KPI-based Activity Planning for People Working in Flexible Processes

Maikel L. van Eck [*], Natalia Sidorova, and Wil M.P. van der Aalst

Eindhoven University of Technology, The Netherlands
{m.l.v.eck,n.sidorova,w.m.p.v.d.aalst}@tue.nl

Abstract. Planning human activities within business processes often happens based on the same methods and algorithms as are used in the area of manufacturing systems. However, human resources are more complex than machines. Their performance depends on a number of factors, including stress, personal preferences, etc. In this paper we describe an approach for planning activities of people that takes into account business rules and optimises the schedule with respect to one or more KPIs. Taking a task list, a set of rules or constraints and a KPI calculation model as input, we automatically create an executable model that captures all the possible scheduling scenarios. The state space of this executable model is explored to find an optimal schedule.

1 Introduction

Scheduling is a well-known type of optimisation problem that occurs in many different contexts, e.g. flexible manufacturing systems, transportation and personnel planning [2,4,9]. Scheduling problems exist in many variations, but often a number of jobs or activities have to be performed by a set of resources while obeying certain constraints. The goal is to divide the activities over the resources and time so that their execution optimises one or more performance measures. In general, scheduling is complex and NP-hard. Different approaches have been developed to deal with this complexity, providing both exact optimal solutions and heuristic approximations.

An aspect that is often not taken into account when planning human activities is that, unlike machines, human performance changes depending on a number of factors [4]. For example, when a person has too much stress, their performance is decreased [5]. By changing for example the lunch time, the performance might be improved or deteriorated. Planning too many challenging activities after each other can also influence the performance.

It has been shown that there are strong relations between work-related stress and deterioration of productivity, absenteeism and employee turnover [3, 5]. Stress and the associated health issues are a big financial burden for both organisations and society in general, with the European Commission estimating the total yearly financial cost at €25 billion. Fortunately, new technological advances

[*] This research was performed in the context of the TU/e IMPULS project.

Activity	CaseID	Time	Employee	Stress
Problem Intake	1	09:00-10:00	John	2
Problem Intake	2	10:00-11:00	John	3
Repair Product	1	11:00-12:00	John	4
Repair Product	2	12:00-13:00	John	6
Break	-	13:00-14:00	John	5
Document Issue	1	14:00-15:00	John	6
Document Issue	2	15:00-16:00	John	8

(a) John's workday of 7 hours, with a maximal stress level of 8.

Activity	CaseID	Time	Employee	Stress
Problem Intake	3	09:00-10:00	Anna	2
Repair Product	3	10:00-11:00	Anna	3
Document Issue	3	11:00-12:00	Anna	4
Break	-	12:00-13:00	Anna	2
Problem Intake	4	13:00-14:00	Anna	3
Document Issue	4	14:00-15:00	Anna	4
Break	-	15:00-15:30	Anna	3
Repair Product	4	15:30-16:30	Anna	4

(b) Anna's workday of 7.5 hours, with a maximal stress level of 4.

Fig. 1: A scheduled workday for two employees, each with a personal schedule.

and smart sensor technologies enable people to unobtrusively monitor their personal stress levels and become more aware of sources of stress at work and stress patterns [8]. In this paper, we explore how the planning of daily activities can be performed with the goal to better manage stress for the employees involved.

As an example, Fig. 1 shows a scheduled workday for two employees working in a hardware maintenance process, whose stress is being monitored. Both employees have each been assigned to work on two cases, but the activities have been scheduled differently. John's schedule in Fig. 1a results in a shorter workday, while Anna's schedule in Fig. 1b results in less stress. Which personal schedule is better depends on the desired tradeoff between time and stress. Of course, due to the effect of stress on performance, the last two activities in John's schedule may actually take longer than usual, so this information also has to be taken into account in the planning.

In this paper we present an activity planning approach to find optimal schedules with respect to one or more *key performance indicators* (KPIs), e.g. stress or workday length. We focus on flexible environments, where planning can be adjusted to people's needs.

The structure of the rest of this paper is as follows: In Sect. 2 we explain our planning approach. Then we discuss our first implementation in Sect. 3 and evaluate it in Sect. 4. Finally, in Sect. 5 we conclude the paper and state several areas for future work.

