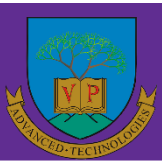
PANNON EGYETEM
MŰSZAKI INFORMATIKAI KAR



PANNONIAN CONFERENCE ON ADVANCES IN INFORMATION TECHNOLOGY

Proceedings of the

Pannonian Conference on Advances in Information Technology (PCIT 2020)

5 June 2020

University of Pannonia, Veszprém, Hungary

Editor: István Vassányi

Cover design: Domonkos Kertész

Published by the University of Pannonia, Faculty of Information Technology

ISBN 978-963-396-144-5

Preface

The Pannonian Conference on Advances in Information Technology (PCIT2020) was organized by the Faculty of Information Technology, University of Pannonia, Veszprém, Hungary together with the Information Technology in Healthcare Work Committee and the Operation Research Work Committee of the Regional Centre of the Hungarian Academy of Sciences, Veszprém, Hungary on June 5, 2020. The scientific program of the conference consisted of classical and interdisciplinary areas of information technology including, e.g., software technology, intelligent systems, image processing, data analysis, process modeling and optimization, medical and industrial applications. After a two-round peer review process, 12 papers of 34 co-authors were accepted for oral presentation at the conference, out of which 10 are included in this proceedings as full papers.

I thank the members of the Scientific Committee for all their efforts in putting together the scientific program, and I thank all authors and participants for their contributions.

Ferenc Hartung

Dean
Faculty of Information Technology
University of Pannonia

PCIT 2020 Scientific Committee

Chair:

Vassányi, István, University of Pannonia

Members:

Bodó, Zalán, Babes-Bolyai University

Borbély, Ákos, Óbuda University

Hartung, Ferenc, University of Pannonia

Heckl, István, University of Pannonia

Kis, Tamás, Hungarian Academy of Sciences Institute for Computer Science and Control

Kiss, Attila, Eötvös Loránd University

Krész, Miklós, University of Szeged

Stark-Werner, Ágnes, University of Pannonia

Szederkényi Gábor, Pázmány Péter Catholic University

Várady, Géza, University of Pécs

Contents

For the full PCIT 2020 conference programme including also the oral presentations without published papers, please see:

https://pcit2020.mik.uni-pannon.hu/index.php/hu/

Hybrid Multimode Resource-Constrained Maintenance Project Scheduling Problem

Zs.T. Kosztyán[1,2,5], *A.Pribojszki-Németh*[1,3,5], *I.Szalkai*[4,5]

[1]Univ. of Pannonia, Dept. Quantitative Methods,
[2] kzst@gtk.uni-pannon.hu, [3] pribojszki-nemeth.aniko@gtk.uni-pannon.hu
[4]Univ. of Pannonia, Dept. Mathematics, szalkai@almos.uni-pannon.hu
[5] 10 Egyetem u., Veszprém, H-8200, Hungary

Abstract: Organizing maintenance tasks is a specific and important field of project and production management. Well-planned, properly scheduled and effectively communicated maintenance tasks result in the accomplishment of more work that is performed more efficiently at a lower cost. In maintenance and production management, agile and lean approaches are becoming more frequently used, where preventive/predictive maintenance (PPM) requires flexible project plans to decide which system components should be maintained to achieve the target system reliability. Maintenance projects have been treated as fixed sequences of corrective/preventive tasks while the agile/lean approach allows and usually requires prioritizing tasks and reorganizing the project structure. The fixed structure of maintenance projects and traditional maintenance project scheduling algorithms are not considered representative of these agile properties. The paper models the preventive resource-constrained maintenance project scheduling problem and aims optimizing its maintainability. A matrix-based algorithm is proposed to apply to both system and project structures that include both traditional and agile/lean project management approaches.

1 Introduction

Advanced maintenance management strategies such as reliability-centered maintenance (RCM), risk-based maintenance management (RBM) and total productive maintenance (TPM) focus on system reliability or the risks of the system, instead of focusing only on the reliability of separate system components. Therefore, the scheduling of preventive maintenance tasks requires the consideration of overall system reliability. The latest maintenance management trends attempt to include both agile and lean approaches. Therefore, the corrective/preventive maintenance tasks can be prioritized to maximize system reliability [1].

The challenge of managing and organizing maintenance tasks as maintenance projects can be summarized as follows. (1) Although maintenance tasks can be assigned to system components based on previous maintenance experience and previous maintenance task lists from machine books, and therefore time, cost and resource demands can be linked to corrective maintenance tasks, increased reliability can only be linked to the system components. Therefore, ***when specifying corrective/preventive maintenance tasks, the increase in***

the reliability of the system component(s) as a result of corrective / preventive tasks should be estimated. (2) The structure of a system specifies the system reliability, and it can be fundamentally different from the structure of the maintenance projects, which include (corrective / preventive) maintenance tasks. Therefore, ***when optimizing a maintenance project, two types of structures (system and project ones) should be linked*** [4]. (3) ***The lean and agile approach allows the considered flexible maintenance projects*** to focus on maximizing system reliability and prioritizing tasks based on those tasks that increase system reliability the most when considering the time, cost, and resource constraints.

In this case, the traditional time-cost tradeoff and/or multimode methods should be extended to address the growth in component reliability and combined with the agile and lean approaches to enable prioritization of the corrective/maintenance tasks to maximize system reliability. To date, no exact or hybrid algorithms are available to support the agile/lean approach in maintenance management. Moreover, agile/lean approaches are more effective than traditional approaches. Therefore, the algorithmic support of the agile/lean maintenance management approaches requires further study. The proposed hybrid algorithm specifies a maintenance project (the set of decided, to-be-realized maintenance tasks that form a predefined task list and propose a sequence of completion (henceforward, project structure)) with an exact method within expectedly polynomial computation time. The proposed exact method for specifying the project structure is combined with a metaheuristic method to select an optimal completion mode.

The proposed method can not only be treated as a hybrid method but can also be considered a hybrid of the traditional (discrete) Multimode Resource Constrained Project Scheduling Problem (MRCPSP) and the agile/lean approach. The algorithmic background of agile and hybrid approaches is a rarely studied field while the use of agile methods is common in the field of software projects. These methods are becoming increasingly popular in other areas, such as maintenance [4] showed how to combine two approaches, such as traditional continuous tradeoff methods, with flexible agile approaches; however, in the case of maintenance we should select a completion mode from discrete technologies. Therefore, instead of using continuous tradeoff methods we should use the MRCPSP in combination with the agile approach. Since the MRCPSP is a generalization of the discrete tradeoff methods, which are usually NP-hard problems, an exact algorithm alone can not be used to solve the problem. Nevertheless, because the selection of tasks and

dependencies can be performed within expectedly polynomial computation time [2], we can propose a faster and better algorithm than a purely metaheuristic method as detailed in [3]. In addition, the proposed method avoids the tradeoff assumption, therefore different types of technology can be considered where the time, cost and resource demands can vary.

In our model we are given first **system components** k_i with **reliability values** $r(k_i)$, a set of possible **tasks** (A) containing **mandatory** and **supplementary** ones for improving $r(k_i)$. We also are given **relations** ($\prec,\sim,\bowtie$) among the tasks: which must be finished before / after, may be handled simultaneously, these relations also may be mandatory or supplementary. For each task $a\in A$ we are given a *set* W_a of **protocols**: a *list* of possible treating methods for the task a. These protocols include cost, time, quality and resource data for each treating.

In PHASE ONE we have to decide which supplementary tasks to be chosen, maximizing certain values (see section 3), fulfilling also some requirements (C_c,C_t,C_{diag}). In PHASE TWOO we have to decide the supplementary relations to fix the completion order of the tasks, minimizing the desired time. In PHASE THREE we have to find a feasible set of protocols $w_a\in W_a$ for the selected tasks a .

The basis of the proposed methods is the *project domain matrix* PDM [2]. Our algorithm and simulations are described in Sections 4 and 5.

Currently, hybrid (i.e. combinations of traditional and agile) approaches are becoming increasingly popular, however, these approaches lack a principled *mathematical foundation* and algorithmic treatment. Our first goal is to fill this gap in this paper, so, for describing exactly the problem and our results we need unfortunately many definitions and notations at first.

2 Mathematical definitions

We are given finite sets $K=\{k_1,...,k_z\}$ of **system components** [we have to maintan] and $A=\{a_1, ...,a_n\}$ of **possible** (maintaning) **tasks**. r:K$\rightarrow$[0,1] is the **reliability** function, i.e. $r(k_i)$ is the reliability **value** of k_i , $\boldsymbol{TSR}(K)\in[0,1]$ is the (total) **system reliability**. (The task-component relation is given in $\mathcal{W}$ below.) The tasks related to system component k_i ($i\le z$) are **mandatory** if $r(k_i)<cr_i$ where $\mathbf{cr}_i$ is a given *critical* reliability value for k_i , other tasks are supplementary. $A^{-}\subseteq A$ contains the **mandatory** and $A^{\sim}:=A\backslash A^{-}$ the **uncertain** or **supplementary** tasks, $|A^{\sim}|=\sigma$. The set of **project scenarios** [*sets* of tasks we choose to do] is $\Xi(A):=\{S\subseteq A: A^{-}\subseteq S\}$.

Among the tasks in A we are given three **relations**: $a_i \prec a_j$ **strict** or **required**: a_j may not be started unless a_i has been completed; $a_i \sim a_j$ means **no dependency** and $a_i \bowtie a_j$ **flexible** dependencies must be resolved (decided) by the algorithm: either to $\prec$ or $\sim$. Clearly $\prec$ is a partial order that excludes cycles like $a_1 \prec a_2 \prec ... \prec a_1$, so we may assume that $a_i \prec a_j \Rightarrow i<j$. A **project structure** [*order* of tasks] determines each flexible relation ($a_i \bowtie a_j$) either to strict ($a_i \prec a_j$) or none ($a_i \sim a_j$). The triplet $X = (S, \prec, \sim)$ represents the **final** project structure of a given project scenario $S \in \Xi(A)$. To summarize: a *project scenario* S is the *set* of the tasks we plan to do, while a *project scenario X* is the order of them when fulfilling.

We use the **matrix representation** $M=[m]_{ij} \in \{X, \emptyset, ?\}^{n \times n}$ of the above input as $m_{i,i}=X$ for $a_i \in A^{-}$, $m_{i,i}=?$ for $a_i \in A^{\sim}$ and for $i \neq j$ we have $m_{i,j}=X \Leftrightarrow a_i \prec a_j$, $m_{i,j}=\emptyset \Leftrightarrow a_i \sim a_j$ and $m_{i,j}=? \Leftrightarrow a_i \bowtie a_j$.

Any finite set of quadruplets $\boldsymbol{W}=\{(t_i,c_i,\mathbf{R}_i,\Delta r_i): i=1,\ldots,m\}$ of nonnegative real numbers and vectors $\mathbf{R}_i \in \{0,1\}^z$ is called a **multimode protocol,** MMp. If for each $a \in A$ we are given a protocol $W_a \subseteq \boldsymbol{W}$ then we call the set $\mathcal{W} = \{W_a : a \in A\}$ **a multimode maintenance management problem** (M^4P). We interpret $\mathbf{w}=(t,c,\mathbf{R},\Delta r) \in W_a$ as paying the **cost** c for solving the task $a \in A$ in **time** t with the set of resources $\mathbf{R}$ with an increase in reliability Δr using the **mode** assigned to $\mathbf{w}$. The desired increase in the reliability of a system component k_i depends on the maintenance tasks $a \in A$ that are assigned to it. One or more tasks can be assigned to a system component, but a task can be assigned to only one system component. For any $S \in \Xi(A)$ and $\mathcal{W}$a **project schedule** is a set $w^{\rightarrow} = \{w_a : a \in S\}$ where $w_a=(t^a,c^a,\mathbf{R}^a,\Delta r^a) \in W_a \in \mathcal{W}$ for each $a \in S$. Let $t_{min}=\min_i t_i$, $t_{max}=\max_i t_i$, $r_{min}=\min_i r_i$, $r_{max}=\max_i r_i$, $c_{min}=\min_i c_i$, $c_{max}=\max_i c_i$ the min/max values of the demands of the tasks, $i=1, \ldots, m$.

The **maintenance activity** $\Delta r^a{}_k$ is the increase in reliability of system component $k \in K$ if $a \in A$ is completed. $\Delta r^a{}_k = \Delta r^a{}_k$ if a is assigned to k, otherwise $\Delta r^a{}_k = 0$. The **total increase** in reliability of the system component k is $\Delta r_k = \Sigma_{a \in S} \Delta r^a{}_k$.

For any $\mathbf{M},\mathbf{N} \in \{X, \emptyset, ?\}^{n \times n}$ $\mathbf{N}$ is an **in-/out-** diagonal **extension** of $\mathbf{M}$ (i.e.$\mathbf{M}$ is a **restriction** of $\mathbf{N}$) if all the "X" and "$\emptyset$" symbols in $\mathbf{M}$ remain unchanged in $\mathbf{N}$ and some "?" *on/off* the diagonal of $\mathbf{M}$ are changed to either "X" or "$\emptyset$". $\mathbf{N}$ is an **in-/out-** diagonal **closure** of $\mathbf{M}$ if $\mathbf{N}$ contains no "?" symbols on/off of the diagonal and $\mathbf{N}$ in/out extends $\mathbf{M}$.

For time bounds, we must consider the $\prec$ dependencies: $P^{\rightarrow} = (a_{i1}, a_{i2}, \ldots, a_{i\ell})$ is a **real path** of length ($\ell \leq n$) if $a_{i1} \prec a_{i2} \prec \ldots \prec a_{i\ell}$ for $1 \leq i_j < n$, $j \leq \ell$. Then

$T_{\min}(P^{\rightarrow},\mathcal{W}) := \Sigma_{a \in P}\, t^{a}_{\min}$. $P^{\rightarrow}$ is a **longest min-path** (or a **critical path**) of **M** if $T_{\min}(P^{\rightarrow},\mathcal{W})$ is *maximal*, assuming P contains only mandatory tasks, the set $\{a_{i1}, a_{i2}, \ldots, a_{i\ell}\}$ contains of **critical activities,** the minimum time is:

$$T_{\min}(\mathbf{M},\mathcal{W}) := \max_{P \text{ real path}} T_{\min}(P^{\rightarrow},\mathcal{W}) .$$

The algorithm will change all ? to either X or ∅ in M in the diagonal in PHASE ONE and in the off-diagonal in PHASE TWO, the resulted matrices are the **in-** and **out-** (*-diagonal*) **closures** of M.

In PHASE THREE we have to find a feasible *project schedule* $w^{\rightarrow}$: we have to select from the completion modes, which influence the budget, project duration, and increase in system reliability.

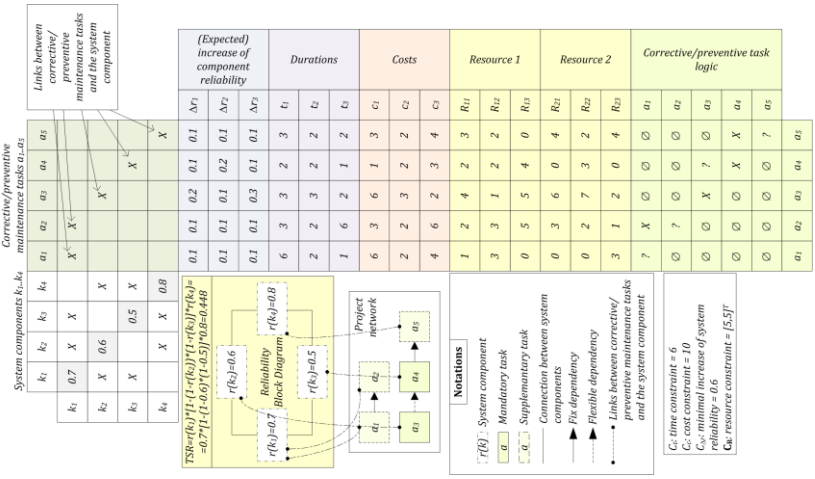


Figure 1. Matrix representation of the hybrid multimode resource constrained maintenance

The first domain of the matrix representation (see Figure 1) describes the reliability diagram of the system, and this domain is symmetric. The diagonal represents the component reliability (or, in another model, the component availability). In the next domain, maintenance tasks are assigned to the system components.

The estimation of the increase in component reliability is based on the completion mode of the tasks.

3. Problems to solve

Now, we define the *hybrid multimode resource-constrained maintenance project scheduling problem* (HMCRMPSP).

In PHASE ONE of our algorithm we have to answer Problem 1 below.

Problem 1 PHASE ONE. We have to find a feasible project *scenario* $S\subseteq A$. Let $C_c, C_t, C_{\Delta r}\in \mathrm{R}^+$ be given such that $C_{\min}(\mathrm{M},\mathcal{W}) \le C_c$, $T_{\min}(\mathrm{M},\mathcal{W})\le C_t$ and $\Delta r_{\max}(\mathrm{M},\mathcal{W},\mathrm{K})\ge C_{\Delta r}$. We have to find an in-closure M′ of M such that

in problem 1/1 : $\Delta r(\mathrm{M}',\mathcal{W},\mathrm{K}) \to \max$,
in problem 1/2a: $C(\mathrm{M}',\mathcal{W}) \to \min$,
in problem 1/2b: $T(\mathrm{M}',\mathcal{W}) \to \min$,
assuming $C_{\min}(\mathrm{M}',\mathcal{W})\le C_c$, $T_{\min}(\mathrm{M}',\mathcal{W})\le C_t$ and $\Delta r_{\max}(\mathrm{M}',\mathcal{W},\mathrm{K})\ge C_{\Delta r}$.

In PHASE TWO we have to find a feasible project *structure* $X=(\mathrm{S},\prec,\sim)$. Since the dependencies are selected to be excluded or included, the selection does not influence the budget and the increase in system reliability.

Problem 2 PHASE TWO. Let M′ be a solution to PHASE ONE and C_t be given such that $T_{\min}(\mathrm{M}',\mathcal{W})\le C_t$. Now find a *structure* (*off-closure* M"of M′) such that $T(\mathrm{M}'',\mathcal{W})\to\min$, assuming $T_{\min}(\mathrm{M}'',\mathcal{W})\le C_t$.

In PHASE THREE, we have to find a feasible *project schedule* $w^{\rightarrow}$, i.e. to select from the completion modes, which influence the budget, project duration and increase in system reliability, the resource *demands* must be considered, too.

Problem 3 PHASE THREE. Let M" be a solution to PHASE TWO, K, $\mathcal{W}$ and C_c, C_t, $C_{R1},C_{R2},\ldots,C_{R\rho}\in\mathrm{R}^+$, $\mathbf{C_R}=[C_{R1},C_{R2},\ldots,C_{R\rho}]^{\mathrm{T}}$ be given *such that* $C_c\ge C_{\min}(\mathbf{M}',\mathcal{W})$, $T_{\min}(\mathbf{M}'',\mathcal{W})\le C_t$, $\mathbf{TPR}_{\min}(\mathbf{M}'',\mathcal{W})\le\mathbf{C_R}$ and $\Delta R_{\max}(\mathbf{M}',\mathcal{W},K)\ge C_{\Delta R}$. Now find $w^{\rightarrow}$ such that

in problem 3/1: $\Delta\mathbf{r}(w^{\rightarrow},K) \to \max$
in problem 3/2a: $\mathbf{c}(w^{\rightarrow}) \to \min$
in problem 3/2b: $\mathbf{t}(w^{\rightarrow}) \to \min$
assuming $\mathbf{t}(w^{\rightarrow})\le C_t$, $\mathbf{c}(w^{\rightarrow})\le C_c$, $\mathbf{TPR}(w^{\rightarrow})\le\mathbf{C_R}$ and $\Delta\mathbf{r}(w^{\rightarrow},K)\le C_r$.

