1 Summary

I am doing a research project in collaboration with Total Energies. I am currently modelling their scheduling process in a way that will allow us to optimize their maintenance operations. My main concern in this process is making the project succeed with the numerous decisions that I have to make but which I feel unequipped for. 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 a correct 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.
- Many topics are outside the scope of my research project due to a lack of the essential skills.
  - Primarily stakeholder management and UX development.
- Uncertainty about the project's financial viability and whether to continue its development within an academic setting.
  - 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.

2 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 how developed a scalable approach to modelling a generic maintenance scheduling system (see section 5). I am modelling something that is similar to what is described in **?** 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.

3 Possible Setup of the External Stay

I propose to have a dedicated contact at Decision Brain during my external stay, with who 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 feasibility of the project and possible course corrections.

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.

4 Personal

I am an introverted person who prefers few but close partnerships. To perform at a high level I benefit immensely from having the right colleagues who provide support and complement the qualities I may lack. I focus on designing systems on a higher level, often feeling worried if I do not believe that I or my team is working on tasks that are not aligned with our final goals. This is good for inspiring and setting direction, but it is clear to me that people with other qualities are also needed to make a project work and be successful.

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

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

Roadmap: Project Parts

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

5 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 meta-heuristics to coordinator state in real-time. The latest version is shown in figure 1.

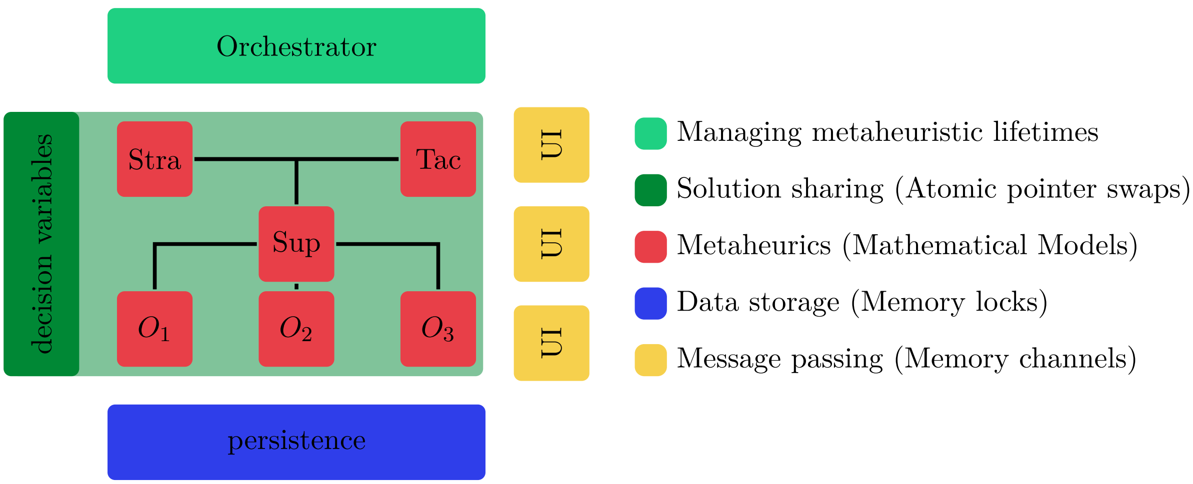

Figure 1: High level architecture of the scheduling system. Persistence stores all data, whether from SAP, user input, or other systems; the Orchestrator manages the lifetimes of the metaheuristics; the metaheuristics does the actual optimization; the decision variable are all stored together and are shared among all optimizing meta-heuristics, 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.

Key Lessons:

- Message passing between metaheuristics are unworkable, e.g. a microservice architecture is 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/meta-heuristics 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 that this architecture enable 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.

