Towards Requirements Analysis for Autonomous Agent Behaviour

Sorabain Wolfheart de Lioncourt[1] and Michael Luck[2]

[1] Department of Computer Science,
University of Warwick,
Coventry, CV4 7AL, UK
bane@dcs.warwick.ac.uk
[2] Dept of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ, UK
mml@ecs.soton.ac.uk

Abstract. The importance of methodologies for the construction of agent-based systems has recently begun to be recognised, and an increase in efforts directed at addressing this concern is currently being seen. Yet the focus of the majority of such work is on the *design* aspects of methodology or on the *higher-level* aspects of analysis. Of no less importance, however, are the behavioural requirements and specification of autonomous agents, which in some sense precede these phases of the development process. In this paper, we provide a detailed analysis of these requirements, and offer a preliminary view on how to focus design on meeting these requirements.

1 Introduction

An intelligent autonomous agent is expected to act for extended periods without human intervention. Although the agent is free to set its own goals and decide how best to achieve them we, as designers or developers, will have particular roles in mind for the agent, and will expect it to act accordingly. This paper details how we might specify behavioural constraints on an autonomous agent, and how to motivate the agent to direct its behaviour accordingly, while still retaining the benefits of autonomy.

At any moment an agent may have several feasible actions that it can perform, and it needs to select the appropriate action according to the external stimuli and its internal needs. Mechanisms for deciding which action is most appropriate are normally termed *action selection mechanisms* [10, 5, 2, 8]. For some types of behaviour the external stimuli are enough to determine the appropriate action, as in obstacle avoidance [4]. For other types of behaviour both external and internal stimuli need to be taken into account, such as dealing with the short-term need for food [12]. In some cases we wish the agent's behaviour to exhibit goal-directedness. For example, if an agent finds itself in a situation where there is no food in the local area then the appropriate behaviour is to move to a new

area. Where an appropriate behaviour requires a significant amount of directed action over time we need the concept of an intention [3, 9] as a special kind of internal drive, otherwise the agent will have to constantly notice that it needs to move to a new area, and may dither as the external simulti and internal needs change [7, 11]. We refer to a mechanism for generating, assessing, and dropping goals and intentions as a *goal management mechanism.*

In this paper we examine these issues in more detail, using the example of a multi-agent solution to a highly simplified emergency services coordination problem. The problem facing the designer is to motivate autonomous ambulance agents to pick up and deliver patients to hospital in such a way as to minimize some performance measure.

In the next section, we look at the different types of behavioural requirement we might impose on an autonomous agent.

2 Behavioural Requirements

If we choose to field an autonomous agent in a particular domain then it must be the case that the environment is too complex or time-consuming for us to develop a control system using conventional software engineering methods. Instead of prescribing the appropriate action to take under all possible circumstances we delegate the responsibility for choosing the appropriate actions to the agent itself. An autonomous agent architecture that is capable of directing its own behaviour towards maintaining a set of requirements in a complex environment would greatly reduce the development time and costs of such systems.

In this section we look at the requirements we may place on agent behaviour, and introduce a performance measure that allows us to measure how effective an agent is at meeting the requirements over time. The kinds of behavioural requirement are similar to those described in [1].

We argue that without an adequate specification of the behavioural requirements, or without any general measure of how well an agent meets these requirements, any agent development process will remain ad-hoc and impossible to generalize.

2.1 Avoidance and Maintenance Requirements

The first type of constraint on agent behaviour we consider is where the agent is required to avoid or maintain particular states of affairs over some timeframe (up to and including the lifetime of the agent). Such requirements include maintaining homeostasis in a simulated biological entity [6], or avoiding situations that puts the agent in danger. In our example, an autonomous ambulance agent must avoid running out of fuel at any point over its lifetime. Maintenance and avoidance requirements are equivalent – a requirement to avoid some set of states (such as any state where the fuel level is zero) is the same as a requirement to maintain the complement of those states (any state where the fuel is greater than zero).

2.2 Periodic Requirements

A periodic requirement is one where we wish the agent to perform a set task periodically over some timeframe. Such requirements may include information gathering, performing diagnostics, or general maintenance. For an autonomous ambulance agent, a periodic requirement will be to check and service the ambulance to reduce the chance of a breakdown.