4 Complexity of the problem

Kosztyán 2015, 2018 [2,3] has shown that the number of possible project structures is 2^d where d is the number of flexible dependencies. If uncertain task completion exists, the number of possible project scenarios is 2^u where $0\le u\le n$. A project scenario has $2^{di}\le n(n\text{-}1)/2$ project structures. Therefore the number of possible project structures is $\mathrm{N}=\sum_{i:=1}^{2^u}2^{d_i}$. If every task can be

completed via m possible modes, the total number of possible project structures is $\mathrm{M} = \mathrm{N}n^m = n^m\Sigma_{i:=1}^{2^u} 2^{d_i}$. So, brute force algorithms cannot be applied even in small problems. For example, if only *one* project scenario exists, all dependencies are flexible and there are only 3 completion modes (n=8, d=n(n-1)/2=28, m=3), then $\mathrm{M} = 2^{28}*82 = 4{,}294{,}967{,}296$.

In *phase one* and *phase two*, the decision tree is a binary heap. Every node in this tree has sharp bounds for the time and cost demands and for the increase in system reliability. In this case, the expected computation time is $O(u+d)$ (see [2]). Since the output of *phase two* is an MRCPSP, in the final phase, we can use heuristic or exact methods to evaluate the specified project structure. Unfortunately, in this case, the decision tree is not a binary heap but an m -heap, where m is the number of modes. Therefore, no fast tree traversal algorithm exists for finding the best project schedule within polynomial time. Since few alternatives usually exist in practice, in the final phase, we implement ACO in accordance with [4] to determine the project schedule.

5 Simulation

The aim of this section to compare the impacts of the project and structural parameters or the maintainability (for both serviceability and repairability). The goal is to compare the effectiveness of traditional long-term shutdowns, where (almost) all system components are checked and/or improved, and novel continuous maintenance approaches, which involve fewer maintenance tasks that can be completed in parallel.

Project plans are generated by the free project generator ProGen [5] and one production/maintenance project is from the empirical project database [6]. In terms of the possible constraints there are 291,600 possible settings, and we generate 20 system structures for each. Therefore, 5,832,000 maintenance projects were solved.

The tradeoff assumption is ignored when analyzing the results of the proposed method. The performance of the maintainability methods is compared when different target functions are applied and flexible tasks and flexible dependencies are allowed (Fig. 1a) and when flexibility is not allowed, i.e. every corrective/preventive task in the maintenance project is decided to complete (Fig. 1b). Figure 1b) shows the results of a long-term system shutdown, while Figure 1a) shows the result of the agile/lean approaches, where

to increase system reliability/decrease the budget/decrease the project duration, flexible (nonmandatory) tasks and dependencies can be excluded from the project.

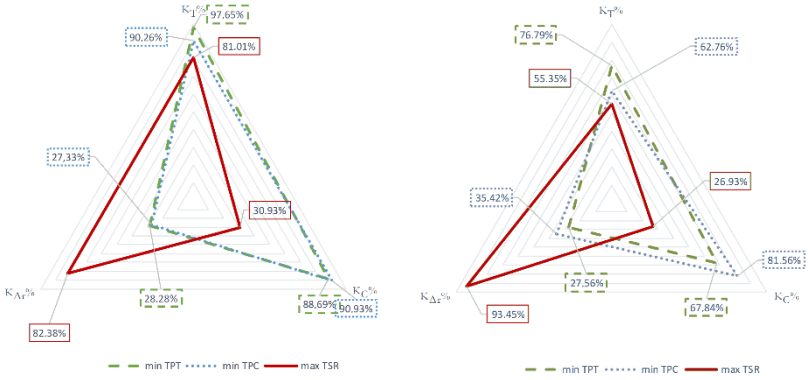


Figure 2: Results in terms of repairability $K_{\Delta r}\%$ and serviceability $K_C\%$, $K_T\%$, **a)** Flexible tasks/dependencies are allowed **b)** Flexibility is not allowed

6 Conclusion

In this paper a novel problem, namely, the hybrid multimode resource-constrained maintenance project scheduling problem is defined. This problem integrates system reliability and preventive maintenance tasks. The proposed method can improve both the repairability and the serviceability. It can organize a maintenance project to minimize the maintenance project tasks while maintaining a specified budget and achieving the desired increase in system reliability (serviceability). At the same time, it can maximize the reliability of the system when keeping time/cost/resource constraints (repairability). After applying the proposed matrix-based maintenance management method the set of maintenance tasks (result of phase 1), their sequence of completion (result of phase 2) and the time/cost/growth of system components (result of phase 3) can be specified.

Since not all the system components will be maintained in a preventive maintenance project scheduling problem, the problem specifies a flexible project that consists of not only mandatory but also supplementary tasks. The simulation test showed that a more reliable system can be effectively maintained and that continuous maintenance is more effective.

The developed matrix-based method and the proposed exact algorithm may be important and essential components of a maintenance project expert system. This method can be applied in a computer-aided maintenance system (CMMS), where reliability/availability values are generated from a diagnostic system, but this method can schedule the maintenance tasks.

The proposed algorithm is a static algorithm; however, the system reliability and the reliability of system components decrease as a function of time. By considering the time dependence and integrating the proposed model with forecast methods, this model can be extended to support predictive maintenance, and a new *predictive maintenance project scheduling problem* can be specified.

References

[1] **Zs.T.Kosztyán, A.Pribojszki-Németh, I.Szalkai:** *Hybrid Multimode Resource-Constrained Maintenance Project Scheduling Problem*, in: Operations Research Perspectives, 6 (2019), 100129, 13 pp., https://doi.org/10.1016/j.orp.2019.100129

[2] **Zs.T.Kosztyán**: *Exact algorithm for matrix-based project planning problems*, Expert Syst. Appl. vol. 42, pp. 4460-4473, 2015; https://doi.org/10.1016/j.eswa.2015.01.066

[3] **Zs.T.Kosztyán:** *Serviceability of large-scale systems*, Simulation Modelling Practice and Theory 84 (2018) 222-231.

[4] **H.Li, H.Zhang:** *Ant colony optimization-based multimode scheduling under renewable and nonrenewable resource constraints*, Automation in Construction 35 (2013) 431-438.

[5] **R.Kolisch, A.Sprecher:** PSPLIB - *a project scheduling problem library*: ORSEP *operations research software exchange program*, European Journal of Operational Research 96 (1) (1997) 205-216.

[6] **J.Batselier, M.Vanhoucke:** *Construction and evaluation framework for a real-life project database*, International Journal of Project Management 33 (3) (2015) 697-710.

The board packing problem: Packing rectangles into a board to maximize profit

G. Dosa[1], L.M. Hvattum[2], T. Olaj[3], Zs. Tuza[4,5]
[1]University of Pannonia, dosagy@almos.uni-pannon.hu
H-8200 Veszprém, Egyetem u. 10, Hungary
[2]University of Molde, Lars.M.Hvattum@himolde.no
P. O. Box 2110, N-6402 Molde, Norway
[3]University of Pannonia, tomas@olaj.no
[4]University of Pannonia, tuza@dcs.uni-pannon.hu
[5]Rényi Institute, 1053 Budapest, Reáltanoda u. 13–15, Hungary

Abstract: **We introduce the board packing problem, defined on a rectangular board to be partially covered by a set of rectangles. The board corresponds to some physical area where we can make investments, and the rectangles correspond to potential investments. By placing a rectangle on the board, on one hand we pay for the use of the rectangle, and on the other hand we make profit. The total profit minus the costs of the placed rectangles should be maximized. We have proved that the problem is NP-hard. We introduce a mixed-integer programming model, and solve some instances with the commercial solver CPLEX. We also propose some ideas on how bigger instances can be handled by heuristic methods.**

Introduction

We introduce a packing problem we call the board packing problem (BPP, for short). A rectangular board with m rows and n columns is given. Let $M = \{1, \ldots, m\}$ be the set of rows and $N = \{1, \ldots, n\}$ be the set of columns. Each position $(i, j) \in M \times N$ of the board has an integer value $g_{i,j}$ called gain (where the gain can be also negative).

A set K of rectangles (or items) are given, each $k \in K$ with a given height $h_k \in \{1, \ldots, m\}$, width $w_k \in \{1, \ldots, n\}$, and cost $c_k \geq 0$.

The objective is to purchase rectangles to place on the board so as to maximize the profit, which is equal to the values of the covered positions minus the cost of the purchased rectangles. We assume that the rectangles will be placed so that their sides are parallel to the sides of the board.

0	0	0	0	0	0
0	1	1	1	0	0
0	1	1	1	0	0
0	1	1	1	1	0
0	0	1	1	1	0
0	0	1	1	1	0
0	0	0	0	0	0

Figure 1: An example board filled with 0 and 1

We assume that rectangles cannot overlap. This is an important restriction, as illustrated by Figure 1. In the example shown, there are $m = 7$ rows and $n = 6$ columns, and $g_{i,j} \in \{0, 1\}$. Consider two squares (so $K = 2$), both of size 3. Then, all positive values of the board (the positions where $g_{i,j} = 1$) can be covered only if the squares overlap each other.

Application

An application of the BPP is when the board represents some area in a city or a country. The rectangles represent facilities that can be built in certain positions. The covered g values represent the profit that is realized from the activity of the facilities. A negative g value means that it is preferable to not cover the point, for example if the location is already occupied. If it is, in spite of that, covered, some amount of penalty or expropriation cost must be paid. The purchasing cost of rectangles corresponds to the cost of installing a new facility.

Previous related models from the literature and new results

The first model in the literature that address a *similar* problem is probably the maximal covering location problem (MCLP), defined by Church and ReVelle [8]: A finite set I of users is given, each user $i \in I$ having a given demand $a_i \geq 0$. Also given is a set J of facilities. We want to find p facilities in the set J so that we maximize the covered demands. A point i is covered if there is at least one chosen facility so that its *distance* from i is not bigger than R, the fixed positive cover radius. The distance between two points is measured by some norm if the demand points are in the two-dimensional plane, or it can be

the shortest distance if the demand points are given as vertices of a graph. There are many methods to solve optimally or near-optimally the problem in different settings, see e.g. [22].

The BPP belongs to this class of problems, but there are significant differences: In our model, R (the cover radius) is not fixed: it is smaller for a smaller square and bigger for a bigger square, so in our case R depends on the facilities. On the other hand, the BPP has the restriction that the demand points are in a grid.

Since the introduction of the MCLP, a large number of related papers has appeared. It is claimed in the introduction of [5] that MCLP is known to be NP-hard. This claim is supported by a reference to a paper of Megiddo et al. [20]. In fact, [20] considers a model where the demand points are vertices of a graph, and the facilities should be put into the 'points' of the graph (where a point of a graph can be some vertex or some 'inner point' of some edge also). In this version the model is clearly NP-hard because it is NP-hard to find a minimum dominating set in a graph. (Let the demand points be the vertices, all weights a_i are unit, and the covering radius is uniformly $R = 1$.) This NP-hardness reduction works for general graphs but not for grids where we want to cover the board with rectangles. As a consequence, the intractability of the BPP does not follow in this way.

Our main theoretical result in this direction is that also the BPP is NP-hard, even for some rather restricted classes of instances. Contrary to MCLP, the proofs which strongly use some constructions from Masek [18], Bereg et al. [3], and Fowler et al. [13], are far from being trivial, and space limitation does not allow us to include them here.

Theorem 1 *[10] The BPP problem is NP-hard even when restricted to instances from either of the following classes:*

(a) The g values are restricted to be either 2 or a large negative value.

(b) The g values are 0 or 1, and all rectangles of K are squares with the same fixed size at least 2.

There are two main areas in the related literature. In one of them, the problem is approached from a geometric point of view, where a given rectilinear object in the plane should be covered by a certain number of smaller items; see [1], [2], [3], [4], [5], [7], [9], [14], [15], [16], [17], [18], [19], [21], and the papers cited therein. The other main branch is the wide area of (some kind of) facility location, see e.g. [6, 11, 12, 23].

A MIP formulation and solution with CPLEX

To create a binary integer programming model for the problem, we define A_k as the set of possible coordinates[1] (i, j) for the top left corner of rectangle k. Let p_{ijk} be the profit if rectangle k is placed with its top left coordinates at (i, j). That is, $p_{ijk} = \sum_{u=i}^{i+h_k-1} \sum_{v=j}^{j+w_k-1} g_{uv} - c_k$. Let x_{ijk} be a binary variable, taking the value 1 if rectangle k is placed with its top left corner at (i, j), and 0 otherwise. Let B_{ijk} be the set of positions (u, v) for the top left corner of rectangle k that will result in position (i, j) being covered by that rectangle. Then, given that rectangles are not allowed to overlap, we get:

$$\max \sum_{k \in K} \sum_{(i,j) \in A_k} p_{ijk}\, x_{ijk} \tag{1}$$

$$\sum_{(i,j) \in A_k} x_{ijk} \leq 1, \qquad k \in K \tag{2}$$

$$\sum_{k \in K} \sum_{(u,v) \in B_{ijk}} x_{uvk} \leq 1, \qquad i \in M, j \in N \tag{3}$$

$$x_{ijk} \in \{0, 1\}, \qquad k \in K, (i, j) \in A_k \tag{4}$$

Here, the objective function (1) maximizes the profit from the covered positions. Constraints (2) make sure that only one location is chosen for each rectangle. To make sure that rectangles do not overlap, constraints (3) are defined for each location, with the left-hand side counting the number of rectangles that cover the location.

Testing the capabilities of CPLEX

The model for BPP given above is defined using binary variables. As the board size is $|M| * |N|$ and the number of rectangles is $|K|$, the number of binary variables is $O(|M| * |N| * |K|)$, and the number of constraints in the MIP formulation is $O(|M|*|N|+|K|) \sim O(|M|*|N|)$.

Table 1 shows the running times for different input classes, where M, N, and K mean the number of rows of the table, the number of columns, and the number of rectangles, respectively. For sake of simplicity the items are chosen to be equal squares, and S means the size of a square.

[1] By the corner with 'coordinates' (i, j) we mean the cell in the i^{th} row and j^{th} column of the rectangular board.

Table 1: Results of running CPLEX to solve the BPP

IC	M	N	K	S	Avg. tot c	Avg. secs	Avg. tot Obj
1	10	10	4	5	224	0.934	73
2	10	10	4	5	12	0.441	46
3	10	10	4	5	107	0.552	54
4	10	10	4	5	12	0.591	13
5	20	20	6	6	699	1.098	219
6	20	20	6	6	42	1.108	163
7	30	30	8	8	1557	1.147	464
8	40	40	9	10	2823	1.590	819
9	40	40	10	10	2989	1.704	932
10	100	100	10	10	3999	10.932	1270

The cost c_k of a rectangle is chosen randomly. The g values are chosen uniformly at random from the interval $[0, 9]$ and then rounded down. For each input class we generated ten instances and after running the solver we took the average of the running times, as shown in the table together with the average of the optimum values. All instances are solved to optimality.

Other Algorithms

Since the problem is NP-hard and becomes intractable for bigger instances, it is worth thinking about the option of approaching the BPP with other methods. First of all, let us mention that a simple brute force method is also able to solve the problem if its size is small. However, an instance with a board of size 10×10 and 4 squares of size 5 takes longer than 1,000 seconds to complete, so only relatively small problems can be solved in this way.

Another option is the application of a greedy construction heuristic: First we put the rectangles into a certain order. This can be e.g. a decreasing order according to the areas of the rectangles. Then we place the first rectangle in the list to its best position (only taking into account this only one item for a moment); we can do this step by brute force. Then we place the second item to its best location, avoiding any overlap with the previously placed item, and so on. This procedure clearly takes polynomial time only. Another option is to apply some kind of metaheuristic, like tabu search or a genetic algorithm.

Acknowledgments

We acknowledge the financial support of Széchenyi 2020 programme under the project No. EFOP-3.6.1-16-2016-00015, and of the National Research, Development and Innovation Office – NKFIH under the grant SNN 129364.

References

[1] E. Allender, L. Hellerstein, P. McCabe, T. Pitassi, and M. Saks, Minimizing DNF formulas and *AC* circuits given a truth table, Electronic Colloquium on Computational Complexity, Report No. 126, 22 pages, 2005.

[2] L. J. Aupperle, H. E. Conn, J. M. Keil, and J. O'Rourke, Covering orthogonal polygons with squares, 26th Annual Conference on Communication, Control and Computing, pages 97–106, 1988.

[3] S. Bereg, S. Cabello, J. M. Diaz-Banez, P. Pérez-Lantero, C. Seara, and I. Ventura, The class cover problem with boxes, Computational Geometry, 45(7), 294–304, 2012.

[4] O. Berman and D. Krass, The generalized maximal covering location problem, Computers & Operations Research, 29, 563–581, 2002.

[5] R. Blanquero, E. Carrizosa, and B. G.-Tóth, Maximal Covering Location Problems on networks with regional demand, Omega, 64(C), 77–85, 2016.

[6] O. Cakir and G. O. Weselowsky, Planar expropriation problem with non-rigid rectangular facilities, Computers and Operations Research, 38, 75–89, 2011.

[7] S. Chaiken, D. J. Kleitman, M. Saks, and J. Shearer. Covering regions by rectangles, SIAM J. Algebraic and Discrete Methods, 2, 394–410, 1981.

[8] R. Church and C. ReVelle, The Maximal Covering Location Problem, Papers of Regional Science Association, 32, 101–18, 1974.

[9] J. C. Culberson and R. A. Reckhow, Covering polygons is hard, SFCS '88, Proceedings of the 29th Annual Symposium on Foundations of Computer Science, pages 601–611, October 24–26, 1988.

[10] G. Dosa, L.M. Hvattum, T. Olaj, Zs. Tuza, The Board Packing Problem: Complexity results, manuscript, 2020

[11] Z. Drezner, On the rectangular p-center problem, Naval Research Logistics, 34, 229–234, 1987.

[12] L. Dupont, M. Lauras, and C. Yugma, Generalized covering location problem with multiple-coverage: Exact and heuristic method, 7th IFAC Conference on Manufacturing Modelling, Management and Control, International Federation of Automatic Control, June 19–21, pages 442–447, Saint Petersburg, Russia, 2013.

[13] R. J. Fowler, M. S. Paterson, and S. L. Tanimoto, Optimal packing and covering in the plane are NP-complete, Information Processing Letters, 12(3), 133–137, 1981.

[14] D. S. Franzblau and D. J. Kleitman, An algorithm for covering polygons with rectangles, Information and Control, 63, 164–189, 1984.

[15] E. Győri, A minimax theorem on intervals, Journal of Combinatorial Theory Ser. B, 37(1), 1–9, 1984.

[16] J. Y.-T. Leung, T. W. Tam, C. S. Wong, G. H. Young, and F. Y. L. Chin, Packing Squares into a Square, Journal of Parallel and Distributed Computing, 10, 271–275, 1990

[17] P. R. S. Mahapatra, P. P. Goswami, P. P. Goswami, and S. Das, Covering points by isothetic unit squares, Proceedings of the 19th Annual Canadian Conference on Computational Geometry, CCCG 2007, August 20-22, 2007, Carleton University, Ottawa, Canada, 4 pages, 2007.

[18] W. J. Masek, Some NP-complete set covering problems, unpublished manuscript, 1978.

