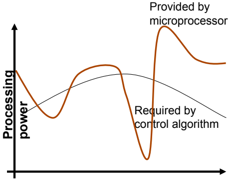
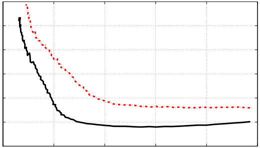
An Intuitive Algorithm for Control with Limited Communication and Processing Resources

Daniel E. Quevedo

Department of Electrical Engineering (EIM-E)
Paderborn University, Germany
dquevedo@ieee.org

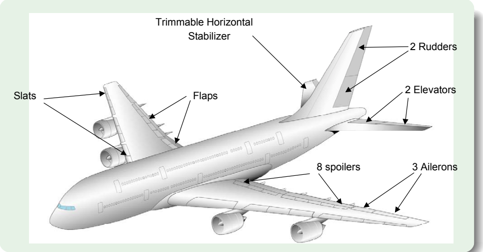
March 2016





Motivating Example:

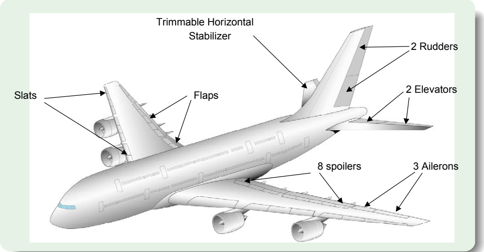
Communication and Processing in an A380 Aircraft



- The Avionics Full Duplex Switched Ethernet Network serves as a backbone to low level networks, e.g., based on CAN.
- Fly-by-wire requires 500 km of cables and many interconnects. This adds to weight, cost and possible fire hazards.
- Fly-by-wireless is being considered.
  Due to dropouts and delays, communication links are not transparent and need to be taken into account in the design.

Motivating Example:

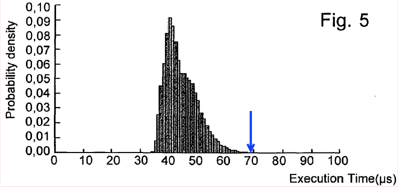
Communication and Processing in an A380 Aircraft

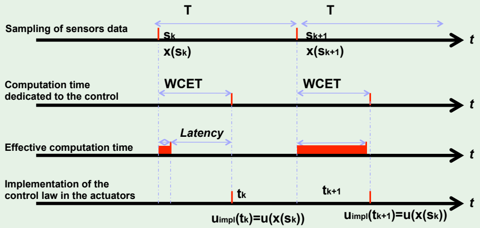


- Eight control computers (and a back-up module) need to divide their attention to various loops, including
  - attitude control,
  - direction of flight,
  - engine controls.
- Processing power available for, and required by, each loop is time-varying.

Traditional Control Design: hard real-time

Execution time distribution

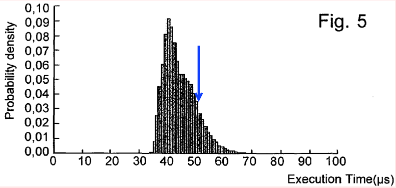


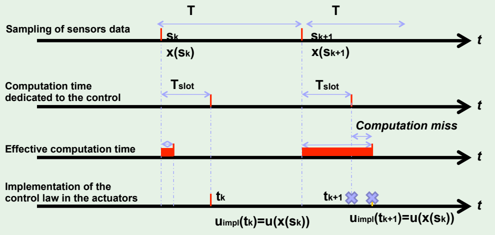


- By using a deadline larger than the worst-case execution time (WCET) a deterministic control loop is obtained.
- As processing architectures become more complex, execution time distributions tend to have longer tails. Thus:
  - The WCET becomes more difficult to determine.
  - Computing and communication resources are idle more often: Inherent conservatism leads to oversized and heavy components.

Control with short deadlines (2013 Airbus patent)

Execution time distribution



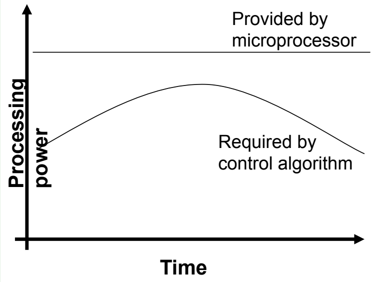


- Shorter deadlines lead to a stochastic loop which uses processor and communication resources more efficiently.
- At times, processing resources are insufficient for evaluating the control policy.
- Once a maximum number of consecutive deadline misses is reached, an auxiliary processor is called upon.

