PhD in Smart Computing
Progress Report - 1st year

Tommaso Papini

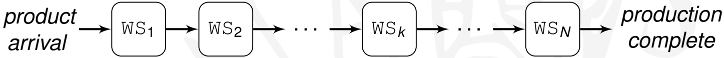
Department of Information Engineering, University of Florence, Italy
tommaso.papini@unifi.it

31st October 2017

- Analysis of assembly lines
- A hybrid technique for MRP transient analysis
- Other projects
  - the LINFA project
  - Activity Recognition for Ambient Assisted Living
- Research plan for the next year

Analysis of assembly lines

Assembly line

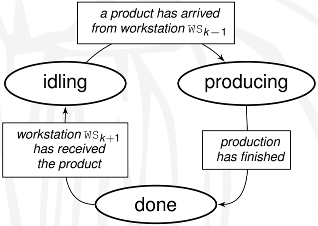


N sequential workstations $\mathtt{WS}_1, \ldots, \mathtt{WS}_N$

- with transfer blocking
- and no buffering capacity

Workstation $\mathtt{WS}_\mathtt{k}$ can be in one of three states

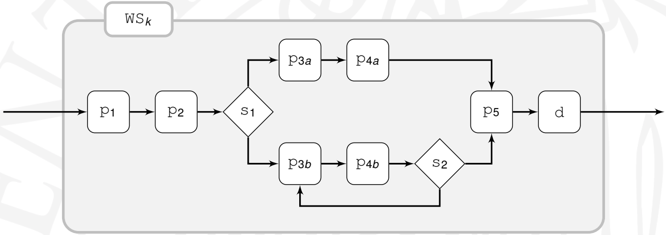
- *producing*: $\mathtt{WS}_\mathtt{k}$ is working on a product
- *done*: $\mathtt{WS}_\mathtt{k}$ is done working on a product
- *idling*: $\mathtt{WS}_\mathtt{k}$ is waiting for a new product



Workstation

Each workstation $\mathtt{WS}_k$

- has no internal parallelism
  - at most one item being processed in each workstation
- can implement complex workflows
  - sequential/alternative/cyclic phases with random choices
- and has GEN phases' durations



The last phase has no duration and encodes the *done* state

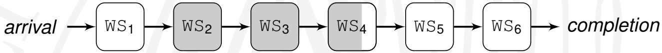
Underlying stochastic process

The underlying stochastic process of each isolated workstation is a Semi Markov Process (SMP)

- due to GEN durations
- and the absence of internal parallelism

The whole assembly line finds a renewal in any case where

- every *done* station is in a queue before a bottleneck
- and everything else is *idling*



Inspection with partial observability

The assembly line can be inspected by external observers

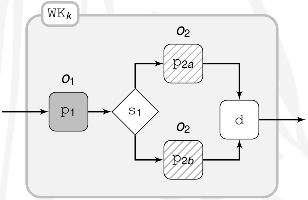
- the line can be considered at steady-state at inspection
- there can be ambiguity about the current phase

An observation is a tuple $\omega = \langle \omega_0, \omega_1, \dots, \omega_N \rangle$

- ω_0 indicates if a new product is ready to enter the line or not
- $\omega_k = \langle \sigma_k, \phi_k \rangle$ refers to $\mathtt{WS}_k$
  - σ_k indicates if $\mathtt{WS}_k$ is *idle*/*producing*/*done*
  - ϕ_k identifies the set of possible current phases

Two kinds of uncertainty

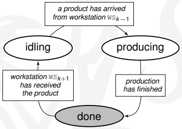
- about the actual current phase
  - discrete
- about the remaining time in the current phase
  - continuous



Performance measures

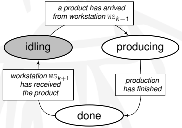
Time To Done

- The remaining time until workstation k, according to observation ω, reaches the *done* state



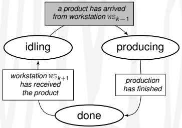
Time To Idle

- The remaining time until workstation k, according to observation ω, reaches the *idling* state



Time To Start Next

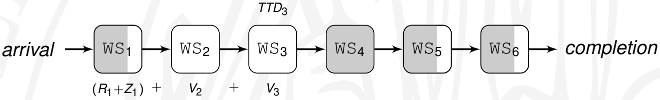
- The remaining time until workstation k, according to observation ω, starts the production of a new product



