SHUNTING FREIGHT CARS WITH OWN POWER UNITS

Elias Dahlhaus[1]

In this paper it is shown that shunting freight cars can be simplified significantly if the freight cars have their own power unit. Freight with own power units are extensively discussed in the project FlexCargoRail.

Key words: FlexCargoRail, merge sort, permutation graph coloring

1 Introduction

In future freight cars will get their own power devices. Still freight cars will be drawn by locomotives, when they are operated on the main lines. But for the first or last mile they will be moved by their own devices and controlled by an automatic system. There is a project dealing with this topic, called FlexCargoRail (see [1] and [5]).

Moreover, the shunting process can be simplified when changing the train. There will be no need of a hump to speed up the shunting process. In this paper, it is shown that further additional possibilities are opened to get trains sorted. The essential advantage is that it is possible to merge several sorted sequences of cars to one sorted sequence in one sorting step. This is not possible in traditional hump yards.

Just at the last main rail yard, it is necessary to sort the cars in a certain sequence. For example in the simple case they have their final destination stations on one line, the have to be sorted in reverse order to their final destination stations. The cars are still moved to their destination stations by a locomotive. Only the last mile from their final destination stations to their final positions (e.g. a factory or a loading platform) they move by themselves. Consider the following example.

The last main rail yard is Munich North. Cars have to be delivered in Mühldorf, Garching, Hörpolding, and Traunstein. The train visits the stations in exactly this sequence. Then cars with destination Mühldorf have to be put to the rear of the train. Cars with destination Garching appear as the second last block, cars to Hörpolding as the second block, and cars to Traunstein as the first block. In Hörpolding, there is a branch to an industrial area of Traunreuth. The last mile from Hörpolding to Traunreuth, the cars will move by themselves.

In chapter 2 it is shown how to merge several sorted sequences to one sequence. This is a standard technique used in algorithm design.

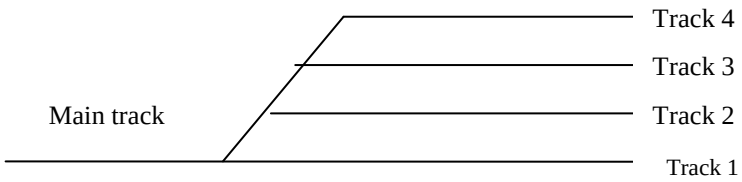
In chapter 3 it is shown how we can sort an unsorted sequence in one sorting step. The number of tracks is minimized. It will be seen that the problem is the same as that of colouring so called permutation graphs.

In chapter 4 the sorting process is presented for the case that the number of tracks is restricted.

[1] Dr. Elias Dahlhaus, Darmstadt University of Technology and DB Systel, , tel.: +49-6151-669852, E-mail: dahlhaus@algo.informatik.tu-darmstadt.de

2 Merging several sorted trains to one

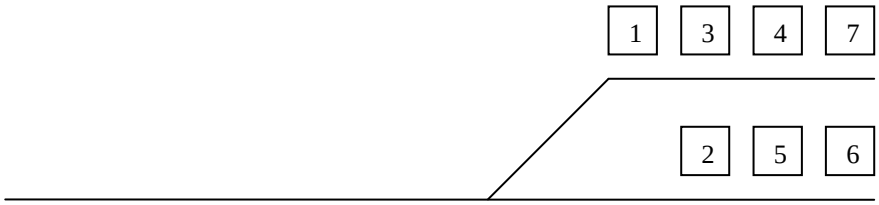
Train 1 arrives in track 1, train 2 arrives in track 2, train 3 arrives in track 3 etc. Track 1, track 2,…..,track n are parallel tracks that merge to one main track.



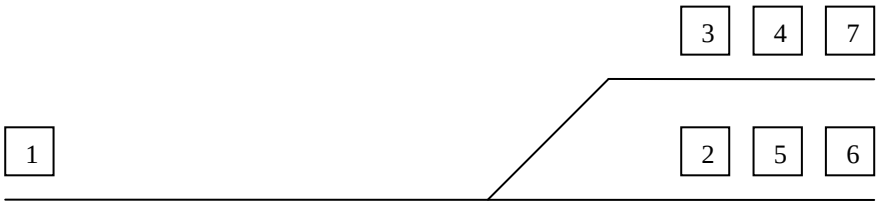
After arrival all cars are uncoupled and are able to move by themselves. The position of each car in the merged train is fixed. We number the cars by the position in the merged train. The cars of each train appear in sorted order.

