}$will be 0.

Case 2: $N_c > t_2/\overline{S_2}$

In this case, the agent cannot deliberate about all *Nc* events which occur in a cycle. When an event numbered by *k* occurs, the number of the remaining deliberations in the *deliberation queue* for the previous (*k*-1) events can be estimated by:

$$N_{r2}(k) = k - 1 - k * \frac{\overline{interval} * R_2}{\overline{S_2}} \tag{4}$$

Because there are no priorities for the deliberations, the waiting time for the deliberation *k* is: $\mathrm{N_{r2}}(k) * \frac{\overline{S_2}}{R_2} - 1$.

So the average waiting time for deliberations is:

$$\begin{aligned} &\overline{WT_2} = \tfrac{1}{N} * \textstyle\sum_{i=\alpha}^{N} \left[\mathrm{N_{r2}}(i) * \tfrac{\overline{S_2}}{R_2} - 1 \right] \\ &= \tfrac{N-\alpha+1}{N} * \left[\tfrac{(N+\alpha)}{2} * \left(\tfrac{\overline{S_2}}{R_2} - \overline{interval} \right) - \tfrac{\overline{S_2}}{R_2} - 1 \right] \end{aligned} \tag{5}$$

Where, α is a minimum integer value satisfying $N_{r2}(\alpha) \geq 0$. α can be calculated by:

$$\alpha = \frac{\overline{S_2}}{\overline{S_2} - \overline{interval} * R_2} \tag{6}$$

3.3 Average Waiting Time for Execution

Two cases are discussed according to whether the agent can complete executing all new intentions in a cycle in which the intentions are generated.

Case 1: $t_2/\overline{S_2} \leq t_3/\overline{S_3}$

In this case, the agent can complete more intentions ($t_3/\overline{S_3}$) than deliberations ($t_2/\overline{S_2}$) in a cycle. So the average waiting time for execution is: $\frac{1}{m} * \sum_{i=1}^{m} (i-1) * \overline{S_3}$= $\frac{\overline{S_3}}{2} * (m-1)$, where $m = t_2/\overline{S_2}$. Then we get:

$$\overline{WT_3} = \frac{\overline{S_3}}{2} * \left(\frac{t_2}{\overline{\overline{S_2}}} - 1 \right) \tag{7}$$

If $t_2 \leq \overline{S_2}$, this means that the agent cannot generate a new intention in a single cycle. The agent can always complete the previous intention before the next one is generated. Thus we get $\overline{WT_3}$=0.

Case 2: $t_2/\overline{S_2} > t_3/\overline{S_3}$

The agent can generate more intentions than it can execute in a cycle. Two sub cases are considered.

Case 2.1: $N_c \leq t_2/\overline{S_2}$

In this case, the agent can complete deliberations about all new events in a cycle in which the events are detected. Two more sub cases are here.

Case 2.1.1: $N_c \leq t_3/\overline{S_3}$

In this sub case, the agent can complete execution of all new intentions generated in the cycle. Thus the $\overline{WT_3}$ is:

$$\begin{aligned} &\overline{WT_3} = \left(\tfrac{1}{N_c} * \textstyle\sum_{i=1}^{N_c} \left((i-1) * \overline{S_3} \right) \right) \\ &= \tfrac{\overline{S_3}}{2} * \left(\tfrac{t}{\overline{interval}} - 1 \right) \end{aligned} \tag{8}$$

If $t \leq \overline{interval}$, $\overline{WT_3}$will be 0.

Case 2.1.2: $N_c > t_3/\overline{S_3}$

In this sub case, the agent cannot execute all new intentions generated in the cycle.

For a new intention numbered by *k*, the number of previous remaining intentions in the *intention queue* can be estimated by:

$$N_{r3}(k) = k - 1 - k * \frac{\overline{interval} * R_3}{\overline{S_3}} \tag{9}$$

Because the priorities are uniformly distributed, the highest priority of the remaining intentions can be estimated as:

$$P_{high} = \frac{N_{r3}(k)}{k-1} * P_{\max} \tag{10}$$

An intention with highest priority is executed immediately. Thus, we can see that the probability that the intention *k* can be executed immediately is:

$$\begin{aligned} &Pro(P_k > P_{high}) = 1 - P_{high}/P_{max} \\ &= \tfrac{k}{k-1} * \tfrac{\overline{interval} * R_3}{\overline{S_3}} \end{aligned} \tag{11}$$

Where, *Pk* is priority of intention *k*.

