1 Summary

I am a third year PhD fellow at DTU Construct and DTU Management and I am currently involved in a research project in collaboration with Total Energies. The focus of my thesis is modelling their scheduling process in a way that will allow for optimizing their maintenance operations by informing key decision makers in real time. While I feel the product could be of great utility, I face a number difficult problems which I have outlined below. I believe that an external stay at Decision Brain will provide the expertise and guidance needed to making the project succeed or informing the future direction of my career path. Below is a list of my main issues so far that I have struggled to solve by myself.

My Main Issues:

- Developing an API without a frontend to get user feedback, especially one the dynamic components of the API.
  - Stakeholders struggle to grasp the concept without a visual representation, making it challenging to communicate the API's functionality. Developing both is time consuming.
- I am lacking essential skills such as effecitive stakeholder management and UX development.
  - Working with Decision Brain I hope to learn how to be effective at what I do best and understand what it takes to rely on people that have the essential skills do not.
- Uncertainty about the project's financial viability, step towards practical application, and how to manage a software project such as this one.
  - I think that the team at Decision Brain will be able to provide guidance here.

Incentives for Decision Brain:

- Gain a fresh perspective on developing scheduling solutions.
- Opportunity to evaluate and potentially recruit a skilled researcher.
- Access to a partial implementation of innovative scheduling code.
- Acquire personal contacts within Total Energies that are responsible for managing and developing maintenance scheduling solutions internally e.g. Total Digital Factory.

I believe an external stay at Decision Brain could greatly improve my personal skills and the future financial value of the project. I suspect my expertise in designing, scoping, and building scheduling systems will contribute to Decision Brain's knowledge base as well.

2 Project Introduction

The project aims to make a real-time scheduling system that is able to **optimize** and **coordinate** state across different decision making actors. This is done by modelling stakeholders decisions individually and then implementing novel coordination mechanisms between the models. To solve this problem I have developed an API that runs on Total Energies production servers so that each relevant stakeholder is able to interact with the part of the schedule that he is responsible for through dedicated API endpoints. Each of these API endpoints (not fully implemented) have a corresponding simple frontend which includes upload functions for data not available in the Total Energies IT systems, simple live statistics of the current state of the scheduling application such as resource consumption, and .xlsx export functions.).

The main rationale for this approach is that you can only optimize schedules to a certain extend before you need to coordinate between stakeholders. This is usually due to practical things such as, safety, regulation, human factors, handling errors due to low quality data. Solving this requires a diverse range of skills of which I only posses a few. It is my hope that Decision Brain would be interested in a collaboration that would prove mutually beneficial for both parties.

3 Goals of the External Stay

The most significant goals of the external stay are:

- Assess the feasibility of implementing my Ph.D. project in an industry setting.
- Integrate my application into a test environment at Decision Brain, if applicable.
- Get expert feedback on my scheduling methodology.
- Learn best practices for implementing scheduling solutions in industry.

It is my belief that I have developed a scalable approach to modelling a generic maintenance scheduling system (see section 6). I am modelling something that is similar to what is described in [Palmer, 2019] which is a source that has a more practical orientation than most academic works.

My code operates on backend SAP tables and user inputs so I believe that there may be a possibility of integrating my code into a system at Decision Brain if this is deemed valuable. Integrating the system at Decision Brain would allow us to evaluate its potential financial value and determine whether pursuing a full implementation could be mutually beneficial.

4 Possible Setup of the External Stay

I propose to have a dedicated contact at Decision Brain during my external stay, with whom I can discuss ideas and seek help from.

- First month: Determine if I can integrate my application in a relevant project at Decision Brain.
- Second month: Work on implementing the scheduling system in Decision Brain with weekly or biweekly feedback.
- Third month: Assess the strength of the project and future collaboration directions with Decision Brain.

The main incentive for Decision Brain to partake in this project would be to get a new perspective on how to develop scheduling solutions, a hiring opportunity, getting a partial implementation of scheduling code if that is deemed relevant.

5 Personal

I am an introverted person who prefers few but close partnerships. I would say my greatest strengths are:

- Care intensly about the quality of the products that I am working on. I will call people out for bad technical decision making and hold people accountable for project decisions.
- Maticulate about understanding the problem and afterwards building the best solution.
- Creating reliable and automatic systems, whether for coding applications, scripting workflows, creating development setups (Linux and Nix), drafting documents (LaTeX or Typst), etc.

