Smart Short Term Capacity Planning: A Reinforcement Learning Approach

Manuel Schneckenreither, Sebastian Windmueller, Stefan Haeussler

Smart short term capacity planning: A reinforcement learning approach

Manuel Schneckenreither[1][0000−0002−4812−4665], Sebastian Windmueller[1], and Stefan Haeussler[1][0000−0003−2589−1367]

Department of Information Systems, Production & Logistics; University of Innsbruck, Innsbruck, Austria stefan.haeussler@uibk.ac.at

Abstract. Capacity planning is an important production control function that significantly influences firm performance. Especially, in the short term, we face a dynamically changing system which calls for an adaptive capacity planning system that reacts based on the current state of the shop floor. Thus, this paper analyzes the performance of a reinforcement learning (RL) algorithm for overtime planning for a make-to-order job shop. We compare the performance of the RL algorithm to mechanisms that set overtime-hours statically or randomly over time. Performance is measured in total costs which consist of overtime, holding and backorder costs. The results show that our tested benchmarks can be outperformed by the RL algorithm, where the major savings were achieved due to less needed overtime.

Keywords: Reinforcement learning · capacity planning · simulation.

1 Introduction

Capacity planning is an important production control function that significantly influences firm performance. It is often divided into long, mid and short term capacity planning [27,14,1]. Long-term capacity planning focuses on yearly resource requirements for manufacturing including plant locations and capacities, planning with suppliers and establishing new technologies or processes. In the medium-term the focus is on monthly or quarterly resource requirements, such as the amount of workforce, raw materials and inventories. In the short term, capacity planning is made on a daily or weekly basis. The task is to match resources, work centers and jobs based on the specific job's requirements [1], where one of the main issues is to balance overtime, holding and backorder costs. This paper focuses on short term capacity planning for a make-to-order manufacturer. Due to the short term nature, we are facing a dynamically changing system with e.g., machine failures, delays and other factors interrupting the manufacturing process. Thus, it would be beneficial to have an adaptive planning system that plans capacities based on the current state of the shop floor. This paper addresses this problem by proposing a smart short term capacity planning approach using Machine Learning. More precisely, we test the performance of a reinforcement learning algorithm in comparison to some static approaches with regard to holding, backorder and overtime costs.

2 Literature review

This literature review is divided into two parts: First, we review the literature on short term capacity planing and second we review the literature on applications of reinforcement learning algorithms in short term production planning and control problems.

There are quite some research papers that use overtime as a way to overcome the problem of high lateness costs, where most authors use heuristics and mathematical programming. Yang et al. [28] look into a single machine shop floor environment that is purely deterministic. Their proposed priority algorithm, for balancing the use of overtime and regular time, showed close to optimal solutions, however is limited by the low shop floor complexity and the deterministic environment. A similar environment is considered by Jaramillo and Erkoc [7], where a finite set of jobs runs in a single machine job shop. Their proposed heuristic managed to beat a mathematical model in CPU-time and cost performance. Ornek and Cengiz [15] employ multiple stacked linear programming models to design a dynamic production planning system that controls lot sizes, alternative job routing and overtime decisions which results in a capacity feasible material plan for a job shop environment. Yuan et al. [29] develop a production planning model that determines planned lead times and production lot sizes to minimize inventory and overtime costs in a job shop production system. Finally, Chen et al. [1] propose a model for make-to-order companies which allows for selecting only the most profitable orders while maintaining delivery date adherence. They suggest using overtime and outsourcing for short term capacity planning, although back-ordering is not allowed in their mixed-integer programming model. They highlight the viability of their approach to small problems sizes, but state that more efficient algorithms are needed to tackle the capacity planning issue on an industrial scale.