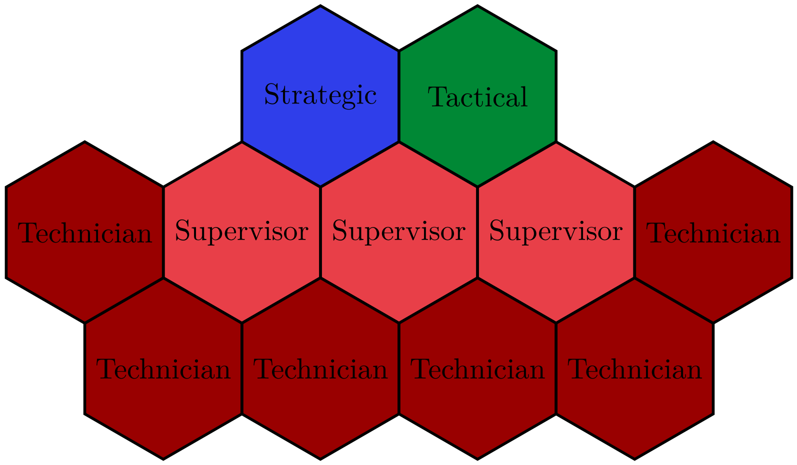

Figure 2: Each meta-heuristic (Actor-based Large Neighborhood Search, see algorithm 1 on page 6) is here shown as a hexagon. Each meta-heuristic is based on a mathematical model each of which are found in the appendix. **Notice**: this system is not hierarchical, each metaheuristic reads the solutions 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 with the weekly and daily scheduling are treated differently.

Key Lessons:

- Modelling each responsible decision maker with its own model makes stakeholder integration easier.
- Extending a smaller model/meta-heuristic 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 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.

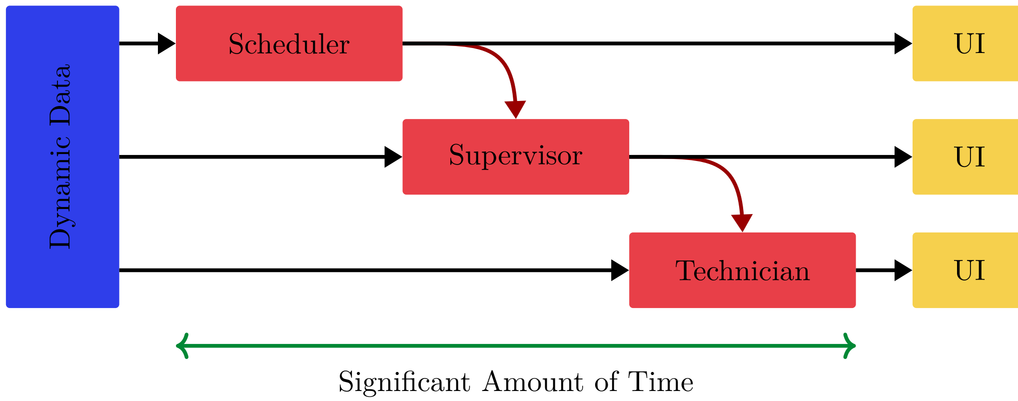

Figure 3: Illustrates the bottlenecks and communication challenges inherent in hierarchical models, where lower-level processes depend heavily on higher-level outputs.

This project takes the approach shown in figure **??**. 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 meta-heuristics finds. Also, through UI components stakeholder can interact with the optimization process that corresponds to his part of the larger maintenance scheduling process.

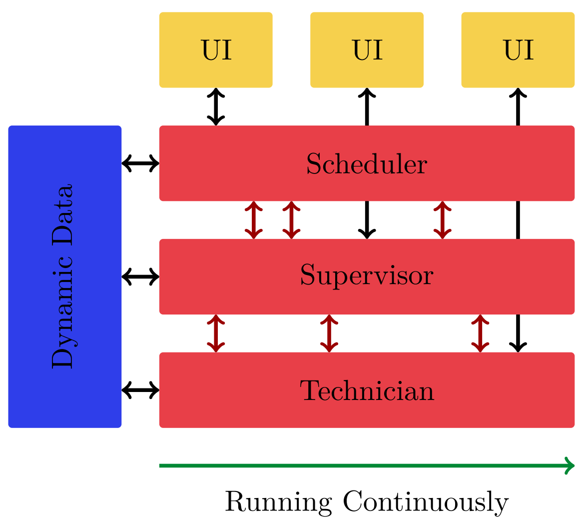

Figure 4: Having metaheuristics continually running and coordinating state you can make a setup that is more robust by dynamically accepting inputs from the relevant stakeholder.

