A graph theoretical approach to shunting problems

L. Koci, G. Di Stefano

Dipartimento di Ingegneria Elettrica,

Università dell’Aquila, Italy

AMORE meeting, 1-4 October, Leiden, Holland

Train depot algorithms

- Instance

Outcoming trains ← Depot ← Incoming trains

Train depot algorithms

- Goal
  - Arrange the trains on a minimum number of depot tracks
  - Put the trains in a “correct” order to minimize shunting operations

Train depot algorithms

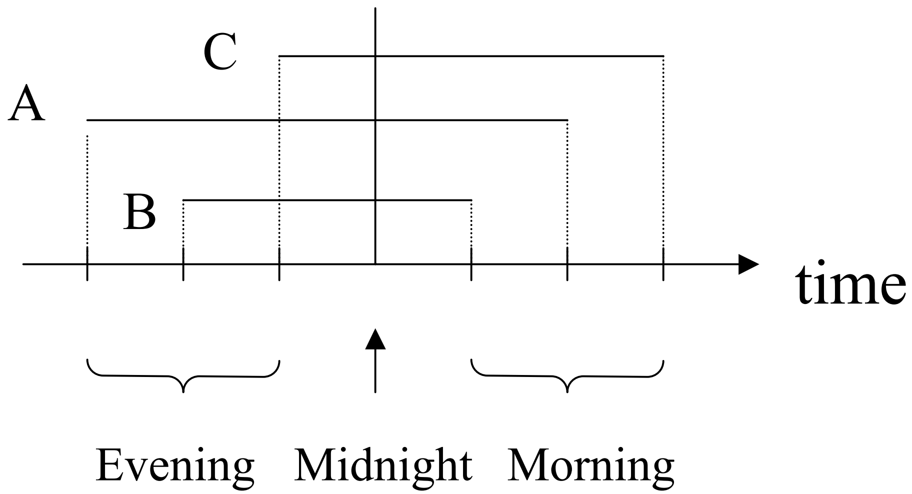
- Constraints
  - Trains
    - sequences, types, lengths
  - Tracks
    - topologies, lengths

Train depot algorithms

- Methods
  - Combinatorics
  - Graph theory (reduction to graph problems)
  - Heuristics

Train scheduling

- Train arrival and departure time

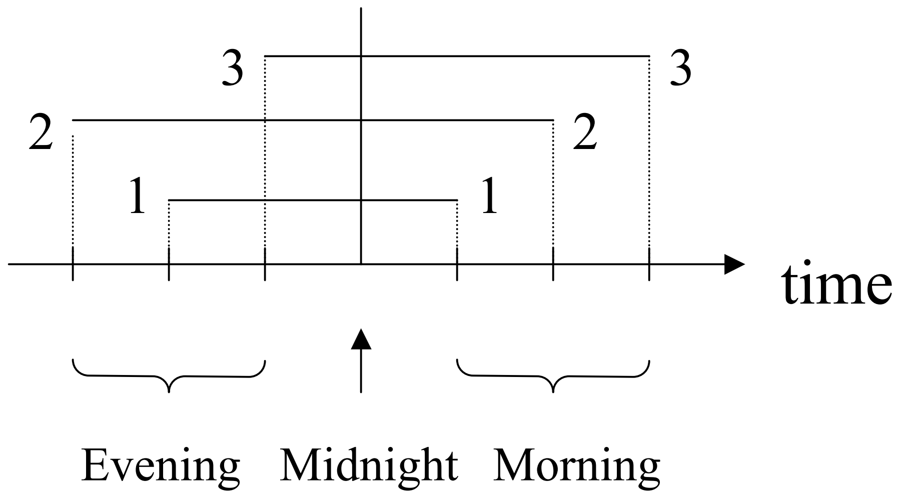


A: first *incoming train*
B: second
C: third

B: first *outcoming train*
A: second
C: third

Train numbering

- Train arrival and departure time



2: first *incoming train*
1: second
3: third

1: first *outcoming train*
2: second
3: third

Train depot algorithms

Train assignment

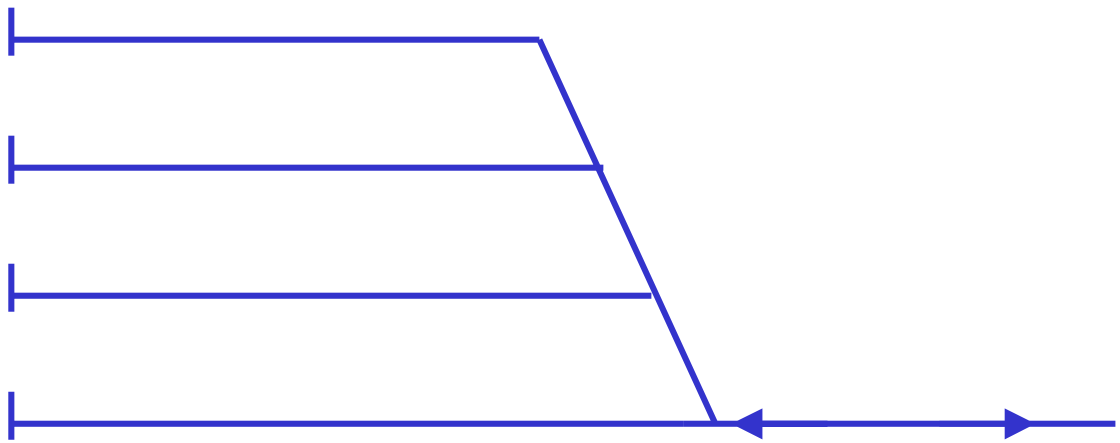
- Trains are numbered from 1 to N
- Incoming train permutation $\pi = [\pi_1 , \pi_2 \ldots \pi_N]$
  - Each train π_i is represented by an integer
- Outgoing train sequence S = [1, 2, ... N]

$$[1, 2, \ldots N] \longleftarrow \text{Depot} \longleftarrow [\pi_1 , \pi_2 \ldots \pi_N]$$

ex. [3, 7, …, 5]