2.3 One-off Tasks

The previous two types of requirement both relate to matters that persist over time. As well as such requirements, an agent may be acting as a part of a system where it will be delegated one-off tasks to perform. Such tasks may include those that should be completed as soon as possible, or before a given deadline. A new one-off task is created every time an emergency call is received by the system, with the goal of ferrying the patient to a hospital.

2.4 Types of Violation

Clearly we would ideally prefer that an agent never violates any of its requirements, but in general this may not be possible. Circumstances may arise where the agent cannot possibly maintain all of its requirements, but is faced with a choice between which requirements to fulfil, and which to let slip. Before examining how to specify preferences between violations, we examine the different types of violation that can occur.

Flexible Violation A flexible violation is one that can be rectified by an appropriate choice of actions. For example, an ambulance agent can put off its regular maintenance checks and servicing during times of crisis, and perform them later instead.

Trap Violation A trap violation is one which can never be rectified, once violated it remains violated for all time. For example, a patient may urgently require an organ transplant, and a suitable organ becomes available at another hospital. The ferrying of the organ to the patient is highly time-critical, and after a certain deadline has passed the organ is rendered useless. Allowing this deadline to pass cannot be rectified by any future action (another organ may become available, but this would form a seperate one-off task to ferry it).

Lethal Violation A lethal violation is similar to a trap violation in that it will remain violated for all time. But more than this, a lethal violation means that the agent becomes incapable of any action from the time of violation onwards. For an ambulance, allowing itself to run out of fuel is a lethal violation as it will be unable to continue on its current assignments, or undertake any new ones (in this simulation we do not allow the recovery of such ambulances).

We would normally expect an agent to prefer flexible violations over trap and lethal violations; and to prefer trap violations to lethal violations. We would not expect an ambulance to prefer carrying a non-urgent patient to ferrying a vital organ; nor would we normally wish an ambulance to run itself out of fuel in an attempt to ferry a vital organ.

3 Specifying Preferences between Violations

When the agent is faced with no way of satisfying all of its requirements it needs a way to choose which requirement violations are most preferrable. In general we cannot simply specify a strict preference ordering between requirements such that higher requirements are never violated in preference to lower ones, as there are many different levels of violation and we may be faced between choosing between sets of violations (suffering the violation of more than one requirement). It may be preferrable to suffer a short duration flexible violation of a particularly important requirement in favour of trap violations of several lower level requirements.

The solution proposed by the authors is to provide the agent with a method of deriving a quantitative measure of the *badness* of a violation (violation cost). The choice between requirement violations is then a matter of choosing the course of action that leads to the minimum expected badness.

3.1 Violation Cost

In general we will be dealing with agents with an indeterminate life-span. In order to measure their performance meaningfully over any period of time, we will derive a measure of *instantaneous* violation cost for each requirement that is not being satisified at that moment in time. The agent's performance over any interval can then be calculated by a summation of the instantaneous violation costs accrued over that interval. The alternative to the accruement of instantaneous costs is to impose the overall cost of a violation either after it has been rectified, or once a course of action that will lead to this overall cost has been committed to. However, this kind of solution will penalize an agent if the period its performance is being measured does not cover the complete violation. For example, if a patient has not been recovered by the end of a performance measurement period, then an agent that only accrues costs after rectification of a violation will not have been penalized for leaving the patient unrecovered. However, an ambulance that performed *better* and collected the patient before the end of a performance measurement period would have accrued the cost of recovering that patient, and would have fared worse in the measurement period, despite actually doing better. A similar problem occurs if the cost is accrued when a violation is committed to, where an agent that puts off committing to anything is measured as performing better, although in practice is performing worse. The only way to avoid such problems is to take on a continuous instantaneous measure of violation cost, which is the approach taken in this paper.

Definition 1. *For an agent with n requirements $r_1, r_2, \ldots, r_n$, and associated cost functions $c_1, c_2, \ldots, c_n$, the instantaneous violation cost of being in a state s is given by*

$$\mathcal{V}(s) = \sum_{i=0}^{n} c_i(s)$$

If the environment's state changes over time with the function, $s(t)$, then we can calculate the overall violation cost over a given interval by the formula given in Definition 2. This measure can be used to compare different agent architectures, or action choices under the same conditions.

Definition 2. *The overall violation cost of an agent whose state at time, t, is given by $s(t)$ over the interval $[t_1, t_2]$ is given by*

$$\mathcal{C}(s) = \int_{t_1}^{t_2} \mathcal{V}(s(t))\ dt$$

Clearly we can exploit this simple relationship both ways, being able to derive an overall violation cost function from instantaneous cost functions, or reversing the process and being able to recover an instantaneous cost function from an overall cost function (by simply taking the instantaneous cost to be the derivative of the overall cost function).