The procedure to merge sorted sequences to one sorted sequence is applied as described for example in [2]. Repeatedly the cars on the tops of track 1,…, track n are considered and the one with the lowest number is selected and moves to the main track.

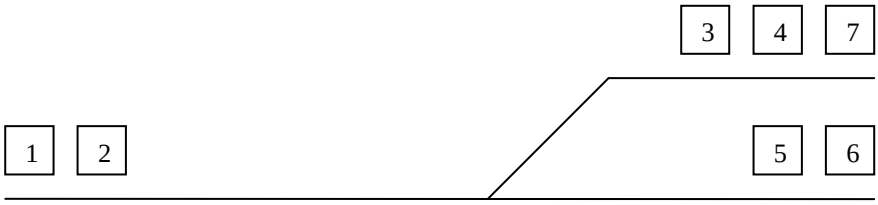
In the following example train 1 and train 2 arrive at a rail yard and the numbering is as in the following figure.



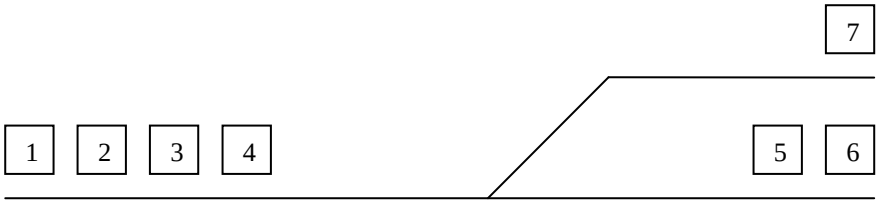
First, car number 1 is moved to the main track.



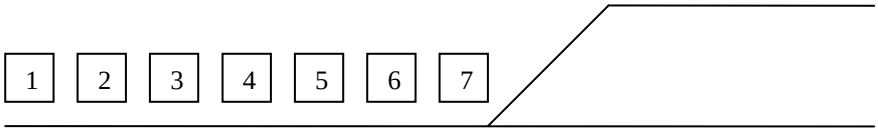
Then car number 2 is the car on the top with lowest number and is moved to the main track.



Then 3 and 4 are moved to the main track.



Finally, all cars are in sorted order.



One merging step is enough to get sorted trains merged into one sorted train.

To get the cars sorted by a humping procedure, much more steps or tracks are necessary. We assume that the first train with the cars 2, 5, and 6 is humped first and the first train with the numbers 1,3,4, and 7 is humped secondly. Then it is the same as if the sequence 1,3,4,7,2,5,6 is humped. The minimum number of tracks that are needed to sort the sequence in one step is determined by the ‚inverse sequence' (the i-th element of the inverse sequence is the position of i in the original sequence). In this example, the inverse sequence is 1,5,2,3,6,7,4. The number of tracks is the number of intervals of the inverse permutation that are monotonically increasing (see for example [3]). In this example they are the intervals (1,5), (2,3,6,7), and (4). In this example three tracks are needed. It is possible to get examples of two sorted trains that need any number of tracks to get sorted by a humping procedure.

3 Sorting an unsorted sequence in one step

Here a permutation representing the sequence of cars as they arrive is given, and it has to be sorted, e.g. (4,2,5,1,6,3) is given and has to be transformed into (1,2,3,4,5,6). One proceeds in the following way (compare also [4]).

- The permutation is split into a minimum number of monotone subsequences. In this example, they are (4,5,6), (2,3), and (1).

- For each monotone subsequence, we provide a track, and the cars belonging to a particular monotone subsequence, are put to the corresponding track. Note that the cars at each track are in sorted order. In above example, cars 4,5, and 6 are put to track 1, cars 2 and 3 to track 2, and car 1 to track 3.
- The cars are merged as in chapter 2 to one sorted sequence.

It remains to demonstrate how to split a permutation into a minimum number of monotone sequences. (see [4]).

