# Agent-Based Systems
# Tutorial 1

Michael Rovatsos

**Q1** For which of the following applications would an agent-based solution be appropriate?

1. A word processing software suite
2. An MP3 player with wireless networking capabilities
3. An automated monitoring system for a national railway network

Consider, for each application, a single-agent and a multiagent solution separately. Give reasons for your answer.

***Solution suggestions:*** *Criteria for using agent/multiagent solutions that should be discussed here are the following:*

- *decentralisation of data and control, asynchronous computation by distributed components*
- *self-interested components representing different users with different goals*
- *need to delegate complex tasks to autonomous agents and not to tell them exactly what to do in each step*
- *interaction between several components, particularly when patterns of interaction and communication cannot be predicted at design time*

*Additional criteria often mentioned are:*

- *exploiting redundancy by having several agents for the same task (particularly relevant in safety-critical applications)*
- *applicability of agent notion as a "natural" metaphor (e.g. when using organisational abstractions in the design of systems or in the case of "lifelike"/"believable" user assistant agents)*
- *increased scalability through high degree of encapsulation (agent capabilities complex to implement once, but once that has been done they should be able to interact with arbitrary numbers of other agents)*

*For the examples, the following answers would be appropriate (I only mention certain arguments, you can go systematically through the list above to discuss further ones):*

1. *Word processor: neither agent nor multiagent-based solution is appropriate. Individual tasks (like tabbing, creating tables, spell-checking etc.) are too simple, document shouldn't change without user input, only very little potential for concurrency (e.g. background spell-checking, but agents not really necessary for that). [As an*

*aside, you can mention the Microsoft "paperclip" assistant, this wasn't well-received by the public at all, although it originally even incorporated very smart machine learning capabilities.]*

2. *Wireless-enabled MP3 player: agent-based solution appropriate (might detect other players, exchange data, negotiate with them, etc.), should be proactively doing that (user wants to listen to music undisturbed); multiagent solution probably not necessary (device would probably only do – at most – a couple of things at a time); point out that there is a difference between modelling the MAS consisting of several such players, and modelling an individual agent for one player.*
3. *Railway monitoring system: a multiagent system solution is appropriate here – agents can control different nodes (organisational design may be suitable including hierarchies), redundancy can be exploited to ensure responsiveness even if some nodes have inaccurate data or in case of node failure, many different tasks, can be quite complex, different nodes might represent different railway companies and actually be self-interested, system can be easily extended if network changes physically. Single-agent solution probably not appropriate, complexity too high for a monolithic system.*

**Q2** Prove or refute the following statements:

1. For every purely reactive agent, there is a behaviourally equivalent standard agent.
2. For every standard agent, there is a behaviourally equivalent purely reactive agent.
3. For every state-based agent there is behaviourally equivalent standard agent.
4. For every standard agent there is behaviourally equivalent state-based agent.
5. Every utility function defined over runs can be expressed by a utility function defined over states.
6. Every utility function defined over states can be expressed by a utility function defined over runs.

***Solution suggestions:***
*Two agents $Ag_1$ and $Ag_2$ are called behaviourally equivalent with respect to environment $Env$ iff $\mathcal{R}(Ag_1, Env) = \mathcal{R}(Ag_2, Env)$. If this is true for any environment $Env$, the are simply called behaviourally equivalent. Since this definition is based on sets of runs, the proof technique is by set inclusion, i.e. we pick a random element $r \in \mathcal{R}(Ag_1, Env)$ and show that is is contained in $\mathcal{R}(Ag_2, Env)$ for given $Ag_1$, $Ag_2$ and arbitrary $Env$ to show the "$\subseteq$" direction, and then we pick a $r' \in \mathcal{R}(Ag_2, Env)$ to show the converse "$\supseteq$". From $\mathcal{R}(Ag_1, Env) \subseteq \mathcal{R}(Ag_2, Env)$ and $\mathcal{R}(Ag_1, Env) \supseteq \mathcal{R}(Ag_2, Env)$ we conjecture that $\mathcal{R}(Ag_1, Env) = \mathcal{R}(Ag_2, Env)$.*

