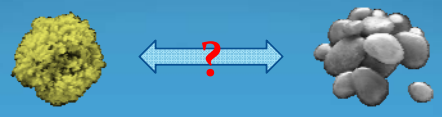
How Hard is Control?

Some reflections inspired by the work of Paul Caspi

Karl-Erik Årzén, Lund University



My Connection to Paul Caspi

- Through his work
- Through ARTIST
- Through our joint partcipation in the ARTIST EU/UNU-IIST School on Embedded Systems, Suzhou, China

ARTIST2

Connections to the Synchronous Approach

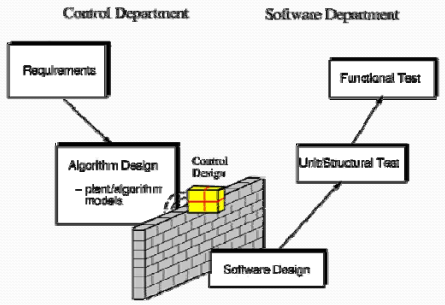
- Only a very very indirect connection to the synchronous approach
- Telelogic, the company that owned SCADE during a period of time, originates from Lund University
- Currently being bought by IBM
- New EC FP7 project ACTORS
  - Based on the CAL Actor data-flow language from Ptolemy 2
  - Ericsson, Xilinx, ...

ACTORS

Paul's Contributions to Control

- *"Between Control and Software"*
- Some examples:
  - Approximation theory for embedded control that captures robustness towards implementation effects (sampling, delays, jitter, distribution, ..)
  - Quasi-Synchronous approach to distributed control
  - Synchronous data-flow languages
  - .....
- Main themes: Synchronicity and Time

Software and Control: The Common Case



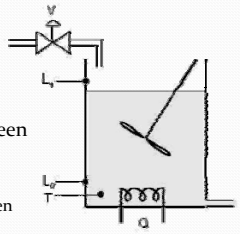
What do we mean by hard?

Hard Deadlines

- In computer science feedback control loops are traditionally modeled as
  - Periodic activities with period T
  - Hard deadlines (D)
    - D=T
    - D <= T
  - Jitter in input-output latencies often handled through buffering
- In control feedback loops are modeled with
  - Periodic sampling
  - Negligible input-output latency
  - Constant input-output latency

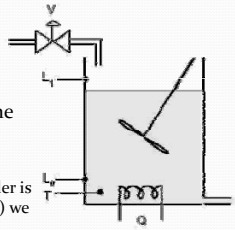
An Example Problem

- Buffer tank for raw material
- Goal 1: Maintain desired temperature
  - PI controller
- Goal 2: Always keep the level between L_o and L_1
  - Event-based sequence control
  - Open V when level below Lo, keep open until level above L1



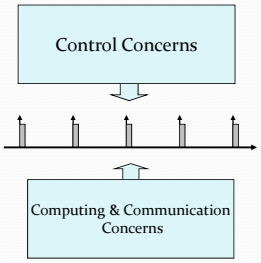
An Example Problem

- The periodically sampled PI-controller is very robust towards temporal non-determinism
  - Jitter in sampling
  - Input-output latencies with jitter
- For the discrete-event controller the deadlines are truly hard
  - E.g. Overflow
  - However, if the discrete-event controller is implemented using sampling (polling) we are back again in the first case
- **Why is it then we use the periodic hard deadline model for these??**



Reasons for time-triggered

- Well defined interface between control and computing community (separation of concerns)
- Simple and deterministic
- Better suited for formal approaches
- Control theory available
  - Sampled Control Theory
- Dependability
- .....



- All excellent reasons!

