Available online at www.sciencedirect.com

ScienceDirect

IFAC-PapersOnLine 48-23 (2015) 380–387

Feedback and Time are Essential for the Optimal Control of Computing Systems

Eric C. Kerrigan *

* *Imperial College London, UK, (e-mail: e.kerrigan@imperial.ac.uk)*

Abstract: The performance, reliability, cost, size and energy usage of computing systems can be improved by one or more orders of magnitude by the systematic use of modern control and optimization methods. Computing systems rely on the use of feedback algorithms to schedule tasks, data and resources, but the models that are used to design these algorithms are validated using open-loop metrics. By using closed-loop metrics instead, such as the gap metric developed in the control community, it should be possible to develop improved scheduling algorithms and computing systems that have not been over-engineered. Furthermore, scheduling problems are most naturally formulated as constraint satisfaction or mathematical optimization problems, but these are seldom implemented using state of the art numerical methods, nor do they explicitly take into account the fact that the scheduling problem itself takes time to solve. This paper makes the case that recent results in real-time model predictive control, where optimization problems are solved in order to control a process that evolves in time, are likely to form the basis of scheduling algorithms of the future. We therefore outline some of the research problems and opportunities that could arise by explicitly considering feedback and time when designing optimal scheduling algorithms for computing systems.

Keywords: Optimal control, scheduling, real-time systems, computing systems, sensor networks, network programming, communication networks, distributed computing

1. OPTIMAL COMPUTERS USE CONTROL

Computing systems may be composed of reliable and efficient *components*, but reliability of the overall *system* comes at the cost of large inefficiencies due to over-engineering. In embedded computing applications, such as avionics, engineers are constantly seeking to reduce the size, weight and power of the systems, which results in significant savings in energy, cost, maintenance and improved safety. In traditional data centres, for example, a server can draw 70–90% of its maximum power when it is not doing any work. Information and Communication Technologies (ICT) were responsible for producing at least 7% of worldwide electricity consumption in 2008 and this figure is expected to rise to more than 14% by 2020 (Vereecken et al., 2010). Though the GeSI SMARTer 2020 report (GeSI, 2012) claims that ICT-enabled solutions has the potential to reduce greenhouse gas emissions by 16.5% in 2020, there is clearly a need for computing systems to reduce their own contribution to these emissions.

What is lacking, however, is a complete theory that allows engineers to understand how various components interact with each other and what effect this has on the overall system behaviour. By building on recent results from control theory and mathematical optimization, disciplines where *feedback* enables engineers to design robust and efficient systems, it is possible to design computing systems that can be one or more orders of magnitude more energy efficient, cheaper, faster, smaller and reliable than today.

Every computing system today employs *feedback* in some form or other to guarantee a certain level of performance and reliability in the presence of *uncertainty*, such as unpredictable work-loads, computational and communication delays, data losses and component failures. Problems that require feedback algorithms arise in a variety of contexts in computing systems (Hellerstein et al., 2004, 2009):

- Data, tasks and resources (such as processors, storage and communication networks) need to be managed to achieve a certain quality of service, guarantee that computations are correct and ensure that tasks are completed before deadlines.
- Minimization of power consumption and overheating protection by dynamic voltage and frequency scaling and smart scheduling of jobs.
- Estimating the workload and available resources, as well as the status and completion rate of jobs.
- Guaranteeing resilience of the system in the presence of faults and cyber attacks.

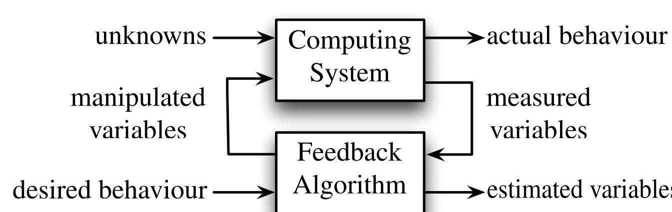

Fig. 1. Feedback-based computing system.

Figure 1 shows the key components of a typical feedback-based computing system. Unknowns, such as future work-loads and data losses, act on the system. It is often pos-

10.1016/j.ifacol.2015.11.309

sible to measure key variables, such as power consumption, duration of a computation or resource utilisation. The feedback algorithm might use these measurements to update a model of the computing system to correct for errors in the estimates of the measurements. If the actual behaviour of the system is different from the desired behaviour, the feedback algorithm updates the values of certain manipulated variables, such as the processor clock frequency or memory allocation, until the behaviour is as desired or the estimates are sufficiently accurate.