[19] N. Megiddo and K. J. Supowit, On the complexity of some common geometric location problems, SIAM Journal on Computing, 13(1), 182–196, 1984.

[20] N. Megiddo, E. Zemels, and L. Hakimi, The maximum coverage location problem, SIAM Journal on Algebraic and Discrete Methods, 4(2), 253–261, 1983.

[21] M. Mukherjee and K. Chakraborty, A polynomial-time optimization algorithm for a rectilinear pratitioning problem with application in VLSI design automation, Information Processing Letters, 83, 41–48, 2002.

[22] M. G. Resende, Computing approximate solutions of the maximum covering problem with GRASP, Journal of Heuristics, 4, 161–177 (1998)

[23] C. S. ReVelle, H. A. Eiselt, and M. S. Daskin, A bibliography for some fundamental problem categories in discrete location science, European Journal of Operational Research, 184 (2008), 817–848.

Parameter optimization of Q-Learning Motivated Algorithm in process scheduling

Gy. Ábrahám[1], Gy. Dósa[1], T. Dulai[1], Á. Werner-Stark[1]
[1]University of Pannonia, Faculty of Information Technology,
abraham.gyula@virt.uni-pannon.hu, dosagy@almos.uni-pannon.hu,
dulai.tibor@virt.uni-pannon.hu, werner.agnes@virt.uni-pannon.hu
8200 Veszprém, Egyetem Street 10. Hungary

Abstract: **In the case of industrial processes the processing time and cost are important factors. To make such a process more effective the resource-based scheduling is inevitable. An algorithm was developed for resource-based scheduling which includes some reinforcement learning aspects to make the scheduling more effective and avoid the pure heuristic solutions. This method is called Q-Learning Motivated Algorithm (QLM) and its several parameters must be set correctly. The parameter setting can be done manually, but it can be hard to find an appropriate setting and it takes too long time. To automatize the parameter setting process we chose the simulated annealing heuristic.**

Introduction

Our problem belongs to the area of process scheduling. In process scheduling problems, we consider several processes, each consist of several tasks, and a set of resources. The resources need to be assigned to the tasks of the processes such that several constraints are satisfied. In our study, the objective is to minimize the overall execution time of all processes (the makespan).

Process scheduling is a computationally hard problem. Such hard problems are often solved by heuristic methods. We developed an algorithm that uses a reinforcement learning algorithm called Q-Learning. Our method is called Q-Learning Motivated Algorithm (QLM). [3] The scheduling problem we are dealing with can be described as follows.

There are n tasks and m machines. The machines have their own machine times for each task, so machines are unrelated. These machines must be assigned to tasks in a way to minimize the time of the whole process. Between tasks, precedence constraints can be defined which strictly determine the processing order of the tasks. At any time, any machine can process at most one task, and the execution of a task cannot be preempted. As an illustration, a small example is given in Figure 1. The tasks are scheduled on the machines

in certain time slots such that the precedence constraints are satisfied. The precedence constraints are not shown on the figure because it would make it too crowded. These constraints are the following:

- J_7 is preceded by J_8 and J_9.
- J_{10} is preceded by J_{11}, J_{12} and J_{13}.
- J_{14} is preceded by J_{15}, J_{16}, J_{17} and J_{18}.
- J_{19} is preceded by $J_{20}, J_{21}, J_{22}, J_{23}$ and J_{24}.

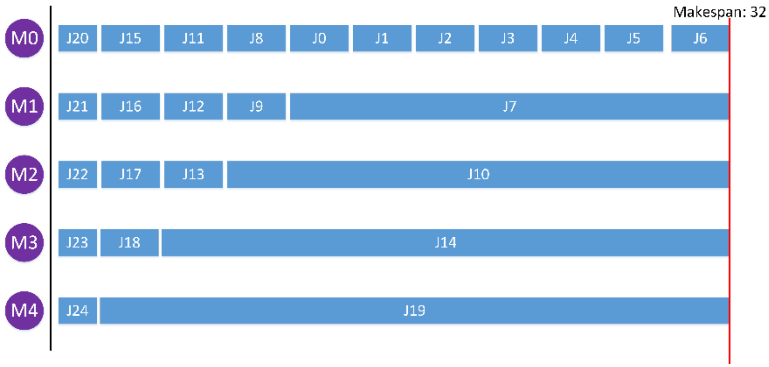


Figure 1: An example schedule

This problem contains 25 tasks and 5 machines, where each machine has its own execution time for each task. In Figure 1 an optimal solution can be seen with 32 time units. This is a very special input where the optimal solution is difficult to find. In easier cases, the algorithm always gives an optimal solution.

To solve this scheduling problem the developed QLM was used. The QLM algorithm has several parameters that must be set before scheduling. All these parameters are related to reinforcement learning.

Hermann et al. [1] and Liu and Yang [2] are the only papers we are aware of that deal with unrelated machine scheduling with precedence constraints represented by chain graphs.

In [1] three kinds of lower bounds (LB1, LB2, and LB3) and several heuristics for this problem are introduced. A case study on 33 inputs compares the solutions of the heuristics to the maximum of the lower bounds.

Liu and Yang [2] propose a more efficient heuristic, and their method is also applicable to more general problems than those are considered here.

The aim of this work is to optimize the parameters of the QLM algorithm and this work is a continuation of [3]. Some preliminary investigations which are related to this work can be found in [4]. [5] also deals with parameter

optimization, which is related to a similar, combinatorial optimization problem.

Parameters of QLM

The QLM algorithm is based on the Q-Learning reinforcement learning technique. So, the parameters of QLM are identical to that of the Q-Learning algorithm. The parameters are the following:

- $\alpha \epsilon [0,1]$
- $\gamma \epsilon [0,1]$
- $R: \mathbb{R} \rightarrow \mathbb{R}$
- number of iterations

Our QLM method can be summarized as follows.

First, the algorithm determines a good order for the tasks based on the Q-values. Q-value is computed for pairs of tasks determines how good if a task is followed by another task. This order is computed based on probabilities that come from the Boltzmann distribution. After the order is computed the tasks are scheduled greedily and the makespan is calculated. Based on the makespan the reward is also calculated which is basically the reinforcement in the algorithm. In the final step the Q-values are updated. These steps are repeated for a certain number of iterations (this is described in detail in [3]).

The α parameter is the learning rate (step size), it determines to what extent the newly acquired information overrides the old one. If this parameter is 0, the agent avoids new information and learns nothing. If it is 1 the agent only considers the new information and avoids prior knowledge (experience).

The γ parameter is the discount factor. In reinforcement learning discount factor determines the importance of future rewards. If it is set to 0, the agent is short-sighted. If it is 1, the agent will strive for long term reward. The γ is used to make the agent more interested in the immediate reward rather than the future reward.

For a lot of problems, the α is set to 0.8 and γ is set to 0.9 or 0.99. Most of the problems can be solved with good results with these settings.

The R reward function in our case is a linear function that maps the current makespan into a positive or negative value. If the current makespan is greater than the best makespan so far, the reward is positive, if the current makespan is lower the reward is negative.

The iteration number describes the size of an epoch during the algorithm computes one schedule.

Parameter optimization

The parameter optimization is a method to optimize parameters of an algorithm to make it more effective. The optimization is the automation of setting parameters. To optimize parameters several heuristics are used, e.g., local search, simulated annealing, or Tabu search.

In our solution, the simulated annealing was used. Simulated annealing (SA) is a heuristic search algorithm. It is useful for finding global optimum in the presence of large numbers of local optimum. The SA uses the objective function of the problem and generates solutions (or instances) from problem configurations. Figure 2 shows the basic concept of SA.

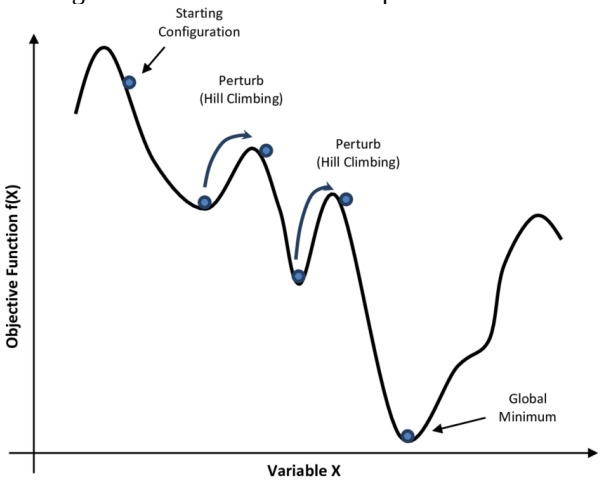


Figure 2: Simulated annealing [6]

In our case, we compute the makespan several times (10 times) with the same parameter settings (configuration). This vector of makespans performs as an "instance" for the SA algorithm.

There was no parameter setting that would give the algorithm the optimal solution. The best solution to the QLM computed is 34 time units.

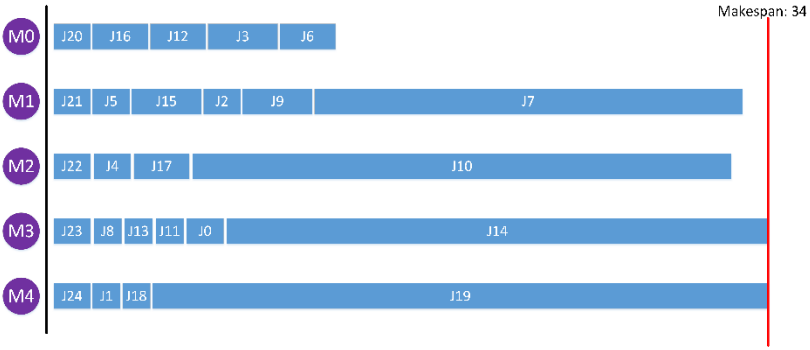


Figure 3: Result of QLM

In Figure 3 a computational result can be seen. Due to the precedence constraints, it is hard to find the optimal solution. The optimal solution can be achieved only if the algorithm follows the next steps:

- the first five tasks are $J_{20}, J_{21}, J_{22}, J_{23}$ *and* J_{24} (arbitrary order)
- the sixth task is J_{19}
- the next four tasks are $J_{15}, J_{16}, J_{17}, J_{18}$ (arbitrary order)
- the eleventh task is J_{14}
- the next three tasks are J_{11}, J_{12}, J_{13} (arbitrary order)
- the fifteenth task is J_{10}
- the next two tasks are J_8, J_9 (arbitrary order)
- the eighteenth task is J_7
- and finally, the remaining tasks are in arbitrary order.

If the algorithm can't find this order the solution will not be optimal.

The optimization algorithm runs the QLM algorithm with given parameters (these are randomly initialized), and the QLM was placed inside a simulated annealing algorithm where the parameters of QLM are optimized.

During the simulated annealing the parameters are changed randomly which means the algorithm selects one parameter randomly to change; to decrease or to increase. The direction of change is also determined randomly. The simulated annealing algorithm used the following probability to compute whether to accept the worse solution.

$$P_{accept} = e^{\frac{v_{old} - v_{new}}{T}}, \text{ where}$$

$$\delta = v_{old} - v_{new}$$

The rule is simple. If $\delta \geq 0$ then the new value is lower or equal than the old one, which is good because we want to minimize the makespan. The new instance is accepted. If $\delta < 0$ means the new value is greater than the old one.

In the latter case, the algorithm computes the probability of acceptance of the new instance.

The optimization process used the current makespan to describe the goodness of the actual instance and this makespan was compared to the best makespan so far. Unfortunately, this kind of solution did not show convergence in parameters. In the next solution, the instance was changed. An instance was represented by a super-instance which includes 10 simple instances. Each simple instance includes 3 makespans. Inside a simple instance each makespan was computed with the same parameter setting. In the new solution, a super-instance was described as follows: among the simple instances which instance includes the best possible makespan. For example, if there are two super-instances *A* and *B*, and each of them has 10 simple instances. In the case of *A*, the best possible makespan shows up in

the second simple instance and in the case of *B* it shows up in the eighth instance, then *A* is better than *B*.

Based on our experiments this kind of solution is sensitive for the parameter setting. Three parameter settings were selected with respect for the reward function.

- A: $(-5, -1, 10)$
- B: $(-10, 0, 20)$
- C: $(-1, 0, 5)$

These numbers are the rewards. The first number is given if the current makespan is worse than the best so far. The second number is given if there is no difference, and the third one is for the improvement. These rewards are given inside simple scheduling.

In the case of A setting, the possible best makespan is found very quickly, usually before the fifth iteration. The following result shows ten iterations with iteration numbers. Each iteration comes from a ten-size iteration batch and represents that iteration which includes the best possible makespan.

$$(I_1, I_3, I_2, I_4, I_1, I_5, I_3, I_2, I_5, I_1)$$

The average of iteration numbers is 2.7 which means the algorithm found the best possible makespan in the early iterations.

The B setting gave a medium result. The best possible makespan was found between the fifth and seventh iterations.

$$(I_5, I_5, I_5, I_6, I_7, I_8, I_8, I_5, I_4, I_6)$$

The average is 5.9.

The C setting gave the worst result. In this case, the best possible makespan was found very slowly, usually in the last iterations.

$$(I_6, I_7, I_6, I_8, I_{10}, I_{10}, I_9, I_7, I_9, I_{10})$$

The average in this case is 8.2.

Conclusion

To optimize the parameters of the QLM algorithm is quite hard because most of the time the result of scheduling was not influenced by the changes of the parameters. A new instance type was introduced called super-instance which includes several simple instances. Due to this the description of the goodness of a super-instance was introduced, too. A super-instance is better if the best possible makespan can be found among the first iterations and worse if it is among the later iterations. We found that the reward function can show a little sensitivity in this case and a reward configuration was found that gave good results compared to two other settings.

The convergence in a parameter is a very rare situation and most of the time the parameters did not show any kind of convergence. Of course, there is a connection between the parameters and the results but in reinforcement learning the values from the literature work very well in most cases. The basic rule based on the literature it is enough to set alpha and gamma parameters to high (0.8 and 0.9, respectively) and divide the rewards into parts based on the problem. In our case the reward function was split into three parts: negative change, no change and positive change.

As far as we know Q-Learning has never been applied for this type of problem before, except in our previous work [3], where our QLM algorithm is introduced and investigated and also compared to other heuristic algorithms.

Acknowledgment

We acknowledge the financial support of Széchenyi 2020 under the EFOP-3.6.1-16-2016-00015. Dósa was also supported by the National Research, Development and Innovation Office -- NKFIH under the grant SNN 129364.

References

[1] J. Herrmann, J. M. Proth, N. Sauer, “Heuristics for unrelated machine scheduling with precedence constraints,” European Journal of Operational Research, 102, pp. 528-537, 1997.

[2] C. F. Liu, S. L. Yang, “A heuristic serial schedule algorithm for unrelated parallel machine scheduling with precedence constraints”, Journal of Software 6(6), pp. 1146-1153, 2011.

[3] Gy. Ábrahám, P. Auer, Gy. Dosa, T. Dulai, A. Werner-Stark, “A Reinforcement Learning Motivated Algorithm for Process Optimization,” PERIODICA POLYTECHNICA-CIVIL ENGINEERING 63:4, pp. 961-970, 2019.

[4] Gy. Ábrahám, Gy. Dosa, A. Werner-Stark, T. Dulai, “Learning the parameters of a reinforcement learning algorithm for process optimization”, Veszprém Optimization Workshop, pp. 11-11, 2019.

[5] A. Benko, Gy. Dosa, Zs. Tuza, “Bin Packing/Covering with Delivery solved with the evolution of algorithms,” BIC-TA, pp. 298-302, 2010.

[6] O. Ghasemalizadeh, S. Khaleghian, S. Taheri, “A review of optimization techniques in artificial networks,” Int. J. of Adv. Res. 4 (9), 2016.

Identifiability analysis and parameter estimation of a SEIR epidemic model

Gergely Horváth[1] and Gábor Szederkényi[1]
[1]Faculty of Information Technology and Bionics, Pázmány Péter Catholic University, H1083 Budapest, Práter u. 50/a

Abstract: **In this contribution, we study the identifiability of SIR and SEIR type epidemic models. We show that the parameterized model is globally structurally identifiable if the measured output is the cumulative number of infected people. Based on the identifiability results, we perform the parameter estimation using the publicly available data of the COVID-19 epidemic in Italy, taking into consideration the time-varying effects of the restrictions.**

Introduction

The dynamical modeling of epidemic processes is of key importance in understanding, predicting and controlling disease spread. The most commonly used lumped epidemic models are the SIR (susceptible - infected - recovered) and SEIR (susceptible - exposed - infected - recovered) which can be written in the form of smooth (polynomial) nonlinear ODEs. It is important to remark that these models are compartmental, and therefore, formally they can be written as chemical reaction networks, where the 'reactions' are transitions between different compartments [1]. Usually we do not have up-to-date information about all state variables of an epidemic process, especially on the number of susceptible and exposed individuals. Probably the most reliable information source is the cumulative number of infected people which is not part of the SIR or SEIR model, but it is easy to append it as an additional differential equation. Since the differential equation of the cumulative infected number does not contain all the model parameters, it is crucial to check the structural identifiability of the model, i.e. to investigate if this observed output contains enough information for identification purposes. There are numerous results in the literature on the identifiability of compartmental and disease models [2]. However, we haven't encountered the same calculation that is summarized in this

paper. Therefore, the aims of this contribution are: 1) the structural identifiability analysis of SIR and SEIR models from the cumulative infected number, and 2) the parameter estimation of a SEIR model using the data of the COVID-19 pandemic in Italy.

Basic notions: identifiability and epidemic models

Identifiability of dynamical models in the form of nonlinear ODEs

The rough definition of the structural identifiability of parametric dynamical models is the following [3]. Let us have the parameterized system operator, $M(\boldsymbol{p})$, and $\hat{\boldsymbol{p}}$ and $\boldsymbol{p}^*$ two parameter vectors. If the equality $M(\hat{\boldsymbol{p}}) = M(\boldsymbol{p}^*)$ implies $\hat{\boldsymbol{p}} = \boldsymbol{p}^*$ then our model is structurally globally identifiable (SGI), which means that for any observed output behaviour, we have one unique solution for the parameter vector. We remark that this definition allows the analysis of autonomous models, since it is not necessary (but possible) to include manipulated inputs into the model. If the given model does not prove to be SGI, then further analysis is required on a local basis, such that we can get unique solution for a given part of the parameter vector. If finitely many solutions exist for the parameters then the model is structurally locally identifiable (SLI). In the case when the model is neither SGI nor SLI, then it is structurally unidentifiable (SUI). It might happen that only a subset of parameters can be uniquely identified, in this case it is advisable to design an updated measurement procedure for the unidentifiable parameters, or modify the model, because SUI can be an indicator for a bad choice of parametrization.

To determine the structural identifiability of a dynamical model, several methods can be used [3]. One possible methodology that we will apply is a differential-algebra-based approach which gives a constructive necessary and sufficient condition for identifiability [4]. According to this, an autonomous dynamical model given in the form of polynomial ODEs is globally identifiable if and only if by differentiating, adding, scaling and multipying the equations, the model can be rearranged to the parameter-by-parameter linear regression form:

$$P_i\left(y; \frac{d}{dt}\right)\boldsymbol{p}_i - Q_i\left(y; \frac{d}{dt}\right) = 0, \quad i = 1, \ldots, d \tag{1}$$

where $\boldsymbol{p} \in \mathbb{R}^d$ is the parameter vector, $\frac{d}{dt}$ is the differential operator,

and P_i and Q_i are differential polynomials. We note that $\boldsymbol{p}_i$ in Eq. (1) can be transformed parameters of the original physical ones with one-to-one correspondence. There are several freely available symbolic software tools to check identifiability (usually, on models of moderate size due to the high computational complexity). We used the recent edition of the DAISY package for this purpose [5].