1. *Let $Ag_1 : E \to Ac$ a purely reactive agent. We define a standard agent $Ag_2 : \mathcal{R}^E \to Ac$ by setting $Ag_2(e_0 \overset{\alpha_0}{\to} \cdots \overset{\alpha_{u-1}}{\to} e_u) = Ag_1(e_u)$ for every $r = e_0 \overset{\alpha_0}{\to} \cdots \overset{\alpha_{u-1}}{\to} e_u \in \mathcal{R}^E$. Now we prove that the sets of runs generated by both agents are identical.*
   *"$\subseteq$": Let $r \in \mathcal{R}(Ag_1, Env)$. We can write $r$ as $e_0 \overset{Ag_1(e_0)}{\to} \cdots \overset{Ag_1(e_{u-1})}{\to} e_u$ since $Ag_1$ is a purely reactive agent. But since $Ag_2(e_0 \overset{\alpha_0}{\to} \cdots \overset{\alpha_{u-1}}{\to} e_u) = Ag_1(e_u)$ by definition for every $u$, $Ag_2(e_0 \overset{\alpha_0}{\to} \cdots \overset{\alpha_{i-1}}{\to} e_i) = Ag_1(e_i) = Ag_1(e_0 \overset{\alpha_0}{\to} \cdots \overset{\alpha_{i-1}}{\to} e_i)$ holds for any $1 \leq i \leq u$, and therefore $r \in \mathcal{R}(Ag_2, Env)$.*

*"$\supseteq$": Let $r' = e_0 \overset{\alpha_0}{\rightarrow} \cdots \overset{\alpha_{u-1}}{\rightarrow} e_u \in \mathcal{R}(Ag_2, Env)$. It holds that $Ag_2(e_0 \overset{\alpha_0}{\rightarrow} \cdots \overset{\alpha_{i-1}}{\rightarrow} e_i) = Ag_1(e_i)$ for all $1 \leq i \leq u$, so we can write $r'$ as $e_0 \overset{Ag_1(e_0)}{\rightarrow} \cdots \overset{Ag_1(e_{u-1})}{\rightarrow} e_u$ and thus $r' \in \mathcal{R}(Ag_1, Env)$.*
*Since we have defined a standard agent $Ag_2$ that produces the same runs in the same (arbitrary) environment as a purely reactive (fixed, but arbitrary) agent $Ag_1$, we can always find a standard agent that is behaviourally equivalent to a purely reactive agent.*

2. *Proof by contradiction. We build a standard agent $Ag_1$ in an environment with states $\{e_0, e_1\}$ and actions $\{\alpha_0, \alpha_1\}$ such that $Ag_1(e_0 \overset{\alpha_0}{\rightarrow} e_1) = \alpha_1$ and $Ag_1(e_1 \overset{\alpha_0}{\rightarrow} e_1) = \alpha_0$. From all possible agent designs for a purely reactive agent $Ag_2$ in this environment, it has to either hold that $Ag_2(e_1) = \alpha_1$ or $Ag_2(e_1) = \alpha_0$. Let us assume $Ag_2(e_1) = \alpha_1$ then $Ag_2(e_0 \overset{\alpha_0}{\rightarrow} e_1) = Ag_2(e_1) = \alpha_1$ and $Ag_2(e_1 \overset{\alpha_0}{\rightarrow} e_1) = Ag_2(e_1) = \alpha_1$ so $Ag_2$ cannot be behaviourally equivalent to $Ag_1$. By using the same argument for $Ag_2(e_1) = \alpha_0$ we can show that no such $Ag_2$ can exist. If there is at least one standard agent for which no purely reactive agent exists then this disproves the claim.*
3. *Assume $Ag_1$ is a standard agent as before. A state-based $Ag_2$ is defined by functions $see : E \rightarrow Per$, $action : I \rightarrow Ac$, $next : I \times Per \rightarrow I$ such that $Ag_2(e_0 \overset{\alpha_0}{\rightarrow} \cdots \overset{\alpha_{u-1}}{\rightarrow} e_u) = \alpha'_u$ iff $see(e_u) = p_u$, $next(s_u, p_u) = s_{u+1}$ and $action(s_u) = \alpha'_u$. Define the percept set of $Ag_2$ as a one-to-one mapping of all environments to percepts i.e. $Per = E$ and the set of internal states as the set of all percept state sequences $I = Per^*$. Then, for every run $r = e_0 \overset{\alpha_0}{\rightarrow} \cdots \overset{\alpha_{u-1}}{\rightarrow} e_u$ of $Ag_1$ we set: $see(e_i) = e_i$, $next(e_1 \cdots e_i, e_i) = e_1 \cdots e_i e_{i+1}$ and $action(e_1 \cdots e_i) = \alpha_i$ for all $1 \leq i \leq u$. It is easy to show that $Ag_2(e_0 \overset{\alpha_0}{\rightarrow} \cdots \overset{\alpha_{u-1}}{\rightarrow} e_u) = Ag_1(e_0 \overset{\alpha_0}{\rightarrow} \cdots \overset{\alpha_{u-1}}{\rightarrow} e_u)$ and with this that the two agents are behaviourally equivalent.*
4. *Assume $Ag_1$ is a state-based agent defined by functions $see : E \rightarrow Per$, $action : I \rightarrow Ac$, $next : I \times Per \rightarrow I$ such that $Ag_2(e_0 \overset{\alpha_0}{\rightarrow} \cdots \overset{\alpha_{u-1}}{\rightarrow} e_u) = \alpha_u$ if $see(e_u) = p_u$, $next(s_u, p_u) = s_{u+1}$ and $action(s_u) = \alpha_u$. Since the actions generated by $Ag_1$ are uniquely determined by percept and internal state sequences as, we can easily set $Ag_1(e_0 \overset{\alpha_0}{\rightarrow} \cdots \overset{\alpha_{u-1}}{\rightarrow} e_u) = \alpha_u$ for some standard agent $Ag_2$ and show it is behaviourally equivalent to $Ag_1$. It can be left to the students as an exercise to show that each run can be mapped to such a percept/state sequence and the action outcome defined accordingly for $Ag_1$ to be $\alpha_u$.*
5. *and 6.: Utility functions defined over runs are strictly more expressive than those defined over environment states. The proof is almost identical as that for parts 1 and 2 (a simple counterexample can be found for a run-based utility function that cannot be expressed as a state-based utility function).*

