European Journal of Operational Research 242 (2015) 865–874

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Production, Manufacturing and Logistics

A continuous buffer allocation model using stochastic processes

Oliver Kolb, Simone Göttlich*

Department of Mathematics, University of Mannheim, 68131 Mannheim, Germany

ARTICLE INFO

Article history:
Received 10 February 2014
Accepted 31 October 2014
Available online 20 November 2014

Keywords:
Production systems
Buffer allocation
Sampling
Nonlinear optimization

ABSTRACT

The buffer allocation problem consists of a dynamical description of the underlying production process combined with stochastic processing times. The aim is to find optimal buffer sizes averaged over several samples. Starting from a time-discrete recursion we derive a time-continuous model supplemented with a stochastic process. The new model is used for simulation and optimization purposes as well. Numerical experiments show the efficiency of our approach compared to other optimization techniques.

1. Introduction

Production systems are organized in a way that the best possible output is reached and storage costs are minimized. However, this objective is usually influenced by external factors such as fluctuating demands, supply bottlenecks or individual product specifications. In most instances, the overall goal is to minimize the total buffer load, where the production dynamics is induced by processing order inequalities and restrictions on the system capacities. Typical applications we think of include the motor vehicle industry in the case of automotive parts as well as other kinds of processing industries.

Mathematical models provide a powerful tool to study and analyze production systems. Especially in the case of the buffer allocation problem, see Alfieri and Matta (2012), Burman (1995), Dallery and Gershwin (1992), Demir, Tunali, and Eliiyi (2014), Dolgui, Eremeev, Kovalyov, and Sigaev (2013), Gershwin (1987), Gershwin and Schor (2000), Gershwin and Tan (2009, 2010), Gürkan (2000), Helber, Schimmelpfeng, Stolletz, and Lagershausen (2011), Matta (2008), and Tan and Yeralan (1997) for an overview. One can basically distinguish between discrete and continuous models. Discrete models are characterized by discrete time periods and work pieces while continuous models instead rely on continuous time and approximate quantities, see e.g. Armbruster, Degond, and Ringhofer (2006), Degond and Ringhofer (2007), Göttlich, Herty, and Klar (2006), Göttlich, Herty, and Ringhofer (2010). In this paper, we focus on a model connecting these two perspectives from a numerical point of view.

In Section 2, a discrete time recursion formula is introduced which can be used for the tracking of individual work pieces through a production system, also called discrete event simulation (DES). The work pieces have particular processing times which are randomly distributed. Evaluation measures help to determine buffer loads and the current work in progress (WIP). The goal is to minimize the total buffer size in the production system. The resulting optimization problem can be exactly reformulated as a mixed integer linear program (MIP), and as illustrated in Matta (2008), Stolletz and Weiss (2013), and Weiss and Stolletz (2013), there is a strong need for adapted solution methods to tackle the underlying MIP—even for smaller test cases. Optimization techniques such as Benders decomposition must be applied to solve the models within an acceptable time frame. In further literature, the classical buffer allocation problem (cf. Demir et al., 2014 and the references therein) is based on a similar idea leading to a NP-hard combinatorial optimization problem. Without additional heuristics there is nearly no chance to deal with large-scale problems.

Another modeling approach, also implying a different optimization as indicated in Fig. 1, will be our major concern. It is well known that discrete event models provide the most accurate description of the underlying dynamics for simulation purposes. However, it is a microscopic model, meaning that the computation times highly depend on the number of work pieces to be considered. An optimization problem based on such a model at first leads to simulation-based optimization procedures only and is usually extremely costly, see Göttlich et al. (2006). In order to derive a suitable optimization framework, one can introduce an associated mixed integer linear program. Although this program is still dependent on individual parts, the optimization can be done in a well-defined way using common Branch and Bound algorithms. At best, global optimality can be reached.

Within this article we are interested in models on a macroscopic scale, or more precisely, we rigorously derive a continuous model which avoids the dependence on single parts and discrete options

* Corresponding author. Tel.: +49 621 1812438.
E-mail addresses: kolb@math.uni-mannheim.de (O. Kolb), goettlich@math.uni-mannheim.de (S. Göttlich).

http://dx.doi.org/10.1016/j.ejor.2014.10.065
0377-2217/

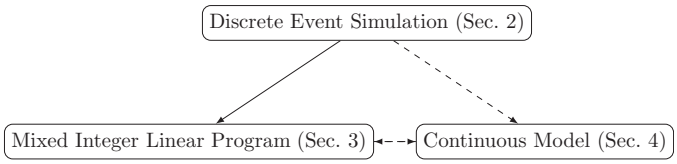

Fig. 1. Connection between existing models and the presented approach.

for the buffer capacities. The benefit of such a continuous model is that it can be used for fast simulation and also optimization. For the optimization we apply nonlinear methods leading to qualitatively very good results close to the mixed integer solution.

The paper is organized as follows: First, based on the discrete time recursion described in Section 2, we formally derive a time-continuous model in Section 4 and also describe its numerical treatment (Section 5.1). Then, using nonlinear optimization techniques as in Göttlich, Kolb, and Kühn (2014), Kirchner, Herty, Göttlich, and Klar (2006), Kolb (2011), and Kolb and Lang (2012), we are able to solve the reformulated problem by a gradient method (cf. Section 5.2). The quality of the results is very promising as pointed out in Section 6.

2. Discrete model

In this section, we start with the description of a time discrete model called discrete event simulation that is suitable for evaluating queuing systems. Similar approaches can be found in Armbruster et al. (2006), Stolletz and Weiss (2013), and Weiss and Stolletz (2013).

