Local Search Algorithms for Scheduling Vehicles in Cyclic Flexible Manufacturing Systems

Masterarbeit

zur Erlangung des akademischen Grades
eines Master of Science

an der Karl-Franzens-Universität Graz

vorgelegt von

Martin Gutjahr, BSc.

am Institut für Operations Research

Begutachter: Ao.Univ.-Prof. Dipl.-Ing. Dr.techn. Johann Kellerer

Graz, 2017

Abstract

Local Search Algorithms for Scheduling Vehicles in Cyclic Flexible Manufacturing Systems

xxx Untertitel der Arbeit englisch xxx

The problem of scheduling jobs and machines in flexible manufactoring systems has been addressed in scientific papers for a vast amount of complex variations. One of the big issues in this category of problems is, that even very basic variations of the problems connected with such systems are NP-complete. Most papers focus on trying to schedule the jobs on the machines optimally. This thesis tries to approach the underlying problem from an entirely different perspective: The schedule of jobs at each stage is considered given, only the amount and starting times of transportation vehicles can be varied. In addition to the constraint of the jobs being in a set order, the transportation vehicles are only allowed to follow a set circuit, visiting all the machines in the order in which the jobs need to be processed, to finally return to the depot and start a new lap. In order to solve the problem, a heuristical approach is made: an Iterative Local Search for a two-dimensional objective function minimizing the amount of machines and the makespan. This heuristic is then applied in two ways, with differences in the pertubation phase. In addition to the general heuristical approach, a Brute Force algorithm is used to solve a test case to be able to compare the solutions and run-times of the heuristics with an approach which returns the optimal solution. The heuristic variations and the brute force algorithm were implemented using Python.

Kurzfassung

xxx Titel der Diplomarbeit Deutsch xxx

xxx Untertitel der Diplomarbeit

Das Problem des Scheduling von Jobs und Maschinen in flexiblen Produktionssystemen wurde bereits in einer großen Anzahl an wissenschaftlichen Papern in vielen Variationen behandelt. Eines der großen Probleme dieser Art von Problemen ist, dass schon relativ einfache Variationen der auftretenden Probleme NP-vollständig sind. Während die meisten Paper versuchen die Jobs optimal auf den Maschinen anzuordnen versucht diese Arbeit das Problem aus einer anderen Perspektive zu betrachten: Die Schedule der Jobs auf den Maschinen wird als gegeben angenommen, nur die Anzahl und die Startzeiten der Transportfahrzeuge können verändert werden. Zusätzlich zur Bedingung dass die Jobs in einer vorgegebenen Reihenfolge bearbeitet werden müssen, dürfen die Transportfahrzeuge nur einem vorgegebenen Kurs folgen, welcher alle Maschinen in der Reihenfolge, in der die Jobs verarbeitet werden, besucht um schließlich wieder zum Depot zurückzukehren und eine neue Runde zu starten. Zur Lösungsfindung wird eine heuristische Herangehensweise verwendet: eine Iterative Lokale Suche für eine zweidimensionale Zielfunktion zur Minimierung der Fertigstellungszeit und der Anzahl an Transportfahrzeugen. Diese Heuristik wird dann in zwei Varianten angewandt, mit jeweils unterschiedlichen Pertubationsphasen. Zusätzlich zum heuristischen Ansatz wird ein Brute-Force-Algorithmus zur Lösung einer Testinstanz verwendet, damit die Ergebnisse und die Laufzeiten der Heuristik mit einem Ansatz, der die Optimallösung erhält, verglichen werden können. Die Heuristikvariationen und der Brute-Force-Algorithmus wurden in Python implementiert.

Inhalt

Vorwort und Danksagung

xxx Optionales Vorwort mit Danksagung... xxx

Eidesstattliche Erklärung

Ich erkläre ehrenwörtlich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benutzt und die den Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen inländischen oder ausländischen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht. Die vorliegende Fassung entspricht der eingereichten elektronischen Version.

Graz, am xxx. xxx 201x

(Max Mustermann)

Abkürzungsverzeichnis (xxx optional xxx)

Die folgende Auflistung gibt einen Überblick der wichtigsten in dieser Arbeit vorkommenden Begriffe, Akronyme, Abkürzungen und Formelzeichen:

	Abkz.	**Erklärung**
A	AHS	Allgemeinbildende Höhere Schulen
B	Bac	Bachelor (Fachwissenschaft)
	BHS	Berufsbildende Höhere Schulen (HTL, HLW, HAK)
	BK	Brückenkurs
	BORG	Oberstufenrealgymnasium
	Bsp/Bspe	Beispiel/Beispiele (Übungsaufgaben)
E	ECTS	Leistungspunkte im European Credit Transfer System
G	Gym	Gymnasium
P	P & P	paper and pencil (Fragebogen in Papierform)
	PS	Lehrveranstaltungstyp Proseminar (siehe UE)
U	UE	Lehrveranstaltungstyp Übung
	UGO	Uni Graz Online - Onlinesystem der Uni Graz (LV-Anmeldungen etc.)
	Uni	Universität (z. B. Uni Graz: Karl-Franzens Universität Graz)
V	V.I.	Vollständige Induktion
	VO	Lehrveranstaltungstyp Vorlesung
	VU	Lehrveranstaltungstyp Vorlesung mit Übung
W	WS	Wintersemester

1. Introduction

The field of scheduling problems has been a topic of interest for researches of various fields for dozens of years already. Through revelations of technology and economy however the field of new problems expands at a similar rate to the field of solved ones. While simpler approaches have been solved for some time now, globalisation not only leads to scheduling problems between factories that are distributed worldwide, but also to having bigger factories than ever before, resulting in a higher amount of machines and more jobs to be scheduled. With technological advancements those bigger factories are not simply staying the same problems enlarged in quantity of machines and jobs, but sophisticated new systems are being developed to be able to increase production speed, volume or reduce the amount of manual labour needed. The idea of offering multiple machines at a single stage of the production process lead researchers to the model of the Flexible Flow Shop. As with all flow shop problems the basic idea is that all jobs have to be processed on a set amount of machines in the same order. As this can relatively easily lead to bottlenecks due to most jobs taking a considerable amount longer on a given machine, the idea of parallel machines at a single stage is introduced. Each job being delivered to a stage with parallel machines may be assigned to either of those machines to be processed, its schedule for which machines to visit becomes flexible.

The scheduling problem focused on in this paper expands the idea of the flexible flow shop to make use of an additional technological advancement, which use is quickly spreading throughout all kinds of industrial fields: Automation. Automation is becoming increasingly relevant as being able to replace workers with automated machines promises a significant competitive edge. In this paper the above mentioned problem of scheduling the jobs in a flexible flow shop environment is set to be given. The focus here lies on the problem of transportation for the jobs between the machines. With the idea of flexible flow shop including multiple machines at each stage, the simplification of the transportation time between machines being zero becomes less and less realistic. As human resources are costly and skilled workers would be wasted on simple transportation tasks, automation is looked towards to give a hand with this problem. Machines such as conveyor belts or automatic guided vehicles (AGVs) are supposed to automatically transport the jobs between the machines without any human assistance other than service and routing.

For this thesis a currently relatively unexplored set-up is considered: A cyclic flexible manufactoring system. The cyclic system is distinct from the standard version in that the depot is starting point and finish for all jobs in the system. This entails the possiblity of being able to send AGVs on a steady cycle throughout the whole production area without any waiting time for possibly unfinished jobs. Waiting for a part to carry to the next stage would be a possiblity, but it would remove one of the key features this approach offers: It is collision free. In addition the position of each vehicle at any given time can already be set in stone solely by fixating the point in time at which the AGV leaves the starting point for the first time.

While the goal of the paper by Blazewicz and Pawlak was to solely minimize the amount of AGVs, here the approach is changed a bit: By fixating the amount of machines in an iteration of the meta-heuristic and simply focusing on the scheduling of a given amount of machines to minimize the makespan, a set of solutions for different amount of AGVs is received and a cost function depending on the amount of machines and the amount of overall delay in the for a given deadline can be used to select a final solution. If the goal remains to minimize the amount of AGVs for production without delay, the cost function can be set to cost any constant c for each machine and Big-M per unit of delay.

2. Characteristics of the problem

The problem of a cylcic flexible manufactoring system has certain limitations, which will be expanded and described in this chapter. The baseline of the overall problem is that vehicles, that constantly go in a cycle, are used to carry jobs from one machine to the next. Extensive reasearch has been made in various directions considering flexible manufactoring systems. While many of these papers focus on scheduling the tasks of each of the jobs on the machines, this paper will only consider a flow shop approach in which the machines are already sorted in the order, in which the jobs need to be processed. The cyclic version of this problem sees the AGVs being routed to drive on a steady course with start and finish being the same point, therefore the AGVs are constantly repeating laps of said track. This in turn enables us to know the position of an AGV at any given point in the system solely by setting a starting time at which the AGV is "released". The actual position in the system is then acquired by calculating the overall time minus the starting time modulo the lap time.

$$Pos_{mt} = (T_m + t) \bmod l$$

Since the schedule of the jobs on the machines themselves is considered to be given, after setting the AGVs to automatically pick up the job which is next in the schedule on the next machine and waiting in the outgoing queue of the current machine, the only input needed to be able to achieve a solution is a starting time for each of the AGVs.