Machine learning can be divided into the categories of supervised learning, unsupervised learning and reinforcement learning [25]. In supervised learning the algorithms are presented with input-output tuples and seek to find a function that best describes this data, where best is defined as the least squared error. In unsupervised learning the algorithms are given data without output labels. They use the structure of the data to cluster it or learn latent variables in Bayesian models. Finally, in reinforcement learning (RL) the input/output pairs are actually never presented to the algorithm, but a function rewards the outcome of previously (by an agent) chosen actions. Thus, it models the human learning capabilities by trial-and-error as the agent usually starts without prior knowledge [25,19]. There are numerous highly-sophisticated algorithms available [13,12,24,6]. With regard to RL techniques applied to short term production planning problems, several papers were published. To the best of our knowledge there are only two studies that use RL techniques for order release planning. Paternina-Arboleda and Das [16] use an R-learning approach called 'SMART' (developed in [2]) to release orders in a single product, serial flow line. They compare its performance to conventional order release policies (e.g., Kanban

and CONWIP) and show that the RL algorithm yields less inventory and is more agile, meaning that it can react more quickly to the dynamic production environment. Schneckenreither and Haeussler [21] use several different value iteration RL algorithms (Q and R-learning) to make periodic order release decisions for a flow shop production system. They show that their approach outperforms static order release mechanisms by yielding lower costs, lateness and standard deviation of lateness.

With regard to dynamic scheduling, there is a growing increase in the use of RL algorithms where most papers focus on value iteration algorithms. Paternina-Arboleda and Das [17] use a relaxed SMART algorithm (R-learning) for a three product, stochastic lot scheduling problem. They show that their approach is computationally feasible and is able to reduce the inventory while keeping the backorders at low levels. Wang and Usher [26] use Q-learning for a job routing problem in a 10 machines job shop and show that the RL algorithm performs equally or better than the benchmark heuristics. An interesting extension is presented by Qu et al. [18] who show that Q-learning can be applied to consider scheduling together with optimal assignment of multi-skilled workers. Furthermore, some papers apply deep Q-learning to scheduling problems where the early works of Zhang and Dietterich [30,31] apply RL for payload scheduling of NASA space shuttles. They find that RL outperforms an approach that combines heuristics and simulated annealing by yielding a lower makespan. Finally, [3,10,9] apply a policy iteration algorithm for scheduling in a job shop where the former two consider eight machines and three production stages. Their RL algorithm outperforms their benchmark: a two stage decision rule that prioritizes orders with regard to their waiting time and then selects the machine with the least workload. In [3] the RL algorithm is tested to a number of scheduling benchmark problems from the OR library ranging from 5 resources and 10 jobs to 15 resources and 30 jobs and modify them by including stochasticity. They show that for deterministic and stochastic scenarios the RL algorithm outperforms scheduling rules such as FIFO and SPT. However, to the best of the authors' knowledge there is no application of RL to short term capacity planning problems.

3 Job shop model and short term capacity planning models

We use a hypothetical job shop in a rolling horizon environment similar to the study of [22] (see Figure 1). We consider a make-to-order restricted job shop, where all arriving orders are collected in an order pool. The orders are processed at the work centers M1, M2 and M3 (each containing a queue and a machine), and the final products are stored in the finished goods inventory (FGI) until their due date. There are three machines and six different product types and all machines can process each product type and each machine can only process one job at a time. We expect the incoming orders to be uniformly distributed among the product types. The routing of each product type is given by the edge