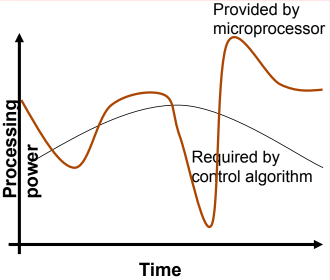
More general situation:

Processing Power Mismatch in Networked and Embedded Systems

Traditional



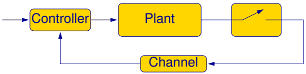
Networked/Embedded



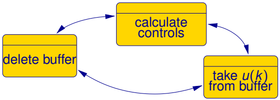
- The traditional assumption about the processor always being able to execute the control algorithm during the available computation slot may break down.

This talk

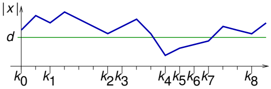
1. presents a modeling framework for closed-loop control, when processing and communication resources are limited,



2. describes an intuitive algorithm to synthesise such loops,



3. illustrates how stability can be analysed using random-time state-dependent drift conditions.



Some previous works of Interest

Stochastic Networked Control

- Hespanha, Dahleh, Sinopoli, Teel, Heemels, etc

Event-triggered Estimation and Control

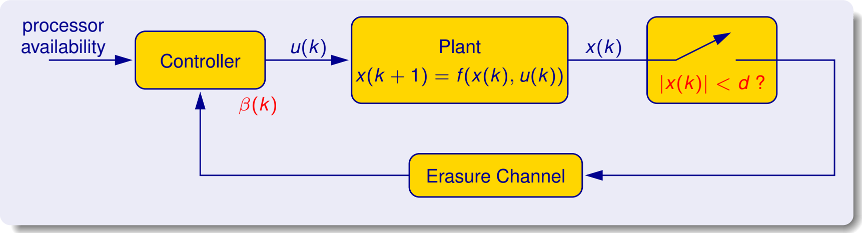
- Transmit and compute only when an event occurs: Åström, Tabuada, Lemmon, Blind, etc.

Control with limited Processing Resources

- Computation-performance tradeoffs for MPC: McGovern, Cervin
- Allow for deadline misses: Seuret
- “Anytime Control” (control is refined on-line, calculations can be terminated at any time): Bhattacharya, Greco, Bicchi

Outline

System Model



- The quantity d is a design parameter which trades communication channel utilization for control performance.
- Transmission between sensor and controller node is through a delay-free link with dropouts:

$$\beta(k) = \begin{cases} 0 & \text{if } x(k) \text{ is received with errors (a dropout occurs),} \\ 1 & \text{if } x(k) \text{ is received error-free,} \\ 2 & \text{if the sensor did not transmit at time } k \text{ (i.e., } |x(k)| < d\text{).} \end{cases}$$

- We are interested in a situation, where a "good" state-feedback controller $\kappa\colon \mathbb{R}^n \to \mathbb{R}^p$ has been pre-designed.
- Processing resources for control may, at times, be insufficient to evaluate κ within the pre-allocated time-slot of length τ.
- A direct implementation of the control policy κ, yields the

Baseline Algorithm

$$u(k) = \begin{cases} \kappa(x(k)), & \text{if } \beta(k) = 1 \text{ and } \kappa(x(k)) \text{ was evaluated} \\ & \qquad \text{between times } kT \text{ and } kT + \tau, \\ \mathbf{0}_p, & \text{otherwise,} \end{cases}$$

where $u(k)$ with $k \in \mathbb{N}_0$ denotes the plant input which is applied during the interval $[kT + \tau, (k+1)T + \tau)$.

Numerical Example

Plant model and controller

- Plant model with i.i.d. disturbance distributed as $\mathcal{N}(0, 1)$:

$$x(k+1) = -x(k) + 0.1\sin(x(k)) + u(k) + w(k), \quad x(0) = 20.$$

- Control policy is taken as:

$$\kappa(x) = x - 0.1\sin(x) + \rho|x|, \quad \rho = 0.9.$$

Resources

$$\mathbf{Pr}\{\beta(k) = 1 \,|\, |x(k)| \geq d\} = 0.5$$
$$\mathbf{Pr}\{\text{processor is available} \,|\, \beta(k) = 1\} = 0.8$$

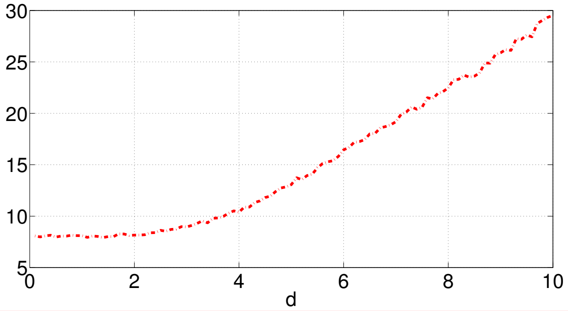
The closed loop is characterised by:

$$x(k+1) = \begin{cases} \rho|x(k)| + w(k), & \text{if processor is available and } \beta(k) = 1, \\ -x(k) + 0.1\sin(x(k)) + w(k), & \text{otherwise.} \end{cases}$$

Empirical Cost

$$\frac{1}{500}\left(\sum_{k=201}^{700} x^2(k)\right),$$

averaged over 1000 realisations.