Before it is possible to formulate an approach to solve this problem, some rules have to be set first, especially considering the handling of jobs in outgoing queues (jobs which have been completed at a machine but have not yet been picked up by an AGV). Since the system is set to be in constant movement, a machine cannot simply wait for a job not yet finished. Therefore a rule has to be set as to which job is to be picked out of those waiting in line. This rule requires the machine to pick up the next job in chronological order that is both scheduled to follow the job currently being worked on at the next stage, and waiting in the outgoing queue of the current machine e.g. if the first job is currently being processed on a machine at the next stage, the second job would be next in line. If that job is not available for transportation, by not being in line yet or already being transported, the third is considered and so on.

The original problem which laid the foundation for this paper, as described by AUTHOR in PAPER, differs from the problem the actually examined problem in another feature: In the paper by AUTHOR the goal was to minimize the amount of vehicles it takes to be able to deliver all jobs in time to satisfy a given schedule with the amount of allowed laps being given. The approach in this thesis is that there is no given deadline known beforehand. The idea is to offer multiple solutions with differing amounts of AGVs given. Each of these solutions will assign a makespan to a given amount of vehicles, which is defined by the sum of time units passed from the start

of the problem until all jobs have been completed and have been transported back to the depot. These solutions can then be ranked e.g. via a cost function that takes vehicles and time needed into consideration or, if the goal is to finish ahead of a given deadline, the cost function can be set to infinite when exceeding the threshold, to ensure that only the solution with the fewest vehicles which finishes in time will be selected.

As described in the previous paragraph, this paper reformulates the original problem posed in the paper by Blazcewicz et al to be multidimensional. This approach has been chosen because it increases the flexibility of the whole approach. While the approach with a given deadline that must be met is certainly not only suitable for a rare case, as these problems occur very often in all kinds of fields, a two-dimensional approach which also looks at the total makespan introduces the possibility of a deadline that may be broken for a cost (late surcharge) or even simpler a model in which the return is based on producing as fast as possible.

3. History of the problem in literature

filler

4. Choosing an approach

The problem of minimizing the amount of vehicles in a circular flexible flow shop was found to be NP-complete by AUTHOR (Unpublished Paper). If this holds true, then the problem being optimized in this thesis is also NP-complete. Therefore the approach was to find a suitable heuristic that would be able to provide a relatively decent solution, while maintaining a low running time. The selection of heuristical and meta-heuristical approaches for scheduling problems is vast, for flexible flow shop problems alone there have been multiple heuristical approaches e.g. AUTHORPAPER and AUTHORPAPER

The idea was to find a heuristical approach, which would be able to provide an approach specifically useful to the problem of scheduling AGVs in the circular system. For this reason the local search meta-heuristic was chosen as an underlying approach. Local search has a great advantage: Defining the neighborhood of a scheduling problem is relatively easy. In this thesis the neighborhood is defined by all possibilities to move the starting time of an AGV by a single unit. For any given amount of vehicles the amount of possible moves inside the neighborhood is therefore two times m with m being the number of machines, since every machine can start a unit of time earlier or later. In the event of a machine starting at the last possible starting point, starting right at the beginning of the time window is considered to be a neighbour and vice versa. This basic outline of the local search provided the corner stone for the algorithm finally implemented.

The basic implementation described above however proved to be too simple as problems with the multidimensionality of the objective function required a more specialized approach with an algorithm tailored to distinguish better between the quality of received solutions. In the original problem the goal was solely to minimize the amount of AGVs needed to finish on time. If however the problem is changed to have two objectives, being the minimization of the makespan and the minimization of the amount of vehicles needed, a new problem arises: Because a higher amount of machines will generally result in a lower makespan, the solutions gained from different neighbourhoods are now ineligible for unreflected comparison.

Because of this problem a new tool, developed for multidimensional objective functions, is introduced: A pareto frontier. AUTHOR2 wrote about the idea of a pareto frontier extensively and showing that it offered the possibility of displaying the optima of a multidimensional problem and allowed for the choice of a preferred solution through implementation of an additional objective function (e.g. a cost function of the two dimensions displayed) or simply by personal preference. The pareto frontier was chosen as an instrument for this thesis because of a reason already touched previously: Due to the nature of scheduling problems increasing the number of AGVs available to transport the jobs between the machines will in its optimal configuration always yield a makespan that is better or as a worst case equal to the makespan received when calculating with one less AGV. This can be proven by finding the optimal solution for a given amount of AGVs n and then increasing the amount. All AGVs are assigned exactly the same starting time as the previously

attained optimal solution for n AGVs. The newly added AGV can be freely allocated to any starting time. This results in all jobs being picked up either at the exact same time or, due to the newly added AGV, earlier than previously. Therefore the total makespan will in the worst case stay the same(e.g. the newly added machine does not transport any job in the system or the earlier transport of a job simply results in longer time in queue due to the processing time of the previous job)

In this thesis the received pareto frontier will not be filtered through the introduction of a cost function, as this simply picks one of the solutions from the list and is highly subjective. Instead a comparison will be made for each amount of machines up to a set limit between the heuristical solution and an optimal solution obtained through a Brute Force Algorithm. The difference in run time and target value will then be used to determine whether the heuristic offers a significant reduction in the first, while maintaining only a slight reduction in the latter.