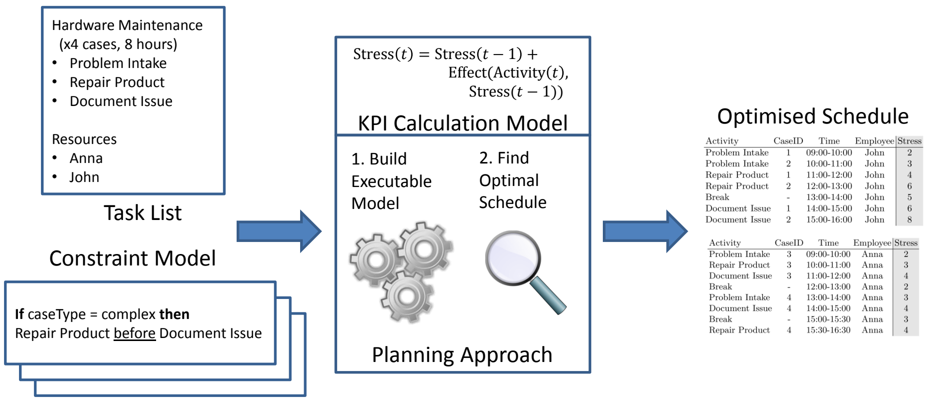
2 Conceptual Approach

A graphical overview of our approach is shown in Fig. 2. We first discuss the input required and then we describe the approach itself.

2.1 Required inputs

We take a task list, the time period in which the activities should take place, a constraint model and a KPI calculation model as input.

The *task list* tells us what activities should be scheduled together with the available resources to divide the activities over. Instead of listing planned activities, the task list can also be extracted from a historical event log. This can be



Activity	CaseID	Time	Employee	Stress
Problem Intake	1	09:00-10:00	John	2
Problem Intake	2	10:00-11:00	John	3
Repair Product	1	11:00-12:00	John	4
Repair Product	2	12:00-13:00	John	6
Break	-	13:00-14:00	John	5
Document Issue	1	14:00-15:00	John	6
Document Issue	2	15:00-16:00	John	8

Activity	CaseID	Time	Employee	Stress
Problem Intake	3	09:00-10:00	Anna	2
Repair Product	3	10:00-11:00	Anna	3
Document Issue	3	11:00-12:00	Anna	4
Break	-	12:00-13:00	Anna	2
Problem Intake	4	13:00-14:00	Anna	3
Document Issue	4	14:00-15:00	Anna	4
Break	-	15:00-15:30	Anna	3
Repair Product	4	15:30-16:30	Anna	4

Fig. 2: An overview of the activity planning approach.

done to see how to improve on a previous execution according to one or more given KPIs. The task list defines the size of the scheduling problem.

The *constraint model* is a collection of rules that have to be upheld for a schedule to be viable. They may describe the order in which activities are executed, specify which people may do what activities, or impose other restrictions on activity executions, e.g. an activity can only be done at a certain location or time. An example of such a rule is "For complex cases, *Repair Product* has to be done before *Document Issue*". Therefore, the schedule in Fig. 1b is only valid if case 4 is a simple case. Rules can even specify that additional activities, not mentioned in the task list, have to be executed. For example, if two activities from the task list are scheduled to be executed sequentially by the same person but at different locations, then that person will need to *travel* between the locations. To specify which people may do what activities the constraint model can define each employee's skills and the competencies needed for each activity [4]. The constraint model also specifies the durations of activities, which may vary depending on different factors. Parameters such as stress levels may influence these durations and the constraint model can describe how, based on patterns found by mining personal historical logs obtained with smart technologies [1,8].

The KPI calculation model defines the quality metrics for schedules. It specifies how the value of one or more specific KPIs can be calculated for a given (part of a) schedule. Examples of KPIs are the length of a working day of people involved, the maximal stress level, and the stress level at the end of the working day. If more than one KPI is specified then a tradeoff needs to be made. This can be done by giving each KPI an importance weight and normalising the KPI values, or by constructing a Pareto front of schedules and allowing the end-user to make the tradeoff [2].

2.2 Planning Approach

The goal of the planning approach is to take the input described above to produce an optimal schedule, or a set of optimal schedules. This approach should ideally not require the end-user to have scheduling expertise or to provide additional input.