The applied disease models

We study the following classical SIR (left) and SEIR (right) models

$$S:\ \dot{x}_1 = -k_1 x_1 x_2 \quad (2) \qquad S:\ \dot{\bar{x}}_1 = -\bar{k}_1 \bar{x}_1 \bar{x}_3 \quad (6)$$

$$I:\ \dot{x}_2 = k_1 x_1 x_2 - k_2 x_2 \quad (3) \qquad E:\ \dot{\bar{x}}_2 = \bar{k}_1 \bar{x}_1 \bar{x}_3 - \bar{k}_2 \bar{x}_2 \quad (7)$$

$$R:\ \dot{x}_3 = k_2 x_2 \quad (4) \qquad I:\ \dot{\bar{x}}_3 = -\bar{k}_3 \bar{x}_3 + \bar{k}_2 \bar{x}_2 \quad (8)$$

$$C:\ \dot{x}_4 = k_1 x_1 x_2 \quad (5) \qquad R:\ \dot{\bar{x}}_4 = \bar{k}_3 \bar{x}_3 \quad (9)$$

$$C:\ \dot{\bar{x}}_5 = \bar{k}_2 \bar{x}_2 \quad (10)$$

where the state variables labelled by S, E, I, R and C denote the proportion of susceptible, exposed, infected, recovered and cumulative infected people within the total population, respectively, i.e. their unit is [persons/total population]. Moreover, $\{k_1, \bar{k}_1\}$, $\{k_2, \bar{k}_3\}$ and $\bar{k}_2$ are positive real model parameters representing the infection probability, the recovery rate, and the transition rate between the E and I compartments, respectively. The units of the parameters can be easily given knowing the unit of x. It is visible, that the SEIR model essentially inserts a delay for newly infected individuals before they become infectious. Here we do not track the number of deaths separately, all infected people are formally transferred to the R compartment at the end of the infection.

Obviously, $\{k_1, \bar{k}_1\}$ depend on the population density, external circumstances such as air pollution and weather, and most importantly (at least for our study), on the introduced restrictions to prevent the spread of the disease.

Identifiability analysis

The only observed outputs are $y = x_4$ and $\bar{y} = \bar{x}_5$ for the SIR and SEIR models, respectively, for which the identifiability analysis of the

two models has been carried out. Due to the limited extent of the paper, we can only show the main points of the calculations for the SIR model. The n-th derivative of any time function f will be denoted by $f^{(n)}$. From Eqs. (2)-(5) we can see that $y^{(1)} = k_1 x_1 x_2 = -x_1^{(1)}$ and from this we also have that $x_2 = \frac{y^{(1)}}{k_1 x_1}$. Taking the second derivative of y gives

$$y^{(2)} = -k_1 y^{(1)} x_2 + k_1 x_1 x_2^{(1)} = -k_1 y^{(1)} x_2 + k_1^2 k_1 x_1^2 x_2 - k_1 k_2 x_1 x_2 \qquad (11)$$

while the re-arranged form of the third derivative is

$$y^{(3)} = \frac{(y^{(2)})^2}{y^{(1)}} - 2k_1 (y^{(1)})^2 - k_1 k_2 y^{(1)} x_2, \qquad (12)$$

where the only variable to be eliminated is x_2, which can be expressed from the above equation and substituted to the fourth derivative of the output. It can be shown by straightforward calculations that the final form of this is the following:

$$y^{(4)} - \frac{3y^{(2)}y^{(3)}}{y^{(1)}} + \frac{2(y^{(2)})^3}{(y^{(1)})^2} = -2k_1 y^{(1)} y^{(2)} - k_2 \left(y^{(3)} - \frac{(y^{(2)})^2}{y^{(1)}} \right) - 3k_1 k_2 (y^{(1)})^2 \qquad (13)$$

where there are no parameters on the LHS, and the RHS is in a linear regression form with respect to $\{k_1, k_2, k_1 k_2\}$. The identifiability of the SEIR model can be checked similarly, although with more cumbersome calculations. The DAISY software also confirms the global structural identifiability of both the SIR and SEIR models with respect to all the parameters from the outputs y and $\bar{y}$, respectively.

Parameter estimation

Having the result that the parameters can be at least theoretically uniquely determined, the parameter estimation for the SEIR model has been performed for the public data of the COVID-19 epidemic in Italy published by the European Centre for Disease Prevention and Control at [6] from 22 February to 10 April, 2020, assuming a population of 60 million. Using public reports, estimates and test results, it was assumed that the true number of infected people at any time instant is 20 times more than the officially recorded figure. We also checked the values 10, 30 and 40 for this multiplier, and the qualitative properties of the model solution (e.g. the computed peak time)

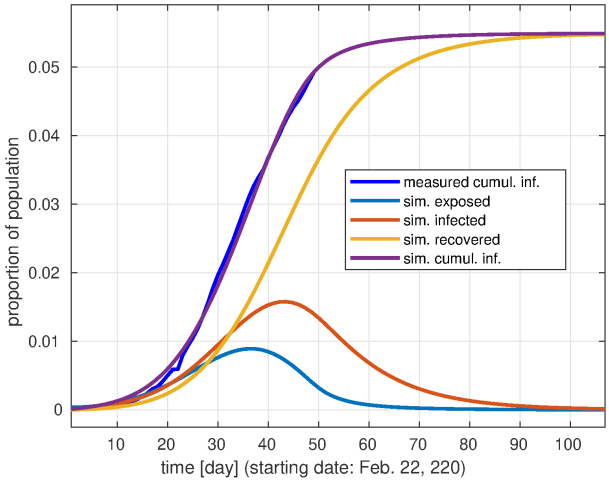


Figure 1: SEIR model fit and simulation results

were very similar in all cases. The criterion function for the estimation was the common quadratic one: $V_N(\boldsymbol{p}) = \frac{1}{N}\sum_{i=1}^{N}(\hat{C}(k,\boldsymbol{p}) - y(k))^2$, where $\hat{C}$ is the model computed and y is the measured output, and N is the number of measurements. In [7] analyzing the data from Wuhan, China, the authors assumed a decreasing piecewise constant function of k_1 (determined by hand calculations), where the intervals depend on the introduction times of different restriction operations. To introduce the minimal number of additional parameters, we used a linear approximation of the decreasing k_1 as $k_1(t) = \beta_0 - \alpha \cdot t$, where $\beta_0, \alpha > 0$. The parameters were estimated using the combination of the Nelder-Mead simplex search algorithm (used for an initial estimate) and the pattern search method which is able to handle bound constraints for the parameters and has guaranteed convergence. The point estimates for the model parameters (suppressing their units) were the following: $\alpha = 0.0108$, $\beta_0 = 0.5409$ which implies that $0.01 \leq k_1(t) \leq 0.5409$, moreover, $k_2 = 0.216$, $k_3 = 0.1$. All these values belong to the ranges published in the literature. The model fit and the prediction of the model beyond 10 April, 2020 (after day 49) can be seen in Fig. 1.

Summary and further work

It was shown using differential algebra tools that classical SIR and SEIR models in nonlinear ODE form are structurally globally identifiable using the cumulative number of infected cases as observed output. The numerical parameter estimation was also performed and the obtained parameter values were well within the range published in the literature. The model is intended to be used for the analysis of restrictive operations introduced to slow down disease spreading.

Acknowledgements

The work of G. Horváth was supported by the grant EFOP-3.6.2-16-2017-00013. Gábor Szederkényi acknowledges the support of the projects EFOP-3.6.3-VEKOP-16-2017-00002 and NKFIH 131545.

References

[1] P. Érdi and J. Tóth. *Mathematical Models of Chemical Reactions. Theory and Applications of Deterministic and Stochastic Models.* Manchester University Press, Princeton University Press, Manchester, Princeton, 1989.

[2] K. Roosa and G. Chowell. Assessing parameter identifiability in compartmental dynamic models using a computational approach: application to infectious disease transmission models. *Theoretical Biology and Medical Modelling*, 16(1), 2019.

[3] E. Walter and L. Pronzato. *Identification of Parametric Models from Experimental Data.* Springer, 1997.

[4] L. Ljung and T. Glad. On global identifiability for arbitrary model parametrizations. *Automatica*, 1994.

[5] G. Bellu, M. P. Saccomani, S. Audoly, and L. D'Angio. DAISY: A new software tool to test global identifiability of biological and physiological systems. *Computer Methods and Programs in Biomedicine*, 88(1):52–61, 2007.

[6] Covid-19 cases worldwide. https://www.ecdc.europa.eu/en/publications-data/download-todays-data-geographic-distribution-covid-19-cases-worldwide.

[7] H. Wang et al. Phase-adjusted estimation of the number of coronavirus disease 2019 cases in Wuhan, China. *Cell Discovery*, 6, 2020.

Solving an extended line balancing optimization problem using dynamic programming

András Éles[1], István Heckl[2]
[1]Department of Computer Science and Systems Technology, University of Pannonia, [1]eles@dcs.uni-pannon.hu, [2]heckl@dcs.uni-pannon.hu
8200 Veszprém, Egyetem utca 10, Hungary

Abstract: **In this work, a method is proposed to solve a line balancing optimization problem with the utilization of dynamic programming. In modern industry, workflow in manufacturing plants is often implemented with the concept of assembly lines. Although utilization of robots in assembly lines is rapidly spreading, there are many cases where human workforce is better, or the only option. This may be the case when electronic devices are assembled. Assigning available workers to the appropriate segment of assembling a product shall be balanced to achieve the best productivity. This is generally not a trivial problem therefore software tools supporting these decisions could be advantageous. A case study is presented here where the optimally balanced assignment is determined. A dynamic programming approach is developed to answer the question for each possible number of workers present. The problem formulation is extended, for example, worker efficiency, penalty for assigning too many tasks for a single worker, can be taken into account.**

Introduction

In modern manufacturing the production of a given product is broken down to a large number of simple tasks. These tasks are performed using assembly lines. As the tasks are usually very simple, a worker may perform a number of consecutive tasks. Naturally the tasks should be assigned to workers in such a way that the load of each worker should be similar to obtain the best productivity. That is why this assignment problem is called line balancing. There would be (at least) one bottleneck worker, whose time spent on a single product item is the highest. This bottleneck is the objective which is to be minimized, also called the cycle time, and is the shortest period in which product items can be completed. At some company this problem have to be solved quite often, in some cases daily, because either there are frequent changes in product demand or the available workforce changes, for example, due to sick leaves.

Assembly Line Balancing (ALB) problems have a vast literature, and can be categorized in several ways. Different solution practices exist based on whether task times are deterministic or stochastic, and whether the assembly line is straight-type or U-type [1]. Other important factors include whether a single, or multiple different kinds of products are to be manufactured, and whether the line is paced or synchronized [2]. In simpler cases, a single product is manufactured, the assembly line is straight-type, paced with a fixed cycle time, data are deterministic and there are no additional resource-considerations [3].

Mathematical programming approaches, especially Mixed-Integer Linear Programming (MILP) models are a common tool for optimization problems, and used thoroughly for line balancing as well. A great advantage of these approaches is that a wide range of constraints and goals can be easily incorporated into a single model [4]. Another approach is the P-Graph framework [5], which is capable of exploiting the combinatorial nature of optimization problems, and reporting not only a single, but the first N best solutions for a problem [6].

In the problems we intended to solve, the assembly line is organized on a per-worker, rather than a per-workstation basis. A single product is produced in a straight line. Problem data are deterministic, and based on estimations of the business-as-usual situation in the plant. Precedence relationships between tasks are not arbitrary, but restricted to a fixed order. This special case gives us the opportunity to use effective algorithms targeting global optimality. This is in contrast with literature examples where order is also optimized, which requires heuristics.

This problem specification was recently tackled by the P-Graph framework as well [6]. Our motivation is to provide a faster and more robust solution. The computational complexity of the P-Graph approach is not polynomial in general, and is difficult to estimate for particular problems. A decent line balancing engine is in need, which can be included in more complex optimization algorithms in the future regarding the manufacturing plant, also considering scheduling, multiple products, workstations and assembly lines for example. A dynamic programming algorithm is proposed, which involves neither the P-Graph framework nor mathematical programming.

A schematic for the line balancing problem is shown in Figure 1. We can see that the product with 5 tasks can be produced with a cycle time of 10 by 2 workers, but only a slightly better cycle time of 8 by 4 workers, as the worker performing the last task is always the bottleneck.

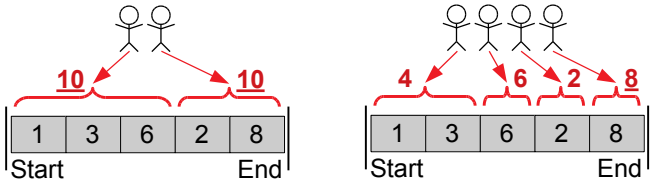


Figure 1. Simple line balancing problem involving a product recipe of 5 tasks and 2 or 4 workers to manufacture it.

Methodology

Given a product recipe of n tasks (numbered from $[0, n[$), we have to assign each of k workers in order (numbered from $[1, k]$) to a consecutive subset of (possibly zero) tasks so that each task is assigned to exactly one worker. The maximum time a worker spends with a single product item is to be minimized. Let us denote by $T_{u,a,b}$ the time worker u spends by performing tasks in $[a, b[$ and by x_u the first task worker u actually performs in an assignment. The objective is formulated in Equation (1) where max is over all the workers and min is over all possible assignments.

$$Z = \min_{0=x_1\le x_2\le\cdots\le x_{k+1}=n} \left(\max_{1\le u\le k} T_{u,x_u,x_{u+1}} \right) \tag{1}$$

Let us denote by $F_{u,v}$ the objective if only the first u workers are present and the product recipe consists of only the first v tasks. $F_{u,v}$ is defined for all $0 \le u \le k$ and $0 \le v \le n$. Known values for $F_{u,v}$ are $F_{u,0} = 0$, and $F_{0,v} = \infty$ $(v \ge 1)$. Equation (2) presents a recursive formula for $F_{u,v}$, which is the key in the dynamic programming approach. The problem for u workers and v tasks is decomposed into two subproblems. The first x_u task is performed by the first $u-1$ workers, and the rest $x_{u+1} - x_u$ tasks are performed by worker u. Whichever produces the higher working time for a single worker, it is selected by the maximum, and $F_{u,v}$ is the minimum over all possible choices of x_u.

$$F_{u,v} = \min_{0\le x_u\le v} \left(\max \left\{ F_{u-1,x_u}, T_{u,x_u,v} \right\} \right) \; \forall\, 1 \le u, v \tag{2}$$

This method guarantees optimality, as at least one optimal solution is provenly included among all cases enumerated. In a simple line balancing problem, $T_{u,a,b}$ could be calculated by summing up the constant time requirement of all tasks in the interval $[a, b[$, independent of worker u, as already presented in Figure 1. However, an extended problem is addressed here which can take into account the following factors. All of these additional parameters are nonnegative.

- An efficiency ratio ϕ_u for each worker u, which acts multiplicatively with the constant time requirement of tasks done by the worker u.
- Penalty for cutting the product recipe at a specific point. For example, assigning tasks v and $v+1$ to workers u and $u+1$ would extend their times spent by an additional η_v for worker u and θ_v for worker $u+1$.
- Penalty for merging the recipe. For example, a penalty κ_v is added to any worker who does both tasks v and $v+1$. Note that we must either merge or cut between consecutive tasks v and $v+1$.
- Penalty for assigning too many tasks for a single worker. This can be expressed as $\lambda_{u,x}$, which depends on the worker and number of tasks, and adds to the total time of worker u spent on a single product item.
- Additional fixed penalties if a worker spends at least a given amount of time working on a single product item. Multiple such limits and fixed corresponding penalties can be specified.

We assumed a low number of the last type of additional penalties, which makes the evaluation of $T_{u,a,b}$ possible in $O(1)$ time including all the factors mentioned above. This gives an $O(n^2k)$ complexity for evaluating $F_{n,k}$ even if calculations are performed straightforward based on the formula. This is better than applications using MILP formulations or P-Graph approaches. However, the complexity can be further reduced by better implementation, exploiting monotonicity of the terms in the formula, and the fact that practical examples are not arbitrary. These possibilities are subject to future work.

An advantage of the algorithm is that the problem is solved not only for k workers, but any first u of the k workers, to optimality. Note that this is in contrast with the P-Graph approach, where a number of the best solutions can be reported for a fixed number of workers instead.

Note that different efficiency ratios for workers suggest that decision on their ordering near the assembly line shall be part of the optimization procedure. However, investigating this would be computationally costly and therefore in this solution method, we assume their order is given a priori. Including ordering decisions either with dynamic programming or mathematical programming approaches is subject to future work.

Implementation

The proposed method for the line balancing problem specification described was implemented as a pure C++ program with Qt Creator. No external tools are required other than standard libraries and language

functionalities available in the C++11 standard. This is an advantage over mathematical programming and P-Graph based approaches which usually depend on the general purpose solvers chosen.

The line balancing problems must be given in a text file regarding a special syntax. This syntax supports all the functionalities mentioned in the methodology. Parameters not appearing in the data files are defaulted, usually to zero values.

The algorithm does preliminary calculations first for fast lookup of $T_{u,a,b}$ values, then proceeds with the evaluation of $Z = F_{k,n}$.

Case Study

The basis of the line balancing problem presented here is a collection of real-world examples involving the assembly of electronic devices. One of such problems is presented here as a case study. The product recipe includes 58 tasks, ranging between an estimated 3 to 30 seconds to perform. The problem is extended with all the mentioned factors.

At most 10 workers stand by the assembly line, having two possible profiles: adept and normal. Each worker profile describes a different collection of efficiency, task count and total time penalties. Adept workers are slightly faster. Note that cutting and merging penalties defined for tasks are global, i.e., independent of workers chosen. Note that based on the assumption on not optimizing for worker order, the profile of the worker at any index u in the line is determined a priori.

The code was compiled and the test was performed in Ubuntu Linux 18.04.4 LTS operating system. The computer was a Lenovo ThinkCentre M83 desktop computer having an Intel i7-4770 3.40 GHz CPU and 16 GB RAM. Execution succeeds in 1 to 1.5 milliseconds including file operations on an SSD. Solution statistics are shown in Table 1. The efficiency is the product of the number of workers and the objective, which informally tells how effective workers are on average. The total column is the total amount of time spent by the workers on a product item. The bottleneck is the worker determining the objective, spending the most time on an item. Its exact job, the $[a, b[$ interval of tasks to be performed is also shown there.

We can observe that the objective is decreasing with the number of workers increasing, roughly inversely proportionally, as expected. The efficiency and total time of the workers is worsening by introducing more workers. This may be due to the fact that the first worker is adept, and there are many cutting and other penalties which add up. Therefore, assigning

more workers for a better result shall be reconsidered in environments where workforce is scarce and there are competing activities to do.