Reasons against

- Can be rigid and inflexible
- May imply over-provisioning of resources to cater for worst-case scenario → problematic in severely resource-constrained embedded applications
- Can be incompatible with event-based legacy software
- Difficult to achieve exactly in e.g., distributed systems
- Model overly restrictive

- **Also good reasons!**
- **However, alternative implementation techniques cause temporal non-determinism**
  - **Sampling jitter**
  - **Jitter in input-output latencies**

Research Approaches

- Ignore it
  - Far too common!
- Constructive Approach
  - Define new models of computation, implementation techniques, scheduling techniques, etc that overcome the shortcomings
- Analytical Approach
  - Develop new models and analysis techniques that help us decide if the non-determinism is harmful or not

e.g Loosely T-T Approach

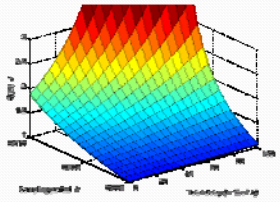
Approximation Theory

Control Performance

- How does temporal non-determinism effect control performance?
- In general,
  - Sampling jitter → ☹
  - Input-output latencies → ☹
  - Jitter in input-output latencies → ☹
  - A short time-varying latency is in most cases better than a longer, but constant, latency
- **Can we get some quantitative measures?**

Quantify Performance: Using Jitterbug

- Statistical analysis
- Quadratic cost functions
- Linear systems

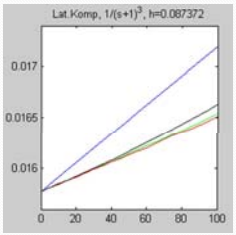


Performance Evaluation

- Batch of typical plant transfer functions
- LQG and PID with and wo delay compensation
- Four different latency distributions
  - Constant = δ_{max}
  - Uniform
  - Normal
  - End-point-distribution
    - Latency equal to 0 or δ_{max} with equal probability

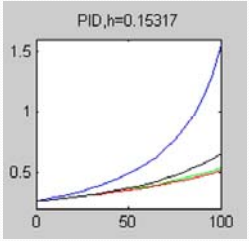
- Only centralized SISO

Typical Performance: LQG



- Designed to minimize the same criterion that is used in the cost function
- The average delay decides performance
- Affine approximation good

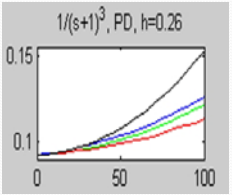
Typical performance: PID



- The average delay decides performance
- Quadratic approximation give better fit

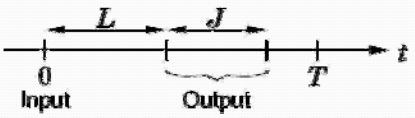
Exception

- For controllers with high gain at high frequencies the end-point-distribution gives the worst performance
  - P(I)D
  - LQG
- Latency jitter can be viewed as a high frequency disturbance



Quantify Performance: Jitter Margin

- Extension to the phase margin / delay margins
- A measure of how much time-varying input-output latency a control loop can tolerate before becoming unstable
- Jitter margin $J_m(L)$: the largest J for which stability can be guaranteed for a constant latency of L



Jitter Margin

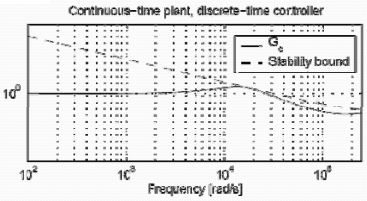
- Graphical frequency interpretation
- Magnitude curve of the Bode diagram of the complementary sensitivity function

$$\left|\frac{P_{\text{alias}}(\omega)K(e^{i\omega})}{1+P_{\text{ZOH}}(e^{i\omega})K(e^{i\omega})}\right| < \frac{1}{\sqrt{J}\left|e^{i\omega}-1\right|}, \quad \forall \omega \in [0,\pi]$$

"Closed Loop System" (complementary sensitivity function)

"Straight Line"

Jitter Margin



- Avoid large resonance peaks
- Not too high bandwidth
- Sufficient high-frequency roll-off
- Has been used to derive tuning rules for PID (Johansson et al, KTH)

Event-Based Control

- What if we relax the assumption that control always should be periodic?
  - Control only when an **event** has occurred, e.g., a threshold crossing
  - Reduced resource utilization
  - Most likely closer to how nature performs feedback
  - Several practical observations have reported that event-based control can perform as good or better than time-based control
  - But, very very little theory (so far)
  - No real understanding for when it is applicable

Event-Based == YES

Event-Based == ??

