AI-assisted Schedule Explainer for Nurse Rostering*

Demonstration

Kristijonas Čyras
Ericsson Research, Sweden
kristijonas.cyras@ericsson.com

Amin Karamlou
University of Austin, Texas, US
karamlouamin@gmail.com

Myles Lee
Oracle Inc., UK
myles.lee15@imperial.ac.uk

Dimitrios Letsios
King's College London, UK
dimitrios.letsios@kcl.ac.uk

Ruth Misener
Imperial College London, UK
r.misener@imperial.ac.uk

Francesca Toni
Imperial College London, UK
f.toni@imperial.ac.uk

ABSTRACT

We present an argumentation-supported explanation generating system, called Schedule Explainer, that assists with makespan scheduling. Our stand-alone generic tool explains to a lay user why a resource allocation schedule is good or not, and offers actions to improve the schedule given the user's constraints. Schedule Explainer provides actionable textual explanations via an interactive graphical interface. We illustrate our system with a proof-of-concept application tool in a nurse rostering scenario whereby a shift-lead nurse aims to account for unexpected events by rescheduling some patient procedures to nurses and is aided by the system to do so.

ACM Reference Format:

Kristijonas Čyras, Amin Karamlou, Myles Lee, Dimitrios Letsios, Ruth Misener, and Francesca Toni. 2020. AI-assisted Schedule Explainer for Nurse Rostering. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 3 pages.

1 INTRODUCTION

Mathematical optimisation affords effective techniques for solving problems formulated via resource constraints and objective functions. Makespan scheduling is one such fundamental discrete optimisation problem [7] concerning effective resource allocation. It underlies numerous real-life applications, from massive scale industrial sheet production [8] to nurse rostering in clinical settings [11]. Mathematical formulations of scheduling problems allow for development of highly-effective solvers (e.g. GLPK [1]). Yet, the same mathematical intricacies make solvers black-boxes: their workings and outcomes are hardly explainable even to experts, let alone lay users. As such, the solvers are not interactive for their users either.

Explainability in scheduling includes, but is not limited to, supporting a user of a system equipped with an optimisation solver in understanding why the system behaves as it does. To be (at least partially) explainable, such a system should first and foremost be able to justify why a solution is good or not. On top of that, it should afford some form of interaction in order for the user to modify the solution and the system to further explain whether and why the resulting one is good. Explainability is of critical importance in clinical settings [10] such as managing nursing personnel. Typical queries in scheduling applications to nurse rostering are:

- What if nurse A were to do job Y rather than job X?
- Why is job X assigned to nurse A rather than nurse B?
- Why is the schedule (not) good?

Recently, a novel paradigm ArgOpt of argumentation for explainable scheduling was proposed [4]. ArgOpt combines (computational) argumentation [2, 9]—a branch of Knowledge Representation and Reasoning [3] within the field of AI—with optimisation to explain the goodness of schedules. Specifically, abstract argumentation (AA) [5] affords an intermediate layer between a solver and a user, capturing the scheduling problem and the mathematical conditions underlying the goodness of schedules. The AA representation of the problem and its solutions allows to formulate and extract formal argumentative explanations which are in turn transformed into user-friendly natural language explanations about a given schedule.

Čyras et al. showed ArgOpt meets crucial desiderata of soundness and completeness as well as cognitive and computational tractability of explanations [4]. They established that formal explanations can be extracted efficiently and illustrated turning them into template-based natural language explanations. However, the paradigm is described only in principle, without detailing its implementation in practice. We tackle the latter issue in this paper.

Building upon the ArgOpt paradigm, we present an AI-assisted system, *Schedule Explainer*, that provides usable explanations in makespan scheduling easily and with clarity. Our system integrates the following components: a) an optimisation solver, allowing to instantiate and solve makespan scheduling problems; b) an AA layer, capturing the mathematical properties of schedules, thereby enabling formal definition and extraction of explanations; c) a user interface, supporting concise actionable explanations and interaction with schedules given, indifferently, by the solver or the user. To complement the general user interface useful for dealing with generic makespan scheduling problem instances, we specifically design a graphical user interface (GUI) for illustrating the potential application of Schedule Explainer in clinical settings, specifically nurse rostering. Schedule Explainer with nurse rostering GUI thus enables a lay user to meaningfully interact with an optimisation solver in allocating resources depending on the user's needs, ensuring the goodness of schedules and providing textual, audial and visual explanations thereof. To our knowledge, it is a unique interactive proof-of-concept tool for explainable nurse rostering in particular, as well as for explainable scheduling more generally.

