Optimization of Hybrid Control Systems in Manufacturing*

Michael Fall

Institute of Automatic Control Engineering
Technische Universität München

michael.fall@mytum.de

*D. Pepyne and C. Cassandras: Optimal Control of Hybrid Systems in Manufacturing
Proceedings of the IEEE, Vol. 88, No.7, July 2000

Content

Problems related to Manufacturing Processes

- Consider the manufacturing process of a metal-making company:
- Metal strips undergo various operations during the production process (rolling, milling, machining metals,...)

Supervisory control: which operations?
sequence of operations?

- Example process: oven heating with a defined *heating profile*

1. Slowly heating of ingots to a desired temperature
2. Holding the metal-strips at a certain temperature level
3. Controlled cooling (annealing)

} **Time consuming** processes to achieve a certain **quality**

Process related control: When to switch operation times?

Integration of **process control** into the plant-wide **scheduling.**

A Hybrid System Framework for Manufacturing

How to achieve the integration of process control into plant-wide scheduling?

- Suitable *process model* required
  - Trade off job completion times vs. quality aspects
  - Applicable to various processes
  - Deal with discrete events and continuous states

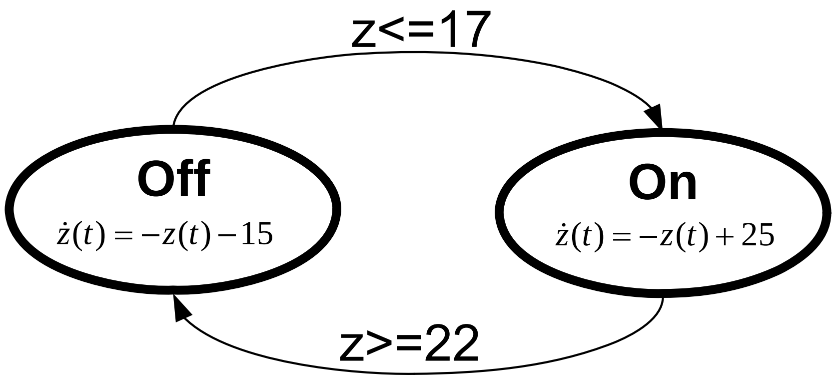
Solution Approach: Introduction of a *Hybrid System Framework*

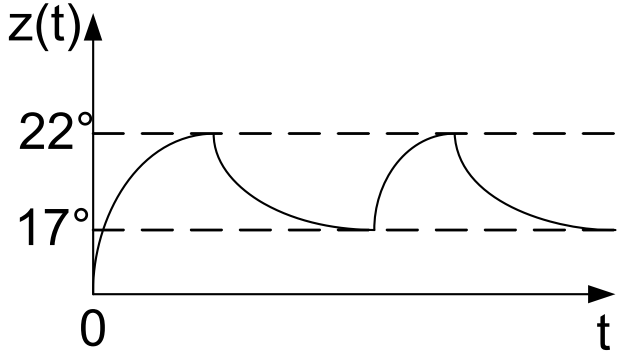
- Generalization:
  - Representation of certain tasks <-> „Jobs“
  - Devices to process on tasks <-> „Servers“
- *Hybrid* nature of the system
  - Description of physical characteristic (shape, functionality, quality)
  - Description of process start and stop times

General remarks on Hybrid Systems

- Example: A simple thermostat as a hybrid system

  Hybrid System: combination of *event-driven* with *time-driven dynamics*
  - Discrete states: Q={0,1}
  - Transitions depending on continuous variables
  - In each state: continuous dynamics and constraints z $\in IR^N$

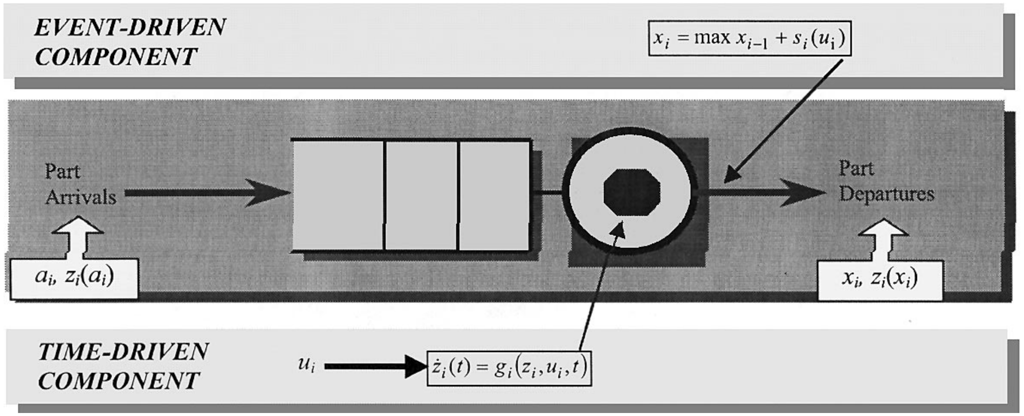




- In General, various types of modeling framework for hybrid systems:
  - Queuing system framework
  - Extension of event-driven models to allow time-driven activities