Simply generating all permutations of activities divided over time and resources is not practically feasible because there are too many possibilities, while most result in invalid schedules due to violated constraints. A common scheduling technique is to use integer programming, which explores the solution space without explicitly enumerating all possible schedules. However, setting up an integer program is a complex task, especially because of history-dependent variables such as the location of a person, stress level and stress-dependent performance.

Therefore, we propose a planning approach consisting of two stages. First, an executable model that generates valid schedules is automatically constructed from the input described above. Second, this model is used to find the optimal schedule(s).

This approach is similar to the ones in [7, 9] on scheduling in flexible manufacturing systems (FMS) and on compliance checking of interrelated compliance rules. In [9], an FMS is modelled as a Petri net (PN), after which a schedule is generated by analysing the state space of the PN. However, the construction of the PN is done manually and the FMS modelling constructs are not as flexible as the rules in the constraint model described above. In [7], a compliance checking framework is described where compliance rules are specified using a formal language and the resulting constructs are translated to Coloured Petri nets (CPNs) [6]. A collection of such CPNs is then composed into one executable model on which the compliance of a given sequence of activities with the rules is checked. However, time is not explicitly modelled and activities that occur in multiple rules occur multiple times in the composed CPN, so it is not possible to use the composed CPN to generate activity schedules.

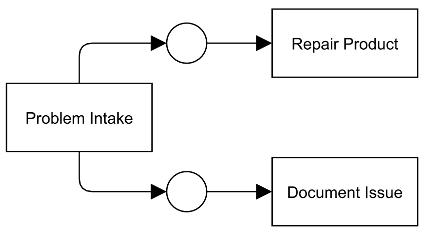
3 Approach Implementation

We have created a preliminary implementation of the proposed activity planning approach as a plug-in in the process mining tool ProM [11]. In this section we discuss how we construct the executable model, specified as a CPN, and how the state space of the executable model is explored to find the optimal schedule.

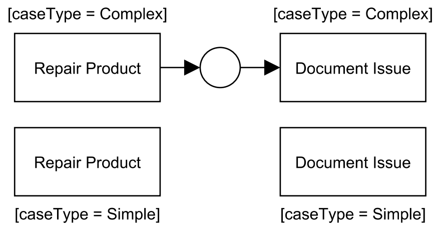
3.1 Building an Executable Model

The executable model is created by combining the inputs described in Sect. 2. It represents the search space of schedules, which is then the input for finding the optimal schedule. To create an executable model, the rules of the constraint model have to be combined and translated to the representation of the executable model.

There are many different ways to express rules or constraints in constraint models [7, 10] We can describe rules in a natural language or use formalisms like LTL. It is also possible to use process models that precisely define what is or is not allowed when executing relevant activities [1]. These models can be



(a) *Problem Intake* occurs before *Repair Product* & *Document Issue.*



(b) If the *caseType* is *Complex*, then *Repair Product* occurs before *Document Issue.*

Fig. 3: Two rules from a constraint model expressed as CPNs.

constructed either by hand or mined from event data related to the involved processes. Another option is to represent each rule as a pattern in a process modelling language, as a mix between creating one big model and describing each rule using a formal rule language.

We choose to model the individual rules of the constraint model as CPNs. One reason for this choice is that CPNs are expressive, but single rules are easier to model than entire processes. Another reason is that CPNs provide us directly with executable models, so we only need to combine them. Two examples of constraints modelled as CPNs are shown in Fig. 3. Activities can be supplied with guards that restrict the conditions relevant to the activity execution, e.g. *caseType* referring to the complexity of a case. When combing rules and creating the schedules we also take into account these guards.

Combining the rules in the constraint model is done using an adapted form of the synchronous product defined for PNs [12]. The adapted synchronous product matches activities by name and then merges not only their dependencies and relations, but also their guards. Guards are merged by taking the conjunction of the guards of the merged activities. Initially, the task list is used to create a basic PN that can execute all required activities exactly as often as needed, which is then sequentially composed with each rule using the synchronous product. The executable model obtained after this step is enhanced with the modelling of resources and time. This results in a single executable model that captures all the rules from the constraint model, as shown by Fig. 4

In our implementation we choose to merge the KPI calculation model with the executable model. Each KPI is tracked separately and their value is updated upon each activity execution. This makes measuring the KPI explicit, even for partial schedules, which helps when finding an optimal schedule. However, the main reason for merging the KPI calculation is that some KPIs, like stress, need to be modelled anyway because they affect the duration of activities.