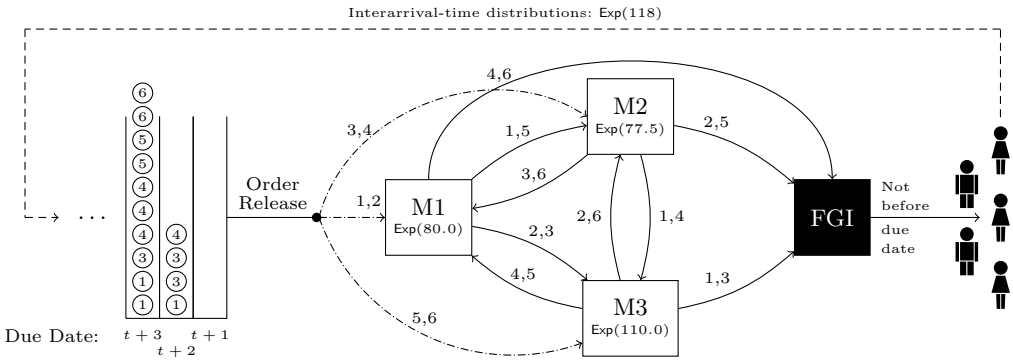

Fig. 1. Job Shop Simulation Model with routing, processing time distributions and demand interarrival time distributions [22].

names, where the number corresponds to the product type. Therefore, each job has to go once through every machine before it is completed, but the routing differs between the six product types and there are no return visits. Thus, each job has to pass three production steps before it is finished (see Figure 1). A period is set to 16 hours (that is 960 minutes) and at the beginning of each working day, jobs get released to the shop floor following a backward infinite loading (BIL) logic which releases orders based on static lead time (3 periods) which is subtracted from the due date. The system employs a first-come-first-serve rule for the scheduling of jobs. Once a job has been finished, it gets moved to the finished goods inventory where it waits to be shipped at its designated due date. If the job is finished before its due date, it has to wait until the due date is reached and then gets shipped (early shipping is prohibited). If the job is finished after its designated due date, it gets shipped right away.

The processing times are exponentially distributed and are given under the corresponding node labels of the machines in Figure 1. The interarrival times of orders arriving at the system are exponentially distributed with a mean of 118 minutes. Orders that arrive to the system are assigned a due date which indicates the time at which the order has to be shipped. The due date slack is always ten periods. The interarrival times were set to yield an utilization of 95% at the bottleneck work center (M3) if no overtime is considered. The simulation model was implemented using Python 3.8.

Using overtime. Overtime is modelled as a decrease in processing time [11], which we only apply to the bottleneck machine M3. As one period or work day has 16 hours, an overtime of 12.5% refers to an additional 2, 25% refers to an additional 4 and an overtime of 50% refers to an additional 8 hours of available capacity.

Algorithm 1: PPO, Actor-Critic Style (Adapted from [24])

```
while stopping criteria is not fulfilled do
    for actor=1,2,...,N do
        Run policy π_θold in environment for T timesteps
        Compute advantage estimates Â_1, ..., Â_T
    end
    Optimise surrogate L_t(θ), with K epochs and minibatch size M ⩽ NT
    θ_old ← θ
end
```

Costs. The cost parameters are set by assuming an increase in value from WIP holding (WIP_C: \$1 per order and period) to the final goods inventory (FGI) holding costs (FGI_C: \$4 per order and period). The backorder costs are set very high (BO_C: \$16 per order and period) due to the MTO environment. If overtime was used during a period, a base cost of \$8 multiplied with the amount of extra hours is incurred to pay the additional hours that workers had to put in. Overtime costs (OT_C) are the only costs that are incurred globally as a flat-rate cost, instead of a cost per individual job.