Modeling of a Single-Stage Manufacturing Process

- Representation of a manufacturing process as a *single-server queuing system*



Structure
- Infinite storage capacity
- Non-preemptive server

Queuing discipline
- First-Come-First-Served Principle (FCFS)

- Queuing system dynamics:

Physical state: Time-driven differential equations during job processing

Temporal state: Discrete event dynamics to describe start and stop times

Hybrid model

Goal: To formulate and solve optimal control problems that trade off cost on the physical and temporal states

Control Policy for a Hybrid System Framework

Control Policy:

Determine how the jobs are being processed through the system optimally

(Assume: Job sequence / job arrival times assigned by an external source)

- Sub-problems that need to be solved:

  1. Compute control trajectories for optimally steering the physical system state — *-> nonlinear optimal control*

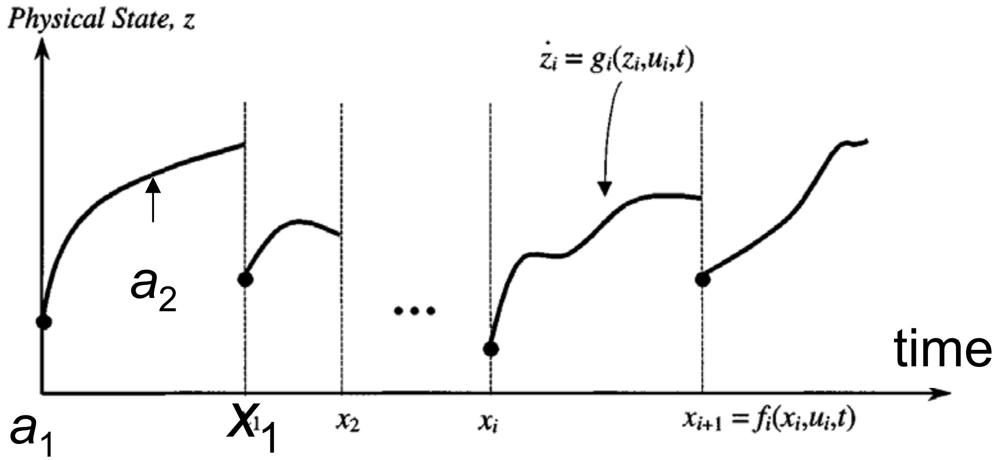
  2. Choose the optimal processing time for each job — *-> discrete-event dynamic system performance*

  3. Determine the order of job-processing
  4. Consider the sequence of servers for each job — *-> scheduling methods*

- All 4 subproblems are tightly coupled together in a hybrid system

Interpretation of the Hybrid System Framework

Discrete event system with time-driven dynamics:



- Time driven dynamics:

$$\dot{z}_i(t) = g_i(z_i, u_i, t)$$
$$z_i(\tau_i) = \zeta_i$$

$z_i(t)$: dynamics of the physical states

u_i:control variable -> time-**in**dependent

ζ_i:initial state; τ_i:processing start time

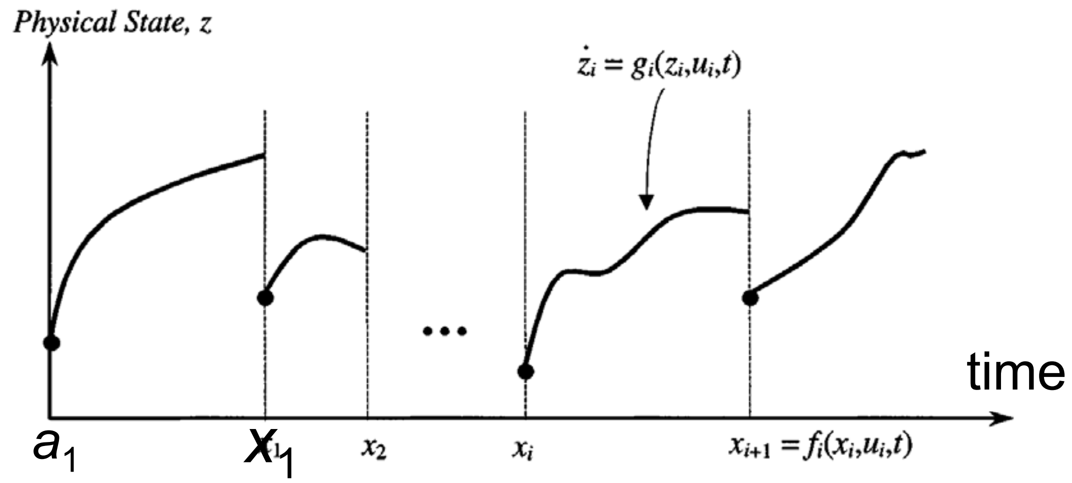
- Event-driven dynamics: evolution of the temporal states

$$x_i = \tau_i + s_i(u_i) = \max(x_{i-1}, a_i) + s_i(u_i)$$

$x_i(t)$: job completion times

$s_i(u_i)$:processing time

Interpretation of the Hybrid System Framework



a_i: job arrival times

x_i: job completion times

- Exogenous (uncontrolled) arrival events, controlled departure events
- Each job must be processed until it reaches a certain quality level
  „Stopping rule“:

$$s_i(u_i) = \min[t \geq 0 : z_i(\tau_i + t) = \int_{\tau_i}^{\tau_{i+t}} g_i(s,u_i,t)ds + \zeta_i \in \Gamma_i]$$