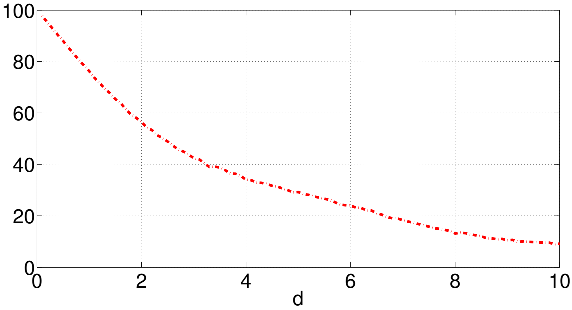


Larger thresholds give a worse control performance!

Channel Utilisation (%)

$$\frac{\text{Total number of time steps at which } \beta(k) \neq 2}{\text{Total number of time steps}},$$

averaged over 1000 realisations.



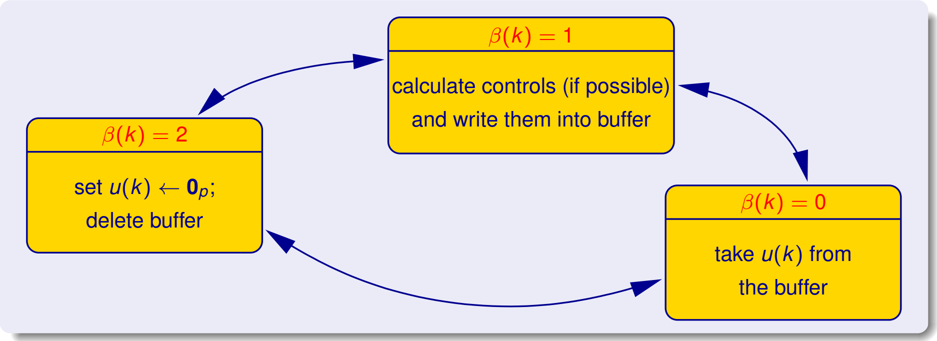
Larger thresholds lead to less communication uses!

Whilst the baseline algorithm is simple, it is by no means clear that it cannot be outperformed by more elaborate control formulations.

- We will next present an intuitive control algorithm.
- The purpose is to make efficient use of the communication and processing resources available.
- The algorithm calculates sequences of tentative future inputs.
- These are stored in a local buffer.
- Buffered values may be used when, at some future time steps, the processor availability precludes any control calculations, or a dropout occurs ($\beta(k) = 0$).

Outline

Event-triggered Anytime Control Algorithm



- If $\beta(k) = 1$, then $x(k)$ is used to calculate $N(k) \in \{0, 1, \dots, \Lambda\}$ tentative control values using κ:

$$u_0(k) = \kappa(x(k))$$
$$u_1(k) = \kappa\big(f(x(k), u_0(k))\big), \quad \text{etc.}$$

- This sequence is stored in a local buffer of size Λ. Its contents may be used when the processor is unavailable or when $\beta(k+\ell) = 0$.

Comments

- The algorithm does not require prior knowledge of processor availability. Therefore, the control task can be preempted.
- The buffer state $b(k)$ provides the current plant input, $u(k) = b_1(k)$.
- If $N(k) > 1$, then $b(k)$ also contains suggested future inputs.
- If the buffer runs out of tentative plant inputs, then $u(k) = \mathbf{0}_p$.

We introduce the

effective buffer length

$$\lambda(k) = \begin{cases} N(k) & \text{if } N(k) \geq 1, \\ \max\{0, \lambda(k-1) - 1\} & \text{if } N(k) = 0 \text{ and } \beta(k) \in \{0, 1\}, \\ 0 & \text{if } \beta(k) = 2. \end{cases}$$

Example

Suppose that $\Lambda = 4$ and that $\{N(0), N(1), N(2), N(3)\} = \{4, 0, 1, 2\}$.