The waiting time for the intention *k* can be shown as:

$$WT_3(k) = \Pr o\, (P_k \leq P_{high}) * blocktime(k) \tag{12}$$

Where, the blocktime(*k*) is the time needed to execute the intentions with a higher *priority* value than intention *k* or intentions with a same *priority* value but generated earlier than the intention *k*. A blocked intention is expected to get executed after all the events in the sequence are input.

The number of remaining intentions when all the N intentions are generated is:

$$N_{r3}(N) = N - 1 - N * \frac{\overline{interval} * R_3}{\overline{S_3}} \tag{13}$$

The average waiting time of the $N_{r3}(N)$ intentions after all events occur is calculated by:

$$\frac{1}{\mathrm{N_{r3}}(N)} * \frac{1}{R_3} * \sum_{i=1}^{\mathrm{N_{r3}}(N)} (i-1) * \overline{S_3} = \frac{\mathrm{N_{r3}}(N)-1}{2} * \frac{\overline{S_3}}{R_3} \tag{14}$$

To calculate the blocktime(k), we also need to estimate the time between the generation of intention k and when intentions for all events are generated. Then blocktime(k) is calculated as:

$$\begin{aligned} &blocktime(k) = (N-k)\,\overline{interval} \\ &-\overline{WT_1} - \overline{S_1} - \overline{WT_2} - \overline{S_2} + \frac{N_{r3}(N)-1}{2}\frac{\overline{S_3}}{R_3} \end{aligned} \tag{15}$$

With equations (12, 11, 15), we can calculate the average waiting time by:

$$\begin{aligned} &\overline{WT_3} = \frac{1}{N} * \sum_{k=\alpha}^{N} WT_3(k) \\ &= \frac{1}{N} * \sum_{k=\alpha}^{N} \left(1 - \frac{k}{k-1} * \frac{\overline{interval} * R_3}{\overline{S_3}}\right) * blocktime(k) \end{aligned} \tag{16}$$

Where, α is a minimum integer value satisfying $N_{r3}(\alpha) \geq 0$. α can be calculated by:

$$\alpha = \frac{\overline{S_3}}{\overline{S_3} - \overline{interval} * R_3} \tag{17}$$

Case 2.2: $N_c > t_2/\overline{S_2}$

In this case, in a cycle, the agent cannot complete deliberations about all events in a cycle in which the events occur. Similar to the previous case, we get:

$$N_{r3}(k) = k * \left(1 - \frac{\overline{S_2} R_3}{R_2 \overline{S_3}}\right) - 1 \tag{18}$$

$$Pro(P_k > P_{high}) = 1 - P_{high}/P_{max} = \frac{k}{k-1} * \frac{\overline{S_2} R_3}{R_2 \overline{S_3}} \tag{19}$$

We can see that when $t_2/\overline{S_2}$is very close to $t_3/\overline{S_3}$, the probability is close to 1, which means that the new intention can be executed soon.

The blocktime(k) is calculated by:

$$blocktime(k) = (N-k) * \frac{\overline{S_2}}{R_2} + * \frac{N_{r3}(N)-1}{2} * \frac{\overline{S_3}}{R_3} \tag{20}$$

Similar to equation (16), we can calculate the average waiting time by:

$$\overline{WT_3} = \frac{1}{N} * \sum_{k=\alpha}^{N} \left(1 - \frac{k}{k-1} * \frac{\overline{S_2} R_3}{R_2 \overline{S_3}}\right) * blocktime(k) \tag{21}$$

Where, α is a minimum integer value satisfying $N_{r3}(\alpha) \geq 0$. α can be calculated by:

$$\alpha = \frac{R_2 * \overline{S_3}}{R_2 * \overline{S_3} - \overline{S_2} * R_3} \tag{22}$$

4 EXPERIMENT

In order to see how closely these equations are able to approximate the real performance of the agents, we make the following comparison experiments.

The time allocation mechanism for the BDI agent is t_1=1, t_2=2, and t_3=4. Thus according to equation (2), the estimated $\overline{WT_1}$ is 3. A total number of *N=1000* events will be created. The detection time of all events is 1 time unit. The *priority* value is selected randomly from 1 to 5.

In the following we will design various experiments with different settings for events deliberation time, execution time, and intervals. The statistics gotten through simulation experiments will be compared to the values estimated by using the equations in previous analysis.