Γ_i: desired quality „level“

- Consider Job1:
  - Job arrival time: a_1
  - Job removal from the server: x_1

during Interval (a_1, x_1) job execution according to:

$$\dot{z}_1(t) = g_1(z_1, u_1, t)$$

Formulation of the Optimal Control Problem

- Conflicting optimization goals:
  - Quality aspects to satisfy customer demands
  - Job completion deadlines

  } Hybrid system framework: Time/Quality tradeoffs

- Optimal Control objective:

  Choose a control policy $\pi=\{u_1,\ldots, u_N\}$ to minimize an objective cost function:

$$\min_{\pi} J = \sum_{i=1}^{N} [\Theta_i(u_i) + \Psi_i(x_i)]$$

J: cost function

Θ_i: cost on control u_i

Ψ_i: cost on job completion x_i

- Multistage optimization problem
- No explicit cost on $z_i(t)$, but the stopping rule $z_i(t)= \Gamma_i$ counts!

Formulation of the Optimal Control Problem

Class 1 problems:

$$\min_{\pi} J = \sum_{i=1}^{N} [\Theta_i(u_i) + \Psi_i(x_i)]$$

- control $u_{(i)}$ is interpreted as the processing time
- $J(\Theta_i, \Psi_i)$ trades off quality vs. Job completion times
- Conditions:
  - Θ_i, Ψ_i : strictly convex, monotonically decreasing
  - $s_i(.)$ is linear with $s_i(u_i)=\alpha u_i$
- Example:

$$s_i(u_i) = u_i$$

$$\Theta_i(u_i) = \frac{1}{u_i}$$

$$\Psi_i(x_i) = (x_i - \delta_i)^2$$

u_i: processing time

x_i: job completion time

δ_i: due date for each job

  - Physical state z_i: interpreted as the job-quality
  - Cost on poor quality + cost on missing the due-date

Formulation of the Optimal Control Problem

Class 2 problems:

$$\min_{\pi} J = \sum_{i=1}^{N} [\Theta_i(u_i) + \Psi_i(x_i)]$$

- control $u(i)$ is interpreted as the effort applied to a job
- $J(\Theta_i, \Psi_i)$ trades off job completion times Θ_i vs. processing speed
- Conditions:
  - Ψ_i strictly convex, monotonically increasing
  - $s_i(.)$ is strictly convex, monotonically decreasing
- Example:

$$s_i(u_i) = \frac{q}{u_i}$$

$$\Theta_i(u_i) = u_i^{2}$$

$$\Psi_i(x_i) = \begin{cases} 0, & x_i < \delta_i \\ (x_i - \delta_i)^2, & x_i \geq \delta_i \end{cases}$$

q: desired quality level

u_i: ...e.g. energy

x_i: job completion time

δ_i: due date for each job

  - Quadratic cost on the effort applied to the job (typical approach) + penalizing tardiness

Analysis of the Optimization Problem

Basic variational calculus techniques:

- General Form of the cost function for a discrete-time optimal control problems

$$J(x,\lambda,u)=\sum_{i=1}^{N}\{L_i(x_i,u_i)+\lambda_i[\max(x_{i-1},a_i)+s_i(u_i)-x_i]\}$$

λ: N-dim. vector for the co-state

- Necessary Conditions for Optimality (maximum principle):
  - Stationary condition: $$\frac{\partial J}{\partial u_i}=0 \Rightarrow \frac{\partial L_i(x_i,u_i)}{\partial u_i}+\lambda_i\frac{ds_i(u_i)}{du_i}=0$$
  - State equation: $$\frac{\partial J}{\partial \lambda_i}=0 \Rightarrow x_i=\max(a_i,x_{i-1})+s_i(u_i)$$
  - Co-state equation: $$\frac{\partial J}{\partial x_i}=0 \Rightarrow \lambda_i=\frac{\partial L(x_i,u_i)}{\partial x_i}+\lambda_{i+1}\frac{d\max(x_i,a_{i+1})}{dx_i}$$

Discussion of possible solutions on the Optimization Problem

- Bellmann Principle / Dynamic Programming (DP)
  - Algorithm based on recursion and memorization
  - Enormous computational effort to search over the whole policy space for jobs *i=1...N*
- Two-point boundary-value problem (TPBVP):
  - Nondifferentiability introduced by event-generation mechanism
  - Consideration of the max function:

$$\frac{\partial J}{\partial x} = 0 \Rightarrow \frac{d}{dx_i}\max(x_i, a_{i+1}) = \begin{cases} 0, \; if \;\; x_i < a_{i+1} \\ 1, \; if \;\; x_i > a_{i+1} \end{cases}$$

a_i: job arrival times

x_i: job completion times

- First order approximations might end-up in a local minimum

Introduction of Nonsmooth Optimization with Lipschitz-continuous functions.