Assume, an initial segment of the permutation has been split into a minimum number of monotone subsequences (e.g. (4,2,5,1) has been split into (4,5),(2), and (1)) and the next element is added (in this example, it is 6). If there is a monotone subsequence with top element smaller than the next element, one selects such a monotone subsequence with the larges top element and adds the next element to this subsequence. In this example, one selects the subsequence (4,5) that becomes (4,5,6) after 6 is added. If there is no such monotone subsequence, a new monotone subsequence is opened. It consists of the next element.

For above example, the whole procedure is done as follows.

- Initially, one has the monotone subsequence (4).
- When 2 is added, the new subsequence (2) is opened, i.e. there are the monotone subsequences (2) and (4).
- When 5 is added, it is added to the monotone subsequence (4) with the highest top element 4, i.e. there are the monotone subsequences (2) and (4,5).
- When 1 is added, a new subsequence (1) is opened, i.e. there are the monotone subsequences (1), (2), and (4,5).
- Next 6 is added. Here, all tops of monotone subsequences are smaller than 6. One selects the subsequence (4,5) with the largest top 5. This sequence becomes (4,5,6), i.e. there are the subsequences (1), (2), and (4,5,6).
- Next 3 is added. The top of (4,5,6) is 6. Therefore 3 cannot be appended to (4,5,6). But it could be appended to (1) or (2). Since (2) is the one with the larger top element, 3 is appended to (2). We finally have the monotone subsequences (1), (2,3), and (4,5,6).

4 Number of tracks is restricted

Finally the realistic case is discussed that the number of tracks is less than the number of monotone subsequences of the permutation of cars that has to be sorted. First it should be remarked that, in general, merging two sequences that can be split into two monotone subsequences cannot be merged to one sequence that can be split into two monotone subsequences. It has to be proceeded as follows.

- It is taken care that the cars of the same monotone subsequence appear in one track consecutively (monotone subsequence separation).
- The separated monotone subsequences are merged.

4.1 Separation of monotone subsequences

We have split the initial permutation into N monotone subsequences and n parallel tracks are available. The first N/n monotone subsequences are put into track 1, the second N/n monotone subsequences are put into track 2 and so on. We concatenate the sequences of the parallel tracks to one sequence. This sequence consists of n intervals of sequences that can be split into N/n monotone subsequences. For each of these intervals, the first N/n^2 monotone subsequences are put into track 1, the second N/n^2 monotone subsequences are put to track 2 and so on. Concatenating the cars of track 1 with the cars of track 2 and so on, one gets a sequence consisting of n^2 intervals with N/n^2 monotone subsequences. Continuing the procedure, one gets, after $\log_n N$ steps N monotone intervals. At each track, there are about N/n monotone intervals.

4.2 Merging the monotone intervals

One merges the first monotone intervals of each of the parallel tracks, the second monotone intervals of each of the parallel tracks and so on. The number of monotone intervals becomes about N/n. We put an almost equal number of monotone intervals into each of the n tracks. We again merge the first monotone intervals of all the n tracks then the second monotone intervals of all the n tracks. The number of monotone intervals is again divided by n. After $\log_n$ steps, the whole sequence consists of one monotone interval and is therefore sorted.

As an overall consequence, we can sort a sequence that can be split into N monotone subsequences by $2 \log_n$ back-and-forth movements.

5 Conclusion

It has been shown that known procedures from algorithm theory and graph theory can be used to sort freight cars, provided they have their own power unit and they act as own vehicles. Section 4.2 is only a minor new mathematical contribution. Less resources (tracks and back-and-forth movements) are necessary compared to hump yards.

Reference literature

1. BAIER, M., ENNING, M, FlexCargoRail, ein Fahrzeugsystem für effizienten Einzelwagenverkehr, *Logistik Management 3 (2008).*
2. CORMEN, T.H., LEISERSON, C.E, RIVEST, R.L., Introduction to Algorithms, McGraw-Hill, 1990.
3. DAHLHAUS, Elias, How to get as many cars sorted as possible in one humping step, *ZEL 2008*
4. GOLUMBIC, M.C., Algorithmic Graph Theory and Perfect Graphs, Academic Press 1980.
5. STUHR, Helge, DIEKENBROK, Björn, Einsatzszenarien selbst angetriebener Zustell- und Sortiervorgänge, *Eisenbahntechnische Rundschau 5* pp. 230-233.