Table 1. Results and statistics for the case study problem

Workers	Objective	Efficiency	Total time	Bottleneck
1	595.45	595.45	595.45	$u = 1, [0{,}58[$
2	321.30	642.60	641.40	$u = 1, [0{,}34[$
3	217.20	651.60	645.80	$u = 1, [0{,}23[$
4	172.70	690.80	666.00	$u = 1, [0{,}20[$
5	135.60	678.00	653.65	$u = 3, [24{,}38[$
6	117.30	703.80	685.00	$u = 6, [51{,}58[$
7	102.20	715.40	671.70	$u = 4, [26{,}35[$
8	91.20	729.60	687.55	$u = 3, [17{,}24[$
9	82.30	740.70	687.35	$u = 4, [22{,}26[$
10	74.60	746.00	695.65	$u = 3, [14{,}22[$

Conclusions

An algorithm based on dynamic programming for the line balancing problem with various penalty and worker-specific parameters is proposed. The method is proven to be effective in solving real-world problem instances fast, therefore it is capable of inclusion as part of optimization problems designed for a wider scope. This wider scope potentially includes multiple products, more assembly lines, resource restrictions, and scheduling in a longer term. The algorithm itself can likely be improved as well, or adapted for specific kind of real-world applications to obtain better results, which is also subject to work in the future.

Acknowledgment

We acknowledge the financial support of Széchenyi 2020 programme under the project No EFOP-3.6.1-16-2016-00015.

This research was supported from the Thematic Excellence Program 2019 the grant of the Hungarian Ministry for Innovation and Technology. (Grant Number: NKFIH-843-10/2019)

References

[1] P. Sivasankaran and P. Shahabudeen. "Literature review of assembly line balancing problems", *Int J Adv Manuf Technol*, vol. 73, 1665-1694, 2014

[2] N. Kumar and D. Mahto. "Assembly line balancing: a review of developments and trends in approach to industrial application", *Glob J Res Eng*, vol. 13, 2013

[3] C. Becker and A. Scholl. "A survey on problems and methods in generalized assembly line balancing", *Eur J Oper Res*, vol. 168, 694-715, 2006

[4] R. F. Deckro and S. Rangachari. "A goal approach to assembly line balancing", *Comput Chem Eng*, vol. 17, 509-521, 1990

[5] F. Friedler, K. Tarjan, Y. W. Huang and L. T. Fan. "Graph-Theoretic Approach to Process Synthesis: Axioms and Theorems", *Chem Eng Sci*, vol. 47, 1973-1988, 1992
[6] A. Bartos and B. Bertok. "Production line balancing by P-graphs", *Optim Eng*, 2019

Diagnosis Method for Batch Processes using Time Dependent Decomposition

A. Leitold[1], M. Gerzson[2]
[1]Dept. of Mathematics leitolda@mik.uni-pannon.hu
[2]Dept. of Electrical Engineering and Information Systems
gerzson@almos.uni-pannon.hu
University of Veszprém, Egyetem str. 10., Veszprém, Hungary

Abstract: **A revised diagnosis method for batch technological system is proposed in this paper. The diagnosis method based on the qualitative colored Petri net model of the system but uses sampling time base decomposition in order to make the diagnosis computationally feasible.**

Introduction

Most of the technological systems can be characterized as discrete time discrete event systems therefore they can be described by discrete modelling tools. The aim of our investigation is the fault diagnosis of the technological system based on their colored Petri net model.

The fault diagnosis can be divided into fault detection, isolation and identification. The occurrence of a fault or faults can be detected based on the measured values, the location of faults can be determined with fault isolation methods while the identification methods help to characterize them.

Basics of the diagnosis method

Discrete event discrete time technological systems can be described by colored Petri nets [1]. In case of the classical approach places refer to the technological units and transition refer to technological steps while the different values of variables can be described by colored tokens in general [2].

The CPN model of the diagnosed system and its occurrence graph is used a quite different way in our diagnosis method as it can be seen in the following.

Qualitative range spaces

The measured values and the states of actuators in the technological system are considered as qualitative variables. Their codomain is decomposed into

qualitative ranges and instead of the exact values these ranges are used to characterize them. For a sensor the following range can be defined, e.g.:

$$Q_s = \{err, e^-, 0, L, N, H, e^+\}$$

where *err* refers to the general failure of the sensor (no signal, e.g.), 0, *L*, *N*, *H* refer to minimum, low, normal, high measured value, while e^-, and e^+ refer to outliers because of negative or positive bias error of the sensor. The qualitative range space can be refined, if required. The states of two-state valves and switches can be described similarly.

Events, traces

The operational procedure consists of steps to be performed in a given states of the system. In our approach these steps are called *event*s and their structure is as follows:

$$event_k = (k, \textit{input values}, \textit{output values})$$

where *k* refers to the sampling timepoint number, *input values* are the states of the actuators and the *output values* are the measured values in qualitative form. The steps are ordered by their number. The list of consecutive steps is called *trace*. For diagnostical purposes we distinguish nominal, faulty and characteristic traces where the nominal trace refers to normal, i.e. faultless operation of the system, the different faulty traces describe the work of the system in case of known faults. The nominal and faulty traces should be determined in advance for the diagnosis. The characteristic trace is generated during the actual work of the system and our diagnosis method is based on its comparison with the nominal trace. The types of deviations are as follows:

- *later* or *earlier*: this type of deviation means that a given combination of input and output values occurs later or earlier time stamp in the characteristic trace than in the nominal trace;
- *never-happened* deviations refer to the situation when the given combination of these values does not occur in the characteristic trace at all;
- *greater* or *smaller* var_out_i: these deviations mean that the qualitative value of output variable *i* is greater or smaller than in the nominal trace at the same time stamp.

Colored Petri net model

In our approach the colored Petri net model has special structure, its general form can be seen in Fig 1. [3]. The function of the net elements is as follows:

- *Places* refer to the input and output variables, to the actual fault and to deviations. In case of variables the colored tokens described their

values, the token color on place *fault* refers to type of fault occurring in the system. The token colors on the places *dev* and *never* depend on the deviations from the nominal course.

– There are three *transitions* in our model only. The transitions t_1 and t_3 fire only once during the simulation process to initialize the system and to determine the never-happened events. The task of the transition t_2 is the timing of the simulation i.e. the execution of the steps.
– *Arcs* define the logical connections between the places and transitions and the *arc functions* describe the change of the token colors during the firing of a transition.

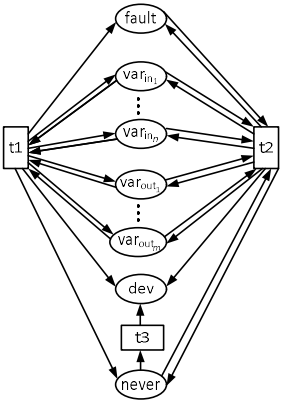


Figure 1. Structure of the generalized colored Petri net model

In case of real technological systems faults occur in probabilistic way. In our model the color of fault token is determined by a fault probability function assigned to the arc between transition t_1 and place *fault*.

Diagnosis based on the occurrence graph

Our diagnosis method is based on the occurrence graph generated by the simulation of the CPN model of the investigated system. The occurrence graph contains all the reachable markings from an initial marking. In our method the initial value of the input and output variables are fixed in advance but the normal or faulty work of the system is determined during the firing of initialization transition. However, all the possible system states are needed for the diagnosis, so the graph has to contain all the nodes belonging to both the normal and the different faulty operational modes. The work of the

system, i.e. the diagnosis can be done with the investigation of the token distribution on the nodes of the graph.

Our proposed diagnostical method is based on occurrence graph of the colored Petri net model of the investigated system and it can be used in on-line and off-line way for detection and identification of faults occurring in the technological system [3].

Time dependent decomposition

In case of a large complex technological system the size of the occurrence graph can be very large and the search on the nodes needs significant computational effort and time. As it was shown in [4] the structural decomposition can be a solution to this problem.

In our recent contribution the use of the decomposition method is introduced in different way. In this case the procedural time dependent decomposition is used similarly to structural decomposition. This decomposition can be done if the operating procedure can be divided into two or more parts and the steps of operating procedure can be classified into these parts unambiguously. In case of batch technologies, the number of units can be low, but the operational procedure contains several steps often. The base of the introduced structural decomposition method is the partition of the complex system into their elements and performing the diagnosis on these units separately.

Case study

A simplified technological process is introduced in this section to illustrate the way of our proposed diagnostical method.

The technological system and the operating procedure

The technological system consists of a tank having an inlet and outlet pipe with valves and equipped with temperature sensor, level switch and heating element. The system can be seen in Fig. 2.

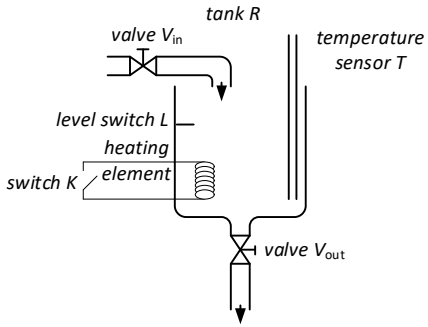


Figure 2: The technological system to be diagnosed

The steps of the operating procedure are as follows. Initially the tank R is empty and the valves V_{in} and V_{out} are closed and switch K is open. As a first step the valve V_{in} is opened and the tank is being filled up until the level switch L gives signal. Normally, the filling up process takes two sampling time periods. The next step is the heating up of liquid. To activate the heating element the switch K is closed and heating up process which takes three sampling periods, normally, is monitored by the temperature sensor T. If the temperature reaches the prescribed value, the switch K is opened and valve V_{out} is opened, too, and the tank R is emptied through its outlet pipe.

The assumed faults are as follows. The valves V_{in} and V_{out} can stick totally or open only half, the level switch doesn't send signal when the liquid level reaches it, the temperature sensor breaks down or it has negative or positive bias error, the switch of the heating element goes wrong and the heating process does not start.

Qualitative range spaces in the technological system

The qualitative range space of the temperature sensor T is the same as it was mentioned in the Part *Qualitative range space*. In our example 0 refers to $T20$, L to $T40$, M to $T60$ and H to $T80$, resp. The valves, the switch of the heater element and the level switch are considered two-state element so their qualitative range spaces are

$$Q_a = \{0, 1\}$$

where 0 refers to closed valves, open heater switch and low-level signal from the level switch, and in case 1 the valves are open, the heater is on and the level is at level switch.

Traces of technological system

As it was mentioned nominal, faulty and characteristic traces are distinguished for the description of the work of the system in normal, faulty and the actual operational mode.

The nominal trace and the states of actuators and measured values by sensors and the steps of operating procedure can be found in Table 1. Series of tables (Table 2.) contain the faulty traces in case of the assumed (i.e. known) faults. In a real system more than one fault can occur at the same time. Our method enables to build in multiple fault into the model and to perform the diagnosis. The traces in the third part of Table 2. give example to this case.

Table 1: The nominal trace of the technological system

nominal trace	*id*	V_1	V_2	*K*	*L*	*T*	*steps*
(0, 0, 0, 0, 0, 0)	0	cl	cl	off	0	T20	initialization
(1, 1, 0, 0, 0, 0)	1	op	cl	off	0	T20	filling up starts
(2, 1, 0, 0, 0, 0)	2	op	cl	off	0	T20	filling up
(3, 0, 0, 1, 1, 0)	3	cl	cl	on	1	T20	filling up over, heating starts
(4, 0, 0, 1, 1, L)	4	cl	cl	on	1	T40	heating
(5, 0, 0, 1, 1, M)	5	cl	cl	on	1	T60	heating
(6, 0, 1, 0, 1, H)	6	cl	op	off	1	T80	heating is over, emptying starts
(7, 0, 1, 0, 0, H)	7	cl	op	off	0	T80	emptying
(8, 0, 0, 0, 0, 0)	8	cl	cl	off	0	T20	emptying is over, process is ready

Colored Petri net model of the system

The structure of the CPN model belonging to our investigated system is the same as it can be seen in Fig. 1. Part of the places refer to actuators (V_{in}, V_{out}, *K*), to sensors (level and temperature) and the colors on them describe their states. Places *dev* and *never* have important role from diagnostic point of view. The place *dev* is for the collection of the deviations between the normal and characteristic trace while place *never* contains the events that haven't happened until a given sampling time point. The role of the transitions and arc functions is the same as it was introduced in the Part *Colored Petri net model*. For the simulation the model the CPNTools [5] was used.

Diagnosis of the technological system

To simplify the diagnosis the sampling time dependent decomposition is done based operating procedure and depicted on the nominal trace. The operation of the system can be divided into two parts: the first part is the

Table 2: Faulty traces – part I

fault V_1 stuck	*fault V_1 half open*	*fault V_2 stuck*	*fault V_2 half open*
(0, 0, 0, 0, 0, 0)	(0, 0, 0, 0, 0, 0)	(0, 0, 0, 0, 0, 0)	(0, 0, 0, 0, 0, 0)
(1, 1, 0, 0, 0, 0)	(1, 1, 0, 0, 0, 0)	(1, 1, 0, 0, 0, 0)	(1, 1, 0, 0, 0, 0)
(2, 1, 0, 0, 0, 0)	(2, 1, 0, 0, 0, 0)	(2, 1, 0, 0, 0, 0)	(2, 1, 0, 0, 0, 0)
(3, 1, 0, 0, 0, 0)	(3, 1, 0, 0, 0, 0)	(3, 0, 0, 1, 1, 0)	(3, 0, 0, 1, 1, 0)
(4, 1, 0, 0, 0, 0)	(4, 0, 0, 1, 1, 0)	(4, 0, 0, 1, 1, L)	(4, 0, 0, 1, 1, L)
	(5, 0, 0, 1, 1, L)	(5, 0, 0, 1, 1, M)	(5, 0, 0, 1, 1, M)
	(6, 0, 0, 1, 1, M)	(6, 0, 1, 0, 1, H)	(6, 0, 1, 0, 1, H)
	(7, 0, 1, 0, 1, H)	(7, 0, 1, 0, 1, H)	(7, 0, 1, 0, 0, H)
	(8, 0, 1, 0, 0, H)	(8, 0, 0, 0, 1, H)	(8, 0, 1, 0, 0, H)
	(9, 0, 0, 0, 0, 0)		(9, 0, 0, 0, 0, 0)

Table 2 (cont.): Faulty traces – part II

fault K wrong	*fault L wrong*	*fault T wrong*	*fault T neg bias error*
(0, 0, 0, 0, 0, 0)	(0, 0, 0, 0, 0, 0)	(0, 0, 0, 0, 0, err)	(0, 0, 0, 0, 0, e^-)
(1, 1, 0, 0, 0, 0)	(1, 1, 0, 0, 0, 0)	(1, 1, 0, 0, 0, err)	(1, 1, 0, 0, 0, e^-)
(2, 1, 0, 0, 0, 0)	(2, 1, 0, 0, 0, 0)	(2, 1, 0, 0, 0, err)	(2, 1, 0, 0, 0, e^-)
(3, 0, 0, 1, 1, 0)	(3, 1, 0, 0, 0, 0)	(3, 0, 0, 1, 1, err)	(3, 0, 0, 1, 1, e^-)
(4, 0, 0, 1, 1, 0)	(4, 1, 0, 0, 0, 0)	(4, 0, 0, 1, 1, err)	(4, 0, 0, 1, 1, 0)
(5, 0, 0, 1, 1, 0)	(5, 1, 0, 0, 0, 0)	(5, 0, 0, 1, 1, err)	(5, 0, 0, 1, 1, L)
			(6, 0, 0, 1, 1, M)
			(7, 0, 1, 0, 0, H)
			(8, 0, 1, 0, 0, H)
			(9, 0, 0, 0, 0, 0)

Table 2 (cont.): Faulty traces – part III

fault T pos bias error	*double faults: V_1_half- V_2_wrong*	*double faults: V_1_stuck-L_wrong*	*double faults: V_1_half-T_pos_bias*
(0, 0, 0, 0, 0, L)	(0, 0, 0, 0, 0, 0)	(0, 0, 0, 0, 0, 0)	(0, 0, 0, 0, 0, L)
(1, 1, 0, 0, 0, L)	(1, 1, 0, 0, 0, 0)	(1, 1, 0, 0, 0, 0)	(1, 1, 0, 0, 0, L)
(2, 1, 0, 0, 0, L)	(2, 1, 0, 0, 0, 0)	(2, 1, 0, 0, 0, 0)	(2, 1, 0, 0, 0, L)
(3, 0, 0, 1, 1, L)	(3, 1, 0, 0, 0, 0)	(3, 1, 0, 0, 0, 0)	(3, 1, 0, 0, 0, L)
(4, 0, 0, 1, 1, M)	(4, 0, 0, 1, 1, 0)	(4, 1, 0, 0, 0, 0)	(4, 0, 0, 1, 1, L)
(5, 0, 1, 0, 1, H)	(5, 0, 0, 1, 1, L)	(5, 1, 0, 0, 0, 0)	(5, 0, 0, 1, 1, M)
(6, 0, 1, 0, 0, H)	(6, 0, 0, 1, 1, M)	(6, 1, 0, 0, 0, 0)	(6, 0, 1, 0, 1, H)
(7, 0, 0, 0, 0, 0)	(7, 0, 1, 0, 1, H)	(7, 1, 0, 0, 0, 0)	(7, 0, 1, 0, 0, H)
	(8, 0, 1, 0, 1, H)	(8, 1, 0, 0, 0, 0)	(8, 0, 0, 0, 0, L)
	(9, 0, 1, 0, 1, H)		

filling up process (framed with red dashed line in Table 3) while the second part is the heating up and emptying (blue dotted line in Table 3).

Table 3: The sampling time dependent decomposition of the nominal trace

nominal trace	*id*	V_1	V_2	*K*	*L*	*T*	*Part I*	*Part II.*
(0, 0, 0, 0, 0, 0)	0	cl	cl	off	0	T20	(0, 0, 0)	(0, 0, 1, 0)
(1, 1, 0, 0, 0, 0)	1	op	cl	off	0	T20	(1, 1, 0)	(1, 0, 1, L)
(2, 1, 0, 0, 0, 0)	2	op	cl	off	0	T20	(2, 1, 0)	(2, 0, 1, M)
(3, 0, 0, 1, 1, 0)	3	cl	cl	on	1	T20	(3, 0, 1)	(3, 1, 0, H)
(4, 0, 0, 1, 1, L)	4	cl	cl	on	1	T40		(4, 1, 0, H)
(5, 0, 0, 1, 1, M)	5	cl	cl	on	1	T60		(5, 0, 0, 0)
(6, 0, 1, 0, 1, H)	6	cl	op	off	1	T80		
(7, 0, 1, 0, 0, H)	7	cl	op	off	0	T80		
(8, 0, 0, 0, 0, 0)	8	cl	cl	off	0	T20		

Assuming the characteristic trace in Table 4. (left) its diagnosis is performed separately for each part. As a first step the framed trace pieces are removed from the characteristic trace and the initial is shifted to 0 in case of the second part. The resulted event list can be seen in right part of Table 4. Using our proposed method not only the size of occurrence graph but search problem is much smaller.

The diagnostic process starts with Part I and the first step is to generate the deviation list between the nominal and characteristic trace. Then this list is searched among the terminal nodes of the occurrence graph belonging to first part. If the token colors on the place *dev* refer this list, then the type of the fault can be concluded from the token color of place *fault* at the same node. In our case the fault is "*fault V_1 half open*". Before the diagnosis of the Part II. the diagnosed fault of Part I. is added to the CPN model of Part II. as initial condition. Then the occurrence graph is generated and the diagnosis is performed as in case Part I. As a result, there is no fault in Part II.

Table 4: The decomposition of the characteristic trace

(0, 0, 0, 0, 0, 0)	Part I.	Part II.
(1, 1, 0, 0, 0, 0)	(0, 0, 0)	(0, 0, 1, 0)
(2, 1, 0, 0, 0, 0)	(1, 1,0)	(1, 0, 1, L)
(3, 1, 0, 0, 0, 0)	(2, 1, 0)	(2, 0, 1, M)
(4, 0, 0, 1, 1, 0)	(3, 1, 0)	(3, 1, 0, H)
(5, 0, 0, 1, 1, L)	(4, 0, 1)	(4, 1, 0, H)
(6, 0, 0, 1, 1, M)		(5, 0, 0, 0)
(7, 0, 1, 0, 1, H)		
(8, 0, 1, 0, 0, H)		
(9, 0, 0, 0, 0, 0)		

As a second example let the characteristic trace in left part of Table 5. be assumed.