I would say my greatest weaknesses are:

- Communication skills, I usually benefit significantly from a proactive and competent teamleader.
- Project management and prioritization is part of my job where I have consistently demonstrated that I have a blindspot.

My hobbies include running (25-30 km per week), reading non-fiction on various topics not directly related to work; and engaging in creative coding projects with friends.

Looking ahead, I aspire to advance my career by developing maintenance scheduling systems either by joining a company like Decision Brain or founding my own company based on the knowledge that I have gained through the Ph.D. program.

Roadmap: Technical Parts

- ☑ Model the Scheduler stakeholder
- ☑ Model the Supervisor stakeholder
- ☑ Model the Technician stakeholder
- ☑ Determine a software architecture
- ☑ Host the API on Total Energies servers
- ☑ Read data from SAP
- ☐ Write data directly to SAP
- ☑ Test output with scheduler stakeholder
- ☐ Test output with supervisor stakeholder
- ☐ Test output with technician stakeholder

Roadmap: Project Parts

- ☑ Study the relevant (practical) literature
- ☑ Interview relevant stakeholders
- ☑ Show prototype concepts to relevant stakeholders
- ☑ Gain support from Total Esbjerg Maintenance Methods
- ☑ Gain support from Subcontractors
- ☑ Gain support from Total Digital Factory
- ☐ External Stay
- ☐ Deliver a functioning scheduling API (difficult)
- ☐ Hand in Ph.D. thesis

6 Technical

This section provides a high-level overview of my current application's architecture and methodologies. It highlights areas where collaboration with Decision Brain can enhance the project's success and offers insights that may benefit Decision Brain's own practices.

This section assumes that the reader is familiar with:

- Fundamentals of software architecture
- Metaheuristic approaches to optimization
- Expertise in programming and application development

Architecture of the Scheduling System

I have spent a significant part of my Ph.D. program refining and testing different architectures to enable metaheuristics to coordinate state in real-time. The applications working title is Ordinator and the latest version of the architecture is shown in figure 1.

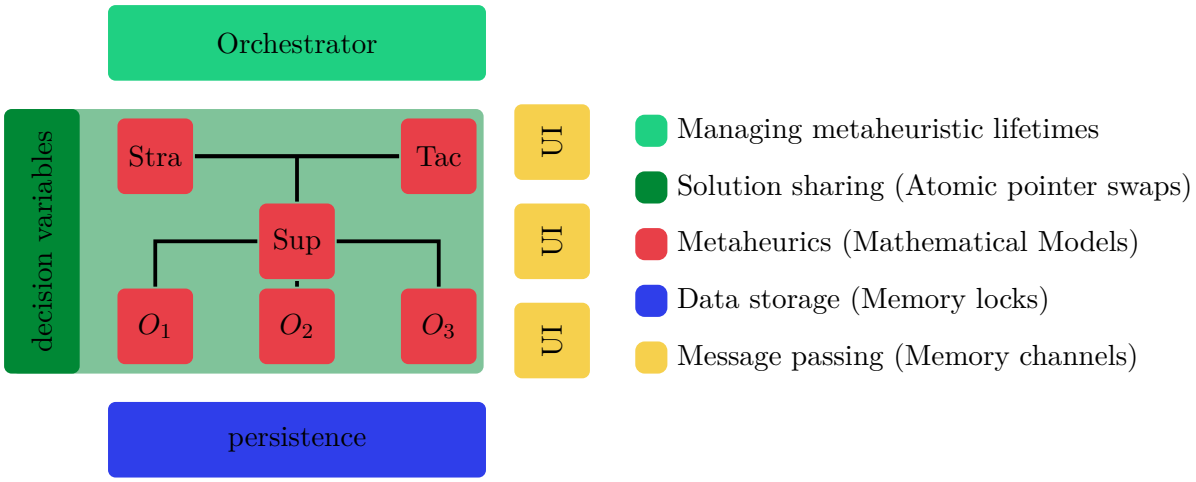

Figure 1: High level architecture of the scheduling system. Persistence stores all data, whether from SAP, user input, or other systems; the Orchestrator manages the lifetimes of the metaheuristics; the metaheuristics does the actual optimization; the decision variable are all stored together and are shared among all optimizing metaheuristics, each algorithm can write to its own state but only read the state of its neighbors; the UI components each communicate with the algorithms that correspond to the individual stakeholder, meaning that the models also specify what each stakeholder is allowed to modify which is crucial for businesses. [Barthélemy et al., 2002]