Experiment 1: Events with same deliberation time, execution time, and fixed intervals

In this case, all the events are assigned with the same deliberation time (2 ticks) and execution time (4 ticks). The intervals between events are a fixed value. We designed five sequences of events with different intervals (1, 2, 4, 7, and 10 ticks). The sequences are input to the agents separately. The average times gotten from analysis and simulation are shown in Table 1. From the table, we can see that in this case, the results from analysis are very close to the real *ART* by simulation. For example, the ratio of difference between the *ART* results to the value gotten from the simulation experiment is only |3006-3004|/3004=0.0666%. The only noticeable difference is shown in italic font. When the fixed interval is 7, for the agent, all events will be inserted exactly before a new cycle is started and detected in next time unit. Thus the events can be detected in next time units and average waiting time for detection is 1. The *response time* for every event is 8.

Table 1. Average time

I*	$\overline{S_2}$	$\overline{S_3}$	Analysis			Simulation			
			$\overline{WT_2}$	$\overline{WT_3}$	*ART*	$\overline{WT_1}$	$\overline{WT_2}$	$\overline{WT_3}$	*ART*
1	2	4	2996	0	3006	3.002	2994	0	3004
2	2	4	2495.5	0	2505.5	3.002	2500.5	0	2510.5
4	2	4	1494.5	0	1504.5	3.002	1499.5	0	1509.5
7	2	4	0	0	10	*1*	0	0	*8*
10	2	4	0	0	10	3.002	0	0	10.002

* I: average interval between events.

Experiment 2: Events with random deliberation time and execution time, fixed intervals

In this case, the deliberation time of the events is chosen randomly in [1, 3]. The execution time of the intentions is chosen randomly in [1, 7]. The intervals are fixed. We still design 5 sequences like in experiment 1. The results for the new sequences are shown in Table 2. It is noticed that the results from analysis are still very close to the results from simulation when the agent is processing congested event sequences.

Table 2. Average time

I	$\overline{S_2}$	$\overline{S_3}$	Analysis			Simulation			
			$\overline{WT_2}$	$\overline{WT_3}$	*ART*	$\overline{WT_1}$	$\overline{WT_2}$	$\overline{WT_3}$	*ART*
1	1.99	3.95	2978.5	0.01	2988.5	3.002	3002.9	46.44	3059.3
2	2.04	3.9	2572.4	0	2582.4	3.002	2612.8	21.97	2644.7
4	1.94	3.9	1391.4	6.26	1407.5	3.002	1408.7	57.69	1476.2
7	1.99	4.06	0	34.7	44.75	*1*	*23*	*100.9*	*132*
10	2.02	4.03	0	0	10.04	3.002	*2.36*	*4.44*	*16.84*

When the fixed interval is 7, the agent is expected to deal with the new event immediately after it is detected since the average time needed to process an event ($\overline{S_1}$+$\overline{S_2}$+$\overline{S_3}$= 7.05) is around 7. However, the deliberation about an event with deliberation time of 3 must be performed in 2 cycles considering t_2 = 2. Due to this, many deliberations about events cannot be processed in the same cycles as the events occur. This induces a bigger $\overline{WT_2}$ in simulation than the expected $\overline{WT_2}$. For the similar reason, $\overline{WT_3}$ from simulation is also larger than the value from analysis.

Experiment 3: Events with same deliberation time and execution time, random intervals

In this case, all events have same deliberation time (2 ticks) and execution time (4 ticks). The intervals between events will be chosen randomly in [1, *max interval*]. We design 4 sequences with *max interval* as 3, 7, 13, 19 respectively. Then the results are shown in Table 3. It is shown in the table that the differences between the results from analysis and simulation are smaller than in last case. The randomness of intervals is the reason for the differences. For example, when average interval is 7.32, the agent is expected to complete each event in each cycle in analysis. However, in simulation, half of the intervals are smaller than 7 and the events cannot be processed in time. This produces a bigger $\overline{WT_2}$.

Table 3. Average time

I	$\overline{S_2}$	$\overline{S_3}$	**Analysis**			**Simulation**			
			$\overline{WT_2}$	$\overline{WT_3}$	ART	$\overline{WT_1}$	$\overline{WT_2}$	$\overline{WT_3}$	ART
2.016	2	4	2487.5	0	2497.5	3.007	2484.7	0	2494.7
4.06	2	4	1464.5	0	1474.5	2.967	1458	0	1467.9
7.32	2	4	0	0	10	2.98	*14.83*	0	*24.8*
10.32	2	4	0	0	10	2.918	*2.338*	0	*12.26*

Experiment 4: Events with random deliberation time, execution time, and intervals

In this case, all the variables will be chosen randomly as in previous experiments. The results are shown in Table 4. The differences are larger than in the last 2 cases due to a bigger extent of randomness.