The Anytime algorithm provides

$$\{b(0), b(1), b(2), b(3)\} = \left\{ \begin{bmatrix} u_0(0) \\ u_1(0) \\ u_2(0) \\ u_3(0) \end{bmatrix}, \begin{bmatrix} u_1(0) \\ u_2(0) \\ u_3(0) \\ \mathbf{0}_p \end{bmatrix}, \begin{bmatrix} u_0(2) \\ \mathbf{0}_p \\ \mathbf{0}_p \\ \mathbf{0}_p \end{bmatrix}, \begin{bmatrix} u_0(3) \\ u_1(3) \\ \mathbf{0}_p \\ \mathbf{0}_p \end{bmatrix} \right\}$$

$\{\lambda(0), \lambda(1), \lambda(2), \lambda(3)\} = \{4, 3, 1, 2\}$, and plant inputs

$$\{u(0), \dots, u(3)\} = \{\kappa(x(0)), \kappa\big(f(x(0), \kappa(x(0)))\big), \kappa(x(2)), \kappa(x(3))\}.$$

If the baseline-algorithm is used, then

$$\{u(0), u(1), u(2), u(3)\} = \{\kappa(x(0)), \mathbf{0}_p, \kappa(x(2)), \kappa(x(3))\}.$$

Outline

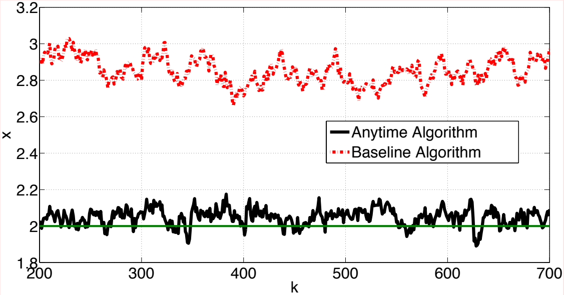
Numerical Example

Resources

- Suppose that $\Lambda = 4$ and that

$$\mathbf{Pr}\{\beta(k) = 1 \,|\, |x(k)| \geq d\} = 0.5$$

$$\mathbf{Pr}\{N(k) = j \,|\, \beta(k) = 1\} = 0.2, \quad j \in \{0, 1, 2, 3, 4\}$$

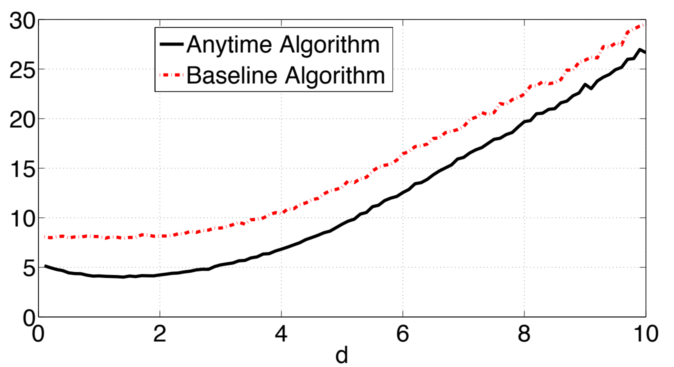


- State trajectory with $d = 2$.
- The anytime control algorithm outperforms the baseline controller.

Empirical Cost

$$\frac{1}{500}\left(\sum_{k=201}^{700} x^2(k)\right),$$

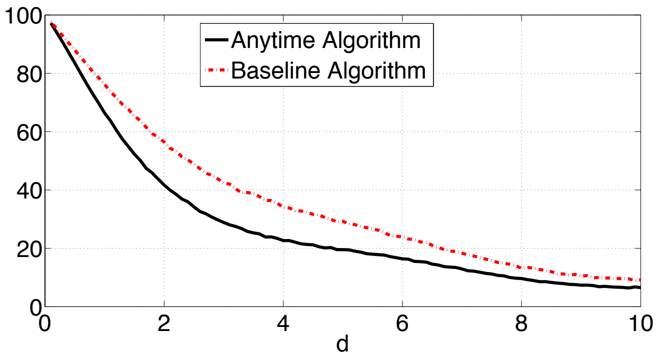
averaged over 1000 realisations.



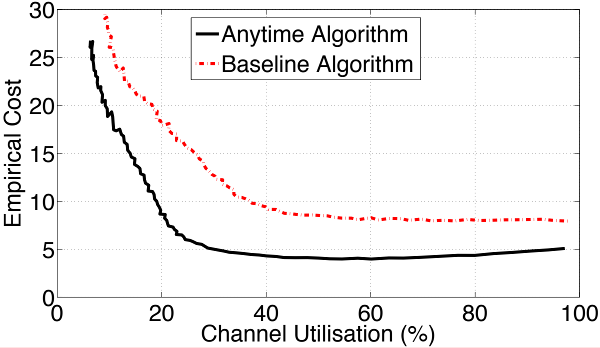
Channel Utilisation (%)

$$\frac{\text{Total number of time steps at which } \beta(k) \neq 2}{\text{Total number of time steps}},$$

averaged over 1000 realisations.