*K. Čyras, A. Karamlou, M. Lee and D. Letsios have pursued research presented here while at Imperial College London and at the time of publishing are its alumni.

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand.

2 SCHEDULER EXPLAINER TOOL

Our tools are publicly available at `github.com/kcyras/aes` (backend) and `github.com/AminKaramlou/AESWebApp` (with the integrated nurse rostering interface). A video introducing Schedule Explainer for nurse rostering is available at `streamable.com/ctztb`. We refer the reader to [4] for details on the ArgOpt paradigm.

Schedule Explainer operates a makespan scheduling problem instance thus. $(\mathcal{M}, \mathcal{J})$ is a pair with a set $\mathcal{J}$ of n jobs (think patient procedures), with processing times p_j for $j \in \mathcal{J}$, to be executed by a set $\mathcal{M}$ of m machines (think nurses). Each job must be processed by exactly one machine and each machine may process at most one job per time. The objective is to find a minimum makespan schedule, i.e. to minimise the last machine completion time.

A schedule is a zero-one matrix of job assignments to machines. Intuitively, a schedule is *feasible* if it meets the assignment constraints (i.e. each and every patient procedure is done by exactly one nurse), and optimal if it also minimises the objective function. As finding optimal schedules is NP-hard, our tool tractably evaluates schedules that are provably approximately optimal: a schedule is *efficient* if it is feasible and satisfies the properties of single and pairwise exchanges which insist that the schedule cannot be improved by exchanging one or two jobs between machines. Schedule Explainer also accommodates *fixed decisions* that insist on specific (non-)assignments (think qualification-dependent procedures).

Explanations about schedules being good in terms of feasibility, efficiency and satisfaction of fixed decisions can be formulated in the language of abstract argumentation. Intuitively, the problem instance, schedules and their properties are captured via argumentation frameworks (directed graphs) whereby explanations are defined as directed paths representing the violations of the goodness of schedules. Our tool filters these by importance: first the feasibility and fixed decisions are considered; then the potential improvements of moving one or two jobs are provided in decreasing order of time saved given the move. Schedule Explainer gives natural language interpretations to the explanations (potentially with voice-over), and accompanies them with visual clues for presentation to the user. Importantly, the explanations are *actionable* in that the user can apply the actions suggested by the tool, such as reassigning jobs, to improve the schedule. Upon taking an action, the tool recomputes whether the schedule is good and gives explanations accordingly.

Backend. The Schedule Explainer's backend allows the user to specify the scheduling problem instance, fixed decisions and a schedule. It also allows for modifications in all of these at any point of interaction. GLPK solver can be used to find an efficient schedule, if needed. Schedule explainer visualises the given schedule via a cascade chart, yields explanations and lists available actions for any selected explanation. The actions are in the form of single or pairwise exchanges of jobs on machines aimed at satisfying fixed decisions and reducing the last machine completion time.

Internally, Schedule Explainer maps the inputs into AA frameworks as detailed in [4], using Boolean tensors for internal representation. The algorithms for checking and explaining efficiency run in time and memory at most quadratic in mn. They can handle tens of machines and hundreds of jobs, generating templated explanations for problems with $mn \leqslant 1000$ well under a second on a virtual machine with dual-core CPU at 2GHz with 2GiB RAM.

Nurse Rostering Application. The Schedule Explainer's nurse rostering GUI is a web application illustrating the backend's functionalities in clinical care settings. User input is tailored to add or remove procedure (job) and nurse (machine) cards one by one. Types of procedures (e.g. blood test) and similar application-specific features can be pre-set depending on the scenaria in question. The outputted explanations and accompanying actions are similarly tailored using pre-set textual templates, verbalised using standard web browser text-to-speech engines. Visual clues pertaining to e.g. the nurses' "emotions" (expressed via static images) and available actions (enriched with icons) help the user to orientate. To ease the cognitive load, the most pertinent explanation is displayed at a time, with the corresponding actions ordered by potential improvement. (See screen capture [0:50–1:33] in the video `streamable.com/ctztb`.)