Table 4. Average time

I	$\overline{S_2}$	$\overline{S_3}$	**Analysis**			**Simulation**			
			$\overline{WT_2}$	$\overline{WT_3}$	ART	$\overline{WT_1}$	$\overline{WT_2}$	$\overline{WT_3}$	ART
2.037	1.98	4.06	2438.5	71.38	2519.9	3.004	2473.6	99.2	2582.9
4.036	2.03	3.98	1523.7	0	1533.7	2.923	1536.6	58.19	1604.7
6.988	2.04	3.94	76.08	0	86.07	3.046	*107.03*	*38.25*	*155.3*
9.978	2.003	3.96	0	0	9.958	2.958	*4.925*	*5.291*	*20.132*

Experiment 5: Events with exponential distributed intervals and random deliberation and execution time

In this case, the intervals between events are exponential probability distributed. The underlying physical process is memoryless. The length of the time interval from the current time to the occurrence of the next event does not depend upon the time of occurrence of the last event (see [10]). This is a frequently used method to simulate the independent events in queueing network research. The cumulative distribution function is shown as:

$$cdf(t) = 1 - e^{-\lambda * t} \tag{23}$$

where, 1/λ is the mean; *t* is the time units.

cdf(*t*) shows the probability that the inter-arrival time between 2 events is less than *t*. So given a number of total events, *sum*, to be used in our experiments, we can decide the number of the intervals with length of *t* time units by:

$$G(t) = (cdf(t) - cdf(t-1)) * sum \tag{24}$$

If the *sum* of the events (N) is provided, the numbers of the intervals with different lengths can be decided. Then the intervals are selected randomly for the events. The current time plus the interval length is the arrival time of the next event.

When we set N as 1000 and 1/λ as 7, the numbers of intervals with different lengths are shown in Figure 2. From the figure, we can see that the number of intervals with same length decreases when the length of the intervals increases. Many intervals (over 60%) are smaller than the average interval, 7.08. This means that congestion between events frequently happens. However in analysis, all events are expected to occur in a fixed interval, 7.08. Thus the results from analysis may be too optimistic compared to the results from simulation.

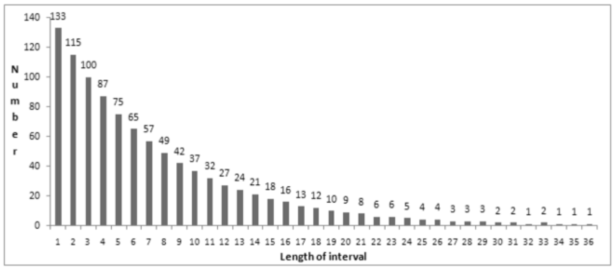

Figure 2. Number of intervals with different lengths

The deliberation time of the events are chosen randomly in [1, 3]. The execution time is chosen randomly in [1, 7]. The results are shown in Table 5. In this case, many events may come in crowd, which raises the uncertainty in agent processing. Thus the differences between the results from analysis and simulation are extended.

Table 5. Average time

I	$\overline{S_2}$	$\overline{S_3}$	**Analysis**			**Simulation**			
			$\overline{WT_2}$	$\overline{WT_3}$	ART	$\overline{WT_1}$	$\overline{WT_2}$	$\overline{WT_3}$	ART
2.05	2.001	4.065	2469.7	35.53	2515	3.052	2482.5	145.3	2637.9
4.01	2	3.92	1487	0	1497	2.965	1398	102.5	1510.4
7.08	2.044	4.034	30.22	0	40.3	3.048	*184.1*	*39.77*	*234*
10.05	1.967	3.954	0	0	9.92	3.018	*8.176*	*6.762*	*24.877*

From the experiments, we can see that the analysis can provide quite good estimations when $\overline{interval}$is smaller than the total time in a cycle of the agent.

However, when $\overline{interval}$ is bigger than the total time in a cycle, the results from analysis are normally too optimistic compared to the results from simulation. The differences are noticeable when $\overline{interval}$ is closer to the total time in a cycle. It is because the analysis is made based on the average attributes of the events. In simulation, the waiting time is affected by the attributes of single events. Simple average values cannot represent the characteristics of the events in the sequence in a precise way.

