Reducing the runtime of an NP-Hard algorithm using deep learning on historical data

Christoffer Lindkvist

September 10, 2025

Abstract

SILKSONG SOON SHAW

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

This thesis is an extension to the Volvo Truck Assembly Line problem (?); Today trucks are placed manually by management workers based solely on their own knowledge, though this is not written down anywhere. The algorithm in the works will using data from Volvo help place the trucks so that there are as few overlaps as possible. My idea is that the algorithm can gain a faster runtime by defaulting to ”safe” combinations which are already used today.

1.2 Research Problem

1.3 Objectives

1.4 How can one intuitively visualize this problem?

1.4.1 UI/UX Problem

The UI will visualize the flow of the assembly line on two axes. One per station, and one in clockcycles.

One clockcycle is the time it takes the theoretical items to make it from one station to the next. Hence the items must be displayed in a way that conveys that some stations take longer and shorter time to complete.

Issues start to appearing when we start to consider that different stations S_n and S_m may take different times to complete. If we then step a clockcycle, our arbitrary unit of time, then we can never keep our items in sync. The main issue is that if we compare the station S_n and S_m we'll see that each station have a different time to finish, then the clockcycle system will not be perfect or even realistic as stations with differing times will each finish in different times and thus an item n might make it to the station S_{n+2} from S_n in the same time it takes item m to make it to the same station as S_{m+1}.

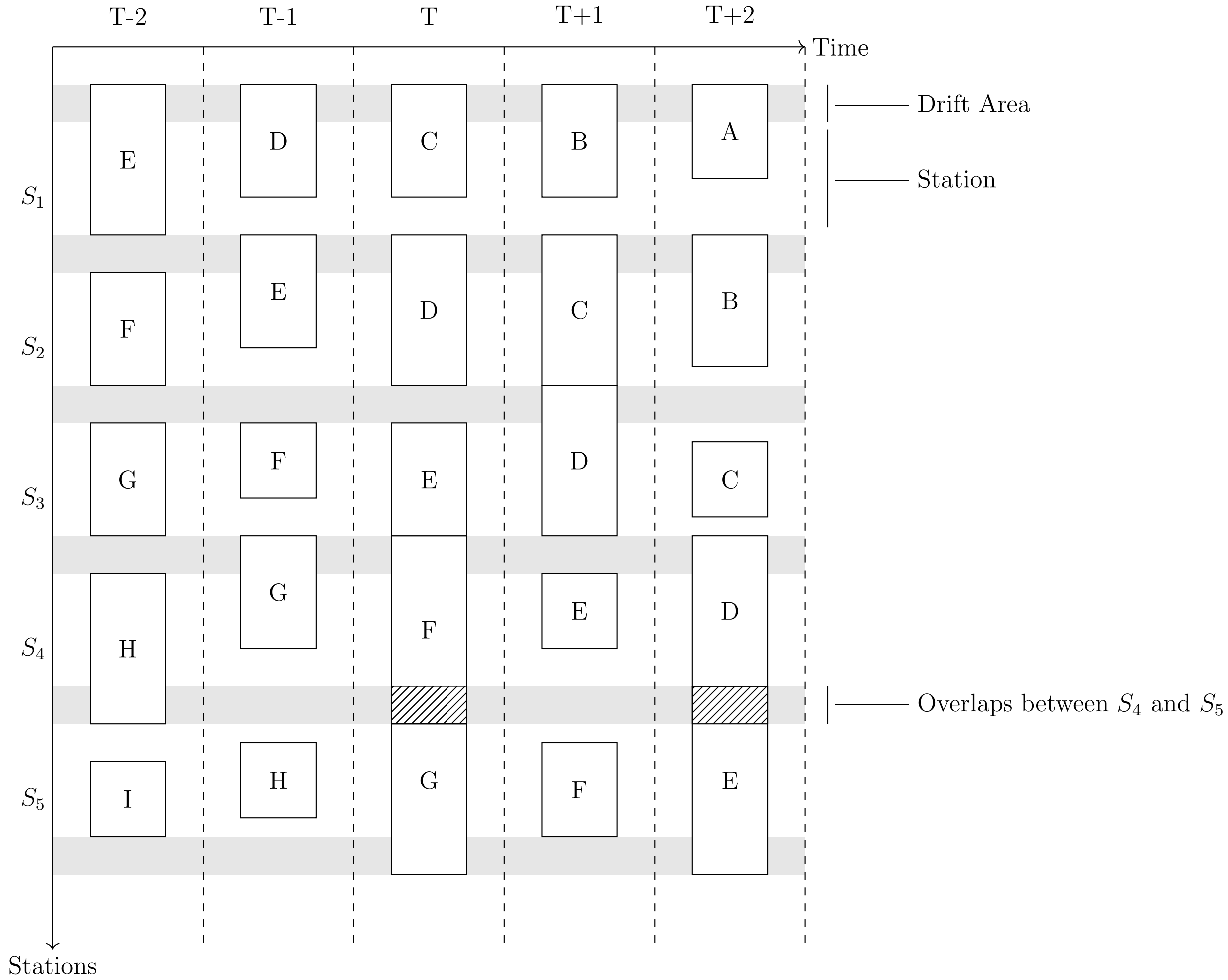

Figure 1.1: Assembly Line Example with Uniform Station and clockcycles

Different items will take a differing amount of time to finish each station. Thus the difficulty in displaying it properly in a UI/UX. If we wish to display each station as uniform sizes, then we also have to stretch the items to make up for it visually. But doing this we have no intuitive way of knowing that S_4 could be 20 seconds long in real life, while S_3 could be 90.