Time To Done

$$\mathrm{TTD}(k,\omega) := \begin{cases} \sum_{\gamma \in \phi_k} P_{k,\gamma,\omega} \cdot (R(k,\gamma) + Z(k,\gamma)), & \text{if } \sigma_k = \textit{producing} \\ \mathrm{TTD}(k-1,\omega) + V(k), & \text{if } \sigma_k = \textit{idling} \\ 0, & \text{if } \sigma_k = \textit{done} \end{cases}$$

- $P_{k,\gamma,\omega}$ probability weight that $\mathtt{WS}_k$ is in phase γ according to ω
- $R(k,\gamma)$ *remaining time* in phase γ of $\mathtt{WS}_k$
- $Z(k,\gamma)$ *execution time* of phases of $\mathtt{WS}_k$ that follow γ
- $V(k)$ *production time* of $\mathtt{WS}_k$

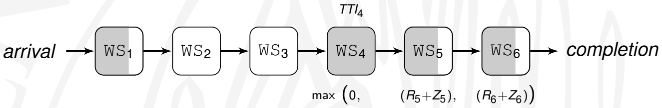


Backward recursive evaluation

Time To Idle

$$\mathrm{TTI}(k,\omega) := \begin{cases} \max\{\mathrm{TTD}(k,\omega), \mathrm{TTI}(k+1,\omega)\}, & \text{if } \sigma_k \in \{\textit{producing}, \textit{done}\} \\ 0, & \text{if } \sigma_k = \textit{idling} \end{cases}$$

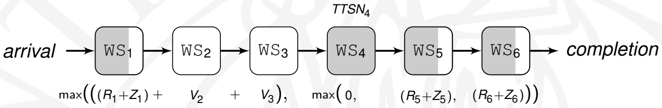
- $\mathrm{TTI}(k,\omega) = \max\{\mathrm{TTD}(k,\omega), \ldots, \mathrm{TTD}(k+n,\omega)\}$
  - $\mathtt{WS}_j$ producing/done $\forall j \in [k, k+n]$
  - either $\mathtt{WS}_{k+n}$ last workstation or $\mathtt{WS}_{k+n+1}$ idling
- $\mathtt{WS}_k$ becomes idle when the bottleneck finishes its production



Forward recursive evaluation

Time To Start Next

$$\mathrm{TTSN}(k, \omega) := \max\{\mathrm{TTI}(k, \omega), \mathrm{TTD}(k-1, \omega)\}$$



Forward and backward recursive evaluation

Disambiguation of observed phases

Resolve observed (producing) phases' ambiguity

- steady-state probability that $\mathtt{WS}_k$ is in phase γ according to ω

Given observation ϕ_k for workstation $\mathtt{WS}_k$

- we compute probability $P_{k,\gamma,\omega}$
- that it was actually γ that produced ϕ_k

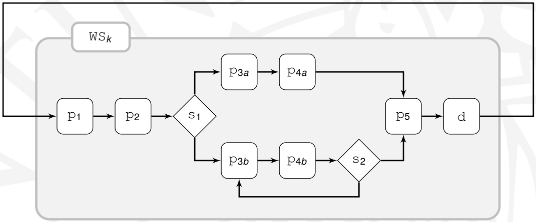
$$P_{k,\gamma,\omega} = \frac{\tilde{\pi}(\gamma)}{\sum_{\gamma' \in \phi_k} \tilde{\pi}(\gamma')}$$

- $\tilde{\pi}(\gamma)$ steady-state probability of phase γ in an isolated model of $\mathtt{WS}_k$

Isolated workstation model

The isolated workstation model represents a workstation repeatedly processing a product

- one product being processed
- after its production, it's moved back to the entry point of the workstation



It can be used for two reasons

- steady-state probabilities of producing phases are independent
- the inspection is at steady-state
  - arrivals and productions can be considerer in equilibrium

Remaining time

Evaluation of $F_{R(k,\gamma)}(t) =$ CDF of $R(k,\gamma)$

- $R(k,\gamma)$ *remaining time* in phase γ of $\mathtt{WS}_k$

Problem!