Empirical Cost versus Channel Utilisation



Performance Gains

For a given transmission rate, the proposed anytime control algorithm reduces the empirical cost by approximately 40-50%.

Outline

Standing Assumptions

Globally Stabilising Nominal Controller

There exist functions $V\colon \mathbb{R}^n \to \mathbb{R}_{\geq 0}$, $\varphi_1, \varphi_2 \in \mathscr{K}_\infty$,[a] a constant $\rho \in [0, 1)$, and a control policy $\kappa\colon \mathbb{R}^n \to \mathbb{R}^p$, such that

$$\begin{aligned} \varphi_1(|x|) \leq V(x) \leq \varphi_2(|x|), \\ V(f(x, \kappa(x))) \leq \rho V(x), \quad \forall x \in \mathbb{R}^n. \end{aligned}$$

[a] A function $\varphi\colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is of *class-*$\mathscr{K}_\infty$ ($\varphi \in \mathscr{K}_\infty$), if it is continuous, zero at zero, strictly increasing, and unbounded.

Open-loop bound

With V as above, there exists $\alpha \in \mathbb{R}_{\geq 0}$ such that[a]

$$V(f(x, \mathbf{0}_p)) \leq \alpha V(x), \quad \forall x \in \mathbb{R}^n.$$

[a] For the numerical example described before, we have $\alpha = 1.1$.

Processor availability is i.i.d.

The process $\{N\}_{\mathbb{N}_0}$ has conditional probability distribution

$$\mathbf{Pr}\{N(k) = j \,|\, \beta(k) = 1\} = p_j, \quad j \in \{0, 1, 2, \dots, \Lambda\},$$

where $p_j \in [0, 1)$ are given.
For other realizations of $\beta(k)$, no plant inputs are calculated:

$$\mathbf{Pr}\{N(k) = 0 \,|\, \beta(k) \in \{0, 2\}\} = 1.$$

Packet dropouts are i.i.d.

The transmissions are Bernoulli with packet transmission success probability

$$\mathbf{Pr}\{\beta(k) = 1 \,|\, |x(k)| \geq d\} = q.$$

Outline

The Baseline Algorithm

- The closed loop is characterised by:

$$x(k+1) = \begin{cases} f(x(k), \kappa(x(k))), & \text{if } N(k) \geq 1, \\ f(x(k), \mathbf{0}_p), & \text{if } N(k) = 0. \end{cases}$$

- We denote via p_0 the probability that the controller is unable to calculate any control input, despite $x(k)$ being available.

Stochastic Stability with the Baseline Algorithm

Suppose that $\mathbf{E}\{\varphi_2(|x(0)|)\} < \infty$ and that

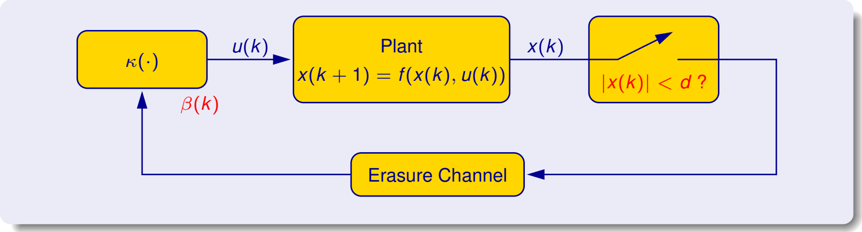
$$\Gamma \triangleq (1-q)\alpha + q\big(p_0\alpha + (1-p_0)\rho\big) < 1.$$

Then there exist finite γ and μ such that

$$\mathbf{E}\{\varphi_1(|x(k)|)\} \leq \gamma\Gamma^k + \mu, \quad \forall k \in \mathbb{N}_0.$$

If $d = 0$, then $\mu = 0$.

Special case: event-triggering with an erasure channel



- If the processor is always available ($p_0 = 0$), then the sufficient condition for stochastic stability reduces to:

$$\Gamma \triangleq (1-q)\alpha + q\rho < 1,$$

where:

- q is the transmission success probability,
- α is the open-loop bound on the plant dynamics, and
- ρ is the closed-loop contraction factor ensured by the control law κ.

