Unifying Models of Data Flow

Tony Hoare
Microsoft Research, Cambridge, United Kingdom

A unifying model is one that generalises a diverse range of more specific theories, applicable to a range of phenomena in the real world. The original models turn out to be special cases, which are (and will remain) useful in their special areas of application. The similarities of the specific theories are codified by a set of algebraic laws which they all share; and they are precisely and concisely differentiated by a choice of laws which they do not share.

We take data flow as our primitive concept. Data held in a central computer memory flows across an interval of time separating an assignment of a value to a variable from an access of the value assigned. Data communicated on a channel flows across space that separates two components of a real or simulated computer network. Acknowledgement, synchronisation and other forms of control flow are treated as data flow in which no information is transferred. This insight enables us treat all these forms of flow uniformly. Furthermore, the treatment is independent of the program which defines and uses the data, and independent of the language in which the program is expressed.

For both memory and communication, we will classify several variations. In the case of computer memory, we will deal with variables that are private to an individual thread and those which are shared between multiple threads. They will interleave accesses to main memory at varying levels of granularity. The hardware of shared memory may conform to strong or weak rules of consistency. Variations of our model of communication include channels that are buffered or synchronised, stuttering or lossy, and overtaking or order-preserving. There are various methods of allocation and disposal of computer resources, including nested allocation of data declared local to a block, as well as dynamically allocated data. This latter is either explicitly disposed by program or recovered automatically in an implementation, for example by a garbage collector. All the models will cover both the sequential and concurrent use of resources by threads and by processes, defined at various grains of atomicity.

This wide range of generality is obtained by concentrating only on the most essential properties of each of the computing concepts. Each definition will concentrate on the primary purpose of the resource and its behaviour, ignoring completely the ways in which a feature might be implemented, either in software and/or by the hardware of a physical computing device. The incompleteness of our formalisation will be unfamiliar to an audience more accustomed to the completeness of other forms of programming language semantics.

The reference complements the current lectures by providing a Unifying Model of Control Flow. It also introduces and uses all the concepts quoted as keywords [see Summer School's Questionnaire].

References

1. C.A.R. Hoare, I. Wehrman, P. O'Hearn. *Graphical Models of Separation Logic.* M. Broy, W. Sitou, T. Hoare (eds.); Engineering Methods and Tools for Software Safety and Security; IOS Press; pp. 177-202; 2009.