**Q3** The game of rock-paper-scissors is a single-shot two-player game in which each agent can pick from either of three moves $R$, $P$ and $S$ simultaneously. The result is evaluated using the following rules:

- $P$ beats $R$
- $R$ beats $S$
- $S$ beats $P$
- all other combinations result in a tie

Assume our opponent (the "environment") plays $P$ with probability 0.4, $S$ with probability 0.5, and $R$ with probability 0.1.

(a) Specify an *optimal* (stochastic) agent for this problem according to the MEU criterion given that the utility for losing the game is -1 and the utility for winning is +1, 0 for a tie.

(b) Can you generalise the result for *any* opponent strategy rather than the one defined in part (a)?

***Solution suggestions:*** *Part (a): In the book (p. 39), the maximum expected utility principle is expressed using probabilities of runs. A simple representation for the possible runs is through two-letter combinations e.g. $PS$ where the first letter stands for "agent move" and the second for "environment move" (so $PS$ means "agent plays $P$, environment plays $S$"). The $3 \cdot 3 = 9$ resulting runs can be grouped together into "win", "lose" and "tie" cases, but this is not necessary, strictly speaking. With this, we can specify the agent's design (or strategy) through two parameters $p, r \in [0:1]$ denoting the probability of the agent two play $P$ and $R$ (the probability of $S$ can be computed implicitly as $1 - p - r$), and the expected utility of a strategy under the given environment $Env$ becomes*

$$\begin{aligned}
EU(p, r, Env) =& u(win) \cdot P(win|p, r, Env) + u(lose)P(lost|p, r, Env) + u(tie)P(tie|p, r, Env) \\
=& (+1) \cdot (P(PR) + P(RS) + P(SP)) + (-1) \cdot (P(RP) + P(SR) + P(PS)) \\
& + 0 \cdot (P(RR) + P(SS) + P(PP)) \\
=& (p \cdot 0.1 + r \cdot 0.5 + (1 - p - r) \cdot 0.4) - (r \cdot 0.4 + (1 - p - r) \cdot 0.1 + p \cdot 0.5) \\
=& 0.1p + 0.5r + 0.4 - 0.4p - 0.4r - 0.4r - 0.1 + 0.1p + 0.1r - 0.5p \\
=& 0.3 - 0.7p - 0.2r
\end{aligned}$$

*What we are looking for is the optimal agent design, i.e.*

$$\max_{p,r \in [0:1]} EU(p, r, Env)$$

*and (luckily), since $EU(p, r, Env)$ is decreasing in both $p$ and $r$, the optimal strategy is $p = 0$ and $r = 0$ (i.e. the agent always plays $s$). Note, however, that the search space of values for $p$ and $r$ might have to be searched exhaustively if the environment was different, or we would have to apply other optimisation techniques.*

*Part (b): No. In the general case (for identical utilities), the above equation reads*

$$\begin{aligned}
EU(p, r, Env) =& (pr' + r(1 - p' - r') + (1 - p - r)p') - (rp' + (1 - p - r)r' + p(1 - p' - r')) \\
=& 3pr' + r - 3rp' + p' - r' - p
\end{aligned}$$

*for probabilities $p'$ and $r'$ of the environment playing $P$ and $R$, respectively, so obviously the optimal agent design depends very much on the quantities of these values.*