3 DISCUSSION

The Schedule Explainer's backend is easily usable by non-experts who know what makespan scheduling is. We have conducted over 30 surveys evaluating the tool's usability and usefulness. The survey takers comprised a varied audience: school leavers and graduates, university students, generally people ranging from having little to extensive knowledge of mathematical optimisation. While the survey results do not present statistically significant results, we found that Schedule Explainer's backend was largely usable and the explanations comprehensible and manageable. The users by and large envisaged the system as potentially helpful for larger tasks.

While the Schedule Explainer's backend is adequate for typical nurse rostering problem sizes in clinical settings, the nurse rostering GUI is not intended to handle real scenaria or depict how the system would appear to the user, e.g. the shift-lead. Rather, it illustrates conceptually how AI-assisted explainable scheduling could arise in a multi-agent setting. We have presented the system to an urgent care centre operational manager for qualitative evaluation and solicited very positive feedback regarding both the need and the potential benefits of such a tool for explainable nurse rostering. Lay users interacting with the system have also found it intuitive, with noticeable contrast in its helpfulness to improve schedules with explanations and actions present versus absent.

We presented a first-of-a-kind, argumentation-supported explainable scheduling system. Among the future challenges we stress Schedule Explainer's adaptation to variations of scheduling problem that are also pertinent to applications including but not limited to nurse rostering [6]. Equally important is further evaluation of the system's usability and usefulness with domain experts.

Acknowledgements

K. Čyras, A. Karamlou, R. Misener and F. Toni were supported by the EPSRC project **EP/P029558/1** ROAD2H: *Resource Optimisation, Argumentation, Decision Support and Knowledge Transfer to Create Value via Learning Health Systems.* D. Letsios was supported by the EPSRC project **EP/M028240/1** *Uncertainty-Aware Planning and Scheduling in the Process Industries.* The authors are grateful to Gabriella Long (UK NHS & Institute of Global Health Innovation) for extensive feedback on nurse rostering in clinical centres.

Data access statement: No new data was collected in the course of this research.

REFERENCES

[1] Andrew Makhorin. 2000. GLPK (GNU Linear Programming Kit). https://www.gnu.org/software/glpk/. (2000). Accessed: 2020-01-13.

[2] Trevor J M Bench-Capon and Paul E Dunne. 2007. Argumentation in Artificial Intelligence. *Artificial Intelligence* 171, 10-15 (2007), 619–641. https://doi.org/10.1016/j.artint.2007.05.001

[3] Ronald J Brachman and Hector J Levesque. 2004. *Knowledge Representation and Reasoning*. Elsevier.

[4] Kristijonas Čyras, Dimitrios Letsios, Ruth Misener, and Francesca Toni. 2019. Argumentation for Explainable Scheduling. *33rd AAAI Conference on Artificial Intelligence* 33 (jul 2019), 2752–2759. https://doi.org/10.1609/aaai.v33i01.33012752

[5] Phan Minh Dung. 1995. On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-person Games. *Artificial Intelligence* 77 (1995), 321–357. https://doi.org/10.1016/0004-3702(94)00041-X

[6] A.T. Ernst, H. Jiang, M. Krishnamoorthy, B. Owens, and D. Sier. 2004. An Annotated Bibliography of Personnel Scheduling and Rostering. *Annals of Operations Research* 127, 1 (2004), 21–144.

[7] Ronald L. Graham. 1969. Bounds on Multiprocessing Timing Anomalies. *SIAM Journal of Applied Mathematics* 17, 2 (1969), 416–429.

[8] Greycon Ltd. 2018. The Art of Fine Planning. https://www.greycon.com/wp-content/uploads/2018/11/opt-Studio-flyer-181109.pdf. (2018). Accessed: 2020-01-13.

[9] Iyad Rahwan and Guillermo Ricardo Simari. 2009. *Argumentation in Artificial Intelligence*. Springer. https://doi.org/10.1007/978-0-387-98197-0

[10] Lucia Sacchi, Stefania Rubrichi, Carla Rognoni, Silvia Panzarasa, Enea Parimbelli, Andrea Mazzanti, Carlo Napolitano, Silvia G. Priori, and Silvana Quaglini. 2015. From Decision to Shared-decision: Introducing Patients' Preferences Into Clinical Decision Analysis. *Artificial Intelligence in Medicine* 65, 1 (2015), 19–28. https://doi.org/10.1016/j.artmed.2014.10.004

[11] D. Michael Warner and Juan Prawda. 1972. A Mathematical Programming Model for Scheduling Nursing Personnel in a Hospital. *Management Science* 19, 4 (1972), 411–422. http://www.jstor.org/stable/2629025