Key Lessons:

- Message passing between metaheuristics are unworkable, e.g. a microservice architecture is a difficult approach. A usable scheduling system needs to be implemented on a single CPU with multi-threading, "normal" best practice for horizontal scaling is difficult as state changes quickly in metaheuristics.
- Optimization problems are difficult due to large and complex solution spaces. Allowing models/metaheuristics to use each others solutions as parameters allows you to keep solution spaces smaller while preserving the ability to model the larger system (sacrificing global optimization which is usually meaningless in practice).

- The operational setting is more complex than it appears and changes faster than you think. Developing large integrated models is difficult as model changes become both more difficult and frequent the larger the model gets. This tends to make such scheduling systems fragile.

The key feature this architecture enables is that we can move away from hierarchical approaches and instead model each stakeholder individually with the responsibilities that exactly that person is responsible for. In figure 2 we see that the system can initialize many instances of each metaheuristic, each corresponding to a stakeholder.

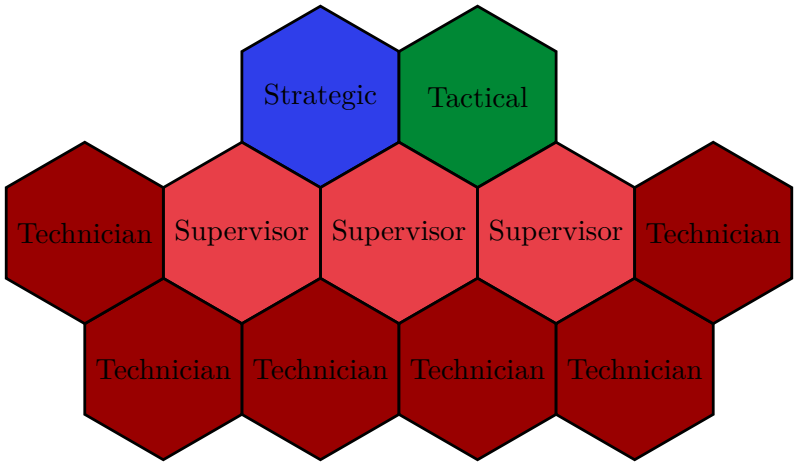

Figure 2: Each metaheuristic (Actor-based Large Neighborhood Search, see algorithm 1 on page 7) is here shown as a hexagon. Each metaheuristic is based on a mathematical model each of which are found in the appendix. **Notice**: this system is not hierarchical, each metaheuristic reads the solutions of the other metaheuristic but they are not dependent on them for their function. Note here that there are two different models for the scheduler, tactical and strategic, where the weekly and daily scheduling are treated separately.

Key Lessons:

- Modelling each responsible decision maker with its own model makes stakeholder integration easier.
- Extending a smaller model/metaheuristic is easier than extending a model that goes across multiple decision-making stakeholders.
- Model setup have both horizontal and vertical scaling, within the limits of a single CPU often a neglected aspect in metaheuristics.
- Maintenance scheduling is 70% coordination and 30% optimization.

Pertually Running Optimization

One of the core principles found doing my interviews with Total Energies employees is that very complex model constraints should be modeled reactively, instead of being encoded into static constraints. There are so many constraints in the real world that encoding them into a model is a lost cause. So the approach taken here is different. Instead of modelling every detail that is needed to make the output of each metaheuristic useful you instead model the basic constraints and then let the stakeholder himself adjust the solution (in Operation Research called ”interactive operation research; in the metaheuristic literature called ”human-guided search”; and in operation management called ”Human-in-the-loop”). In figure 3 I have tried to show the issues that seems to arise when you in practice try to implement operation research approaches in maintenance scheduling. [Meignan et al., 2015]