1.1 Why is the control of computing systems challenging?

Computing systems present a number of significant challenges that stretch the theory and practice of modeling, control and optimization well beyond what is possible with the state of the art:

- Time-scales at which the dynamics evolve range from pico-seconds to hours and even days. In some applications the speed of a feedback algorithm is not important and in others absolutely critical.
- Power consumption can range from tens of MW to less than a pW. The power consumed by the feedback algorithm itself may therefore have to be minimized.
- The cost of a system can range from a few cents to billions of euros. The silicon area and cost of implementing a feedback algorithm may therefore be absolutely critical to the application.
- The same system may have to execute a variety of tasks with mixed criticalities. Some tasks or units are not allowed to miss their deadline or fail, whereas delays and data losses are acceptable in others.
- Demands by an application on the speed or resources can vary by orders of magnitude in a single run. Uncertainty models based only on the worst-case or average-case may therefore not be practical.
- The number of computing units that may interact with each other and the number of tasks may vary from one to billions or more. Feedback algorithms therefore need to be scalable.
- Processing units can all be located on one chip or spread across the world. Feedback algorithms therefore might have to be implemented in a decentralized manner, while guaranteeing correct overall system behavior.
- Computing systems have hybrid dynamics. Methodologies are needed that can cope with the interaction between discrete dynamics on the computation side, e.g. logic and discrete states or events, and continuous dynamics on the physical side, e.g. heat dissipation or energy usage.
- First-principles modeling of computing systems is still very much in its infancy. A mixture of new first-principles and data-driven modeling methodologies needs to be developed.

1.2 Not all computers are on desktops or in data centres

It is often the case that embedded computing systems are mobile and/or part of a sensor network. Typical examples include: (i) automotive electronics or avionics systems, (ii) mapping of traffic or pollution in a city, (iii) automated manufacturing, farming or warehousing, or (iv) UAVs flying in formation to reduce air drag, fighting forest fires or performing remote earth sensing tasks.

In these applications there is often a need for individual nodes to cooperate towards satisfying high level tasks, which require a significant amount of computation power. Consider the fire fighting scenario, for example. The system has to use a numerical model of the fire dynamics to predict how the fire will develop, solve an optimization problem to determine where and when to send each UAV and fire fighting unit, as well as perform local control of each UAV. Because of the unpredictable environmental conditions, the system should be fault tolerant and be able to do the simulation, data processing, coordination and planning by itself in real-time, by combining the processing power of each node in the network, rather than sending all the data to a central high performance computing facility.

Mobile computing systems and sensor networks present many of the challenges to modeling and control mentioned above, with the addition that the quality and structure of the communication network varies with time, hence the topology of the computation network has to change. If the nodes are mobile, then the position and propulsion energy also have to be integrated when determining how best to control the computation, communication and data storage.

The main point to note here is that the computing system often interacts with the physical world and/or vice versa in real-time. Traditional methods for the control of computing systems do not always explicitly acknowledge or take advantage of this fact. It therefore makes sense to consider the modeling and control of the combined cyber-physical system, rather than treating the computing system as separate from the physical system.

1.3 Where are we?

One of the reasons why computing systems are over-engineered is because feedback algorithms in computing systems are usually designed in an ad hoc manner without systematic use of methods from the rich body of control theory, the science of feedback in dynamical systems. This is often also not helped by there being some slight, but important, differences in terminology between the computing and control communities. In computing, the terms ‘dynamic’ or ‘static’ are used where a control engineer would have insisted on using ‘feedback/closed-loop’ or ‘open-loop’, respectively. In control theory, feedback algorithms can be dynamic or static. Sometimes ‘dynamic’ or ‘static’ are used in the computing literature when, respectively, ‘open-loop time-varying’ or ‘open-loop constant’ would have been consistent with the control theory literature. A control engineer would usually agree that a computer engineer’s ‘feedback scheduler’ is a feedback algorithm.

A number of academic and industrial research groups have reported significant improvements in the response times, quality of service, reliability, energy and resource usage of computing systems by systematically implementing control-theoretic feedback algorithms in the design of their computing and software systems (Hellerstein et al., 2004, 2009; Sha et al., 2004).

Research in this area is still very much at an early stage. Many existing techniques for the control of computing