Summary of “Randomized Problem-Relaxation Solving for Over-Constrained Schedules”*

Patrick Rodler[1]
Department of Artificial Intelligence and Cybersecurity, University of Klagenfurt, Austria

Erich Teppan
Department of Artificial Intelligence and Cybersecurity, University of Klagenfurt, Austria

Dietmar Jannach
Department of Artificial Intelligence and Cybersecurity, University of Klagenfurt, Austria

Abstract

We present a general framework for tackling over-constrained job shop scheduling problems (JSSP) where the volume of jobs (orders) exceeds the production capacity for a given planning horizon. The goal is to process as many or as utile jobs as possible within the available time. The suggested framework approaches this optimization problem by solving multiple randomly modified relaxed problem instances, thereby taking a sample in a solution space that covers all optimal solutions. By continuously storing the best solution found so far, the result is a complete anytime algorithm that incrementally approximates an optimal solution. The proposed framework allows for highly parallel computations, and all of its modules are treated as black-boxes, allowing them to be instantiated with the most performant algorithms for the respective sub-problems. Using IBM's cutting-edge CP Optimizer suite, experiments on well-known JSSP benchmark problems demonstrate that the proposed framework consistently schedules more jobs in less computation time compared to a standalone constraint solver approach. Due to the generality of the proposed approach and its applicability to computing minimum-cardinality or most preferred minimal diagnoses, this work has the potential to positively impact the field of model-based diagnosis.

2012 ACM Subject Classification Computing methodologies → Causal reasoning and diagnostics; Computing methodologies → Randomized search; Computing methodologies → Optimization algorithms; Computing methodologies → Parallel algorithms; Computing methodologies → Modeling and simulation; Applied computing → Industry and manufacturing

Keywords and phrases diagnosis computation, randomized diagnosis computation, minimum-cardinality diagnoses, most preferred diagnoses, maximum-probability diagnoses, applications of diagnosis (over-constrained scheduling problems), diagnosis-based optimization, constraint programming, CP Optimizer, job shop scheduling problem, job set optimization problem, operations research, scheduling, industry use cases, minimal subset subject to a monotone predicate (MSMP) problem, problem relaxation, sampling for optimization, anytime algorithm

Digital Object Identifier 10.4230/OASIcs.DX.2024.33

Category Extended Abstract

Related Version *Full Version*: `https://doi.org/10.24963/kr.2021/72` [12]

Funding This work was supported by FFG, contract FO999910235.

[1] Corresponding author.
* See [12] for the Full Paper.

Summary

Introduction and Motivation. Job scheduling is a crucial task in production industries. One important and frequently addressed problem is the *Job Shop Scheduling Problem (JSSP)* [2], which is NP-hard in the general case. Given a set of machines and a set of jobs, where each job consists of an ordered sequence of operations, with each operation having a specified duration and needing to be executed on a particular machine, the JSSP involves finding a schedule that assigns a start time for each operation on its respective machine such that *(i)* each operation of a job can only begin after the preceding operation of the same job is completed, *(ii)* on each machine, the next operation may only start once the current operation is finished, and *(iii)* the completion time (i.e., the time required to process all jobs) is minimized. The (NP-complete [4]) decision version of a JSSP involves a deadline as an additional input, and asks whether there is a schedule satisfying criteria (i) and (ii) with a completion time before the deadline.

Different methods have been applied to approach JSSPs. Among those, constraint programming (CP) has a long and successful history, and present-day CP solving systems are able to handle large-scale problem instances. However, today's often highly dynamic production regimes, supporting, e.g., make-to-order or lean production, lead to optimization problems on top of the underlying JSSPs which may significantly increase computation times.

One typical problem of that type arises when the set of orders (jobs) exceeds the current production capacities with respect to a given planning horizon (e.g., a week). In such a situation the producer is confronted with an over-constrained JSSP, i.e., it is not possible to work off all the orders in time and it must be decided which of the jobs to postpone. We call the task of finding a job set of maximal utility (e.g., revenue) that can be finished within a given planning horizon the *Job Set Optimization Problem (JOP)*, which is NP-hard [1].

When using constraint programming, a straightforward approach to solving JOPs is to adapt the CP encoding of the over-constrained JSSP by adding an optimization statement, which effectuates that the utility of jobs scheduled and finished prior to a given deadline is maximized. We call this the *direct CP approach* to JOP. Since however the JOP represents a hard problem on top of the JSSP, also the most powerful state-of-the-art CP solvers may struggle with the increased problem complexity.