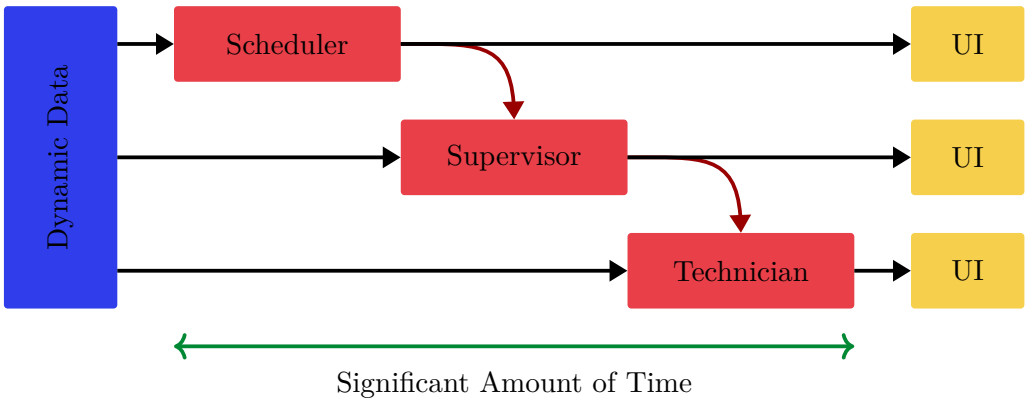

Figure 3: Illustrates the bottlenecks and communication challenges inherent in hierarchical models, where lower-level processes depend heavily on higher-level outputs.

This project takes the approach shown in figure 4. Instead of running an optimization algorithm once and then providing a stakeholder with a single solution, each algorithm runs in perpetuity always optimizing against the latest available information. This means that each algorithm will be able to optimize based on the solutions that the other metaheuristics finds. Also, through UI components stakeholder can interact with the optimization process that corresponds to his part of the larger maintenance scheduling process.

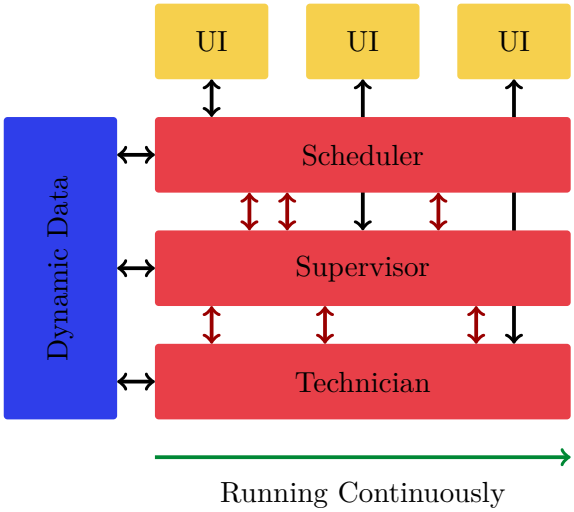

Figure 4: Having metaheuristics continually running and coordinating state you can make a setup that is more robust by dynamically accepting inputs from the relevant stakeholder.

Key Lessons:

- Hierarchical approaches are problematic in practice, as the knowledge and information required for a high-quality and functioning maintenance scheduling process are usually found ”lower in the hierarchy” rather than at higher levels, as managers sometimes implicitly believe.
- Having metaheuristics continuously running computational overhead as you only need to reach initial convergence once. Making the user experience more responsive and consecutive solutions will look more similar. [Alba et al., 2013] [Yang, 2015]
- Designing continuously running metaheuristics means that you have to think carefully about software architecture. Everything has to be scalable, responsive, and dynamic.

7 Appendix

This appendix contains the pseudo-code and mathematical models associated with each metaheuristic implementation.

The mathematical models here are only guiding, as everything is implemented as metaheuristics a one-to-one mapping between the mathematical models and the actual implementation can lose some nuances.

Pseudo Code

All implemented algorithms are based on a variant of the Large Neighborhood Search metaheuristic with some modification to enable both message passing and solution state shared through atomic pointer swapping.

