Summary

I am doing a research project in collaboration with Total Energies. I am in the process of modelling their scheduling process in a way that will allow us to optimize their operations. My main concerns in this process is how to make the project succeed with the numerous decision that I have to make but which I feel poorly equiped to make. My hope is that an external stay at Decision Brain will help me understand what the best appraoch are to making my project succeed. This includes both the Ph.D. and potentially after my graduation. Below is a list of my main issues so far that I have failed to solve by myself.

Main Issues:

- Developing the API without a frontend to get feedback
  - I is hard to get the Stakeholders that are actually doing the work to understand the idea without developing a frontend.
- Many topics are outside the scope of my research project due to a lack of the essential skills.
  - Stakeholder management, and UX development.
- Gauging the financial vaule of the project; whether I should continue implementing or keep it academic.
  - I think that the team at Decision Brain will be able to gauge this in less than a month.

Goals of the External Stay

For me the three most significant goals of the external stay would be:

- Gauge whether my Ph.D. project can be implemented in practice.
- Integrate my application into a test environment at Decision Brain.
- Get competent feedback on my scheduling approach.

I have a somewhat naive belief that I have found a solid scalable approach to modelling a generic maintenance scheduling system (see section below). I am modelling something that is similar to what is described on **?** which is a source that has a more practical orientation than most academic works.

My code work 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 of value. Integrating the system at Decision Brain would be ideal as I think that it would enable us judge the potential financial value of a full implementation of my project. It would also make it clear if it is a project that is worth chasing further by Decision Brain.

Setup of the External Stay

My initial idea of a setup with Decision Brain, would be that I get a stable contact for the duration of the external stay with which I can discuss my ideas and that I can rely on for help.

- 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's IT infrastructure.
- Third month: Assess the feasibility of the project and possible course corrections.

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

Technical

I will provide a high-level overview of what it is that the current application is doing. I believe that this will make it clear how Decision Brain can help to make the project succeed.

Architecture of the Scheduling System

I have spend 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 below.

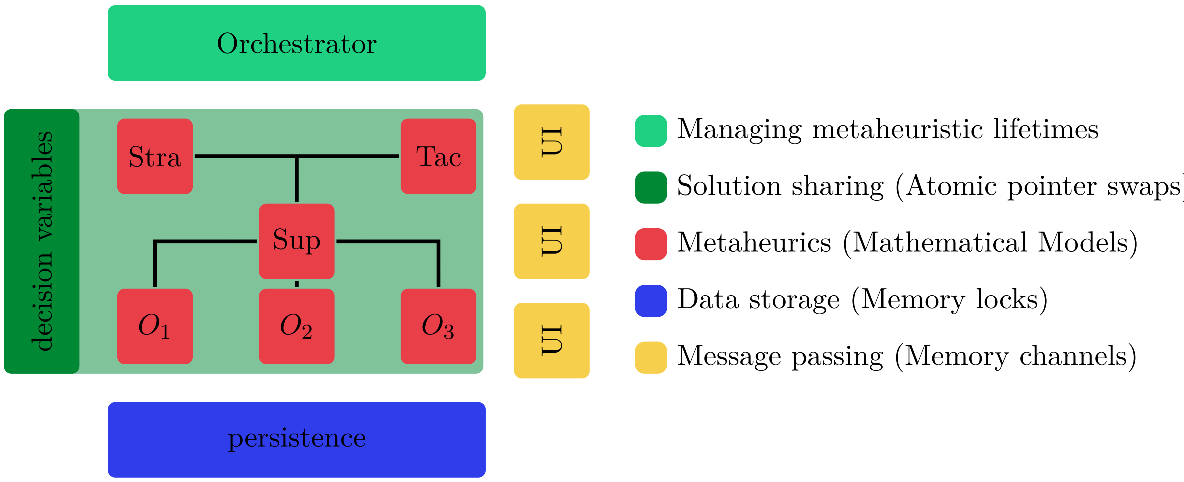

Figure 1: High level overview of the architecture of the scheduling system. Persistance holds all data whether from SAP, user-input, or other systems; the Orchestrator is the part of the system that manages the lifetimes of the metaheuristics that does the actual optimization; the decision variable are all store together are are shared among all optimization 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

Key Lessons:

- Message passing between metaheuristics are unworkable, e.g. a microservice architecture is difficult approach. Usable scheduling system needs to be implemented on a single CPU with multi-threading, "normal" best practice for system horizontal scaling is very difficult.
- 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 system.
- The operational setting is more complex than you think and changes faster than think. Developing large integrated models is a difficult as model changes become more difficult the larger a model gets.

The key feature that this architecture enable is that we can move away from hierarchical approachs and instead model each stakeholder individually with the responsibilities that exactly that person is responsible for.