Example for a Nonsmooth Cost Function

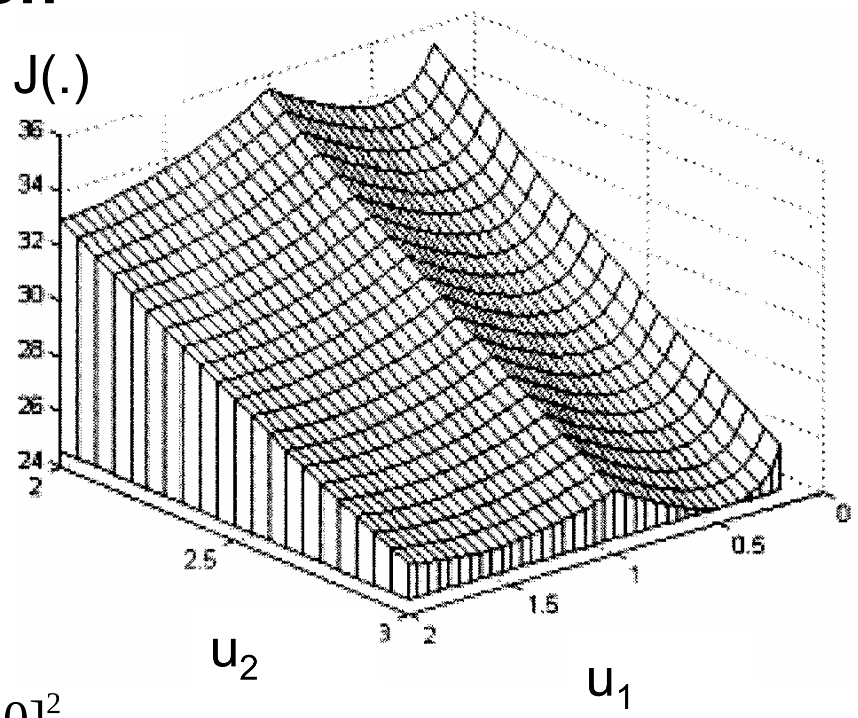
- Class-1 Example with N=2

$$\min_{\pi} J = \sum_{i=1}^{N} [\Theta_i(u_i) + \Psi_i(x_i)]$$

$$\Theta_1(u_1) = \frac{1}{u_1}; \quad u_2(u_2) = \frac{1}{u_2};$$

$$\Psi_1(x_1) = x_1^2, \quad \Psi_2(x_2) = (x_2 - 30)^2$$

$$J(u_{1,}u_2) = \frac{1}{u_1} + \frac{1}{u_2} + (2+u_1)^2 + [\max(2+u_1,3) + u_2 - 30]^2$$



- Surface is not differentiable across the "crease" where $x_1 = a_2$
- J(.) is *not* convex! (although Θ_i, Ψ_i != strictly convex)
- Points of non-differentiability form a critical component in the analysis

  Goal: Exclusion of these jobs

Example for a Nonsmooth Cost Function

- Introduction of critical jobs:

A job *i=1…N* is called critical if $x_i = a_{i+1}$

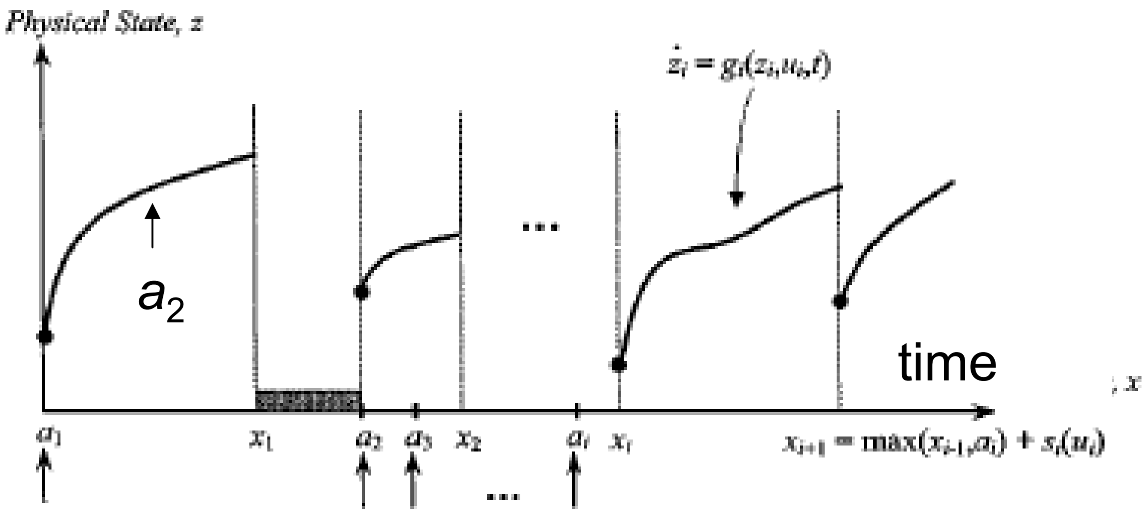
a_i: job arrival times

x_i: job completion times

  - Consequences for the cost function:

$$\frac{\partial J}{\partial \lambda_i} = 0 \quad \Rightarrow \quad x_i = \max(a_i, x_{i-1}) + s_i(u_i)$$