Algorithm 1 Actor-based Large Neighborhood Search

Input Q = message queue
Input P = problem instance
Input X = initial schedule
Input S = shared solution
repeat
    $X^t = clone(X)$
    while $Q.has\_message()$ **do**
        $P.update(S, m)$
        $X^t.destruct(S, m)$
    end while
    $X^t.repair(S)$
    if accept(X^t, X) **then**
        $X.update(X^t)$
    end if
    if $c(X^t) < c(X)$ **then**
        $X.update(X^t)$
        $S.atomic\_pointer\_swap(X)$
    end if
    $Q.push(m)$
until

I have decided to include the mathematical models of the four different metaheuristics. It is not meant to provide a through understanding but show how the larger process can be modelled through a series of smaller models. **Notice:** the "meta variables" sections are the decision variables (and time τ) from other models that are being used in the respective model as a parameters (meaning they are not variables in that specific model and that model cannot then change the value of the variable).

Strategic Model: A Knapsack Variant

Meta variables:

$$s \in S \tag{1}$$

$$\beta(\tau) \tag{2}$$

$$\tau \in [0, \infty] \tag{3}$$

Maximize:

$$\sum_{w \in W(\tau)} \sum_{p \in P(\tau)} strategic\_value_{wp}(\tau) \cdot \alpha_{wp}(\tau)$$

$$- \sum_{p \in P(\tau)} \sum_{r \in R(\tau)} strategic\_penalty \cdot \epsilon_{pr}(\tau)$$

$$+ \sum_{p \in P(\tau)} \sum_{w1 \in W(\tau)} \sum_{w2 \in W(\tau)} clustering\_value_{w1,w2} \cdot \alpha_{w1p}(\tau) \cdot \alpha_{w2p}(\tau) \tag{4}$$

Subject to:

$$\sum_{w \in W(\tau)} work\_order\_work_{wr} \cdot \alpha_{wp}(\tau) \leq \ resource_{pr}(\tau, \beta(\tau)) + \epsilon_{pr}(\tau) \quad \forall p \in P(\tau) \quad \forall r \in R(\tau) \tag{5}$$

$$\sum_{w \in W(\tau)} \alpha_{wp}(\tau) = 1 \quad \forall p \in P(\tau) \tag{6}$$

$$\alpha_{wp}(\tau) = 0 \quad \forall (w, p) \in exclude(\tau) \tag{7}$$

$$\alpha_{wp}(\tau) = 1 \quad \forall (w, p) \in include(\tau) \tag{8}$$

$$\alpha_{wp}(\tau) \in \{0, 1\} \quad \forall w \in W(\tau) \quad \forall p \in P(\tau) \tag{9}$$

$$\epsilon_{pr}(\tau) \in \mathbb{R}^{+} \quad \forall p \in P(\tau) \quad \forall r \in R(\tau) \tag{10}$$

Tactical Model: A Resource Constrained Project Scheduling Problem Variant

---

Meta variables:

$$s \in S \tag{11}$$

$$\alpha(\tau) \tag{12}$$

$$\tau \in [0, \infty] \tag{13}$$

Minimize:

$$\sum_{o \in O(\tau,\alpha(\tau))} \sum_{d \in D(\tau)} tactical\_value_{do}(\tau) \cdot \beta_{do}(\tau) + \sum_{r \in R(\tau)} \sum_{d \in D(\tau)} tactical\_penalty \cdot \mu_{rd}(\tau) \tag{14}$$

Subject to:

$$\sum_{o \in O(\tau,\alpha(\tau))} work_o(\tau) \cdot \beta_{do}(\tau) \leq tactical\_resource_{dr}(\tau) + \mu_{rd}(\tau) \forall d \in D(\tau) \quad \forall r \in R(\tau) \tag{15}$$

$$\sum_{d=earliest\_start_o(\tau)}^{latest\_finish_o(\tau)} \sigma_{do}(\tau) = duration_o(\tau) \quad \forall o \in O(\tau, \alpha(\tau)) \tag{16}$$

$$\sum_{d^* \in D_{duration_o(\tau)}(\tau)} \sigma_{d^* o}(\tau) = duration_o(\tau) \cdot \eta_{do}(\tau) \quad \forall o \in O(\tau, \alpha(\tau)) \quad \forall d \in D(\tau) \tag{17}$$