3.2 Finding an Optimal Schedule

The state space of the executable model described above contains all valid schedules as a final state. The state space is finite, due to the limitation on the number

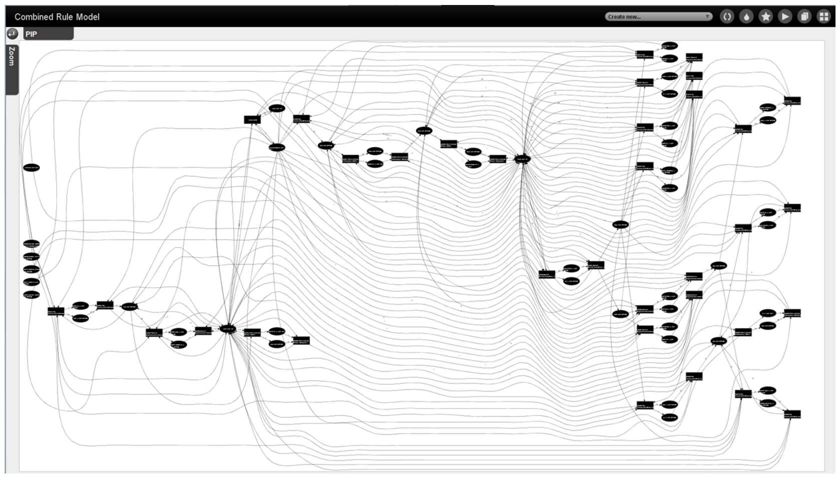


Fig. 4: An automatically constructed executable model in ProM.

of activities that need to be scheduled as well as the bound on the available time to schedule them in. Therefore, one way to find the optimal schedule is to use existing state space exploration techniques to find all final states or deadlocks [9].

Due to the explicit tracking of KPIs and time in each state of the executable model, both constructing a schedule and finding the optimal schedule are straightforward. Given a set of final states, the optimal schedule is the one with the best KPI score or the best tradeoff between KPIs, shown to the user e.g. by creating a Pareto front. Constructing the schedule for a state means traversing the explored state space back to the initial state and recording which activities were executed at what time.

4 Evaluation

We have performed a limited experimental evaluation for the implementation of our approach. A stress monitoring scenario was created, where two employees work at two possible locations on a simple hardware maintenance process. A model explaining the effect of executing activities on the stress of the employees was assumed to be known. Our implementation automatically combined the business rules of the scenario, modelled as CPNs, and scheduled a number of activities, searching for a good tradeoff between stress levels and working time. Two versions of the scenario were tested, one where activity duration did not depend on the stress level and one where stress levels affected performance.

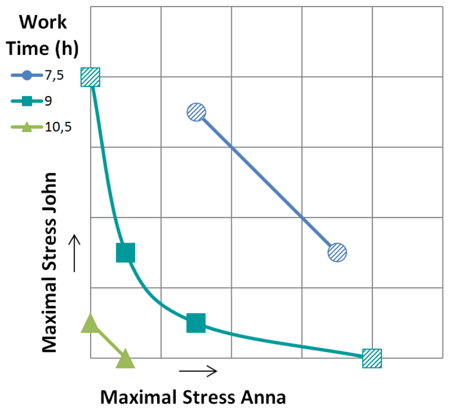
The experimental results are shown in Fig. 5. It is clear from Fig. 5a and Fig. 5b that the number of possible schedules increases exponentially with an increasing number of activities to plan. This also means that the time needed to explore the state space increases exponentially and it quickly reaches a point where the optimal schedule cannot be found within reasonable time. However, it should be noted that the current state space exploration implementation is

Nr. of cases (activities)	Shortest work time	Nr. of valid schedules	Computation time
1 (3)	3 hours	18	0.5 min.
2 (6)	6 hours	160	12.5 min.
3 (9)	7.5 hours	1001	8 hours
4 (12)	?	?	$\gg$24 hours

(a) Planning results with uniform activity durations. The shortest work time is the smallest possible workday length in which all activities can be executed.

Nr. of cases (activities)	Shortest work time	Nr. of valid schedules	Computation time
1 (3)	3 hours	18	0.5 min.
2 (6)	6 hours	130	12 min.
3 (9)	9 hours	396	7 hours
4 (12)	?	?	$\gg$24 hours

(b) Planning results when the effects of stress on performance are considered.



