AI & SOCIETY (2023) 38:963–965
https://doi.org/10.1007/s00146-022-01463-6

CURMUDGEON CORNER

Toward trustworthy programming for autonomous concurrent systems

Lavindra de Silva[1] · **Alan Mycroft**[2]

Received: 14 July 2021 / Accepted: 13 April 2022 / Published online: 8 June 2022

A key focus in AI is building machines and software capable of being autonomous, especially in complex and dynamic environments where, e.g., self-driving cars, trading systems, and social care robots operate. Such autonomous systems are able to independently make decisions and act on them with limited human intervention, balancing the pursuit of long-term goals (proactiveness) with rapid response to environmental changes (reactiveness) (Fisher et al. 2021). The notion of an autonomous system is synonymous with the notion of an 'autonomous software agent' (Fisher et al. 2021), and a class of domain-specific language called an Agent-Oriented Programming Language (AOPL) has proved to be one of the most successful approaches to building such systems. AOPLs provide abstractions over Object-Oriented Programming, by modelling complex systems through the 'intentional stance'—human-like mental attitudes, such as beliefs, goals, and intentions, enabling users understand, explain, predict, and program behaviour by abstracting from the detail (objects, attributes, etc.). Indeed, giving people this ability helps build *trustworthy* AI systems, particularly those that people can trust to have been designed and programmed to be lawful, ethical, and robust,[1] ensuring adherence to applicable laws, regulations, and ethical principles, and operating in a safe, secure and reliable manner.

AOPLs have been studied and developed for over 30 years, with their first implementations proceeding alongside philosophical, conceptual, and early theoretical underpinnings (see, e.g., Bordini et al. 2020). A subfield called Formal Semantics for AOPLs emerged out of a need to describe practical and implementable systems, and bridge the gap between the early theoretical work and successful AOPL implementations. A formal semantics includes both an execution model and a programming language syntax, and they now exist for many of the implemented AOPLs, both for describing their 'decision-making engines', and specifying their input in the form of 'Standard Operating Procedures' (SOPs), comprising the layered steps needed to accomplish high-level operations ('tasks'), such as user requests (Fig. 1), and to respond to environmental changes. Such semantics have enabled rigorously modelling AOPLs and reasoning about their behaviour and decisions, and the ability for programmers and operators to *understand* concretely how an AOPL-based autonomous system will execute the given SOPs, *explain* (e.g., to a regulator) why the system did what it did, or automatically *verify* that a layer(s) in the system is compliant or safe. In contrast especially to systems built using Machine Learning, AOPLs facilitate a layered approach to explanation, with the option to probe deeper layers until the desired level of understanding is reached, e.g., a user could debug an issue by asking the system which top-level goals (adopted tasks) it had, what intention it had for a goal (i.e., which SOP instance it had committed to executing), and if necessary, why that intention rather than another, and why the system held a certain belief (Bordini et al. 2020).

Formal (semantics for) AOPLs have had various extensions over the years, either describing existing implementations or laying the groundwork for new or improved implementations. A key extension has been the ability to execute goals *concurrently*; this can be achieved either by multi-core execution or simulated by multi-tasking on a single-core processor (Fig. 1 illustrates various forms of concurrency). While there has been progress on the work on multi-tasking, it is now critical for formal AOPLs to support notions of concurrency that suit *multi-core/multi-processor* systems, especially in this era as such systems

✉ Lavindra de Silva
Lavindra.deSilva@eng.cam.ac.uk

Alan Mycroft
Alan.Mycroft@cl.cam.ac.uk

[1] Department of Engineering, University of Cambridge, Cambridge, UK

[2] Department of Computer Science and Technology, University of Cambridge, Cambridge, UK

[1] As per the European Commission's Ethics Guidelines for Trustworthy AI.