$$\sum_{o \in O(\tau,\alpha(\tau))} \eta_{do}(\tau) = 1, \quad \forall d \in D(\tau)$$

$$\sum_{d \in D(\tau)} d \cdot \sigma_{do1}(\tau) + \Delta_o(\tau) = \sum_{d \in D(\tau)} d \cdot \sigma_{do2}(\tau) \quad \forall (o1, o2) \in finish\_start_{o1,o2} \tag{18}$$

$$\sum_{d \in D(\tau)} d \cdot \sigma_{do1}(\tau) = \sum_{d \in D(\tau)} d \cdot \sigma_{do2}(\tau) \quad \forall (o1, o2) \in start\_start_{o1,o2} \tag{19}$$

$$\beta_{do}(\tau) \leq number_o(\tau) \cdot operating\_time_o \quad \forall d \in D(\tau) \quad \forall o \in O(\tau, \alpha(\tau)) \tag{20}$$

$$\beta_{do}(\tau) \in \mathbb{R} \qquad \forall d \in D(\tau) \quad \forall o \in O(\tau, \alpha(\tau)) \tag{21}$$

$$\mu_{rd}(\tau) \in \mathbb{R} \qquad \forall r \in R(\tau) \quad \forall d \in D(\tau) \tag{22}$$

$$\sigma_{do}(\tau) \in \{0, 1\} \qquad \forall d \in D(\tau) \quad \forall o \in O(\tau, \alpha(\tau)) \tag{23}$$

$$\eta_{do}(\tau) \in \{0, 1\} \qquad \forall d \in D(\tau) \quad \forall o \in O(\tau, \alpha(\tau)) \tag{24}$$

$$\Delta_o(\tau) \in \{0, 1\} \quad \forall o \in O(\tau, \alpha(\tau)) \tag{25}$$

Supervisor Model: An Assignment Problem Variant

Meta variables:

$$z \in Z \tag{26}$$

$$\alpha(\tau) \tag{27}$$

$$\theta(\tau) \tag{28}$$

$$\tau \in [0, \infty] \tag{29}$$

Maximize:

$$\sum_{a \in A(\tau, \alpha(\tau))} \sum_{t \in T(\tau)} supervisor\_value_{at}(\tau, \lambda_t(\tau), \Lambda_t(\tau)) \cdot \gamma_{at}(\tau) \tag{30}$$

Subject to:

$$\sum_{a \in A_o(\tau, \alpha(\tau))} \rho_a(\tau) = work_o(\tau) \quad \forall o \in O(\tau, \alpha(\tau)) \tag{31}$$

$$\sum_{t \in T(\tau)} \sum_{a \in A_o(\tau, \alpha(\tau))} \gamma_{at}(\tau) = \phi_o(\tau) \cdot number_o(\tau) \quad \forall o \in O(\tau, \alpha(\tau)) \tag{32}$$

$$\sum_{o \in O_w(\tau, \alpha(\tau))} \phi_o(\tau) = |O_w(\tau, \alpha(\tau))| \quad \forall w \in W(\tau, \alpha(\tau)) \tag{33}$$

$$\sum_{a \in A_o(\tau, \alpha(\tau))} \gamma_{at}(\tau) \leq 1 \quad \forall o \in O(\tau, \alpha(\tau)) \quad \forall t \in T(\tau) \tag{34}$$

$$\gamma_{at}(\tau) \leq feasible_{at}(\theta(\tau)) \quad \forall o \in O(\tau, \alpha(\tau)) \quad \forall t \in T(\tau) \tag{35}$$

$$\gamma_{at}(\tau) \in \{0, 1\} \quad \forall o \in O(\tau, \alpha(\tau)) \quad \forall t \in T(\tau) \tag{36}$$

$$\rho_a(\tau) \in [lower\_activity\_work_a(\tau), work_a(\tau)] \quad \forall a \in A(\tau, \alpha(\tau)) \tag{37}$$

Technician Model: Single Machine Scheduling Problem Variant

Meta variables:

$$t \in T(\tau) \tag{38}$$
$$\alpha(\tau) \tag{39}$$
$$\gamma(\tau) \tag{40}$$
$$\tau \in [0, \infty] \tag{41}$$