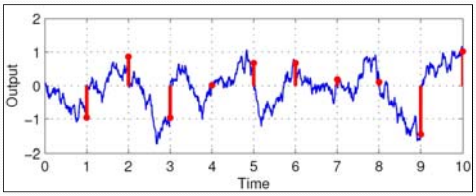
Example: First order system

- Simple problem studied by Åström and Bernhardsson [1999]
- First order process disturbed by white noise

$$dx = axdt + udt + dw$$

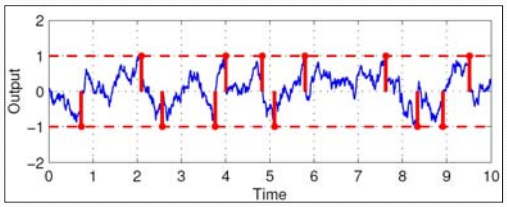
- The controller can "reset" the plant state to zero using impulse control (Dirac control signals)

Periodic Control



- Sampling/control interval: T = 1
- Output variance: V = 1/2

Aperiodic Control



- Event detection threshold: r = 1
- Output variance: V = 1/6

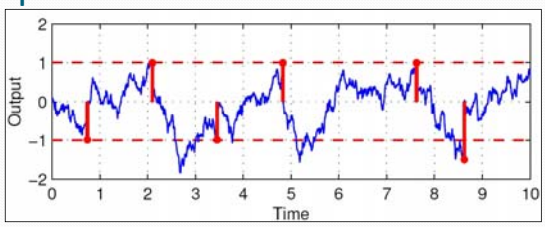
Aperiodic vs Sporadic

Problems with aperiodic control:

- No minimum inter-event time
  - theoretically infinite resource utilization
- Assumes infinitely fast (continuous) sampling

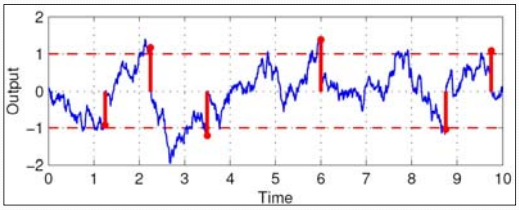
- Alternative: **sporadic control i**ntroduced by Anton Cervin and coworkers
  - Minimum control interval Tc
  - Sampling interval Ts (≤ Tc)

Sporadic Control 1



- Minimum control interval: Tc = 1
- Sampling interval: Ts = 0

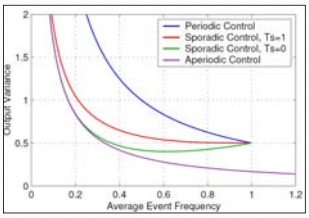
Sproradic Control 2



- Minimum control interval: Tc = 1
- Sampling interval: Ts = 0.25

Comparison

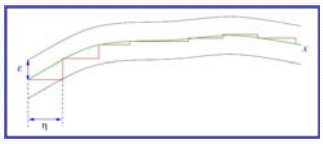
- Compute stationary probability distribution as a function of threshold → output variance, average event frequency



Extensions and Limitations

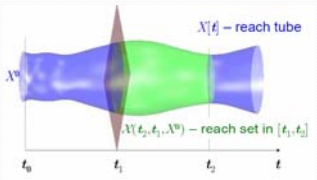
- Extensions
  - Input-output latencies with jitter
  - Measurement noise
  - Load disturbances
- Many unsolved problems:
  - What are the suitable problem formulations / applications?
  - When does event-based control pay off (performance vs design time)
  - Controller synthesis for higher-order plants
  - Implementation/real-time scheduling

Observation



- Approximation theory for embedded control by Paul et al
- Captures effects of sampling, delays, jitter etc
- Appears to have much in common with the set-valued approach to control

Observation



- Set valued initial values and disturbances
- Differential inclusions rather than differential equations
- Aubin, Kurzhanski, Varayia, Mitchell, ...
- Something for the future?

Breaking the wall