Key Lessons:

- Hierarchical appraochs 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.
- Designing continuously running metaheuristics means that you have to think carefully about software architecture. Everything has to be scalable, responsive, and dynamic.

6 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:

$$\begin{aligned}
&\sum_{w \in W(\tau)} \sum_{p \in P(\tau)} strategic_value_{wp}(\tau) \cdot \alpha_{wp}(\tau) \\
&- \sum_{p \in P(\tau)} \sum_{r \in R(\tau)} strategic_penalty \cdot \epsilon_{pr}(\tau) \\
&+ \sum_{p \in P(\tau)} \sum_{w1 \in W(\tau)} \sum_{w2 \in W(\tau)} clustering_value_{w1,w2} \cdot \alpha_{w1p}(\tau) \cdot \alpha_{w2p}(\tau)
\end{aligned} \tag{4}$$

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:

$$\sum_{k \in K(\gamma(\tau))} \delta_{ak}(\tau) \cdot \pi_{ak}(\tau) = activity\_work_a(\tau, \rho(\tau)) \cdot \theta \quad (\tau) \forall a \in A(\tau, \gamma_t(\tau)) \tag{43}$$

$$\lambda_{a21}(\tau) \geq \Lambda_{a1last(a1)}(\tau) + preparation_{a1,a2} \quad \forall a1 \in A(\tau, \gamma_t(\tau)) \quad \forall a2 \in A(\tau, \gamma_t(\tau)) \tag{44}$$

$$\lambda_{ak}(\tau) \geq \Lambda_{ak-1}(\tau) - constraint\_limit \cdot (2 - \pi_{ak}(\tau) + \pi_{ak-1}(\tau)) \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \tag{45}$$

$$\delta_{ak}(\tau) = \Lambda_{ak}(\tau) - \lambda_{ak}(\tau) \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \tag{46}$$

$$\lambda_{ak}(\tau) \geq event_{ie} + duration_{ie} - constraint\_limit \cdot (1 - \omega_{akie}(\tau)) \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \quad \forall i \in I(\tau) \quad \forall e \in E(\tau) \tag{47}$$

$$\Lambda_{ak}(\tau) \leq event_{ie} + constraint\_limit \cdot \omega_{akie}(\tau) \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \quad \forall i \in I(\tau) \quad \forall e \in E(\tau) \tag{48}$$

$$\lambda_{a1}(\tau) \geq time\_window\_start_a(\beta(\tau)) \quad \forall a \in A(\tau, \gamma_t(\tau)) \tag{49}$$

$$\Lambda_{alast(a)}(\tau) \leq time\_window\_finish_a(\beta(\tau)) \quad \forall a \in A(\tau, \gamma_t(\tau)) \tag{50}$$

$$\pi_{ak}(\tau) \in \{0, 1\} \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \tag{51}$$

$$\lambda_{ak}(\tau) \in [availability\_start(\tau), availability\_finish(\tau)] \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \tag{52}$$

$$\Lambda_{ak}(\tau) \in [availability\_start(\tau), availability\_finish(\tau)] \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \tag{53}$$

$$\delta_{ak}(\tau) \in [0, work_{a\_to\_o(a)}(\tau)] \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \tag{54}$$

$$\omega_{akie}(\tau) \in \{0, 1\} \quad \forall a \in A(\tau, \gamma_t(\tau)) \quad \forall k \in K(\gamma(\tau)) \quad \forall i \in I(\tau) \quad \forall e \in E(\tau) \tag{55}$$

$$\theta_a(\tau) \in \{0, 1\} \quad \forall a \in A(\tau, \gamma_t(\tau)) \tag{56}$$