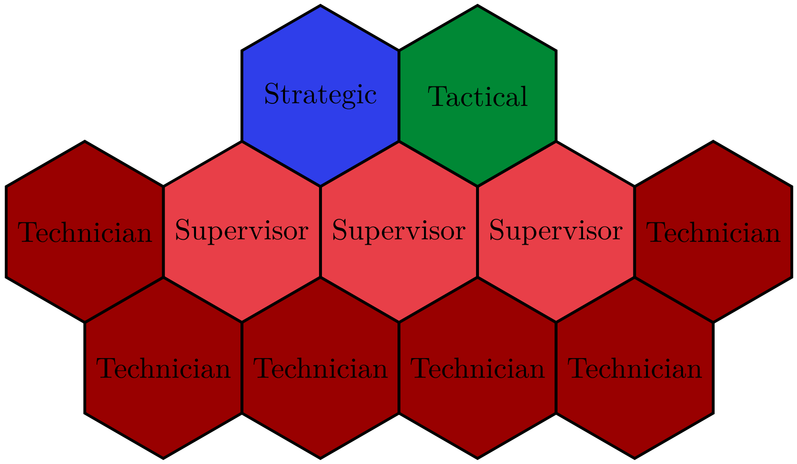

Figure 2: Each meta-heuristic (Actor-based Large Neighborhood Search, see algorithm 1 on page 6) is here shown as a hexagon. Each meta-heuristic is based on a mathematical model each of which are found in the appendix. **Notice**: this system is not hierarchical, each metaheuristic reads the solutions of the other metaheuristic but they are not dependent on them for their function.

Key Lessons:

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

Pertually Running Optimization

One of core principles found doing my interviews is that very complex model constraints should be modeled reactively, instead of being encoded into static constraints. What is meant by this is that there are so many constraints in the real world that encoding them into a model is a lost cause. So the approach that I have 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 reserach; in the metaheuristic literature called ”human-guided search”; and in operation management called ”Human-in-the-loop”). In the figure below I have tried to show the issues that seems to me to often arise when you in practice try to implement operation research approaches in practice.

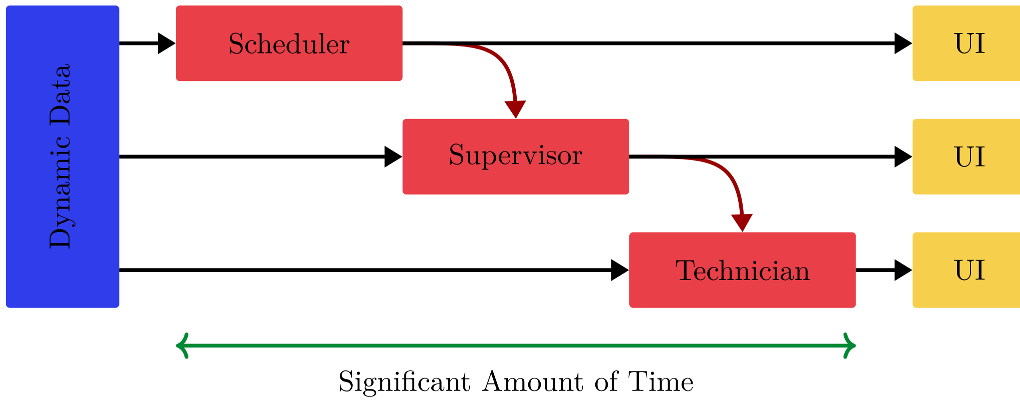

Figure 3: Here we see one of the reasons why hierarchical approaches usually occur when researchers or practitioners try to create a maintenance scheduling sysytem. When you try to model each process as separate entities you will often get a setup where each process is dependent on getting results from the previous model. This works fine going down, but it is simply a horrible solution when information has to go up again.

This project takes a different approach as 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.

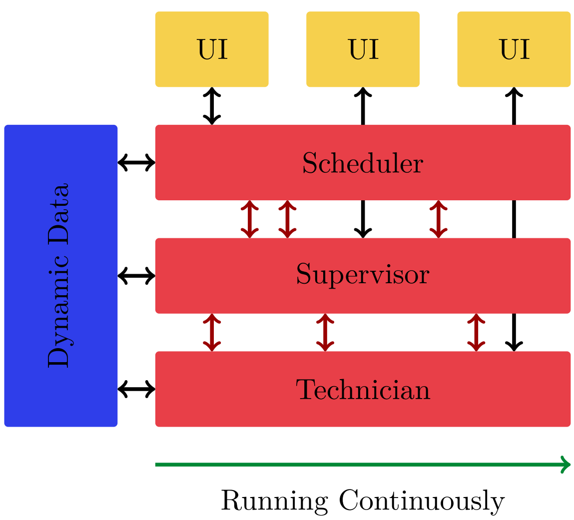

Figure 4: Having meta-heuristics continually running and coordinating state you can make a setup that is more robust to

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 saves alot of computations as you only need to reach initial convergence once. Making the user experience more responsive for the end user and consecutive solutions will look similar.
-

Pseudo Code

All implemented algorithms are based the 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 = SharedSolution
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

Strategic Model

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

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

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:

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

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

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

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

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

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

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

Operational Model

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:

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

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

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

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

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

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

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

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

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

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

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

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

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

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