  - If there are no critical jobs: -> standard gradient-based methods (TPBV-solvers)
    otherwise: -> “Chattering“ across the crease at the minimum



Nonsmooth Optimization

- Objective:

  - To develop a solution that is able to deal with the introduced non-differentiability
  - Optimization of Lipschitz continuous functions

$$|f(x) - f(y)| \leq K|x - y|$$

K: open subset of $/R^N$

- Lipschitz functions: are continuous, but need not be differentiable everywhere

$$x_i = \max(x_{i-1}, a_i) + s_i(u_i) \qquad \textit{is Lipschitz}$$

$$\min_{\pi} J = \sum_{i=1}^{N} [\Theta_i(u_i) + \Psi_i(x_i)] \qquad \textit{is also Lipschitz } (\textstyle\sum \textit{theorem})$$

- In General, Cost Functions in Hybrid Optimal Control problems have discontinuities, but are Lipschitz

Nonsmooth Optimization

How to determine a global extremum?

– Reminder: Continuously differentiable (smooth) functions

- Necessary condition for a point to be a local extremum:
- Global extremum: Hesse-Matrix + boundary conditions!
- Use of gradient-based methods possible

$$\frac{\partial f(x)}{\partial x_i} != 0$$

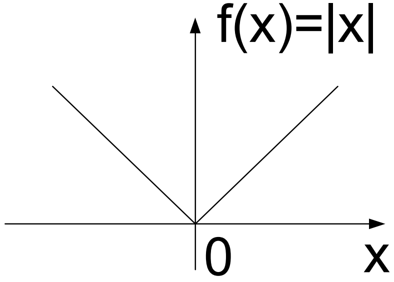
– Lipschitz continuous functions

- Necessary conditions for the optimum as a generalization of the gradient
- Introduction of the subdifferential $\partial f(u)$ of f at u: $\partial f(u)$
- Most important property: **if u is a local extremum of f, then**: $0 \in \partial f(u)$

Solving the optimization problem requires deriving an expression for the subdifferential $J(u_1, \ldots u_N)$.

Subdifferential Derivation

- Example:



$$\left.\begin{aligned}\lim_{x\uparrow 0^-}\frac{\partial f(x)}{\partial x} &= -1;\\ \lim_{x\downarrow 0^+}\frac{\partial f(x)}{\partial x} &= +1;\end{aligned}\right\} \quad \begin{aligned}&\textit{subdifferential}\\ &\partial f(u) = [-1,1]\\ &0 \in \partial f(u)\end{aligned}$$

How to use the subdifferential in our optimization problem?

- Provides a way to check for the optimal solution
- Event-driven dynamics enable a simple elevation of the subdifferential
- Using the left and right derivatives of *J(.)* it can be shown that the optimal control sequence u_i is unique

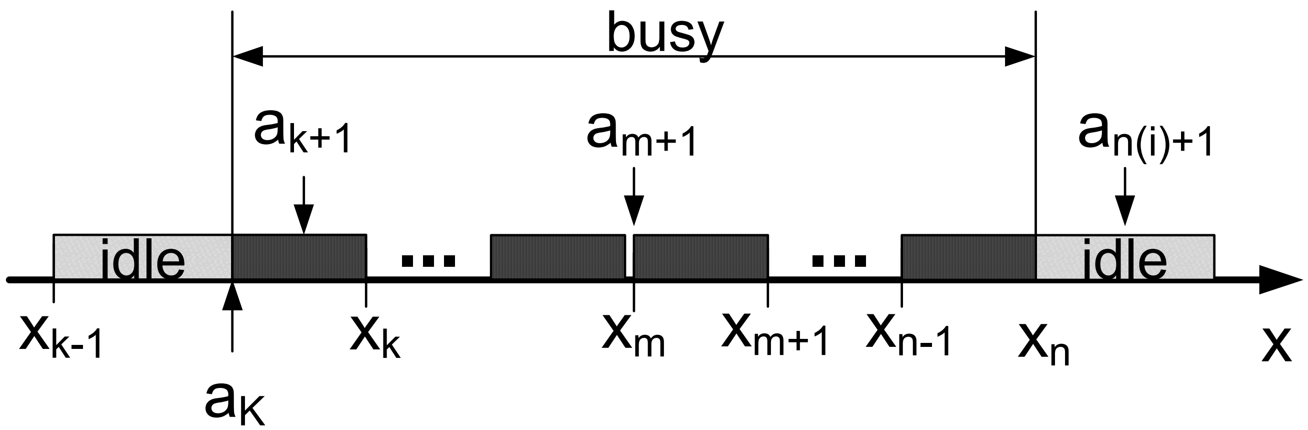
Subdifferential Derivation

Helpful definitions when evaluating the subdifferential

a_i: job arrival times

x_i: job completion times

- Introduction of a *sample path* consisting of:
  - departure times in response to given arrival times
  - idle periods
  - busy periods