Sketch of proof

- The process $\{x(k)\}$, $k \in \mathbb{N}_0$ is Markovian.
- Using the assumptions and writing x for $x(0)$, we have:

$$\begin{aligned}\mathbf{E}\{V(x(1))\,|\,x\} &= \sum_{j=0}^{2} \mathbf{E}\{V(x(1))\,|\,x, \beta(0)=j\}\mathbf{Pr}\{\beta(0)=j\,|\,x\} \\ &< \Gamma V(x) + (\alpha - \Gamma)\varphi_2(d), \quad \forall x.\end{aligned}$$

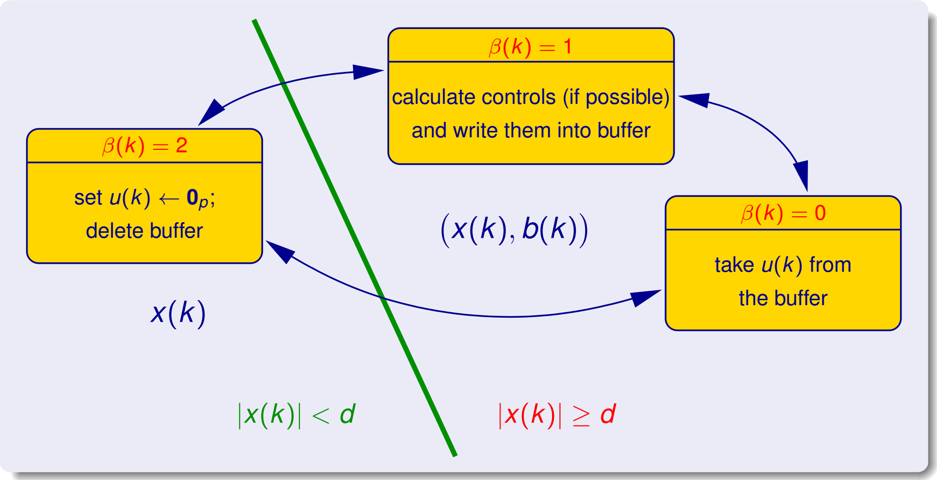
- Thus, using the Markov property, we obtain

$$\mathbf{E}\{\varphi_1(|x(k)|)\,|\,x\} \leq \Gamma^k V(x) + \frac{(\alpha-\Gamma)\varphi_2(d)}{1-\Gamma} < \infty, \quad \forall k \in \mathbb{N}_0.$$

- Taking expectation with respect to the distribution of the initial state $x(0)$ establishes the result.

Outline

Preliminaries



- Due to buffering, $\{x(k)\}$, $k \in \mathbb{N}_0$ is not a Markov process.
- This complicates the analysis significantly.

Preliminaries

- To study stability of the event-based anytime algorithm, we will develop a state-dependent random-time drift condition of the form:

$$\mathbf{E}\{V(x(k_{i+1}))\,|\,x(k_i) = \chi\} \leq D + \Omega V(\chi), \quad \Omega < 1,$$

where $\{k_0, k_1, k_2, \dots\}$ are special random time instants.

- If $\{x(k_i)\colon i \in \mathbb{N}_0\}$ is Markovian, then the above would ensure exponential boundedness at the instants k_i:

$$\mathbf{E}\Big\{V(x(k_i))\,|\,x(k_0) = \chi\Big\} \leq \Omega^i V(\chi) + \frac{D}{1-\Omega}, \quad \forall i \in \mathbb{N}_0.$$

- Depending on
  1. system behaviour in-between instants k_i
  2. the distribution of $\{k_{i+1} - k_i\}$

  boundedness at all instants $k \in \mathbb{N}_0$ may follow.

The Randomly Sampled Process

- We begin our analysis, by denoting the random time steps where the buffer is empty via

$$\mathcal{K} = \{k_i\}, \quad i \in \mathbb{N}_0,$$

where

$$k_{i+1} = \inf\left\{k \in \mathbb{N} \colon k > k_i,\ \lambda(k) = 0\right\}, \quad k_0 = 0.$$

- We also describe the amount of time steps between consecutive elements of $\mathcal{K}$ via

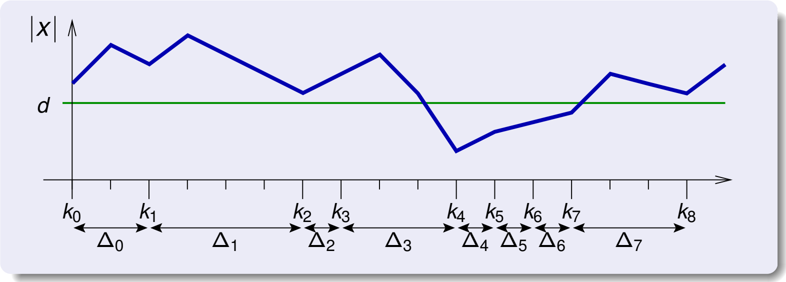
$$\Delta_i \triangleq k_{i+1} - k_i, \quad i \in \mathbb{N}_0$$

- The following property of the randomly sampled process x is key:

Lemma

The plant state sequence at the time steps $k_i \in \mathcal{K}$, namely $\{x(k_i) \colon k_i \in \mathcal{K}\}$, is Markovian.