As pictured in figure 1.1, D will take a lot of time on S_4 and is forced to utilize some time from S_3 and S_4, which works well in a vacuum. The problems start to arise when E also has to utilize some additional time from its previous stations, causing an overlap between D and E at $T + 2$.

The same problem arises with F and G at T as both items need to borrow time from the stations before and after. Thus we run into another overlap.

Do note that on T E does not utilize the drift area which results in it sitting flush with F on the timeline, this may look good on paper but can result in overlap in practice due to the humans at the assembly line occasionally taking a bit longer than presumed. This can be resolved by moving E into the drift area at the end of S_2.

The same issue arises at $T + 1$ where C and D just barely get enough time, but it cannot get resolved by simply moving D forward, as D on $T + 2$ will require all time it can get on S_4.

1.5 The Algorithm

1.5.1 The Heuristic Approach

The problem to properly order manufacturing assembly lines with as few overlaps as possible is an NP-Hard problem.

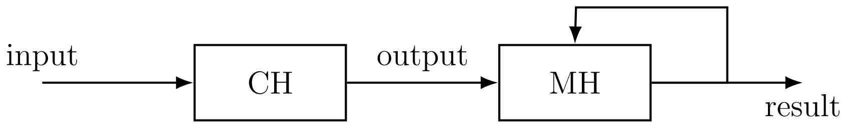

Figure 1.2: Heuristic solution

The Algorithm designed to solve this problem is a heuristic solution that will be made out of a Construction Hieuristic (*CH*) that produces a starting point based on pre-defined constraints, that feeds into a Meta Heuristic (*MH*) that finds a better solution starting from the output of the construction heuristic and self-improving until an acceptable result is returned.

1.5.2 Complementing the Heuristic Approach using Machine Learning

Due to the fact that servicemen today place the items manually using unwritten knowledge (tacit) that they have accumulated over the years.

The idea is that if they have knowledge of a good enough solution from the get-go with some risk of overlap, then we can train a Deep Learning Model (*ML*) on such previous data to give the algorithm a better starting point, thus reducing the runtime of that algorithm.

However it is worth to consider that such an approach can prove redundant or yield worse results if the problem at hand is an easy problem where many solutions can be found quickly, as opposed to a hard problem where a desired solutions may not even be found.

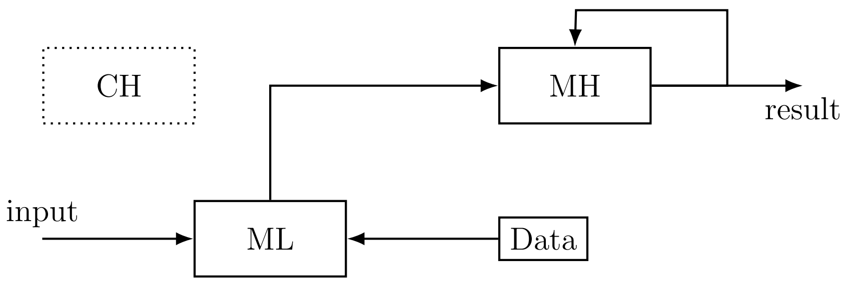

Figure 1.3: ML solution

Chapter 2

State of the art analysis

2.1 LSTM

In previous papers attempting to remedy this problem the methodology used was a Recurring Neural Network (RNN) utilizing Long Short-term Memory (LSTM) [2] Due to the fact that we wish to emulate tacit knowledge that the servicemen use to manually place the items in a ”good enough” fashion.

The LSTM model is unsupervised and thus

Chapter 3

Literature Review

3.1 Theoretical Framework

3.2 Previous Work

3.3 Research Gaps

Chapter 4

Machine Learning

4.1 Why Machine Learning

4.2 Data Collection

4.3 Data Analysis

Chapter 5

Results

5.1 Findings

5.2 Data Presentation

Chapter 6

Discussion

6.1 Interpretation of Results

6.2 Comparison with Literature

6.3 Implications

Chapter 7

Conclusion

7.1 Summary of Findings

7.2 Limitations

7.3 Future Work

Bibliography

[1] J. Abbasi, *Predictive Maintenance in Industrial Machinery using Machine Learning*, Master's thesis, Luleå University of Technology, Department of Computer Science, Electrical and Space Engineering, 2021.

[2] A. Dupuis, C. Dadouchi, and B. Agard, *A decision support system for sequencing production in the manufacturing industry*, *Computers & Industrial Engineering*, vol. 185, p. 109686, 2023. doi: 10.1016/j.cie.2023.109686.

[3] Author, Title, Journal, Year.