5 DESIGN OF BDI AGENTS

In order to design an effective BDI agent, it is preferred to have smaller *ART*. The values of the average waiting time should be smaller. The agent should be able to process all the incoming events in a cycle. So from the analysis, we have the conditions:

$$\overline{S_2} * \frac{t}{\overline{interval}} \leq t_2 \Rightarrow \frac{(t_1 + t_3)}{\frac{\overline{interval}}{\overline{S_2}} - 1} \leq t_2 \tag{25}$$

$$t_2 \leq \overline{S_2} \tag{26}$$

$$t_3 \geq t_2 * \frac{\overline{S_3}}{\overline{S_2}} \tag{27}$$

Equation (2) shows that if t_2 and t_3 are decided, a bigger t_1 will help decrease $\overline{WT_1}$. Considering equation (3), it is preferred that $t \leq \overline{interval}$. From equation (7), a smaller t_2 will help decrease $\overline{WT_3}$. Thus an optimal mechanism to allocate resources is:

$$t_2 = \overline{S_2} \tag{28}$$

$$t_3 = \overline{S_3} \tag{29}$$

$$t_1 = \overline{interval} - t_2 - t_3 \tag{30}$$

We can see that the mechanism will ensure that the BDI agent processes an event in a cycle. At the same time, the agent can spend as much time as possible on detecting.

6 CONCLUSION

In this paper, we present two ways for predicting the responsiveness of BDI agents, simulation and analysis. By analysis, if the basic information of the event sequence and the settings of a BDI agent are known, the average response time can be estimated using the equations. The results of the analysis provide a quick way to estimate the agent performance without making simulation experiments. The comparison of results from simulation and analysis shows that the performances can be well estimated by analysis if the average interval between events is smaller than the total time in a cycle. The predicted average response time may help modulate the allocation mechanism of a BDI agent for better performance. In another case, it can help select one agent which has best performance for a specific environment among several BDI agent candidates.

In future research, we will concentrate on the analysis when an agent is expected to have sufficient time to process events immediately. The analysis will be made based on the unique characteristics of single events, for example, the distribution function of the intervals and how the deliberation and execution time are decided. It is expected to give more precise estimations for such sequences.

ACKNOWLEDGEMENTS

We would like to thank the referees for their comments, which helped improve this paper considerably. This work was partially supported by the grant from Singapore NRF fund.

REFERENCES

[1] I.A. Ferguson, *TouringMachines: An architecture for dynamic, rational, mobile agents*, Phd thesis, University of Cambridge, 1992.

[2] M. Georgeff, B. Pell, M. Pollack, M. Tambe, and M. Wooldridge, 'The belief-desire-intention model of agency', in *Intelligent Agents V: Theories, Architectures, and Languages*, eds., Muller J.P., Singh M., and Rao A., volume 1555, 1–10, Springer-Verlag, Berlin, (1999).

[3] D. Gross and C.M. Harris, *Fundamentals of Queueing Theory*, Wiley, 1998.

[4] F.F. Ingrand, M.P. Georgeff, and A.S. Rao, 'An architecture for real-time reasoning and system control', *IEEE Expert*, **7**(6), 34–44, (1992).

[5] F.P. Kelly, 'Networks of queues with customers of different types', *Journal of Applied Probability*, **12**, 542–554, (1975).

[6] J. Lee, M.J. Huber, E.H. Durfee, and P.G. Kenny, 'Um-prs: an implementation of the procedural reasoning system for multirobot applications', in *AIAA/NASA Conference on Int. Robots in Field, Factory, Service and Space*, pp. 842–849, American Institute of Aeronautics and Astronautics, (1994).

[7] R. Nelson, *Probability, Stochastic Processes, and Queueing Theory - The Mathematics of Computer Performance Modeling*, Springer Verlag, New York, 1995.

[8] A. Pokahr, L. Braubach, and W. Lamersdorf, 'A flexible bdi architecture supporting extensibility', in *the 2005 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT-2005)*, eds., A. Skowron, J.P. Barthes, L. Jain, R. Sun, P. Morizet-Mahoudeaux, J. Liu, and N. Zhong, pp. 379–385, Compigne University of Technology, France, (2005).

[9] A.S. Rao and M. Georgeff, 'Bdi agents: From theory to practice', in *the First International Conference on Multiagent Systems*, eds., Victor R. Lesser and Les Gasser, pp. 312–319, San Francisco, USA, (1995).

[10] J.A. Rice, *Mathematical Statistics and Data Analysis*, Duxbury Press, 1999.

[11] S.M. Veres and J. Luo, 'A class of bdi agent architectures for autonomous control', in *the 43rd IEEE conference on decision and control*, pp. 4746–4751, Institute of Electrical and Electronic Engineers, (2004).

[12] M. Wooldridge, *Reasoning about rational agents*, The M. I. T. Press, Cambridge, MA, 2000.

[13] H. Zhang and S.Y. Huang, 'Are parallel bdi agents really better?', in *the 17th European Conference on Artificial Intelligence*, pp. 305–309, Riva del Garda, Italy, (2006).