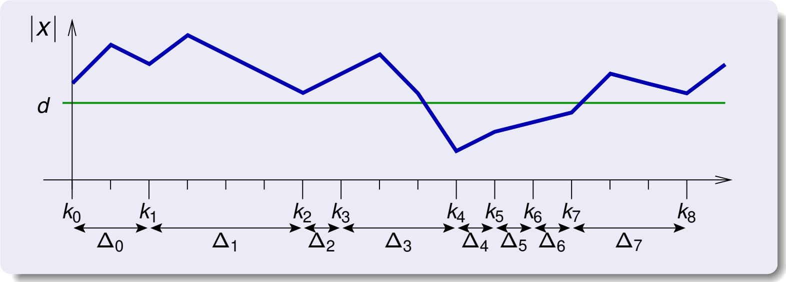
System Behaviour



- In the present disturbance-free case, the plant inputs are simply given by

$$
\begin{aligned}
u(k_i) &= \mathbf{0}_p, \quad \forall k_i \in \mathcal{K} \\
u(k_i + \ell) &= \kappa(x(k_i + \ell)), \quad \forall \ell \in \{1, \ldots, \Delta_i - 1\}.
\end{aligned}
$$

Evolution of $V(x(k_i))$

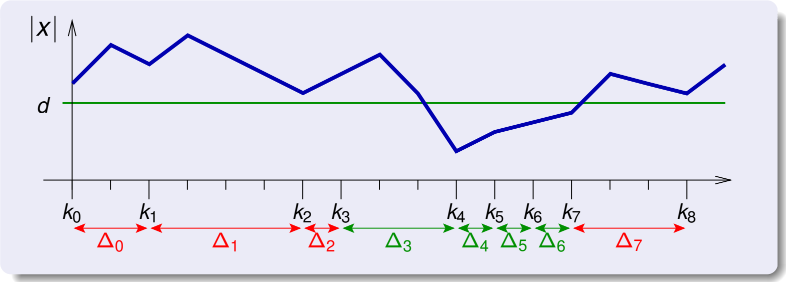


- It is then easy to see that

$$\mathbf{E}\{V(x(k_{i+1}))\,|\,x(k_i)=\chi,\Delta_i=\delta\}\leq \alpha\rho^{\delta-1}V(\chi),\quad \forall\chi\in\mathbb{R}^n.$$

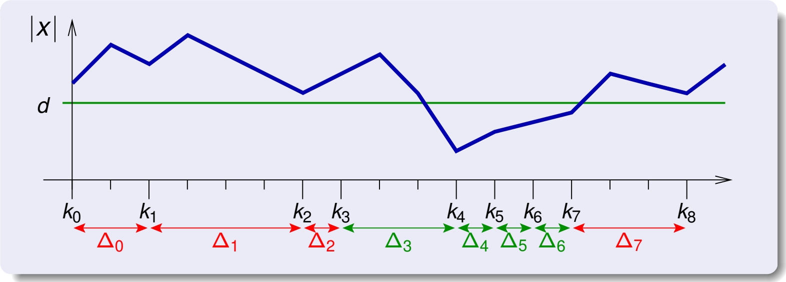
- Unfortunately, due to the event-triggering mechanism, the distribution of $\{\Delta_i\}_{i\in\mathbb{N}_0}$ depends on $x(k_i)$ and is difficult to characterise.
- Hence establishing a suitable drift condition is not so easy!

Conditioning Events



- It turns out to be convenient to distinguish between the two cases:
  1. the buffer is emptied due to lack of resources ($\beta(k_{i+1}) \in \{0, 1\}$), and
  2. it is emptied triggered by the plant state being in the desired region.
- Only the first case influences stability conditions.

Finding an Upper-bound on the Drift



- Accordingly, one can condition on $\beta(k_{i+1})$ and use the law of total expectation to show that

$$\mathbf{E}\{V(x(k_{i+1}))\,|\,x(k_i)\} \leq \varphi_2(d) + \mathbf{E}\{V(x(k_{i+1}))\,|\,x(k_i), \beta(k_{i+1}) \neq 2\}.$$

- Now, we can condition on Δ_i to
  1. establish exponential boundedness at the instants $k_i \in \mathcal{K}$, and
  2. upper-bound $\mathbf{E}\{\sum_{k=k_i}^{k_{i+1}-1} V(x(k))\,|\,x(k_i), \beta(k_{i+1}) \neq 2\}$.
- This leads to...