- remaining times of enabled GEN transitions are *dependent*
- joint probabilities don't allow for a compositional approach

1/3 *Immediate* approximation

- assume that phase γ is inspected at its ending
  - $\tilde{F}_{R(k,\gamma)}(t) = 1 \quad \forall t$
- represents an *upper bound*

2/3 *Newly enabled* approximation

- assume that phase γ is inspected at its beginning
  - $\tilde{F}_{R(k,\gamma)}(t) = F_\gamma(t)$
  - $F_\gamma(t)$ original CDF of the duration of γ
- represents a *lower bound*

Remaining time

$^3/_3$ *Independent remaining times* approximation

- consider the remaining times of ongoing phases as *independent*
- represents a (better) *lower bound*

Theorem: positive correlation & stochastic order

If $\hat{R}$ is an independent version of vector R of positively correlated remaining times of ongoing phases, then $\hat{R} \geq_{st} R$

Steady-state distribution of $\hat{R}(k, \gamma)$
computed according to the Key Renewal Theorem[1]

$$\tilde{F}_{R(k,\gamma)}(t) = \frac{1}{\mu} \int_0^t [1 - F_\gamma(s)] ds$$

- μ expected value of $F_\gamma(t)$

[1] Serfozo, R., 2009. Basics of applied stochastic processes. Springer Science & Business Media.

Execution and production time

Evaluation of $F_{Z(k,\gamma)}(t)$ and $F_{V(k)}$

- $Z(k,\gamma)$ *execution time* of phases of $\mathtt{WS}_k$ that follow γ
- $V(k)$ *production time* of $\mathtt{WS}_k$

CDFs of $Z(k,\gamma)$ and $V(k)$ are computed as transient probabilities

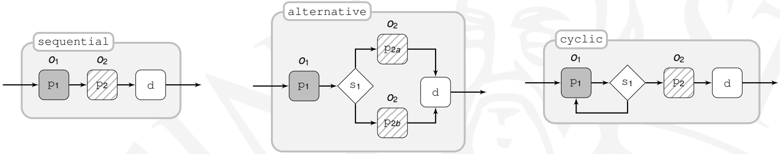
- $F_{Z(k,\gamma)}$ transient probability from phase after γ to final phase of $\mathtt{WS}_k$
- $F_{V(k)}$ transient probability from first phase to final phase of $\mathtt{WS}_k$

Upper/lower bounds for *TTD*, *TTI* and *TTSN* can be evaluated

- *convolution* and *max* operations maintain stochastic order

Case study assembly lines

Sequential, alternative and cyclic workstations

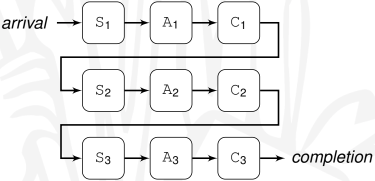


Simple assembly line

- two sequential workstations
- both in phase p_1 at inspection

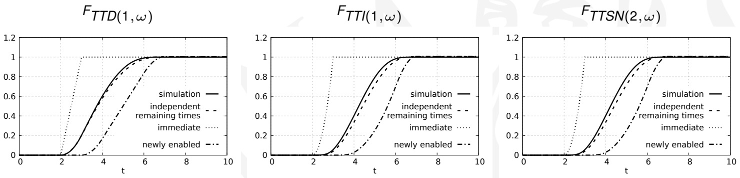
Complex assembly line

- three repetitions
- of sequential/alternative/cyclic ws
- all observed in *producing*



Simple assembly line

TTDone, TTIdle, TTStartNext



TTD, *TTI* and *TTSN* computed in

- 41/45/42 min for simulation
- 0.15/0.18/0.10 s for bounds

Very good approximation results

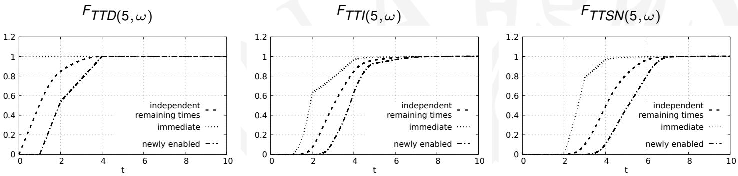
- especially for *independent remaining times*