3.2 Violation Cost Functions

The formulae given in the previous definitions are designed to be as general as possible, where the instantaneous violation cost can depend on any combination of environmental state variables. In practice, the violation cost functions with be considerably simpler than this generality allows, often depending on only a single measurable variable.

Example instantaneous violation cost functions for the simple ambulance agent are shown in Figure 1.

The cost function for a violation of the fuel maintenance requirement is simple: if the agent runs out of fuel then it incurs an immediate cost of 100 (this number has been chosen arbitrarily). Note that this is the instantaneous cost, so it will incur a cost of 100 at every instant beyond the time it ran out of fuel as well, which will penalize an agent that ran out of fuel early on more than one that ran out late it its lifetime.

The periodic requirement to service the ambulance can be violated flexibly, and the service deadline at time $t1$ is soft (that is, there is still reason to get the ambulance serviced after the ideal date has passed). Here the instantaneous violation cost increases steadily, as it becomes increasingly likely that the ambulance will breakdown in proportion to the time that has passed since the last service. The cost function is always increasing, and is bounded above at 100,

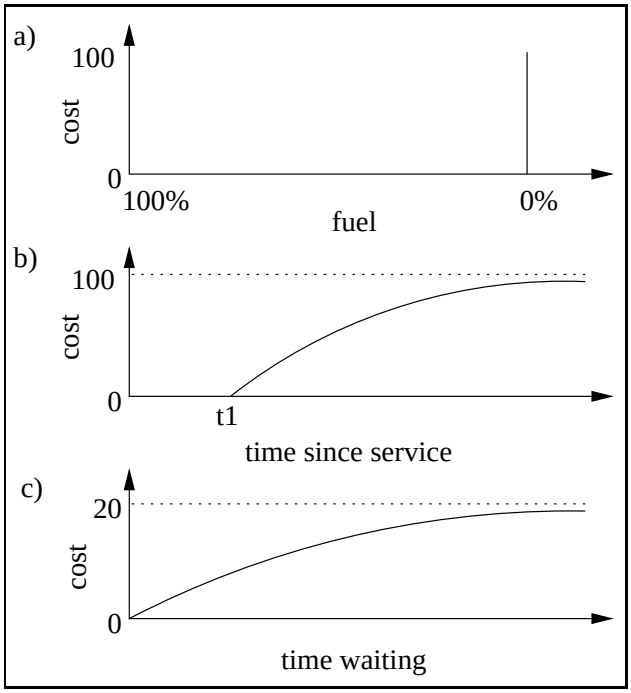

Fig. 1. Cost functions for violations of the ambulance requirements. a) cost of violating fuel maintenance requirement. b) cost of not receiving regular servicing. c) cost of patient not being delivered to hospital.

which is the same level as the instantaneous cost of running out of fuel. This is because if the ambulance actually breaks down then it is effectively the same as if it ran out of fuel (for demonstration purposes, this scenario does not allow the recovery of ambulances that have either broken down, or run out of fuel). Note again that the upper bound on the instantaneous violation cost does not indicate a bounded cost for a violation – the instantaneous violation cost is effectively the derivative of the overall cost function, and in the limit the overall cost function is still increasing with a slope of 100.

Finally, the cost functions relating to one-off tasks to recover patients are of a similar form to the servicing requirement cost function, except that in this case the ideal solution would be if the patient were instantly recovered and placed in hospital. This is unlikely to ever be the case, so an ambulance agent will always expect to incur some cost in recovering a patient. Here the instantaneous cost function is again increasing, this is to ensure that no single patient is ever left waiting for an unreasonable amount of time. If the instantaneous cost function was at a fixed level then ambulances would be striving to minimise the mean waiting time for patients, but would favour patients that are easier to pick up (i.e. are near the hospital and ambulance stations) over those that are far away.

This kind of behaviour could lead to patients being left indefinitely, although the mean waiting time was still minimized. Rather than just wanting to minimise the mean waiting time, we also want to ensure that the maximum waiting time is also kept low. Here we have chosen to bound the maximum instantaneous cost at 20. This number was chosen quite arbitrarily for this example, but represents our desire for ambulance agents to not risk running out of fuel or breaking down in their desire to recover patients. In order to decide to risk running out of fuel an ambulance would have to reason that it will recover 5 patients in doing so, which would otherwise never be recovered. This is unlikely to be the case, even if there is only one ambulance, since the agent deciding this course of action can choose to refuel and service itself and then collect those patients safely.