5. Implementation

Before we are able to try and approach the problem with a local search algorithm, the local search itself has to be defined more clearly. As stated in the previous chapter, the idea is to vary the amount of machines available and find optima for all of those cases to be able to display a pareto frontier. To be able to achieve this it is necessary to adjust the original local search, to be able to distinguish between solutions achieved with varying amounts of AGVs, as the solutions received for the maximum makespan with a given amount of AGVs cannot be directly compared. To circumvent this problem the local search approach is refined to iteratively increase the amount of AGVs, while not changing the amount during an iteration. This new iterative local search consists of a set amount of "local searches"which equals the maximum amount of AGVs permitted. In each iteration the amount of AGVs is increased by a single unit, while staying constant throughout. This approach will return a single solution for each amount of AGVs, which is then displayed in a pareto frontier.

For the actual implementation of the model a few parameters have to be set. These include the number of iterations that the local search is going to run for per amount of machines, the maximum amount of machines and of course all the parameters set by a specific test environment have to be saved, namely the amount of production stages, the production machines for each stage, the amount of jobs, the lap time of an AGV, the time needed by an AGV to traverse between machines, the schedule of jobs on the production machines and of course the process time for each job on each machine. As noted in a previous chapter, the only input variable is the actual starting time for each of the AGVs.

To venture a bit further into the comparability of the heuristic, this paper introduces multiple variations working on the same base construct. The main difference is in the pertubation step which happens after each iteration that has ßtrandedïn a local optimum. The base version was a simple random jump. The starting time of every AGV, except for AGVs starting at 0, is set to a random starting point and the neighbourhood search starts anew. Before each of these pertubations the currently reached solution is compared to the best so far reached solution for the given amount of AGVs and is saved if it is an improvement. Different from this probabilistic approach, the second variation of the heuristic set out to be deterministic, except for the generation of the starting solution. The idea behind the differences in pertubations is to be able to achieve comparability between efficiency and reliability. The deterministic approach sets all starting times of AGVs not starting at 0 to be starting a set amount of units of time early(if the starting time would be negative, the time is instead counted down starting from the last possible start). As local searches with deterministic pertubations generally suffer from the same problem, namely the pertubation either being too high, resulting in almost random jumps, or being too low, resulting in getting stuck at a local optimum not being able to leave, the heuristic is run multiple times with varying pertubation lengths.

6. Mathematical Formulation

In this section the paper will go into depth as to how the problem was approached from a mathematical perspective. To be able to approach the problem with a programmed algorithm, some definitions have to be made beforehand. As described closer in previous chapters, the general idea is that AGVs are released onto a track, where they will run in a neverending circle. Upon reaching a machine or the depot the currently loaded job is unloaded and the job that is scheduled to be processed the earliest on the next machine is selected from the jobs waiting in the outgoing queue at the machine. Jobs unloaded at a machine are placed in the incoming queue. Note that this queue does not have a real order as the schedule of jobs on the machines is given, meaning that even though a job might have been the first to arrive at a machine, it might not get processed at said machine until after some other jobs have been. The objective of the program is to minimize the maximum completion time for a given amount of AGVs V. The objective function can therefore be depicted as:

$$\min C_{max} \tag{6.1}$$

For this formulation a relatively long list of constraints has to be set, which will now be explained in detail. Constraint (1) is set to make sure that every available vehicle has to start before a full lap could have been made, since if a vehicle was to start at a later point in time it would simply miss a full lap that could have been achieved by letting the vehicle start roundtime R units of time earlier.

$s.t.:$

$$\sum_{t=0}^{R} x_{vt} = 1 \ldots \forall v. \tag{6.2}$$

$$x_{vt} \in \{0, 1\} \tag{6.3}$$

Teil I.

Resümee

7. Zusammenfassung

8. Ausblick

Literatur

Verzeichnisse

Abbildungen

Tabellen

Anhang

Anhang A.

Erster Teil vom Anhang

Anhang B.

Zweiter Teil vom Anhang

Anhang C.

Oben beginnende Überschrift, falls nötig