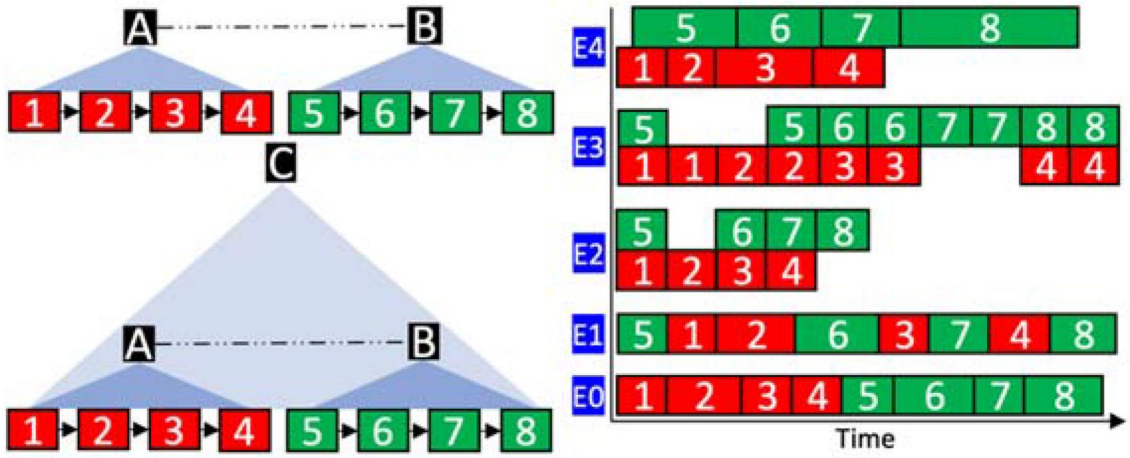

Fig. 1 A and B are tasks achieved by SOPs (triangles) comprising sequences of four operations. The graph shows one possible execution of operations over time for some AOPL execution models, as A and B run as top level tasks or as subtasks within task C. E0–E4 on the *Y* axis correspond to an execution that is: sequential (E0); interleaved (E1); interleaved, synchronous, and with operations taking unit time (E2); as before, and with operations split (shown as repeating numbers) into 'start' and 'end' parts (E3); and asynchronous (E4)

have become ubiquitous—being present in devices including phones and low-cost computers—and autonomous systems are becoming prevalent, and even affordable in new sectors (e.g., smart personal transporters and sophisticated personal drones). Formal AOPLs for contemporary autonomous systems must, therefore, cater for the opportunity to run goals, intentions, and their threads (e.g., subgoals) on separate cores, while balancing this against the need to verify such behaviour, which may well be harder due to the many ways intentions/threads can overlap.

We believe that the development of a formal AOPL that supports suitable notions of concurrency can take much inspiration from the body of work in the general computer science (CS) literature on modeling concurrency in general-purpose programming languages and distributed processes. However, it is quite likely that developing such an AOPL will be a distinct and major strand of work, like the existing strands on concurrency models for C and Java. For example, the CS literature does not seem to, as argued elsewhere, cater for aspects that are fundamental in AI, such as modelling the 'beliefs' of an autonomous system (i.e., information about itself and the environment); modelling the 'preconditions' of operations (specifying when the operation is applicable) and their 'effects'; and a mechanism for inferring the system state that results from executing an operation in the current state (the 'frame problem'). In AI itself, there is some work that takes a step toward supporting concurrency in formal AOPLs, but with the limitation wherein only 'synchronised concurrency' is allowed, i.e., each segment that is executed concurrently [with another segment(s)] must be a single operation, each of which is performed 'in sync' (Fig. 1). While this approach has been used to mimic an AOPL operation that stretches over multiple operations by splitting the former into 'start' and 'end' parts (Fig. 1), splitting may not capture concurrency fully (van Glabbeek 2015).

A more natural approach to modelling such overlaps between AOPL operations (see 'E4' in Fig. 1) has been proposed by the first author in recent work, but this has nonetheless left open many challenging and interesting questions for a concurrency theory for AOPLs. In particular, since many AOPLs take the intentional stance, what are suitable models for concurrently executing *intentions*, and how do they relate to models for executing threads *within* an intention (given that threads are preprogrammed, whereas intentions are influenced by people and the environment)? Core to this question is how to model intentions/threads that interact through a (central) belief base that is *shared*, and the environment being sensed *during* concurrent execution. The AOPL's theoretical properties would also need to be understood, especially its consistency with concurrency semantics in CS, and the 'fragment' of the AOPL that is amenable to verification would need to be ascertained, as well as how to maximise the fragment to achieve increased reliability.

Curmudgeon Corner Curmudgeon Corner is a short opinionated column on trends in technology, arts, science and society, commenting on issues of concern to the research community and wider society. Whilst the drive for super-human intelligence promotes potential benefits to wider society, it also raises deep concerns of existential risk, thereby highlighting the need for an ongoing conversation between technology and society. At the core of Curmudgeon concern is the question: What is it to be human in the age of the AI machine? -Editor.

References

Bordini RH, Seghrouchni AEF, Hindriks K, Logan B, Ricci A (2020) Agent programming in the cognitive era. Auton Agents Multi-Agent Syst 34:37

Fisher M, Mascardi V, Rozier KY, Schlingloff B-H, Winikoff M, Yorke-Smith N (2021) Towards a framework for certification of reliable autonomous systems. Auton Agents Multi-Agent Syst 35:8

van Glabbeek RJ (2015) Structure preserving bisimilarity, supporting an operational Petri Net semantics of CCSP. In: Proceedings of correct system design, pp 99–130