Table 5: Diagnosis in case of two independent faults

(0, 0, 0, 0, 0, 0)	Part I.	Part II.
(1, 1, 0, 0, 0, 0)	(0, 0, 0,)	(0, 0, 1, 0)
(2, 1, 0, 0, 0, 0)	(1, 1, 0)	(1, 0, 1, L)
(3, 1, 0, 0, 0, 0)	(2, 1, 0)	(2, 0, 1, M)
(4, 0, 0, 1, 1, 0)	(3, 1, 0)	(3, 1, 0, H)
(5, 0, 0, 1, 1, L)	(4, 0, 1)	(4, 1, 0, H)
(6, 0, 0, 1, 1, M)		(5, 1, 0, H)
(7, 0, 1, 0, 1, H)		(6, 1, 0, H)
(8, 0, 1, 0, 1, H)		
(9, 0, 1, 0, 1, H)		

In case of the diagnosis of Part I the same result is concluded: valve V_1 opens only half way. But from the diagnosis of Part II we get valve V_2 does not open at all.

As third example let us the characteristic trace in the left part of Table 6.

The diagnosis of Part I results that either the valve V_1 is stuck or level switch are wrong. The effect of these faults is indistinguishable so the set of possible faults can be determined. The diagnosis of Part II can not be performed in this case.

Table 6: Diagnosis in case of two indistinguishable faults

(0, 0, 0, 0, 0, 0)		
(1, 1, 0, 0, 0, 0)	Part I.	Part II.
(2, 1, 0, 0, 0, 0)	(0, 0, 0)	(0, 0, 0, 0)
(3, 1, 0, 0, 0, 0)	(1, 1, 0)	(1, 0, 0, 0)
(4, 1, 0, 0, 0, 0)	(2, 1, 0)	(2, 0, 0, 0)
(5, 1, 0, 0, 0, 0)	(3, 1, 0)	(3, 0, 0, 0)
(6, 1, 0, 0, 0, 0)	(4, 1, 0)	(4, 0, 0, 0)
(7, 1, 0, 0, 0, 0)		
(8, 1, 0, 0, 0, 0)		

Conclusion

A revised diagnosis method was introduced in our contribution, which is based on the qualitative colored Petri net model of the system. The main advantage of our proposed method the sampling time dependent decomposition, which can be useful in case of technological systems consisting of small number of technological units but batch processes take place in them. This time dependent decomposition reduces the size of the occurrence graph used for diagnosis and the size of the searching problem would be smaller. As a result, the diagnosis is faster and computationally feasible.

Acknowledgment

The authors acknowledge the financial support of Széchenyi 2020 under the EFOP-3.6.1-16-2016-00015 and the Thematic Excellence Program 2019 the grant of the Hungarian Ministry for Innovation and Technology (Grant Number: NKFIH-843-10/2019).

References

[1] Jensen, K.: *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use*, Springer-Verlag, Berlin 1996

[2] Gerzson, M.; B. Márczi and A. Leitold. 2012. Diagnosis of Technological Systems based on their Coloured Petri Net Model, *ARGESIM Report* Eds: Troch, I., Breitenecker, F. no. S38 358/1-6

[3] Leitold, A., Márczi, B., Pózna, A.I., Gerzson, M.: Monitoring and Diagnosis of Manufacturing Systems using Timed Colored Petri Nets, *Hung. J. Ind. Chem.* , 2013

[4] Pózna, A.I., Gerzson, M., Leitold, A., Hangos, K.M., : Colored Petri Net based Diagnosis of Technological Systems, in *European Simulation and Modelling Conference,* ESM'2016, Las Palmas, Spain, 2016

[5] CPN Group, University of Aarhus, Denmark: CPNTools 2.2.0 http://wiki.daimi.au.dk/cpntools/

Well-Known Brands for Testing Colour Memory

Cecilia Sik-Lanyi[1], Bettina Miklós[2], Tamás Szőcs-Józsa[3]
[1]University of Pannonia, Faculty of Computer Science,
Department of Electrical Engineering and Information Systems
8200 Veszprem, Egyetem street 10.
[1]lanyi@almos.uni-pannon.hu, [2]bettinamiklos@gmail.com,
[3]tom.szocsjozsa@gmail.com

Abstract: **Test software was developed and used to test users' colour memory. Logos of well-known brands were used for this research. There are two types of tasks in the test-software: on the one hand, the user needs to recognize the real colour brand logo, of which only one is the original colour, the others have a lighter or darker shade. On the other hand, in a colouring part, the user must colour the black and white version of the investigated logos from a colour palette. The users' choices were saved in a database and analysed.**

Introduction

Colours have associations and meanings, for example, red means “in the red” or financial trouble, or it could mean danger or stop. Green means money or “go.” A few colours have similar meanings everywhere (gold, for example, stands for success and high quality in most cultures), but most colours have different meanings in different cultures. For example, in the U.S. white signifies purity and is used at weddings, but in other cultures white is the colour used for death and funerals. Happiness is associated with white, green, yellow, or red, depending on the part of the world. [1]

In today's consumer society, the primary goal of every service provider and retailer is to sell as many services and products as possible. This endeavour created marketing as a corporate activity. The purpose of marketing communication is to inform the customer that the product exists and to create a demand for the product. One part of this process is the use of logos. It is important for a company logo to refer to the company profile and stand out from other companies with its uniqueness. One of the important elements of this is the creation and use of company-specific colours. The logo is essentially a symbol that has been used since ancient times to convey information. These illustrations help people communicate complex things simply. In general, most symbols in the world have the same meaning. Thus, a good logo should not only symbolize the company, but

also create positive feelings in the potential consumer. Colours play a key role in this.

We can find in the Visual Guidelines of Google Marketing Platform: “Our company’s brand is a recognizable identity and the foundation for how we present ourselves in a cohesive manner. It’s a jumping off point for all of our other communications and materials to convey who we are, what we do, and how we add value. Whether you’re creating a chart or sending an email, it’s important to present the best version of our brand in a streamlined way.” [2]

For this research well-known brand colours were used.

The logo of a company is defined in the Public Appearance Guidelines of the company. Fig. 1 shows the colour-code of the Mastercard Branding Requirements and Fig. 2 shows the colour-code of the Goggle.

Figure 1: Visual Guideline of the Mastercard [3]

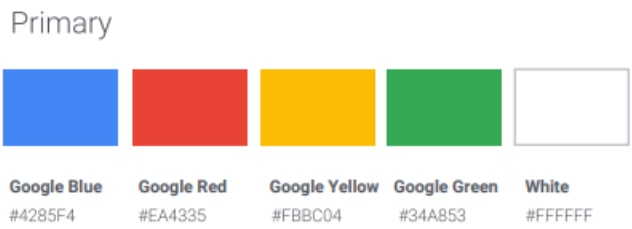


Figure 2: Visual Guideline of Google [4]

A memory colour is the typical colour of an object that an observer acquires through their experience with that object. However, the fact that an observer knows about the typical colours of certain objects (i.e. memory colours) does not necessarily imply that this knowledge influences the way they see the actual object colours. For this reason, it is sensible to disentangle the concept of a memory colour from the idea of a memory colour effect on colour appearance. Hence not all research on memory colour is about the memory colour effect. [5] To the best of our knowledge,

no similar research has been conducted that would have used the colour of brands.

Method

The study lasted from October 2018 to mid of April 2020. Firstly, we have searched the PANTONE colour-code of more than 70 well-known brands. The second step was developing the test software, and the third one was collecting observers.

We have executed the tests in a selected room. It had a bit of light coming from the window on the side of the room, but no artificial light has been used in the selected room. This has helped us prevent reflections on the screen. The brightness level has been kept to the same level during the tests. The observers could hold the device in a natural position, like how they normally use a tablet like this. Also, every observer has received similar instructions on how to complete the test. Twenty observers have fulfilled the tasks in the test-software. None of the observers were colourblind.

Test Software

There are several versions of the Android operating system, the latest is the 10th but we have used the 7th for our development goals. In this version, the manufacturer made the colour calibration option available. This is important for the test software because we did not want to get false results while the test software has been used by the observers.

The main menu is shown on the Fig. 3a and the first menu is the player information (Fig.3b).

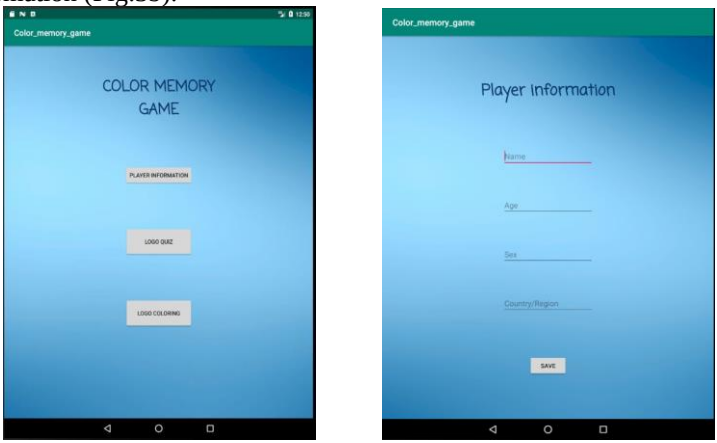


Figure 3: a.) The main menu of the test software, b.) The Player Information screen

The first test is the “Logo Quiz” as it shown on the Fig. 4. The application randomly selects logos from a list of about 50 items. By using colour shift, we have created three new images alongside the original (darkened or lightened). If the user is unfamiliar with the logo and does not want to guess, they can skip the current image with the "Skip" button.
When the quiz is complete, the application stores the player's data and answers in an SQL table. After the 15th logo, a score screen tells the player how many correct answers they have (Fig. 5).

Figure 4: The Logo Quiz Task

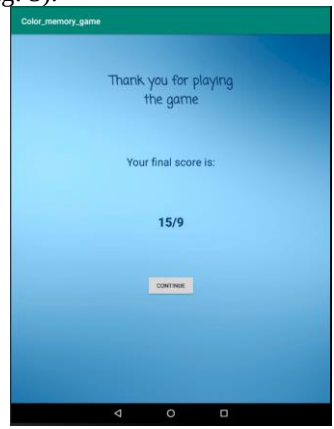


Figure 5: The player’s score

The player gets five pieces of logos randomly during the colouring task. When the player selects one partition of the logo, a grayscale image of the selected partition appears on the gray background (Fig. 6). When they press this button, the colour palette appears, where they can select the colour they want (Fig. 7). On the right-hand side, on the white background, they see the image of the logo it is coloured (Fig.8). If they want to make adjustments, they can still do so at this point. As soon as they click the right arrow, a window pops up where they have to confirm that they want to move on. When they finalize their answers, we save the original colours of the partitions with the new colours and print out the difference of the selected colours from the original colour (Fig. 9). Clicking on the "Reset colours" button will restore the uncoloured grayscale partitions. Pressing the "I don't know this brand" button will give them another image instead of the current one.

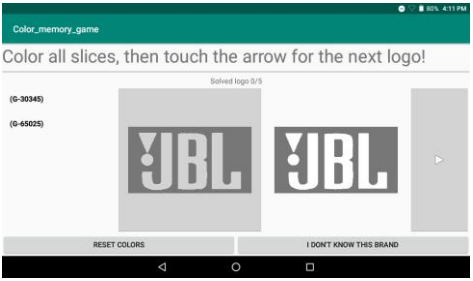


Figure 6: The Colouring Task structure.

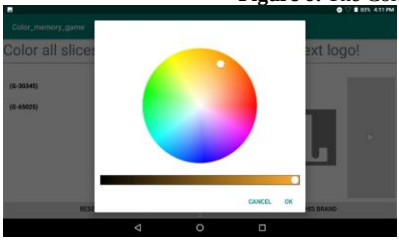


Figure 7: Colour palette in the application

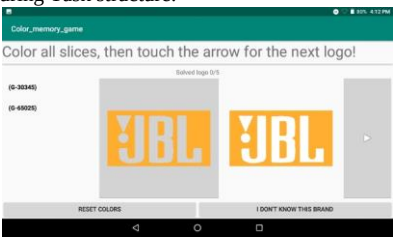


Figure 8: The coloured logo by the user

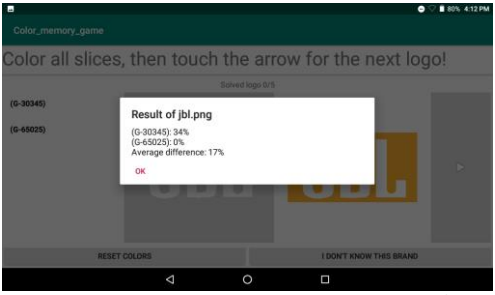


Figure 9: After finalization, the user can see how the colour selected for each partition differs from the original one

Results and Discussion

About 20 people managed to complete both game modes.

Logo quiz statistics:

The quiz mode was completed by 22, all between the ages of 20 and 40, apart from one observer. Based on the data, on average, the original was found 6 times out of 15, 4 times 10% lighter, 3 times 10% darker and about 2 times 20% darker. We have found two outstanding results, in one case the respondent hit the original image 11 times out of 15 and the other respondent hit it 10 times.

Considering all the logos, the results are the following:

- there were 13 logos, only once
- 8, twice
- 13, three times
- 9, four times
- 2, five times
- 2, six times
- and there was only one logo (IKEA) selected seven times as the original.

In addition, it is worth noting that there were logos that were never found by the observers: Acer, Deichmann and Linkedin.
The results of the Logo Quiz are shown by Fig. 10 and Fig. 11.

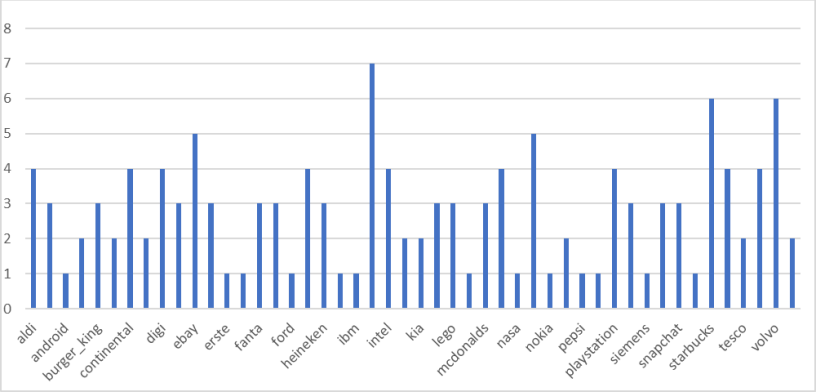


Figure 10: Bar chart of original results by logo

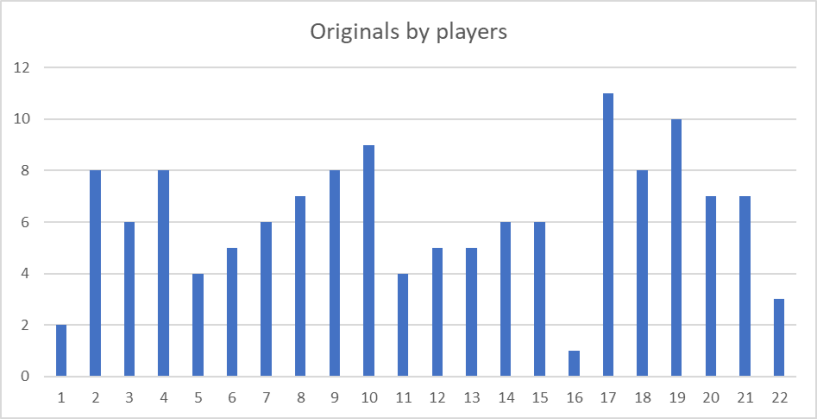


Figure 10: Bar chart of players' original results

Logo colouring statistics:

26 people played with our colouring mode.

Generally, black and white partitions were selected without problems by the observers. In addition, if the logo consisted of only one main colour, the colour could be accurately determined. For example, there was no Facebook logo coloured in green or other than blue. The following logos matched the colour, but not the hue: HP, Ford, Nokia, Siemens, PlayStation, Snapchat, Twitter, Facebook, Android, Coca-Cola, Kia.

For multi-colour logos, the task has proven to be more difficult. Usually two colours and well-known brands were successful (e.g. Tesco, NASA, Pepsi, IKEA, LEGO), but there was an exception (Starbucks).

Logos of three or more colours have already caused difficulty. The DM logo consists of blue, red and yellow colours, which were correctly coloured by 3 out of 4 users, but one user used light blue instead of red colour and red instead of yellow colour.

The Burger King logo, which is also blue, red and yellow, was problematic. One in three people associates colours with good partitions. One person chose black instead of red. The third person chose red instead of blue and black instead of red.

The biggest problem was if it was already 4 colours like the ebay logo. The observers coloured it with such different colours that the average colouring is not like the original one. One in four people correctly coloured the blue letter "b", the others chose a shade of red. The green "y" was coloured only two by green. The yellow letter "a" was only correctly coloured by one person, and no one was coloured red by the letter "e" see Fig. 11.

Figure 11: Nobody coloured the ebay logo well - average colour (left) and original (right)

Conclusion

Unfortunately, the test could not be completed with users of different origins, and the gender ratio is not sufficient to draw conclusions from it. What can be seen from the results, however, is that the more colours players have to remember, the less they remember well. Since the more people coloured, the more similar the average was to the original, so it is likely that more data would have resulted in a slightly different result.

Acknowledgment

The authors would like to thank the financial support of Széchenyi 2020 programme under the project No EFOP-3.6.1-16-2016-00015.

References

[1] S. Weinschenk, 100 Things Every Designer Needs to Know About People, New Riders, Berkeley, CA, 2011 pp. 27-28
[2] https://designguidelines.withgoogle.com/ads-branding/google-marketing-platform/visual-guidelines.html
[3] https://brand.mastercard.com/brandcenter/mastercard-brand-mark.html
[4] https://designguidelines.withgoogle.com/ads-branding/assets/1hquKP2EcBlDaWyaLkoezqd8r9KPjRgnp/visual-guidelines-google-marketing-platform.pdf
[5] C. Witzel and K. Gegenfurtner. „Memory Color" in Encyclopedia of Color Science and Technology, R. Luo (Ed.), Springer Science+Business Media New York 2013

Gait cycle recording using Kinect ONE sensor

P. Müller[1], A. Nagyváradi[2], L. Szabó[2], M. Gerzson[1,3], Á. Schiffer[1]
[1]Dept. of Technical Informatics {muller.peter;schiffer.adam}@mik.pte.hu
[2]Dept. of Systems and Software Technologies
{nagyvaradi.anett;szabo.levente}@mik.pte.hu
[1,2]University of Pécs, Pécs, Hungary
[3]Dept. of Electrical Engineering and Information Science,
gerzson.miklos@virt.uni-pannon.hu, University of Pannónia Veszprém, Hungary

Abstract: **A gait analysis method is introduced in this contribution in order to help the work of physiotherapists and physicians. The motion is captured by Microsoft Kinect ONE sensor (Microsoft Corporation, USA) and the data are elaborated by a self-developed MATLAB (MathWorks, USA) scripts.**

Introduction

Examining a human movement can provide a wealth of information about a patient's medical condition. This method can be used to diagnose abnormal changes (lesions), ability development and monitor the rehabilitation process of people with reduced mobility. There are several approaches to monitor people, among other things with sensors and various imaging and processing devices [1]. In this case a Kinect v2, as known as Kinect ONE sensor, and a self-developed MATLAB based application were used to examine the movement of the lower limbs. The gait of children with and without lesions was recorded in our measurement system and the measured data was used to identify and to compare their gait of patients. During the evaluation, the position of the skeleton model, the associated body joints and angles can be calculated.