- Evaluation of the subdifferential $\partial J(u_1,\ldots u_N)$ $\quad \zeta_i^- = \lim_{x_{m(i)} \uparrow a_{m(i)+1}} \frac{\partial J}{\partial u_i}; \quad \zeta_i^+ = \lim_{x_{m(i)} \downarrow a_{m(i)+1}} \frac{\partial J}{\partial u_i}$

- Optimal Control Sequence *i=1…N* must satisfy: $0 \in [\zeta_i^-, \zeta_i^+] \in \Re^1$

Decoupling properties

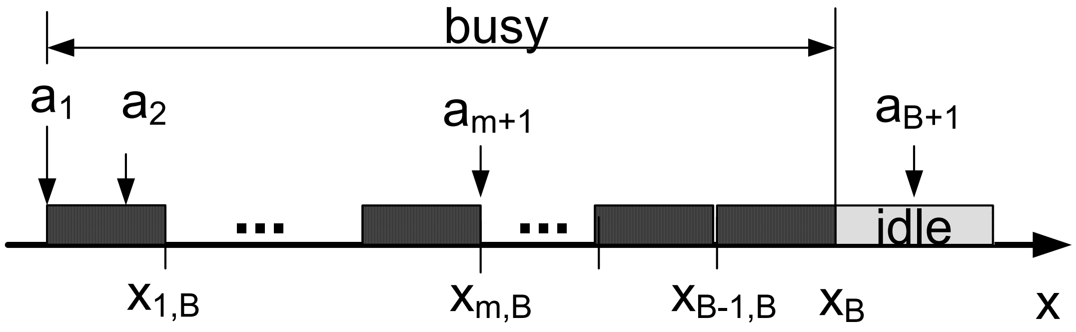
- Decomposition of the optimal state trajectory into fully decoupled segments

$$\min_{u_1 \ldots u_N} J = \sum_{i=1}^{N} J_i = \min(\min_{u_1 \ldots u_P} J_P = \sum_{i=1}^{P} J_i, \ldots, \min_{u_P \ldots u_Q} J_Q = \sum_{i=P+1}^{Q} J_i); \qquad P,Q < N$$

- Decoupling properties according to the event-generating mechanism
  - Idle period decoupling property
    - Optimal control u_i^* : dependent on number of Jobs and on arrival times a_i
    - Controls u_i for individual busy periods can be calculated independently
  - Block related decoupling property
    - Controls u_i for jobs before/after a critical job are independent
- Idea: Solving of the large optimization problem as a series of smaller (independent) subproblems (restrict the number of degrees of freedom)

Critical Job Identification

- For practical problems: Almost any sample path will contain critical jobs
- Considering a busy period containing jobs $i=1...B$ (starting with arrival time a_1)



a_i: job arrival times

x_i: job completion times

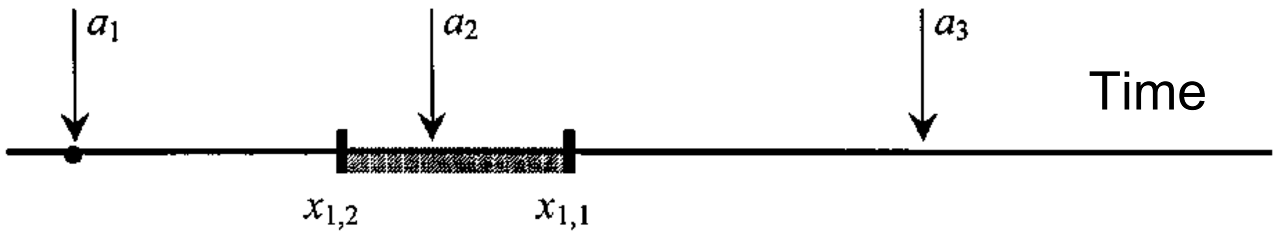
  - ***Optimal*** job departure times $x_{i,B}$ are only dependent on a_1 and B
    -> Pre-Computation of optimal departure times $x_{i,B}$ is possible! *(i=1,...,B-1)*

- Introduction of the critical interval $[x_{i,B},x_{i,i}]$
  Lemma: if any a_{i+1} *e* $[x_{i,B},x_{i,i}]$ then: interval will will include at least one critical job

- Determination of critical jobs:
  Lemma: Depending on job arrival times and on pre-computation optimal times -> statement *whether or not* a job is critical

Critical Job Identification

Example: *job1, … , job3*



a_i: Job arrival times

$x_{i,B}$: Pre-computed optimal job departure times (i=1... B-1)

- B → Number of jobs on the sample path
- i → Index of the *i-th* job to be processed