Maximize:

$$\sum_{a \in A(\tau, \gamma_t(\tau))} \sum_{k \in K(\gamma(\tau))} \delta_{ak}(\tau) \tag{42}$$

Subject to:

$$\sum_{k \in K(\gamma(\tau))} \delta_{ak}(\tau) \cdot \pi_{ak}(\tau) = activity\_work_a(\tau, \rho(\tau)) \cdot \theta \quad (\tau) \forall a \in A(\tau, \gamma_t(\tau)) \tag{43}$$
$$\lambda_{a21}(\tau) \geq \Lambda_{a1last(a1)}(\tau) + preparation_{a1,a2} \quad \forall a1 \in A(\tau, \gamma_t(\tau)) \quad \forall a2 \in A(\tau, \gamma_t(\tau)) \tag{44}$$
$$\lambda_{ak}(\tau) \geq \Lambda_{ak-1}(\tau) - constraint\_limit \cdot (2 - \pi_{ak}(\tau) + \pi_{ak-1}(\tau)) \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \tag{45}$$
$$\delta_{ak}(\tau) = \Lambda_{ak}(\tau) - \lambda_{ak}(\tau) \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \tag{46}$$
$$\lambda_{ak}(\tau) \geq event_{ie} + duration_{ie} - constraint\_limit \cdot (1 - \omega_{akie}(\tau)) \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \quad \forall i \in I(\tau) \quad \forall e \in E(\tau) \tag{47}$$
$$\Lambda_{ak}(\tau) \leq event_{ie} + constraint\_limit \cdot \omega_{akie}(\tau) \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \quad \forall i \in I(\tau) \quad \forall e \in E(\tau) \tag{48}$$
$$\lambda_{a1}(\tau) \geq time\_window\_start_a(\beta(\tau)) \quad \forall a \in A(\tau, \gamma_t(\tau)) \tag{49}$$
$$\Lambda_{alast(a)}(\tau) \leq time\_window\_finish_a(\beta(\tau)) \quad \forall a \in A(\tau, \gamma_t(\tau)) \tag{50}$$
$$\pi_{ak}(\tau) \in \{0, 1\} \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \tag{51}$$
$$\lambda_{ak}(\tau) \in [availability\_start(\tau), availability\_finish(\tau)] \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \tag{52}$$
$$\Lambda_{ak}(\tau) \in [availability\_start(\tau), availability\_finish(\tau)] \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \tag{53}$$
$$\delta_{ak}(\tau) \in [0, work_{a\_to\_o(a)}(\tau)] \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \tag{54}$$
$$\omega_{akie}(\tau) \in \{0, 1\} \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \quad \forall i \in I(\tau) \quad \forall e \in E(\tau) \tag{55}$$
$$\theta_a(\tau) \in \{0, 1\} \quad \forall a \in A(\tau, \gamma_t(\tau)) \tag{56}$$

References

Enrique Alba, Amir Nakib, and Patrick Siarry, editors. *Metaheuristics for Dynamic Optimization*, volume 433 of *Studies in Computational Intelligence*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. ISBN 978-3-642-30664-8 978-3-642-30665-5. doi: 10.1007/978-3-642-30665-5.

J.P Barthélemy, R Bisdorff, and G Coppin. Human centered processes and decision support systems. *European Journal of Operational Research*, 136(2):233–252, January 2002. ISSN 03772217. doi: 10.1016/S0377-2217(01)00112-6.

David Meignan, Sigrid Knust, Jean-Marc Frayret, Gilles Pesant, and Nicolas Gaud. A Review and Taxonomy of Interactive Optimization Methods in Operations Research. *ACM Transactions on Interactive Intelligent Systems*, 5(3):1–43, October 2015. ISSN 2160-6455, 2160-6463. doi: 10.1145/2808234. URL `https://dl.acm.org/doi/10.1145/2808234`.

Richard D. Palmer. *Maintenance Planning and Scheduling Handbook, 4th Edition* . McGraw Hill, 4th edition edition, September 2019.

Shengxiang Yang. Evolutionary Computation for Dynamic Optimization Problems. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, GECCO Companion '15, pages 629–649, New York, NY, USA, July 2015. Association for Computing Machinery. ISBN 978-1-4503-3488-4. doi: 10.1145/2739482.2756589.