We consider a queuing network with K consecutive processors, each of them with a queue of size C_m ($m \in \{1, \dots, K\}$) in front, see Fig. 2. The times $T_{m,n}$ needed by processor m to process piece n ($n \in \{1, \dots, N\}$) follow a processor dependent random distribution and will be sampled accordingly. Note that for the evaluation and the comparison of all presented models (Sections 2–4), we always consider several samples of processing times $T_{m,n}$ in the numerical results in Section 6, but the same samples will be used for all models. Transport times between different processors are neglected and so the work pieces are directly fed into the next production unit.

The main ingredient of a time recursion is the modeling of effects such as free flow, starving and blocking. Let $\tau_{m,n}$ be the arrival time of piece n at the queue of processor m. Assuming all pieces are processed as fast as possible, the arrival times of the considered production line read

$$\tau_{m,n} = \max\{\max\{\tau_{m,n-1}, \tau_{m-1,n}\} + T_{m-1,n}, \tau_{m+1,n-(C_m+1)}\}. \tag{1}$$

The first term,

$$\max\{\tau_{m,n-1}, \tau_{m-1,n}\} + T_{m-1,n},$$

results from the fact that piece n cannot be started being processed at processor $m-1$ before the preceding piece $n-1$ has left processor $m-1$ and therewith arrived at (the queue) of processor m at time $\tau_{m,n-1}$ (free flow) and not earlier than the arrival time $\tau_{m-1,n}$ of piece n at processor $m-1$ (starving). The second term in (1) ensures that if there is enough space in the queue in front of processor m (at least piece $n-(C_m+1)$ has been processed, yielding a free slot), the piece n can be processed directly, otherwise it will be stopped (blocking). For given initial conditions $\tau_{1,n}$ (arrival times at the first processor), processing times $T_{m,n}$ and queue sizes C_m, a simulation/evaluation of the arrival times $\tau_{m,n}$ ($m \in \{2, \dots, K+1\}$, $n \in \{1, \dots N\}$) can be done based on (1) (with $\tau_{m,n} = -\infty$ if $n \le 0$ or $m > K+1$). Here, we typically assume $\tau_{1,n} = 0$ and $C_1 = C_{K+1} = \infty$. Note that we also consider $\tau_{K+1,n}$, as the time when piece n leaves the last processor.

In a next step, cf. Armbruster et al. (2006) and Göttlich et al. (2006), evaluation measures are needed to identify the quantities flux and WIP. Mathematically, we introduce the concept of curves of cumulative counts using the Heaviside function $H(\cdot)$. Then, with

$$H(x) = \begin{cases} 1 & x \ge 0 \\ 0 & x < 0 \end{cases}$$

we denote by

$$U_m(t) = \sum_{n=1}^{N} H(t - \tau_{m,n})$$

the number of pieces having arrived at (the queue of) processor m until time t. Further, we define the flux

$$F_m(t) = \frac{d}{dt} U_m(t) = \sum_{n=1}^{N} \delta(t - \tau_{m,n}),$$

where δ is the Dirac delta function (derivative of the Heaviside function in a distributional sense), and the WIP

$$W_m(t) = U_m(t) - U_{m+1}(t) + W_{m,0}$$

with initial conditions $W_{m,0}$. Obviously,

$$\frac{d}{dt} W_m(t) = F_m(t) - F_{m+1}(t) \tag{2}$$

holds. Due to the limited queue size C_m in front of processor m, the following constraint must be satisfied for the WIP:

$$0 \le W_m(t) \le C_m + 1. \tag{3}$$

The WIP is limited by $C_m + 1$ because C_m pieces may be waiting in the queue and one can be currently processed.

3. MIP approach

The discrete model described in Section 2 can be modeled within a linear MIP (cf. Matta, 2008; Weiss & Stolletz, 2013). Transferred to our notation (regarding names of the variables and also indexing), the relevant constraints within the MIP formulation are

$$\tau_{1,n} = 0 \quad n \in \{1, \dots, N\}, \tag{4}$$

$$\tau_{m,n} \ge \tau_{m,n-1} + T_{m-1,n} \quad m \in \{2, \dots, K+1\}, \quad n \in \{1, \dots, N\}, \tag{5}$$

$$\tau_{m,n} \ge \tau_{m-1,n} + T_{m-1,n} \quad m \in \{2, \dots, K+1\}, \quad n \in \{1, \dots, N\}, \tag{6}$$

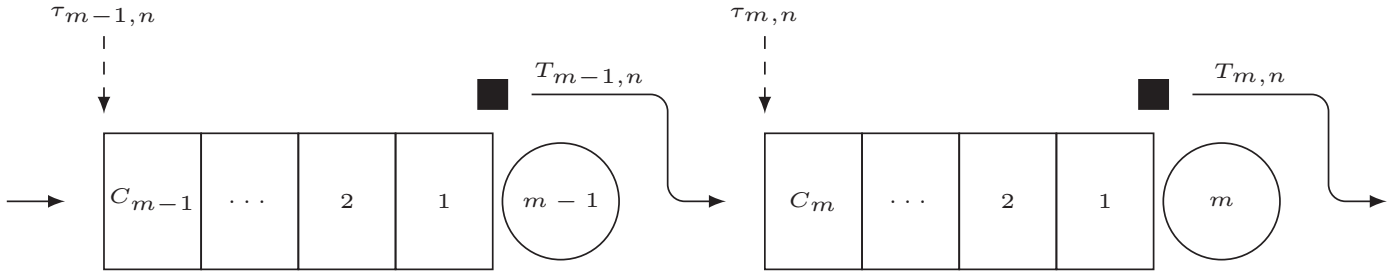

Fig. 2. A serial production line.