Short term capacity planing. The challenge of the problem at hand is to find an optimal balance of the costs incurred through the production, i.e. WIP_C, FGI_C, and BO_C, and the costs accumulated due to overtime OT_C. Therefore, short term capacity planning is a complex timing problem due to the cost structure, the non-linearity of the system and its' periodicity. In this paper we test three different short term capacity planing approaches: (i) three different static benchmarks which either plan with no overtime at all (no_OT) or fix the number of overtime-hours to 2, 4 or 8 (OT_2, OT_4 and OT_8 respectively). (ii) two random mechanisms which randomly choose between either zero, two and four ($OT\_Ran\_0\_2\_4$) or zero, four and eight overtime-hours ($OT\_Ran\_0\_4\_8$) and (iii) one reinforcement learning algorithm. For the later we use the Proximal Policy Optimization (PPO) algorithm of [24], which is an actor-critic policy gradient method that learns from its interaction with the environment. The policy (actor) and state-action function (critic) are represented by an artificial neural network (ANN) while parallel agents gather multiple consecutive experiences of the environment before each update, similar to n-step Q-Learning [12]. Algorithm 1 presents the PPO algorithm. There are N actors that compute the advantage estimates on the old policy $\pi_{\theta_{\text{old}}}$. Therefore, PPO maintains a state value function $V(s, a)$ and a policy function $\pi_\theta(s, a)$. The former evaluates each state-action pair (s, a) by a scalar value by means of the returned reward in the steps after visiting this state action pair, while the later describes the probability to choose the given action a in state s. Using the state value evaluation of T consecutively visited states an advantage is calculated that forms the basis of the policy function update: $\hat{A}_t = -V(s_t, a_t) + \sum_{i=0}^{T-1} \gamma^i r_{t+i} + \gamma^T V(s_{t+T}, a_{t+T})$,

where r_t is the returned reward at time step t and $0 < \gamma < 1$ the discount factor. In particular the advantage estimates for each time step of the agents, computed with the returned rewards and state values, form the surrogate loss. In the most intuitive way this is $L_t(\theta) = r_t(\theta)\hat{A}_t$ with the probability ratio $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$. Hence, it is a simple form of trust region policy optimisation [23]. Therefore, the more stochastic the policy function in regard to the last policy $\pi_{\theta_{\text{old}}}$ the smaller the probability ratio $r_t(\theta)$, as well as the better the value function describes the actual expected discounted future reward the smaller the advantages $\hat{A}_t$ and therefore the surrogate loss, i.e. updates to the policy. We set the discount factor $\gamma = 0.99$ throughout the paper and furthermore use clipping with $\epsilon = 0.2$, for the updates to prevent diverging policies. The loss is back-propagated through the artificial neural network by the Adam optimiser with default parameterisation [8]. We use this algorithm with 2048 experiences between each update with minibatch size 64 and 10 learning epochs on each policy improvement. Overall the policy is trained for 1 million periods.

4 Results

The length of each simulation run was 8000 periods including a warm-up period of 1000 periods and each scenario was run for 30 replications. Table 1 shows the results of the evaluations. The first column denotes the tested short term capacity planning approaches, namely (i) the static approaches are either denoted as OT_k where k represents the set number of overtime-hours or No_OT which plans no overtime at all, (ii) the random overtime approaches $OT\_Ran\_0\_2\_4$ and $OT\_Ran\_0\_4\_8$ which randomly decide whether overtime will be planned or not and (iii) the used RL algorithm denoted as either $PPO\_0\_2\_4$ or $PPO\_0\_4\_8$ depending on the set of valid actions the agent can execute. Column two to six depict the cost-based performance measures in Dollars: the Total Costs, the costs for held finished goods inventory (FGI_C), backorder costs (BO_C), costs for overtime-hours (OT_C) and WIP inventory holding costs (WIP_C). Finally, the last column shows the service level denoted as SL(%) reached in percent which is defined as the percentage of orders that were finished before their due date. The mean of the performance measures are compared and tested using a Wilcoxon signed-rank test at a significance level of $p = 0.05$. All values in Table 1 marked with an asterisk are not significantly different from the best performing model.

One can see that the two RL algorithms ($PPO\_0\_2\_4$ and $PPO\_0\_4\_8$) perform best regarding total costs. The best performing RL algorithm ($PPO\_0\_4\_8$) yields only 80.54% and 80.25% of total costs in comparison to the best static (OT_2) and random approach ($OT\_Ran\_0\_2\_4$) respectively. This is mainly due to the savings in overtime costs where the best RL algorithm yields less than than a third of the other two approaches. Furthermore, it is noteworthy that the WIP costs of the RL algorithms are rather high, only the WIP_C of No_OT are higher. This shows that the RL algorithms seems to find a good balance between highly utilized machines while maintaining good due date adherence.