Approach. As a remedy to this, we propose a framework to tackle the JOP based on the observation that the problem of computing a subset-minimal job set to be postponed *(i)* is a relaxation of the JOP, *(ii)* can be solved using a linear number of CP solver calls, each for one JSSP decision problem, and *(iii)* is not directly supported by current CP solvers. The idea is to compute multiple such subset-minimal job sets in a random way, thus intuitively taking a random sample in a solution space that covers all JOP solutions. By always storing the best found solution, the framework allows to successively approach a JOP solution.

Compared to the direct CP approach, which can be seen as tackling two problems at once, i.e., the (implicit) subset-minimality as well as the optimal utility of the JOP solution, our framework achieves a disentanglement of these two problems by extracting the (efficient and well understood) MSMP [5] (*M*inimal *S*ubset subject to a *M*onotone *P*redicate) reasoning from the solver and leaving to the solver the role of deciding a polynomial number of JSSP instances (for which state-of-the-art solvers are optimized).

Our framework consists of three modules: a *CP solver* for decision versions of JSSP, a sufficiently general [7] *MSMP algorithm* for finding subset-minimal job sets, and a *random number generator* for creating multiple random subset-minimal job sets to enable optimization.

Remarks: *(Anytime Property)* Our approach is an anytime algorithm, i.e., the solution quality increases monotonically throughout the solving process. *(Completeness)* Given sufficient time and a suitable full-cycle random number generator (cf. [13]), the algorithm will always produce a JOP solution. *(Adaptability)* Each module in the framework is viewed as a black-box, allowing integration of different algorithms and benefiting from the latest research in JSSP and MSMP. *(Parallelization)* No information exchange is required between iterations, enabling efficient multi-threaded implementations of our approach. *(Automation)* No manual adaptation of the CP encoding of a given JSSP instance to a JOP encoding is needed. *(Generality)* The approach is applicable to computing minimum-cardinality or most preferred diagnoses for any model-based diagnosis problem as per Reiter's definition [6] (see below).

Evaluation. To compare the proposed framework with a direct CP approach, we conducted evaluations on 100 JOP instances extracted from Taillard's well-known JSSP benchmark suite [15] (where uniform job utility is assumed). We implemented the direct CP approach by IBM's CP Optimizer, a currently leading CP solver [3]; in our framework, we instantiated the CP solver also by CP Optimizer for a fair comparison, the MSMP algorithm by the random input permutation approach for Inv-QX [14] proposed by [9], and the random number generator by the respective Java function. Our results show that the use of the proposed framework *always leads to schedules involving more in-time finished jobs* than the direct CP approach. Specifically, our approach schedules an avg. of 8 % and up to 15 % more jobs for instances with 50 jobs and 15 machines, and 5 % (avg.) and 13 % (max.) more for instances with 100 jobs and 20 machines. This means a reduction in the number of postponed jobs (cf. diagnosis cardinality in model-based diagnosis) by 27 % (avg.) and 63 % (max.). Additionally, our approach is *always significantly faster* as it constantly yields better results (up to 12 % more scheduled jobs) within 1 hour than the direct CP approach achieves in 2 hours.

Relation to Model-Based Diagnosis. The proposed framework to approach the JOP is generally applicable to compute most preferred (or minimum-cardinality) diagnoses for any diagnosis problem $\mathcal{P} := (\textsc{sd}, \textsc{comps}, \textsc{obs})$ as per Reiter's theory [6]. This can be recognized by observing that *(1)* the computation of a minimal diagnosis given $\mathcal{P}$ is an MSMP problem (cf. [7]), just as the computation of a subset-minimal job set that needs to be postponed to satisfy a scheduling deadline is an MSMP problem, and *(2)* the problem of computing a minimal diagnosis is a relaxation of the problem of computing a most preferred (or minimum-cardinality) diagnosis[2], just as finding a subset-minimal job set to be postponed is a relaxation of the JOP. Hence, put in diagnostic terms, our approach involves the computation of a random set of minimal diagnoses for a given diagnosis problem $\mathcal{P}$ while always storing the best found diagnosis so far, aiming to ultimately find one or more most preferred (or minimum-cardinality) diagnoses. According to the classification scheme suggested in [10] (see the definitions there), the diagnosis algorithm based on our approach is sound, complete, not best-first, multiple-solution, not conflict dependent, generally applicable, logics-agnostic, and space-efficient, and it uses black-box reasoning.