4 Choosing Appropriate Requirements

We make two assumptions about each requirement of an agent, namely that:

1. there exists at least one possible situation where the requirement is violated; and
2. at least one of the situations in which the requirement is violated could either have been predicted and avoided, or is a flexible violation that is detectable and can be rectified by an appropriate choice of agent actions.

Both of these assumptions serve to protect the agent from worrying about situations that it can do nothing about. The first assumption means that it must be possible for each requirement to be violated. For example, it would be pointless to give an ambulance agent the requirement *avoid being in two places at once*, since the ambulance can never find itself in such a situation.

The second assumption states that an agent must be able to predict and avoid at least some lethal and trap violations for each requirement (ideally, but not necessarily all). This will depend on the perceptual and physical capabilities of an agent. For example, a vampire with the requirement *avoid having a wooden stake driven into the heart* needs to be able to perceive a would-be vampire slayer with stake in hand, and then be able to run away very fast in the other direction. A vampire without the senses to detect the danger, or the ability to avoid it once it has been detected can do nothing to avoid the requirement being violated.

In the case of a flexible violation the agent need not necessarily be able to predict and avoid the violation before it arises, but should be able to rectify at least some of the possible situations in which a flexible violation has occurred (not necessarily all). This will again depend on the physical capabilities of an agent, it would be reasonable to expect a vampire to be able to rectify a violation of the requirement *avoid sunlight* should it be unfortunate enough to find itself in such a situation, but giving a tree the same requirement would be pointless since the the average tree is powerless to rectify such situations.

If either assumption is not met then the behaviour of the agent with such a requirement will be identical to an agent without the requirement. If requirement serves no purpose in directing the agent, it should be removed.

5 Conclusion

The analysis of requirements for autonmous agents considered above provides a first pass towards the first stage in the software development process. The method provided in this paper gives both a general way for agents to calculate the utility of their actions, and an effective performance measure that is independent of an agent's capabilities.

The violation costs assigned to requirements play a dominant role in prescribing how an autonomous agent should behave. In order for this framework to be successful there needs to be a clear methodology for selecting appropriate values to match the behaviour that the designer has in mind. This is a general problem with many motivational systems [12, 2], where the appropriate parameters for specific agents are often found by trial-and-error rather than by a formal analysis.

References

1. C. Balkenius. The roots of motivation. In J.-A Meyer, H. L. Roitblat, and S. W. Wilson, editors, *From Animals to Animats II*, Cambridge, MA, 1993. MIT Press.
2. Bruce Mitchell Blumberg. *Old Tricks, New Dogs: Ethology and Interactive Creatures.* PhD thesis, Massachusetts Institute of Technology, 1997.
3. M. E. Bratman. *Intentions, Plans and Practical Reason.* Harvard University Press, Cambridge, MA, 1987.
4. R. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2:14–23, 1986.
5. Pattie Maes. How to do the right thing. *Connection Science Journal*, 1(3), 1989.
6. D. McFarland and T. Bösser. *Intelligent Behaviour in Animals and Robots.* MIT Press, Cambridge, MA, 1993.
7. M. Minsky. *The Society of Mind.* Heineman, 1987.
8. Paolo Pirjanian. *Multiple Objective Action Selection & Behaviour Fusion Using Voting.* PhD thesis, Department of Medical Informatics and Image Analysis, Aalborg University, 1998.
9. Anand S. Rao and Michael P. Georgeff. Modeling rational agents within a BDI-architecture. Technical Report 14, Australian Artificial Intelligence Institute, February 1991.
10. J. K. Rosenblatt and D. Payton. A fine-grained alternative to the subsumption architecture for mobile robot control. In *Proc IEEE/INNS Int'l Joint Conf on Neural Networks*, page 65ff, 1989.
11. M. K. Sahota. Action selection for robots in dynamic environments through inter-behaviour bidding. In D. Cliff, P. Husbands, J.-A.Meyer, and S.W. Wilson, editors, *From Animals to Animats III*, pages 138–142, Cambridge, MA, 1994. MIT Press.
12. Toby Tyrrell. *Computational Mechanisms for Action Selection.* PhD thesis, University of Edinburgh, 1993.