Model	Total Cost	FGI_C	BO_C	OT_C	WIP_C	SL(%)
$PPO\_0\_4\_8$	411,234	259,577	66,896	34,450	50,312	93%
$PPO\_0\_2\_4$	417,765*	264,294*	66,369*	37,874*	49,228*	93%*
OT_2	510,609	300,218	57,585	112,000	40,806	94%*
$OT\_Ran\_0\_2\_4$	512,461	297,130	61,735	111,810	41,785	93%*
No_OT	587,826	173,770	312,342	0	101,715	66%
$OT\_Ran\_0\_4\_8$	614,014	340,668	19,972	224,183	29,190	98%
OT_4	614,832	347,913	15,572	224,000	27,347	98%
OT_8	853,620	384,561	3,261	448,000	17,798	100%

*not significant different

Table 1. Costs and service levels of the tested short term capacity planning models, sorted by total costs

5 Conclusion

Short term capacity planning can have a huge impact on the company performance. For this operative task a dynamic model is needed to react to the dynamic production system. Therefore, this paper analyzes the potential of smart short term capacity planning models, more precisely we test a reinforcement learning (RL) model on this task. We use a simulation of a three-stage make-to-order jobshop and compare its performance to simple benchmark heuristics. Performance is measured by by total costs consisting of holding, backorder and overtime costs and the service level which is defined as the percentage of orders finished before its' due date. The results show that the RL algorithm yields the lowest total costs which is mainly due to savings regarding overtime costs.

The main limitations of our study are threefold: (i) the results and findings are only limited to simulated case, (ii) while a promising RL algorithm was tested, there are others that might be even better suited for this task, e.g., [20] presents a very promising RL algorithm for capacity planning and (iii) our experimental design should be extended: First, by testing different demand and processing time distributions and second by comparing the performance of a RL algorithm to state-of-the-art approaches using optimization models such as [15] or [1]. Third, including more sophisticated order release models [5,4,22] to test the interrelation with order release planning is an interesting direction for future research.

References

1. Chen, C.S., Mestry, S., Damodaran, P., Wang, C.: The capacity planning problem in make-to-order enterprises. Mathematical and computer modelling **50**(9-10), 1461–1473 (2009)

2. Das, T.K., Gosavi, A., Mahadevan, S., Marchalleck, N.: Solving semi-markov decision problems using average reward reinforcement learning. Management Science **45**(4), 560–574 (1999)
3. Gabel, T., Riedmiller, M.: Distributed policy search reinforcement learning for job-shop scheduling tasks. International Journal of production research **50**(1), 41–61 (2012)
4. Haeussler, S., Netzer, P.: Comparison between rule-and optimization-based workload control concepts: a simulation optimization approach. International Journal of Production Research **58**(12), 3724–3743 (2020)
5. Haeussler, S., Schneckenreither, M., Gerhold, C.: Adaptive order release planning with dynamic lead times. IFAC-PapersOnLine **52**(13), 1890–1895 (2019)
6. Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., Silver, D.: Rainbow: Combining improvements in deep reinforcement learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 32 (2018)
7. Jaramillo, F., Erkoc, M.: Minimizing total weighted tardiness and overtime costs for single machine preemptive scheduling. Computers & Industrial Engineering **107**, 109–119 (2017)
8. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
9. Kuhnle, A., Röhrig, N., Lanza, G.: Autonomous order dispatching in the semiconductor industry using reinforcement learning. Procedia CIRP **79**, 391 – 396 (2019). https://doi.org/https://doi.org/10.1016/j.procir.2019.02.101, 12th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 18-20 July 2018, Gulf of Naples, Italy
10. Kuhnle, A., Schäfer, L., Stricker, N., Lanza, G.: Design, implementation and evaluation of reinforcement learning for an adaptive order dispatching in job shop manufacturing systems. Procedia CIRP **81**, 234–239 (2019)
11. Land, M.J., Stevenson, M., Thürer, M., Gaalman, G.J.: Job shop control: In search of the key to delivery improvements. International Journal of Production Economics **168**, 257–266 (2015)
12. Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., Kavukcuoglu, K.: Asynchronous methods for deep reinforcement learning. In: International conference on machine learning. pp. 1928–1937. PMLR (2016)
13. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (2015)
14. Olhager, J., Rudberg, M., Wikner, J.: Long-term capacity management: Linking the perspectives from manufacturing strategy and sales and operations planning. International journal of production economics **69**(2), 215–225 (2001)
15. Ornek, A., Cengiz, O.: Capacitated lot sizing with alternative routings and overtime decisions. International Journal of Production Research **44**(24), 5363–5389 (2006)
16. Paternina-Arboleda, C.D., Das, T.K.: Intelligent dynamic control policies for serial production lines. Iie Transactions **33**(1), 65–77 (2001)
17. Paternina-Arboleda, C.D., Das, T.K.: A multi-agent reinforcement learning approach to obtaining dynamic control policies for stochastic lot scheduling problem. Simulation Modelling Practice and Theory **13**(5), 389–406 (2005)
18. Qu, S., Wang, J., Govil, S., Leckie, J.O.: Optimized adaptive scheduling of a manufacturing process system with multi-skill workforce and multiple machine types:

An ontology-based, multi-agent reinforcement learning approach. Procedia CIRP **57**, 55–60 (2016)
19. Russell, S., Norvig, P.: Artificial intelligence: a modern approach. Prentice Hall Upper Saddle River, NJ, USA (2002)
20. Schneckenreither, M.: Average reward adjusted discounted reinforcement learning: Near-blackwell-optimal policies for real-world applications. arXiv preprint arXiv:2004.00857 (2020)
21. Schneckenreither, M., Haeussler, S.: Reinforcement learning methods for operations research applications: The order release problem. In: Nicosia, G., Pardalos, P., Giuffrida, G., Umeton, R., Sciacca, V. (eds.) Machine Learning, Optimization, and Data Science. pp. 545–559. Springer International Publishing, Cham (2019)
22. Schneckenreither, M., Haeussler, S., Gerhold, C.: Order release planning with predictive lead times: a machine learning approach. International Journal of Production Research pp. 1–19 (2020)
23. Schulman, J., Levine, S., Abbeel, P., Jordan, M., Moritz, P.: Trust region policy optimization. In: International conference on machine learning. pp. 1889–1897. PMLR (2015)
24. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017)
25. Sutton, R.S., Barto, A.G., et al.: Introduction to reinforcement learning, vol. 135. MIT press Cambridge (1998)
26. Wang, Y.C., Usher, J.M.: A reinforcement learning approach for developing routing policies in multi-agent production scheduling. The International Journal of Advanced Manufacturing Technology **33**(3-4), 323–333 (2007)
27. Wortman, J., Euwe, M., TAAL, M., Wiers, V.: A review of capacity planning techniques within standard software packages. Production Planning & Control **7**(2), 117–128 (1996)
28. Yang, B., Geunes, J., O'Brien, W.J.: A heuristic approach for minimizing weighted tardiness and overtime costs in single resource scheduling. Computers & Operations Research **31**(8), 1273–1301 (2004)
29. Yuan, R., Graves, S.C.: Setting optimal production lot sizes and planned lead times in a job shop. International Journal of Production Research **54**(20), 6105–6120 (2016)
30. Zhang, W., Dietterich, T.G.: A reinforcement learning approach to job-shop scheduling. In: IJCAI. vol. 95, pp. 1114–1120. Citeseer (1995)
31. Zhang, W., Dietterich, T.G.: High-performance job-shop scheduling with a time-delay td (λ) network. In: Advances in neural information processing systems. pp. 1024–1030 (1996)