Feasible approach

- very fast bounds evaluation
- compared to simulation

Complex assembly line

TTDone, TTIdle, TTStartNext



TTD, *TTI* and *TTSN* computed in

- 0.126/0.123/0.75 s for bounds

Scalable solution

- in a complex scenario
- simulation would be infeasible

A hybrid technique for MRP transient analysis

A hybrid technique for MRP transient analysis

Transient analysis of Markov Regenerative Processes (MRP) employing different techniques for different regenerative epochs

The basics:

- Exact techniques require specific conditions to be met
  - different techniques require different conditions
- Kernel rows of different epochs can be evaluated independently

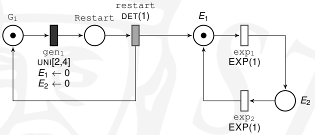
The idea:

- Evaluate each kernel row with a different technique
  - corresponding to the condition enabled in that epoch
  - eventually with an approximate technique, if no conditions are met
- Compute transient probabilities with Markov Renewal Equations

Techniques for MRP transient analysis

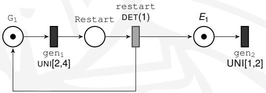
Analysis under enabling restriction[2]

- at most one GEN enabled in each state



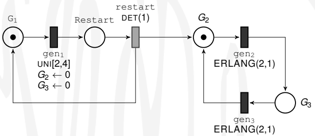
Analysis with stochastic state classes[3]

- a regeneration is always reached within a bounded number of events
  - i.e. no cycles without regenerations
- a.k.a. bounded regeneration



Approximate analysis

- usable when no conditions are met



[2] German, R., Logothetis, D., & Trivedi, K. S. (1995, October). Transient analysis of Markov regenerative stochastic Petri nets: A comparison of approaches. In Petri Nets and Performance Models, 1995., Proceedings of the Sixth International Workshop on (pp. 103-112). IEEE.

[3] Horváth, A., Paolieri, M., Ridi, L., & Vicario, E. (2012). Transient analysis of non-Markovian models using stochastic state classes. Performance Evaluation, 69(7), 315-335.

Classification of epochs

Through **non-deterministic analysis**

- State Class Graphs (SCG) are built
- for each regenerative epoch

By visiting each SCG, epochs are classified

- enabling restriction
  - if at most one GEN is enabled in any state
- bounded regeneration
  - if no cycle is present

Iterative approximate technique

Based on analysis with stochastic state classes

- truncated after enough precision is met

Improvement with heuristics

1. expand at most ν_{start} nodes for non restricted epochs
2. identify the truncated node Φ with highest reaching probability
   - based on steady-state analysis of the embedded DTMC
3. expand at most ν_{iter} nodes from Φ
4. if at least ν_{max} nodes expanded, stop
   - otherwise, return to step 2

The LINFA project

The LINFA project

Smart drug restocking for hospital wards

- minimise overall cost of ordering and stocking drugs
- predict drug usage and possible shortages

The idea:

- Build and solve a Markov Decision Process (MDP) model of the ward
  - actualised at runtime with the current state of the ward
- suggest the optimal strategy top the user

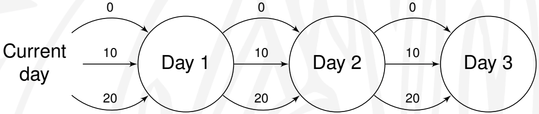
Solution architecture

At the end of the current day

- a new PRISM MDP model is instantiated
  - through a Java module
  - with the ward's current state

The MDP model models:

- stochastic evolution of the ward during each day
- non-deterministic choices (i.e. drug orders)



Other projects

Activity Recognition for Ambient Assisted Living

Other projects

Research plan for the next year

Research plan for the next year

Analysis of assembly lines

Introduction of buffering capacity

- with fixed/variable capacity
- so to model more realistic scenarios

Derivation of additional performance measures

- in the same compositional fashion
- e.g. production time of a certain product in the line
- or of the next N products

Derivation of a more educated *upper bound*

Research plan for the next year
The LINFA project

Model more aspects to refine the ward model

- introduce personalised healthcare protocols
- employ process mining techniques

State-space optimisation

- avoid state-space explosion
- investigate other tools
  - Storm

Research plan for the next year

Activity Recognition for Ambient Assisted Living

Refine model based AR

- exploit fuzzy logic to include support for continuous sensors
  - accelerometer/thermometer/. . .

Identify good AR datasets for AAL

- investigate the literature
- generate new datasets

In order to apply process mining techniques