In fact, as discussed in [11], the JOP can be interpreted as the task of computing a most preferred diagnosis Δ for the diagnosis problem $\mathcal{P}_{\text{JOP}} := (\textsc{sd}_\kappa, \textsc{comps}, \textsc{obs})$ where

- the *system description* $\textsc{sd}_\kappa$ corresponds to the JSSP instance with deadline κ,
- the *set of system components* $\textsc{comps}$ constitutes the original job set J of the JSSP,
- the *abnormality assumption* for a component means removing the respective job from J,

[2] Under the common assumption that the preference function p (e.g., probability) of a diagnosis is strictly antimonotone, i.e., $p(X) > p(Y)$ holds whenever $X \subset Y$ (cf. [8, Sec. 2.1.3]).

- the *set of observations* OBS is empty,
- the *consistency check* whether $\text{SD}_\kappa \cup \text{OBS} \cup \Delta$ is non-contradictory assuming $\Delta \subseteq \text{COMPS}$ to be abnormal and $\text{COMPS} \setminus \Delta$ to be normal amounts to solving a JSSP decision problem,
- the *component costs* are the given (positive) job utilities u_j for $j \in J$, and where

the preference function $p(\Delta)$ for a diagnosis Δ is defined as the sum of the utilities of jobs in $J \setminus \Delta$. Note, for uniform job utilities, minimum-cardinality diagnoses are most preferred.

This essentially means that any generally applicable best-first diagnosis computation algorithm can be applied to tackle the JOP (cf. [10] for a definition and examples of such algorithms). Based on our insights from the scheduling domain, we expect our proposed "optimization by relaxation and randomization" strategy to be particularly useful if the deterministic computation of preferred (or minimum-cardinality) diagnoses turns out to be computationally hard or impractical, and/or if highly parallel implementations are possible.

References

1 Philippe Baptiste. On minimizing the weighted number of late jobs in unit execution time open-shops. *EJOR*, 149(2):344–354, 2003. doi:10.1016/S0377-2217(02)00759-2.

2 Jacek Blazewicz, Klaus H. Ecker, Erwin Pesch, Günter Schmidt, Malgorzata Sterna, and Jan Weglarz. *Handbook on Scheduling: From Theory to Practice*. Springer, 2019.

3 Giacomo Da Col and Erich C. Teppan. Industrial size job shop scheduling tackled by present day CP solvers. In *Principles and Practice of Constraint Programming (CP)*, 2019.

4 Michael Garey, David Johnson, and Ravi Sethi. The complexity of flowshop and jobshop scheduling. *Math. Oper. Res.*, 1(2):117–129, 1976. doi:10.1287/MOOR.1.2.117.

5 Joao Marques-Silva, Mikoláš Janota, and Anton Belov. Minimal sets over monotone predicates in boolean formulae. In *Int'l Conf. on Computer Aided Verification (CAV)*, 2013.

6 Raymond Reiter. A theory of diagnosis from first principles. *Artif. Intell.*, 32(1):57–95, 1987. doi:10.1016/0004-3702(87)90062-2.

7 Patrick Rodler. A formal proof and simple explanation of the QuickXplain algorithm. *Artif. Intell. Rev.*, 55(8):6185–6206, 2022. doi:10.1007/S10462-022-10149-W.

8 Patrick Rodler. Memory-limited model-based diagnosis. *Artif. Intell.*, 305:103681, 2022. doi:10.1016/J.ARTINT.2022.103681.

9 Patrick Rodler. Random vs. Best-First: Impact of Sampling Strategies on Decision Making in Model-Based Diagnosis. In *AAAI Conf. on Artificial Intelligence (AAAI)*, 2022.

10 Patrick Rodler. How should I compute my candidates? A taxonomy and classification of diagnosis computation algorithms. In *European Conf. on Artificial Intelligence (ECAI)*, 2023.

11 Patrick Rodler and Erich Teppan. The scheduling job-set optimization problem: A model-based diagnosis approach. In *Int'l Workshop on Principles of Diagnosis (DX)*, 2020. arXiv:2009.11142. arXiv:2009.11142.

12 Patrick Rodler, Erich Teppan, and Dietmar Jannach. Randomized problem-relaxation solving for over-constrained schedules. In *Int'l Conf. on Principles of Knowledge Representation and Reasoning (KR)*, 2021. doi:10.24963/kr.2021/72.

13 Linus Schrage. A more portable Fortran random number generator. *TOMS*, 5(2):132–138, 1979. doi:10.1145/355826.355828.

14 Kostyantyn Shchekotykhin, Gerhard Friedrich, Patrick Rodler, and Philipp Fleiss. Sequential diagnosis of high cardinality faults in knowledge-bases by direct diagnosis generation. In *European Conf. on Artificial Intelligence (ECAI)*, 2014.

15 Eric Taillard. Benchmarks for basic scheduling problems. *EJOR*, 64(2):278–285, 1993.