There are different methods to record the human gait. Visual markers can be attached to the body and their signals can be processed by optical equipment. Another solution is to use wearable pressure-sensitive sensors or measure the muscle work [2]. Using these techniques, the following disadvantage has to be counted on: from the fact that sensors are fastened to the specific points of human body it directly follows that wires used for the data transmission and free human walk can be interfered.
Naturally this difficulty can be eliminated by using wireless sensor and methods, however this makes the measurement very expensive.

Cameras and other optical equipment can be used for recording the human's movement but image sensors with high framerate and depth sensitive

solutions are required to get suitable data for gait analysis. These devices are excessively expensive, too.

The aim of our research is to develop a method which makes the gait analysis relatively cheap and easy-to-use even for physiotherapists having shallow knowledge in the field of information technology.

Measurement with Kinect V2 sensor

The Microsoft Kinect ONE sensor can meet the requirements formulated in the introduction. This equipment is widely available and relatively cheap [3]. It transmits 11 data streams in real time including RGB video, depth image data and skeleton model up to six people at the same time. From the point of view of our investigation it is very important that the skeleton model consists of 25 connection points and most of them refer to the joints of the human body. Having the X, Y and Z coordinates of the joints, their spatial positions can be determined. Another advantage of the Kinect that it can record the movement and determine the speed of the walk, too. A main disadvantage of the sensor that it cannot be moved during the measurement because the sensor needs a fix reference point to identify people and calculate the distance [2]. This restriction can cause a serious problem in gait analysis because the sensor can detect the human walk on the range of 0.5 to 3.5 meters from the device. This interval is very short because if the patient is not restricted seriously in his or her movement, the steady state gait cycle cannot evolve. In a frequently used solution the patient walks on a treadmill [4], but in a case like this a special, practically non-moveable equipment is required. If the investigation is aimed to analyze sportsmen's motion, it is highly advantageous that the speed of the walk can be controlled easily. Although, in case of disabled people the treadmill cannot be used, the sensory range of Kinect is sufficient to get the data for analysis.

Our measurement layout

Using the Kinect V2 sensor two types of movement were measured, the investigated persons' walking and their standing up from a chair.

In case of walk the investigated person should be about 3.5 – 4 m from the sensor. In case of standing up the optimal distance between the chair and the camera is about 2 – 2.5 m. The first step is the calibration phase, where the identifier of the person is registered. The determination of the id is an important part of the process especially when the investigated persons need help in their motion and the helper must be distinguished during data processing. The start of the recording can be initiated with a wireless equipment in order not to restrict the sweep and not to be moved the camera. As it was mentioned the measurement has a good reliability on the range of

3.5m to 0.5m from the sensor. The recorded data are the 25 joints of the body in the function of the space (X, Y and Z coordinates) and the time. If there are more than one person in front of the camera, based on the body indexes these data can be distinguished from each other. The camera of the equipment is IR-Time of Flight-operated, its resolution is 512*424 pixels and its theoretical speed is 30 fps, but 25 fps can be reached on average according to our experience. In order to keep the accuracy of the measurement the camera image and the binarized picture of color differentiated body indexes can be seen during the measurement.

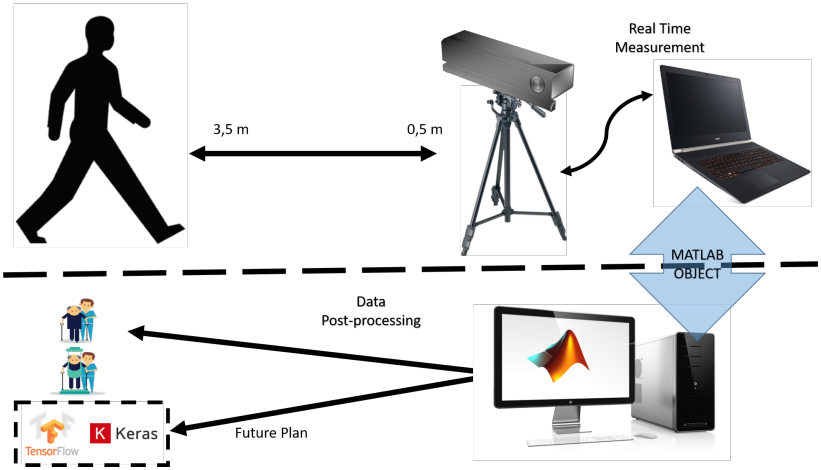


Figure 1: The measurement layout

After a primary conversion the structured data are preprocessed by a script in order to disintegrate and prepare them for further elaboration. This step is the filtering necessary for data transfer to a quicker and higher capacity but not necessarily movable computer in one file. The next step is the filtering in order to reduce the effect of the noise to the data. During the further step of data processing a skeleton model which is based on spatial coordinate points and time can be visualized by a self-developed script. Using this skeleton model, the gait or the standing up can be simulated, the investigated person's movement can be observed.

The evaluated data can be elaborated further using other software packages in order to determine the changes, the efficacy of a treatment, e.g. The data also can be used to statistical or anatomical calculations planned by physiotherapists or physicians [5], [6], [7].

The structure of our measurement layout can be seen in Figure 1.

Evaluation of the measured data

At this stage of our investigation children from our staff were measured under the supervision of a physiotherapist to check our experimental layout. Accordingly, the following results have informative character.

The first person was a 6-year-old boy. In Figure 2 his walk can be seen in the function of time. The Figure 3 contains the angles of hip, knee ankle vs. time curves in the upper diagram, while on the lower diagram the distance vs. time and the position Y vs time curves can be seen.

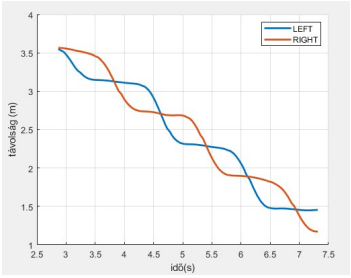


Figure 2: Steps of a 6-year-old boy

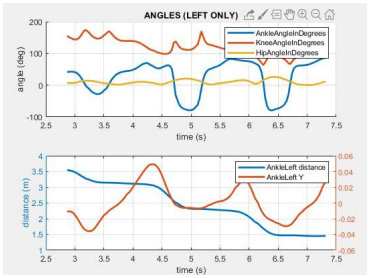


Figure 3: Angles in the course of walk of 6-year-old boy

In Figures 4 and 5 the movement of the left and right ankles and feet, the lower point of spine is depicted in X-Z dimensions.

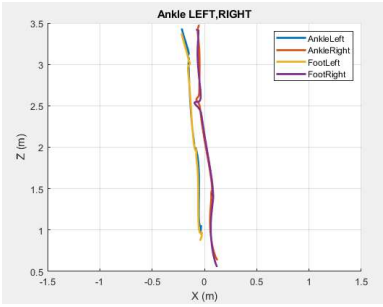


Figure 4: Ankles in X-Z plane 6-year-old boy

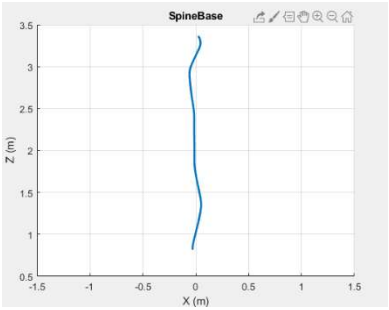


Figure 5: Spine in X-Z plane of 6-year-old boy

Below, as another example, presenting the case of a 12-year-old boy having walking disturbance, the same diagrams can be found.

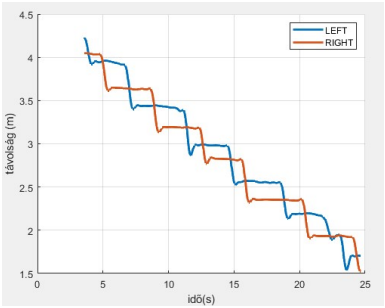


Figure 6: Steps of a 12-year-old boy having walking disturbance

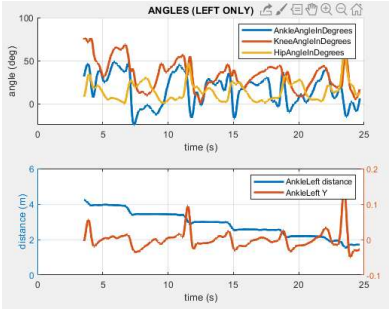


Figure 7: Angles in the course of walk of 12-year-old boy

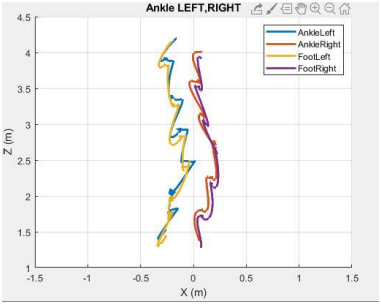


Figure 8: Ankles in X-Z plane 12-year-old boy

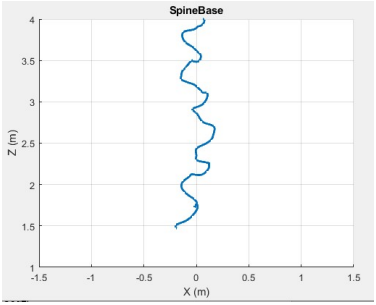


Figure 9: Spine in X-Z plane of 12-year-old boy

Based on the obtained data, the physiotherapists are able to get information about the children's health status. By the help of the processed data, there is an opportunity to predict the duration of rehabilitation process and success rate of the treatment and they are able to identify the early signs that may cause health problems in the future.

Conclusion

A gait analysis method was introduced in our contribution in order to help the work of physiotherapists and physicians. The motion was captured by Microsoft Kinect V2 sensor and the data were elaborated by ~~a~~ self-developed MATLAB scripts. The results can be evaluated by statistical methods in order to test the efficacy of a treatment, e.g.

Acknowledgment

The research project is conducted at the University of Pécs, Hungary, within the framework of the Biomedical Engineering Project of the Thematic Excellence Programme 2019 (TUDFO/51757-1/2019-ITM).

References

[1] T. Guzsvinecz, V. Szücs, C. Sik-Lanyi: "Suitability of the Kinect Sensor and Leap Motion Controller—A Literature Review" SENSORS 19 : 5 p. 1072 , 25 p. 2019

[2] Muro-de-la-Herran A., Garcia-Zapirain B., Mendez-Zorrilla A. Gait analysis methods: An overview of wearable and non-wearable systems, highlighting clinical applications, Sensors, Vol. 14, No. 2, 2014, pp. 3362–3394.

[3] G. Roy, A. Bhuiya, A. Mukherjee, S. Bhaumik "Kinect Camera Based Gait Data Recording and Analysis for Assistive Robotics, Procedia Computer Science, 133. pp 763-771. 2019

[4] X. Xu, R. W. McGorry, L. S. Chou, J. Lin, C. Chang: "Accuracy of the Microsoft Kinect for measuring gait parameters during treadmill walkin. Gait&Posture 42 (2) 145-151 2015

[5] Clark R. A., Pua Y. H., Fortin K., Ritchie C., Webster K. E., Denehy L., Bryant A. L. Validity of the Microsoft Kinect for assessment of postural control, Gait Posture, Vol. 36, No. 3, 2012, pp. 372–377.

[6] F. Coutts , Gait analysis in the therapeutic environment, Manual Therapy, Vol. 4, No. 1, 1999, pp. 2–10.

[7] Surer E., Kose A. Methods and technologies for gait analysis, in Computer analysis of human behavior, Salah A., Gevers T. (Eds), Springer, 2011, pp. 105-123.

Comparing the Reflection and the Refraction of Light Phenomena in Reality to their Counterparts in a Virtual Environment Rendered by Cycles

Tibor Guzsvinecz[1], Tibor Medvegy[2]
[1]Department of Electrical Engineering and Information Systems, University of Pannonia, guzsvinecz@virt.uni-pannon.hu
[2]Institute of Physics and Mechatronics, University of Pannonia, medvegyt@almos.uni-pannon.hu
8200, 10 Egyetem utca, Veszprém, Hungary

Abstract: **This article compares the real-world reflection and refraction of light phenomena to their counterparts in a virtual environment. For this, the authors used the Cycles Render technique in Blender. This technique uses a rendering method called Path Tracing which is a type of Ray Tracing to create photorealistic visuals. To achieve this comparison, the authors modeled a virtual scene in Blender with equal parameters to the scene in reality. A glass container was used and it was filled with water and vegetable oil. Two tests were done with a coin and two with a small stick in each state of the glass container. It can be concluded that the virtual indices of refraction of liquids are similar to reality. However, the number of clearly visible reflections differ.**

Introduction

When creating virtual objects and worlds, it has to be as realistic as possible to be immersive to the user. Therefore, they not only have to be modeled realistically, but they have to be rendered photorealistically as well. Two of the most photorealistic renderers are called Ray Tracing and Path Tracing. The former originated from Ray Casting [1] and undergone many upgrades over the years. The most recent definition is that it traces the path of light back from the virtual camera to the light source(s) [2]. The latter is not different from Ray Tracing, as it is a type of it. Path Tracing uses the Monte Carlo method [3]: the rays only produce a single ray per bounce and they go into a random direction. Afterward, all rays are sampled randomly and the final, photorealistic image is created.

According to Whitted [4], the Ray Tracing rendering method can simulate shadows, true reflection and refraction. The latter two were also investigated by de Greve [5] while writing a ray tracer: to create photorealistic visuals, reflection and transmission should be realistic as seen in Figure 1.

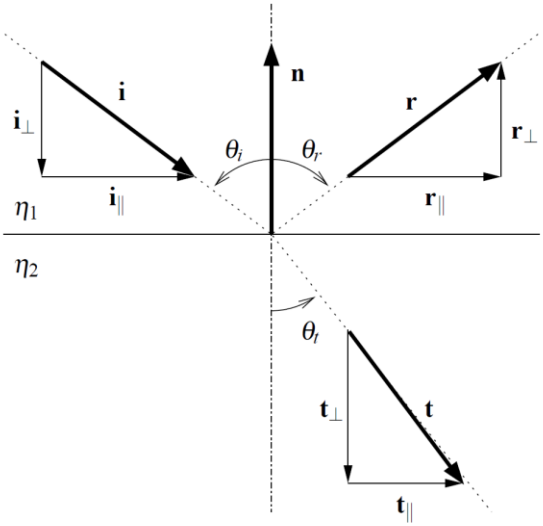


Figure 1. The reflection and refraction of light in reality [5].

To correctly – or realistically – simulate the reflection of light in virtual environments, the angle of reflection has to be equal to the angle of incidence. In the case of the refraction, Snell's Law has to be followed [6]. This law states that the sines of the angles and the products of the refractive indices must be the same.

These photorealistic simulations are important as the users can feel more immersed if the virtual world is realistic: the graphics [7] and the used devices matter [8]. Since Ray Tracing rendering methods – thus, Path Tracing as well – follow the path of light, the geometric optics phenomena can be modeled physically accurately. However, one question comes into mind: how physically accurate is the rendered image? To answer a part of this question, the authors of this article focused on the reflection and refraction of light in the real world and a virtual environment. Therefore, the authors created environments in both realities with the same parameters and rendered the virtual one using Cycles Render [9] in Blender. This rendering technique uses Path Tracing.

Methodology

As mentioned in the introductory section, the advantage of the Ray Tracing rendering methods is that they can model the reflection and refraction

phenomena. Therefore, the authors created an experimental setup, where both phenomena can be easily modeled and investigated. To see whether Path Tracing is physically accurate, the virtual environment and the real environment had to have the same setup and parameters. For the investigation and comparison, the authors used a glass container, water, vegetable oil, a coin and a small stick.

First of all, a glass container with the dimensions of 30 cm * 10 cm * 6 cm was acquired. The thickness of its glass walls was 0.3 cm. After acquiring the container, it was modeled in the software called Blender with the same dimensions. A virtual spotlight-type light source was positioned above the glass container similarly to the scene in reality. Then, a virtual coin was created with the same dimensions as a 10 Hungarian Forint coin. Its diameter is 2.48 cm and its thickness is 0.13 cm. A virtual stick was also created with a diameter of 1.6 cm.

To model the path of virtual light as similarly as possible to reality, the most important information to know is the Index of Refraction (IoR) of the medium which describes the behavior of light at the borders of the medium.

To create the virtual glass material, the authors used a Bidirectional Scattering Distribution Function (BSDF) [10], namely the Principled BSDF. The transmission of the material is set to 100% and its roughness to 0% as they do not affect the angles of refraction and reflection. Thus, the scattering phenomenon in the medium and the roughness of the surface of the object were not taken into account. Its IoR value was set to 1.5 which is the IoR of the used glass in reality. The Solidify modifier was used on the virtual glass to set its thickness and by doing so, correct some of the reflections: without it, the reflections that should have been on the backside of the glass container appeared on the front side.

For the creation of the virtual liquid material such as the water and the vegetable oil, the Principled BSDF was used as well. Originally, the Glossy BSDF and the Refraction BSDF were used on the virtual liquid. However, after the table under the container was created, these BSDFs did not produce correct reflections: if the table was longer than the container, no reflections appeared on the front side of the container. The reflections only appeared on the front side of the glass container if the table had the same length as the container.

To simulate the refraction in the liquids, the IoR values were set to 1.333 and 1.466 in the case of the virtual water and the virtual vegetable oil materials, respectively.

After everything was created, the authors used the following camera parameters that are presented in Table 1 to compare the IoR values between the two realities. Each test was done with an empty glass container, then with water / vegetable oil level of 5 cm. T_n refers to the test number.

Table 1: Camera parameters during the comparison.

	Distance from the Object (x, y, z)	**Camera Angle (x, y, z)**
Coin and Stick T_1	36.8 cm, 9.7 cm, 18.2 cm	60°, 0, 109°
Coin and Stick T_2	29.3 cm, 0.7 cm, 11.45 cm	60.3°, 0.8°, 87.8°

Results and Discussion

Since the authors conducted two tests with both the coin and the stick, this section is broken into two subsections: the first subsection is about the results of T_1 and the second presents the results of T_2. For the figures, both the coin and the stick were put in the container to make the article less redundant.

Results and Discussion of T_1

In this subsection, the results of T_1 is presented and discussed. As mentioned in the Methodology section, three states were created in both realities: a water level of 5 cm, a vegetable oil level of 5 cm in the container, and an empty container. In Figure 2, comparisons are shown between the containers that are empty or filled with water or vegetable oil in both real and virtual environments.

First of all, the empty container is investigated. As there is no liquid, only the reflection phenomenon can be examined. The reflections are almost the same in both cases. The only difference is that in the case of the rendered image, they are more visible. The reflections that are on the furthest side of the container are almost invisible in the real world, but if the reader looks closely, they can be observed.

Afterward, the containers with the liquids are investigated and the results are interesting. While the IoR of the liquids is similar in each case according to the apparent positions of the coin and the stick, the number of reflections in the rendered images differs when the authors examined the stick. When it enters the liquid, a reflection can be noticed in the rendered images. Also, more differences exist, all occurring in the images taken in the real world. One such difference is at the end of the stick in the liquid: a reflection can be noticed on the left side of the container, while in the rendered image it is not visible. Another difference occurs in the case of the filled containers: On the right side of the container, the coin and the stick are reflected, though they are barely noticeable.