(c) The Pareto front of optimal schedules with 3 cases. The shaded solutions are only valid with uniform activity durations.

Fig. 5: The results of the experimental evaluation.

not very efficient. Another observation is that considering the effects of stress on performance imposes additional restrictions on the solutions, so the state space is smaller and the number of valid schedules is reduced. This also means that the computational complexity of finding the optimal schedule is lower.

Fig. 5c contains a Pareto front of optimal schedules when planning 3 cases. The Pareto front shows that a tradeoff can be made in terms of dividing the work over the available people, as well as the available time. Performing the required work in less time causes higher stress levels. The shaded points indicate schedules that are infeasible if the effect of stress on performance is considered.

5 Conclusion

In this paper we have described an activity planning approach that can find an optimal schedule with respect to one or more KPIs. The approach takes a task list, a set of rules or constraints, and a KPI calculation model to create an executable model that can generate schedules. The state space of this executable model is explored to find an optimal schedule.

Unfortunately, evaluation has shown that the current implementation is not suitable for practical purposes. The implementation allows for a lot of freedom in the types of constraints that can be specified and it can automatically construct an executable model out of these constraints. However, exploring the state space of the resulting executable model is very expensive. Due to the large number of possible ways to schedule activities, the state space becomes too big to explore. There are still multiple areas of future work that can make the suggested approach suitable for practical purposes.

One direction of future work is the use of heuristics during the state space exploration. As the stress of people is highly variable and dependent on many factors, it is difficult to model and predict. Searching for an optimal schedule on a flawed stress model will probably not result in an optimal schedule in practice, so a focus on finding good instead of optimal schedules may be more suitable. The use of state space reduction techniques could also speed up the exploration.

Another direction of future work is the use of a different method to find the optimal schedule. The executable model represents the space of valid schedules, and it might be more efficient to translate this model to a different formalism, e.g. a constraint program. While defining a constraint program directly might be error-prone and challenging, the translation is possible, and would enable the use of many optimisation techniques that exist in constraint programming.

Additionally, more research is needed on learning personalised models that predict people's stress in practical situations and how that affects performance.

References

1. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer (2011)
2. Deb, K.: Multi-objective optimization using evolutionary algorithms, vol. 16. John Wiley & Sons (2001)
3. EU-OSHA: Calculating the cost of work-related stress and psychosocial risks (2014), `https://osha.europa.eu/en/publications/literature_reviews/calculating-the-cost-of-work-related-stress-and-psychosocial-risks`
4. Hartmann, S., Briskorn, D.: A survey of variants and extensions of the resource-constrained project scheduling problem. European Journal of Operational Research 207(1), 1–14 (2010)
5. Jamal, M.: Job stress and job performance controversy: An empirical assessment. Organizational behavior and human performance 33(1), 1–21 (1984)
6. Jensen, K.: Coloured petri nets. In: Petri nets: central models and their properties, pp. 248–299. Springer (1987)
7. Jiang, J., Aldewereld, H., Dignum, V., Tan, Y.H.: Compliance checking of organizational interactions. ACM Transactions on Management Information Systems (TMIS) 5(4), 23 (2014)
8. Kocielnik, R., Sidorova, N., Maggi, F.M., Ouwerkerk, M., Westerink, J.H.D.M.: Smart technologies for long-term stress monitoring at work. In: Computer-Based Medical Systems (CBMS), 2013 IEEE 26th International Symposium on. pp. 53–58. IEEE (2013)
9. Lee, D.Y., DiCesare, F.: Scheduling flexible manufacturing systems using petri nets and heuristic search. Robotics and Automation, IEEE Transactions on 10(2), 123–132 (1994)
10. Ramezani, E., Fahland, D., van der Aalst, W.M.P.: Where did I misbehave? Diagnostic information in compliance checking. In: Business Process Management, pp. 262–278. Springer (2012)
11. Verbeek, H.M.W., Buijs, J.C.A.M., Van Dongen, B.F., van der Aalst, W.M.P.: XES, XESame, and ProM 6. In: Information Systems Evolution, pp. 60–75. Springer (2011)
12. Winskel, G.: Petri nets, morphisms and compositionality. In: Rozenberg, G. (ed.) Advances in Petri Nets 1985, Lecture Notes in Computer Science, vol. 222, pp. 453–477. Springer Berlin Heidelberg (1986)