Main Result

Stochastic Stability with the proposed Algorithm

Suppose that $\mathbf{E}\{\varphi_2(|x(0)|)\} < \infty$ and that

$$\Omega \triangleq \sum_{\delta \in \mathbb{N}} \alpha \rho^{\delta - 1} \mathbf{Pr}\{\Delta_i = \delta \,|\, \beta(k_{i+1}) \neq 2\} < 1.$$

Then there exist finite γ, μ such that

$$\max_{k \in \{k_i, k_i+1, \dots, k_{i+1}-1\}} \mathbf{E}\{\varphi_1(|x(k)|)\} \leq \gamma \Omega^i + \mu, \quad \forall i \in \mathbb{N}.$$

If $d = 0$, then $\mu = 0$.

- The conditional distribution

$$\mathbf{Pr}\{\Delta_i = \delta \,|\, \beta(k_{i+1}) \neq 2\} = \mathbf{Pr}\Big\{\Delta_i = \delta \,\Big|\, |x(k_{i+1})| \geq d\Big\}.$$

depends only on p_j and q and can be easily characterised using finite Markov Chain methods (“first return times to $\lambda = 0$”).

Comments

- Our result establishes a condition on
  - plant,
  - control law,
  - channel, and
  - processor availability

  which ensures stochastic stability of the event-based anytime control loop.
- The analysis can be extended to situations where processor and communication resources are not i.i.d., but correlated.
- Under continuity assumptions, related stability conditions can be derived for plant models with disturbances.

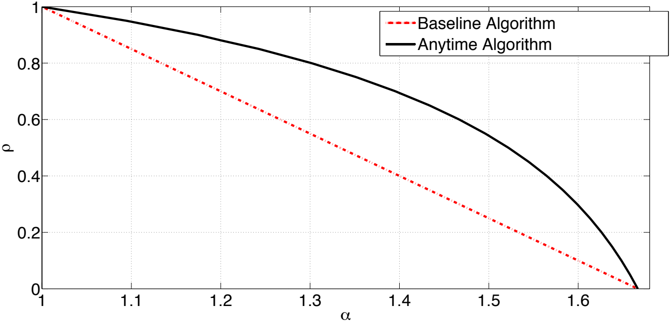
Outline

Resources

- Suppose that the buffer size is set to $\Lambda = 4$ and that

$$\mathbf{Pr}\{\beta(k) = 1 \,|\, |x(k)| \geq d\} = 0.5$$

$$\mathbf{Pr}\{N(k) = j \,|\, \beta(k) = 1\} = 0.2, \quad j \in \{0, 1, 2, 3, 4\}$$

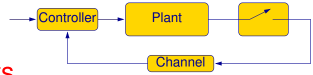


The stability regions in the α-ρ plane established for the anytime control algorithm are larger than those for the baseline controller.

Outline

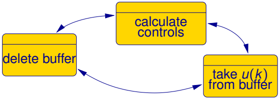
Conclusions

- We have presented a framework for the study of control when processor availability and communication resources are random.
  - The sensor node is event-triggered and transmits data using a communication link prone to dropouts.

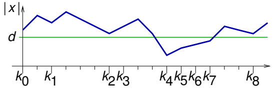


  - The control algorithm is executed with a processor that can provide only time-varying and a priori unknown resources.

- To better utilise the processor, the plant inputs are calculated by an algorithm that provides sequences.



- For general non-linear systems, we used stochastic Lyapunov methods to obtain sufficient conditions for stability.



Future Work

Many research problems remain open, e.g.,

- characterising closed loop performance,
- studying event-based transmission strategies with memory, and
- developing processor scheduling and cooperation strategies.

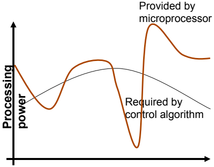
Further Reading

1. D. E. Quevedo, V. Gupta, W.-J. Ma and S. Yüksel, "Stochastic Stability of Event-Triggered Anytime Control," *IEEE Trans. Automat. Contr.*, December 2014
2. D. E. Quevedo, W.-J. Ma and V. Gupta, "Anytime Control using Input Sequences with Markovian Processor Availability," *IEEE Trans. Automat. Contr.*, February 2015
3. B. Demirel, V. Gupta, D. E. Quevedo and M. Johansson,"On the trade-off between control performance and communication cost in event-triggered control," *IEEE Trans. Autom. Contr.*, under review.

Acknowledgements

- A/Prof. Vijay Gupta
  Department of Electrical Engineering
  University of Notre Dame, USA
- Dr. Wann-Jiun Ma
  Department of Mechanical Engineering and
  Materials Science Engineering
  Duke University, USA
- A/Prof. Serdar Yüksel
  Department of Mathematics and Statistics
  Queen's University, Ontario, Canada

PADERBORN UNIVERSITY
The University for the Information Society



Thank You!