Similarly, at the upper left corner on the front of the real-world container, there is a reflection of the stick. This reflection is barely noticeable in the case of the rendered images. However, it is more observable with vegetable oil than with water.

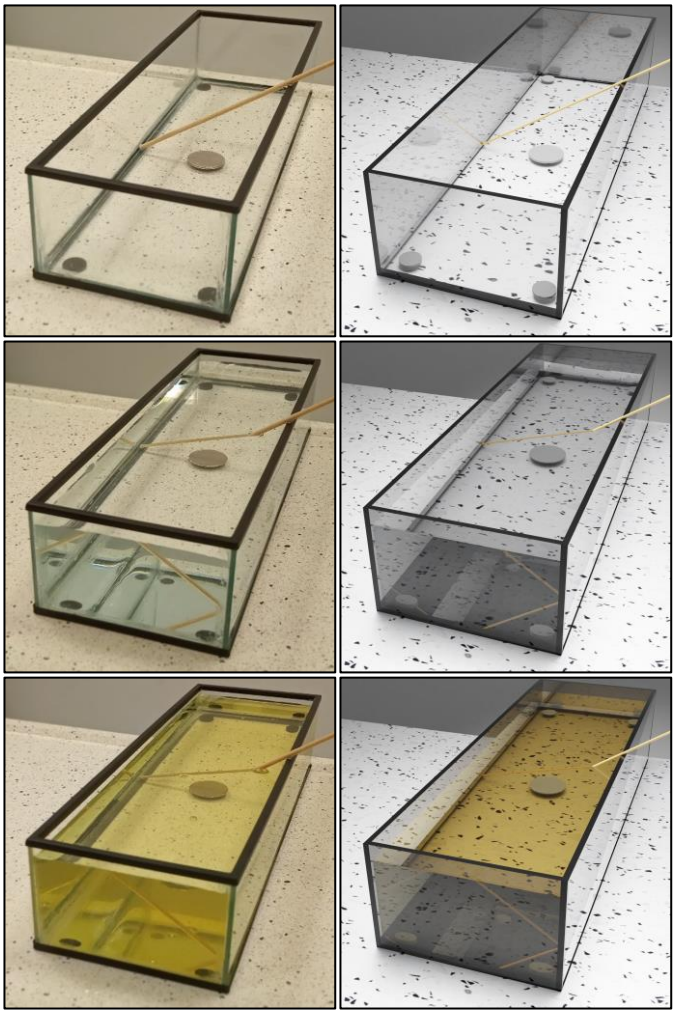


Figure 2. Comparisons between the glass containers that are empty (top), filled with water (middle) or vegetable oil (bottom). The left pictures are from reality, while the ones on the right are rendered.

Results and Discussion of T_2

The results of T_2 are presented and discussed in this subsection. Similarly, to T_1, three states were created in this case as well. In Figure 3, comparisons

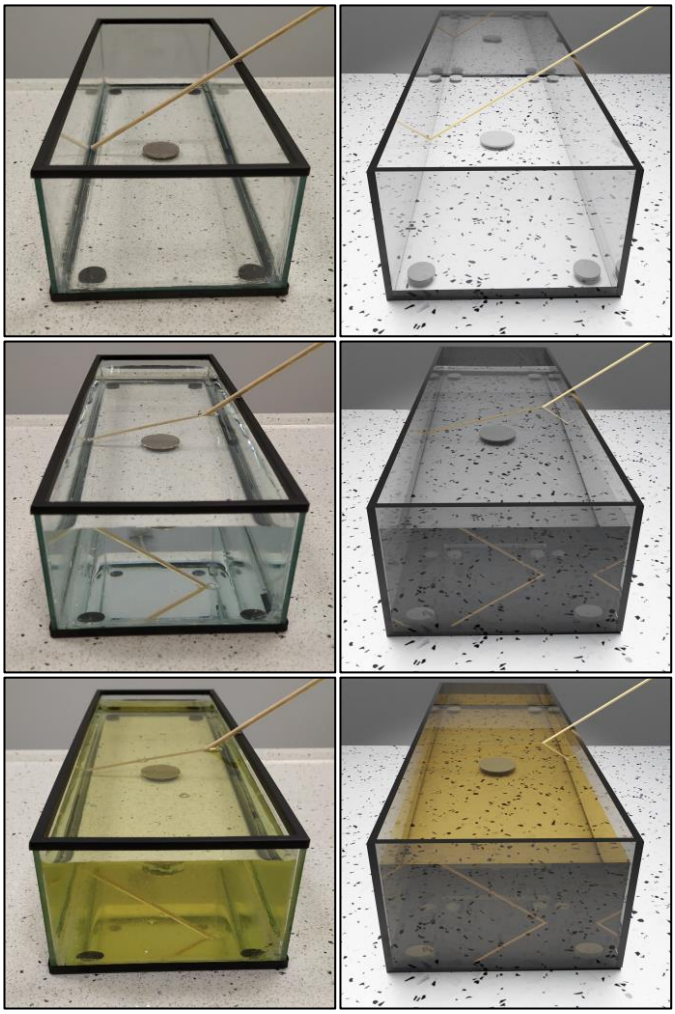


Figure 3. Comparisons between the glass containers that are empty (top), filled with water (middle) or vegetable oil (bottom). The left pictures are from reality, while the ones on the right are rendered.

are presented between the empty containers and those which filled with water or with vegetable oil in both real and virtual environments.

When looking at the empty container, it can be easily noticed that the results are similar to T_1. Reflections exist in both reality and the virtual environment, but the reflections that are on the backside of the rendered container are more noticeable than in reality.

With the liquids, however, there are differences between the results of T_1 and T_2. The first thing to notice is on the left side of the container. The end of the stick is reflected on the glass in all cases, therefore it is possible that it was reflected in the rendered images of T_1 as well, but it was not observable in that angle.

The reflection and refraction numbers also differ between the realities and the tests. In the rendered images the stick is not only refracted in the water, but multiple reflections of it can be observed: one such reflection is when the stick enters the liquid, and the other is of the same phenomenon, albeit in a lower position. One more reflection of the stick can be seen on the backside of the container. This reflection is not present in reality. Another reflection of the stick is noticeable at the right side of the front glass of the container. Half of this reflection can be seen in reality as well when the container is filled with water. Therefore, it is possible that this reflection is also there in the case of the vegetable oil, but it is barely noticeable.

Conclusions

In this article, the authors compared the reflection and refraction of light phenomena in reality to their counterparts in a virtual environment. The goal was to assess whether Path Tracing is physically accurate. To render the virtual environment, the authors used the Cycles Render technique in Blender.

The authors created an experimental setup using a glass container where the same environment could be easily modeled with equal parameters as in reality. Two tests were done both with the water and vegetable oil. A coin and a small stick were put inside the container to test the reflections and refractions.

After the investigation, the authors concluded that the IoR of the liquids in the virtual environment that uses Path Tracing are similar to reality. However, during the comparisons, the authors noticed the differences in the number of reflections. Their visibility also differs between the two realities. In the liquid at the part where there is no glass in front of the stick, the number of reflections is greater when talking about the rendered image. If the glass is

positioned between the stick and the eye (or the virtual camera), its reflections became less noticeable than in reality.

In conclusion, it can be said that Path Tracing is almost physically accurate according to the results of the refraction and reflection tests. The refraction phenomenon is perfectly simulated. However, some flaws exist in the case of the reflection phenomenon. These flaws are not huge, and they may not be noticeable by a person who is not accommodated in the field of physics.

Acknowledgment

The authors would like to thank Sándor Guba for his help in acquiring camera tripods and taking images of the scene in reality. The hardware for the research was supported by the Nation's Young Talent Scholarship of the National Talent Program, under the NTP-NFTÖ-19-B-0090.

References

[1] A. Appel. “Some techniques for shading machine renderings of solids.” in *Proceedings of the spring joint computer conference*, Atlantic City, NJ, USA, 30 April – 2 May, 1968, pp. 37-45

[2] K. Suffern. “Ray Tracing from the Ground up.” CRC Press. 2016

[3] E. Lafortune. “Mathematical models and Monte Carlo algorithms for physically based rendering.” Department of Computer Science, Faculty of Engineering, Katholieke Universiteit Leuven, 1996

[4] T. Whitted. “An improved illumination model for shaded display.” Communications of the ACM, vol. 23, no. 6, pp. 343–349, 1980

[5] B. de Greve. “Reflections and refractions in ray tracing.” 2006

[6] M. Born, E. Wolf. “Principles of optics: electromagnetic theory of propagation, interference and diffraction of light.” Elsevier. 2013

[7] S. Papagiannidis, E. Pantano, E. W. See-To, C. Dennis, M. Bourlakis. “To immerse or not? Experimenting with two virtual retail environments.” Information Technology & People, vol. 30, no. 1, pp. 163–188. 2017

[8] J. Tham, A. H. Duin, L. Gee, N. Ernst, B. Abdelqader, M. McGrath. “Understanding virtual reality: Presence, embodiment, and professional practice.” IEEE Transactions on Professional Communication, vol. 61, no. 2, pp. 178–195, 2018

[9] B. Iraci “Blender Cycles: Lighting and Rendering Cookbook.” Packt Publishing Ltd. 2013

[10] F. O. Bartell, E. L. Dereniak, W. L. Wolfe. “The theory and measurement of bidirectional reflectance distribution function (BRDF) and bidirectional transmittance distribution function (BTDF).” Radiation scattering in optical systems, vol. 257, pp. 154-160, International Society for Optics and Photonics. 1981

Changing your mindset: a desktop application supporting the changeover from ICD-10 to ICD-11

Csongor Miklós Cziráki, Lilla Viktória Kókai, István Vassányi, László Balkányi

University of Pannonia, Medical Informatics R&D Centre
H-8200 Veszprém, Egyetem u. 10.
cz.m.csongor@gmail.com

Abstract: **This paper introduces a desktop framework which aims to help experts change over to the new 11th version of the International Classification of Diseases coding system. The project focuses on the usage of the new coding system for financing the health care delivery system by providing support for the interactive re-definition of the current Diagnosis Related Groups with the new ICD-11 codes. Such a solution will be a useful tool in the near future when the ICD-11 becomes mandatory to use.**

Key words: ICD-11, semantical mapping, healthcare code systems, DRG

Introduction

In more than the past ten years the Word Healthcare Organization (WHO) have prepared a complete overhaul of the International Classification of Diseases (ICD) medical classification, the current 10th version used since the nineties, being obviously outdated. The new version, ICD-11 will be mandatory to be used in every participant country, including Hungary [3]. This new ICD-11 system promises to bring groundbreaking changes to many parts of the healthcare system where disease codes are used. Opposed to the currently used ICD-10, the new version is not just a linear hierarchy, where it is really hard to make all the connections between medically related concepts, but more like a web of concepts, which can be thought of as an oriented, weakly connected graph [1].

In order to simplify public healthcare financing, some countries, like Hungary, use a so-called Diagnostic Related Group (DRG) system in which healthcare case types with similar costs are grouped together. The medical care providers use the DRG system in their monthly reports to reimburse their costs from the National Health Insurance Fund. DRG groups are defined by the diagnoses and medical interventions, like surgical procedures, involved

in the case. Since these case groups are attributed and classified by ICD-10 codes (among others), they will be outdated as soon as the changes take place.

In this pilot project, our goal was to develop a software framework which will allow experts to create new DRG's or update existing ones, supporting an interactive but automated semantical matching between ICD-10 and ICD-11. The two main components of the solution are 1) a relational database that contains the whole BNO-10, ICD-10, ICD-11 and DRG systems as well as any available structured information that connects them, and 2) a desktop application that uses this database as a backend. To our best knowledge, this project is the first initiative in this field in Hungary.

Methods

The backend database

The ICD-11 coding system contains a web of concepts called the 'Foundation' in which nodes van have multiple parents and are identified by URI's, and 'Linearizations', which are composed of Foundation nodes in a single-parent tree structure, with a code assigned to each node in a linear code space. A linearization already published by WHO is the Mortality and Morbidity Statistics (MMS), developed for the classic diagnostic processes. Both in Foundation and the Linearization use the 'is-a' relationship between child and parent nodes. A node in the MMS can also have 'post-coordination' nodes attached to it along pre-defined 'axes' like laterality, etiology, severity, body parts etc [2]. These nodes are also parts of the Foundation concept web, as ICD-11 contains not only diagnosis related concepts, rather, it serves as a general knowledge model for the healthcare domain.

The ICD-10 model is a simple, strictly linear hierarchy with only a few properties like title and code. The presence of a feature, like 'severe', for an MMS code can be expressed concisely via post-coordination, but in ICD-10 a separate code must be used for such purposes. This structural difference makes the mapping between the two coding systems far from obvious.

Though WHO provides web API's for the browsing of the Foundation and the MMS, these are slow and not always reliable, so it was necessary to store all information in a local database. The relational database model was designed to support all the details of the Foundation and the MMS, and the mapping to and back ICD-10 as well as the Hungarian version of ICD-10 called BNO-10. Since the ICD-11 contains qualitatively and quantitatively more medical concepts then the ICD-10, the mapping from ICD-11 to ICD-10 is not entirely possible. For the ICD-10->ICD-11 mapping, the WHO created a few mapping tables that we also included in the database. The

general structure of a DRG was also included in the model. For the sake of clarity every coding system has its own schema in the database.

BNO-10 and ICD-10 are slightly different, there are some cases where the national code is not the same as the international one, mostly due to the different length of cycles of updating. The mapping between the two cannot always be done automatically, an expert is needed to translate and approve that the two codes represent the same diagnostic concept. The last relation to be made is between ICD-10 and the DRG. This is more trivial, since the national healthcare legislation on DRG documents these relations as an attachment.

The final database contains 27 tables separated in 5 schemas for the ICD-11, CD-10, BNO-10, DRG and the connection tables, respectively.

The desktop application

Creating a new DRG requires knowledge of the hospital care costs, and linking the current ICD to the new one requires medical expertise. This means that the user interface of the product should be minimalistic and clean, yet professional.

The software provides two methods to start working on the DRG system:

- Creating a new category in the grouping system from scratch
- Importing an existing category from the database

After one of these has happened the expert can start the editing, by adding custom ‘chapters’ and ‘groups of related diagnosis’, deleting existing ones or editing their properties. A ‘chapter’ only has a title and a number, while a ‘group’ has the following editable properties: group code, group title, simple title, minimum and maximum days of hospitalization, normative days of hospitalization and weight. Once a ‘group’ has been created the user can assign ICD-11 concepts to it, by typing the concept code or definition. The software can usually recommend a connection between an ICD-10 entity and an ICD-11 concept; in these cases the user can even enter the ICD-10 code, then the program automatically assigns the ICD-11 version of the same medical concept. This feature helps a lot to those, who are not yet familiar with the new ICD-11 codes. The process will result in a hierarchy which is easy to browse by a tree viewer commonly seen in a filesystem manager.

Since the ICD-11 system will be unfamiliar for many people, the program provides a separate viewer to navigate between ICD-11 concepts. This view is based on the MMS linearization, represented as a tree graph. The whole

linearization contains thousands of medical concepts, therefore the viewer only shows the parent and the siblings of the selected concept (Fig. 1).

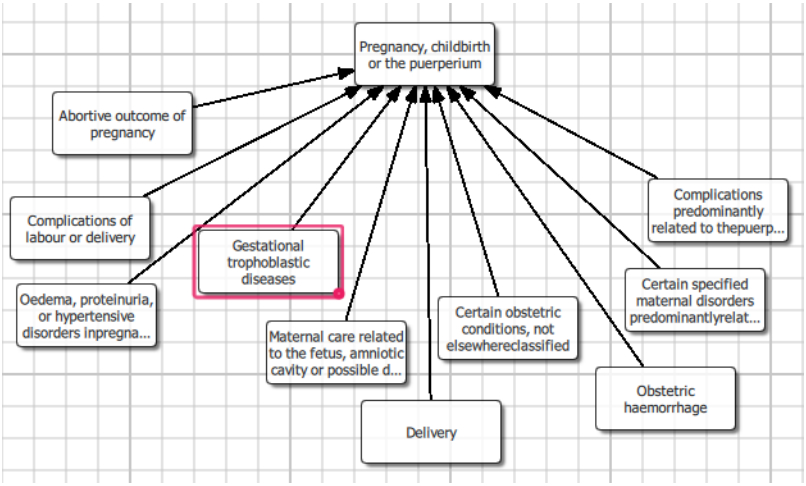


Figure 1: *Example of the ICD-11 viewer. The selected concept is Gestational trophoblastic.*

Implementation technology

For the database technology the Microsoft SQL Server was selected. and for uploading this database we used scripts written in Python. The Python language is very practical for processing data from different sources, because of the wide variety of libraries provided for it. The ICD-11 and ICD-10 data are available for the public by using an API provided by the WHO [5], the DRG and the national version of the ICD-10 is also public, but in text document format (xlsx, docx, dbf) [6]. Python can handle both sources very easily. To connect and upload to the database a pyodbc library is available for the language.

In terms of creating a modern graphical user interface, there were a few technologies to consider. The most popular ones to create desktop applications are JavaFx, C# and C++/Qt. To make a web application usually Angular or React JavaScript frameworks are used. Since it is not important that the software should be widely available and installation is not a problem, we decided to develop a desktop application. The application has to meet the following criteria: It has to look modern, but easily manageable, connection to the database should be simple programmatically and it has to be reasonable fast. Since it handles a lot of data the more control we have over it, the better.

JavaFx is able to produce really modern and clean user interfaces, but managing the database connection is far from trivial and it provides less control over the data than the other options. While C# makes the database management extremely easy, it is not flexible and it is not the best choice for making modern, clean user interfaces. In the end we decided to choose C++ and the Qt widget framework to create the interface. Thanks to its popularity there are many libraries that support database connections. Moreover with Qt widgets it is easy to create modern and appealing interfaces.

Results

The full content of the ICD-11 Foundation and MMS, the published mappings between ICD-10 and the MMS, and the ICD-based definition of a selected DRG class have been modeled, loaded and stored in the backend database. This database is available from the authors for research purposes. The beta version of the desktop app has been finalized and tested on the 289B (Gastro-intestinal hemorrhage) DRG of the 06M main DRG class as an evaluation pilot. The stable version will soon be available for community tests.

Conclusion

The paper presented the first results of a pilot project that developed IT support for the modeling of the semantical relationships among the DRG, BN0-10, ICD-10 and ICD-11 coding systems. The interactive app and the backend database will be a useful tool for the DRG change-over in the near future.

References

[1] Tu, S., O. Bodenreider, C. Celik, C. Chute, S. Heard, and R. Jakob. et al. "A content model for the ICD-11 revision." Technical Report BMIR (2010): 2010-1405.

[2] Tudorache, T., Falconer, S., Noy, N.F., Nyulas, C., Üstün, T.B., Storey, M.A. and Musen, M.A., 2010, October. Ontology development for the masses: creating ICD-11 in WebProtégé. In International Conference on Knowledge Engineering and Knowledge Management (pp. 74-89). Springer, Berlin, Heidelberg.

[3] https://www.who.int/classifications/icd/en/

[4] https://icd.who.int/icd11refguide/en/index.html#1.2.4GeneralfeaturesofICD-11|general-features-of-icd11|c1-2-4

[5] https://icd.who.int/icdapi

[6] http://www.neak.gov.hu/virtualis_rovat/altfin_virt_dok2/archiv_torzsek