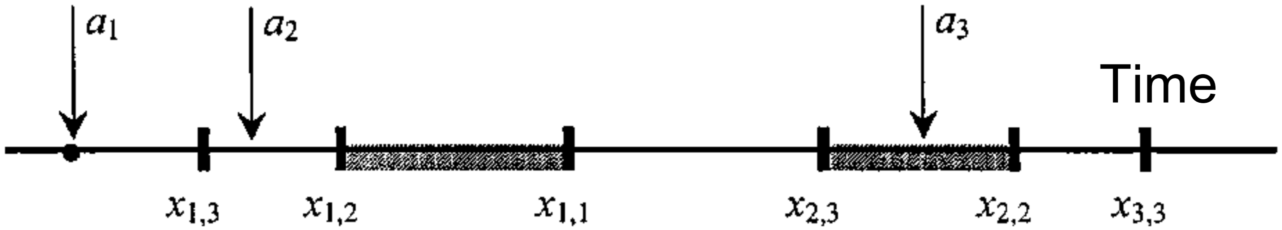
- Arrival time of job a_2 relative to the critical interval $[x_{1,2}, x_{1,1}]$ allows to identify whether job1
  - *1) is critical or not*
  - *2) does end the first busy period*
  - *3) is included in a busy period containing at the least job 1 and 2*

Critical Job Identification

Example: *job1, … , job3*

a_i: job arrival times

x_i: job completion times



- If $a_2 \le x_{1,3}$ && $a_2 \le x_{1,3}$ and $x_{1,3} \le a_2 \le x_{3,3}$ } Job1 is critical

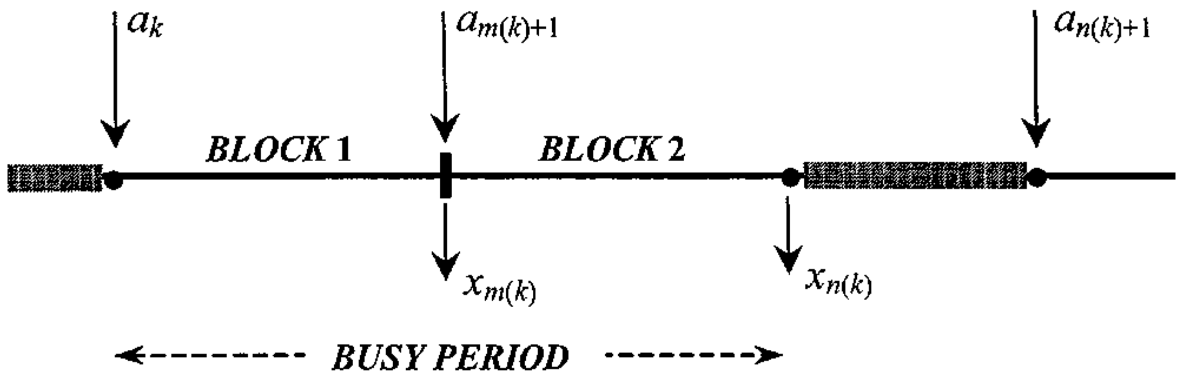
- If $x_{1,3} \le a_2 \le x_{1,2}$ and $x_{2,3} \le a_3 \le x_{2,2}$ } a sign-check needs to be implemented: $\zeta_i^- \cdot \zeta_i^+ < 0$ $0 \in [\zeta_i^-, \zeta_i^+]$

A Recursive Backward Algorithm

- Essential Idea:
  - Decomposition of the overall nonsmooth optimization problem into (smooth) subproblems with reduced dimensionality
  - Use of standard gradient-based solvers for individual subproblems (TPBVP)
  - Calculate each subblock by using terminal constraints (TC)

- Role of critical jobs (points of non-differentiability)

Example



– Two independent solutions (one for each block)
– Necessary condition: Identification of the busy period structure

Determining the busy period structure

Problem: Find a systematic way to identify the busy period structure

- General Approach:
  - Search for the optimal solution over all busy and block periods
  - exhaustive computational effort:

    For jobs N=1...N: 2^{N-1} different busy period structures

    2^{B-1} possible block structures (for jobs $j=1...B$ in a block)

  -> infeasible except for small problems

- Approach by D. Pepyne / C.Cassandras:

  - Identification of the busy period structure by implementing sign-checks
  - Calculation for each job in backward recursive manner
  - Use of efficient gradient-based-methods

A Backward Recursive Algorithm

- Example with N=5 jobs:

- Class-1 cost with a nonlinear service function $s_i(u_i)$:

$$\min_{u1,\dots,u5} J = \sum_{i=1}^{5}[\frac{1}{u_i} + x_i^2]$$

subject to:

$$x_i = \max(a_i, x_{i-1}) + u_i^2$$

J: cost Function

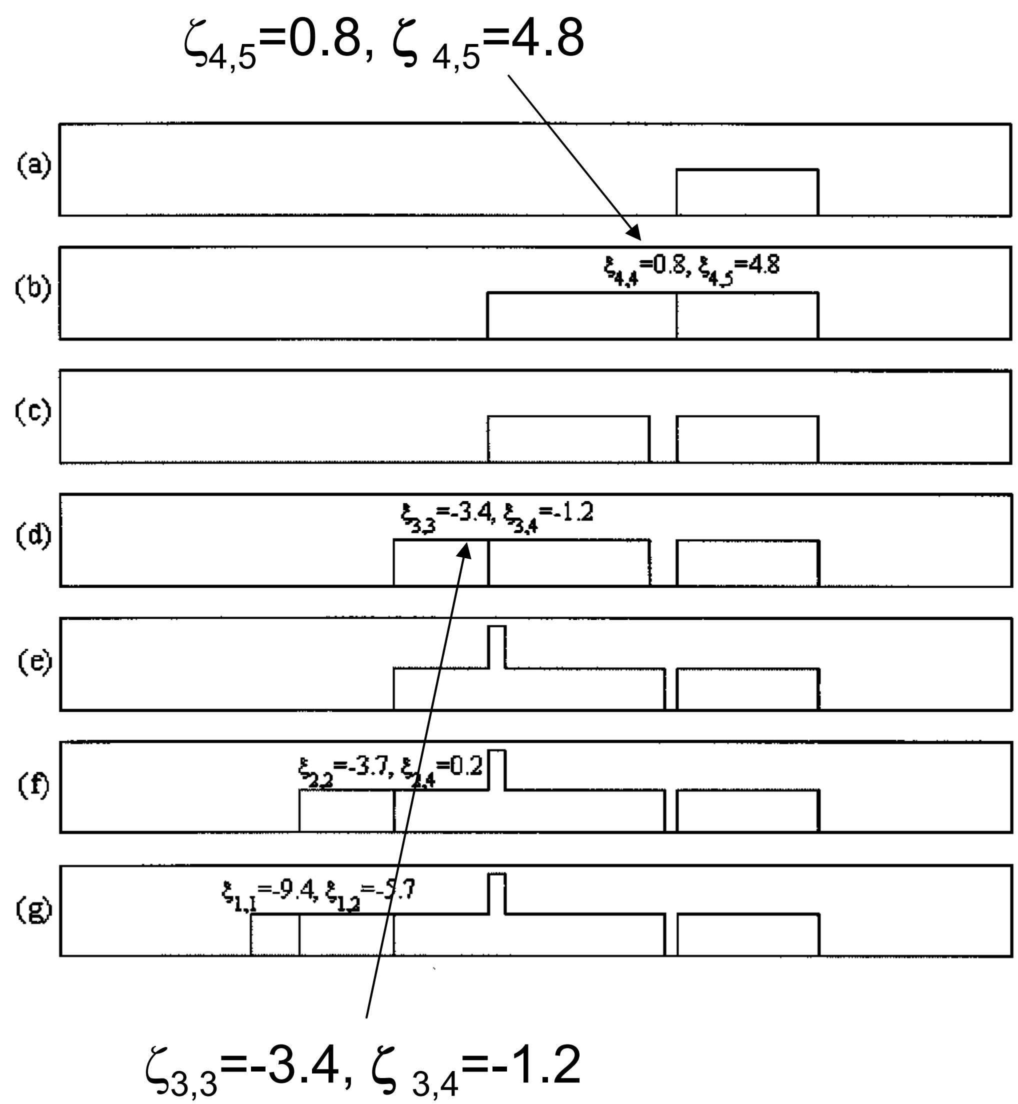
Θ_i: cost on control u_i

Ψ_i: cost on job completion x_i

- *J(.)* is strictly convex -> unique global extremum does exist!
- Input:
  - arrival times $a_1,\dots a_5$
  - *TCs* to identify critical jobs

- Recursive manner: starting with Job N and adding one by one previous jobs
- Implementation of the Algorithm using MATLAB

A Backward Recursive Algorithm



1. Initialization: Solve $P_{5,5}(0)$ to obtain u_5^* and x_5^*

2. Introduction of Job 4: calculate optimal control u_4^* and u_5^* (jobs in isolation)

Coupling properties:

- Computation of the Quantities $\zeta_{4,5}$ and $\zeta_{4,5}$+ sign test
- $\zeta_{4,5}$, $\zeta_{4,5}$ > 0: Decoupling of Job 4+5 into separate busy periods
- Idle Period Decoupling: no need to recalculate u_5^*

3. Introduction of job 3

- $\zeta_{4,5}$, $\zeta_{4,5}$ < 0: Merge of job 3 into busy period of job 4

4. Continue with job2 ...

Conclusion

- Solution of a general optimal control problem related to manufacturing processes
- Introduction of a hybrid system framework combining time-driven with event-driven dynamics
- Quality / time tradeoffs related to manufacturing process lead to a nonsmooth optimization problem
- Solution approach: *Divide and Conquer Scheme*

- Extension towards multistage processes

References

- D. Pepyne and C. Cassandras: Optimal Control of Hybrid Systems in Manufacturing

  Proceedings of the IEEE, Vol. 88, No.7, July 2000
- C. Cassandras, D. Pepyne, Y. Wardi: Optimal Control of a Class of Hybrid Systems

  IEEE Transactions on Automatic Control
- O. Stursberg: Script on lecture „Discrete Event and Hybrid Systems“

  Technical University of Munich