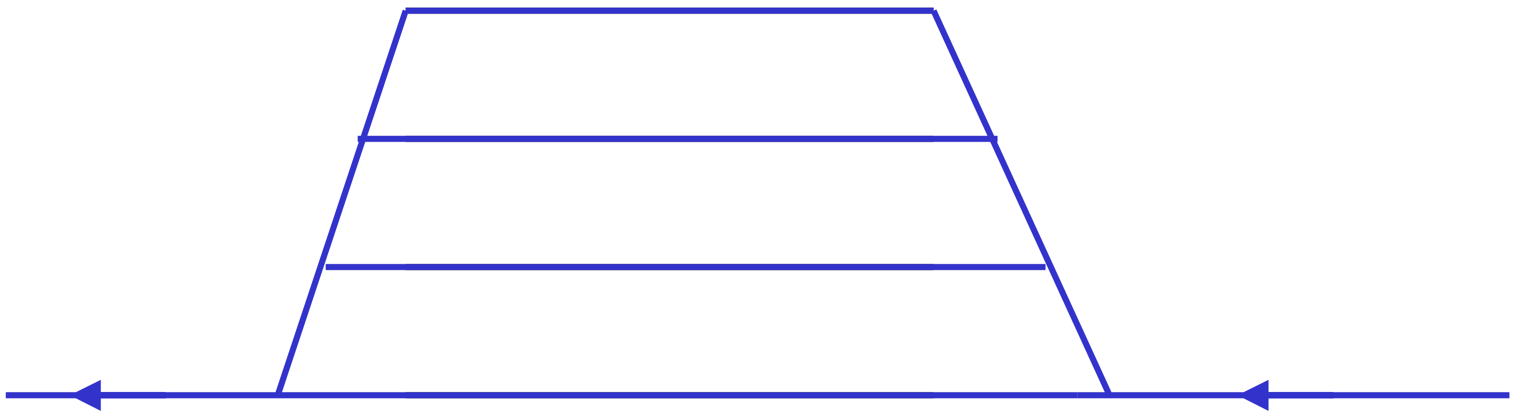
Depot topologies

- *Shunting area*



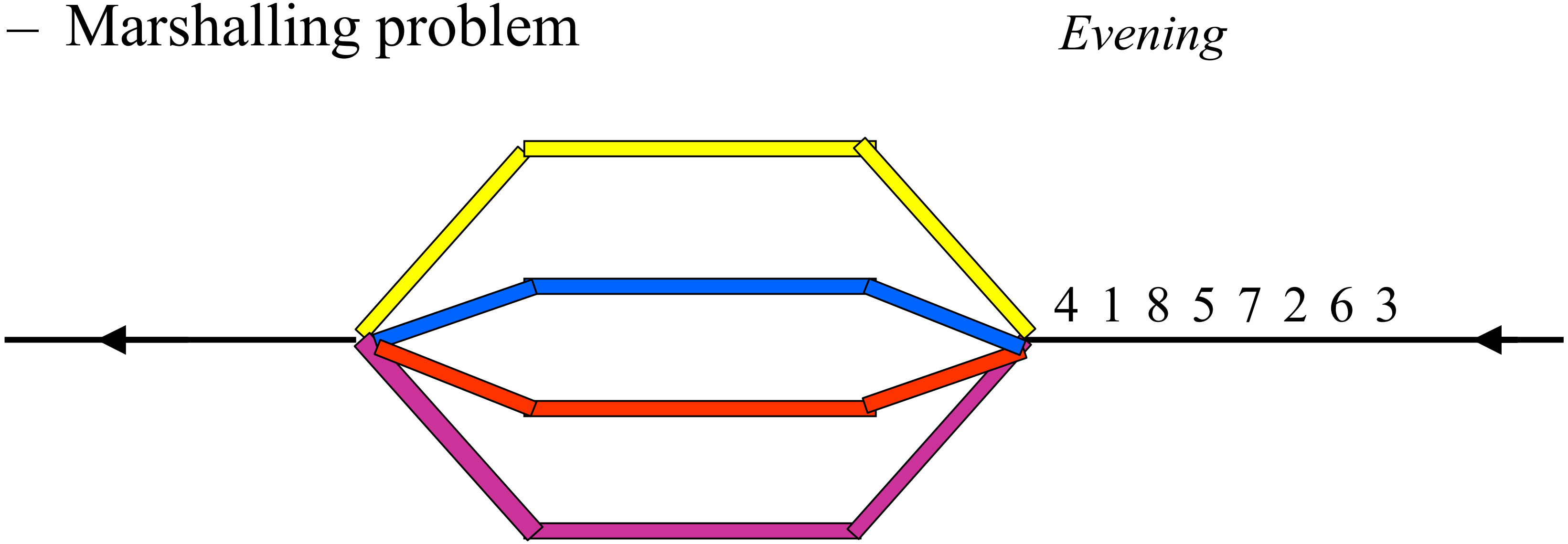
Depot topologies

- *Marshalling area*



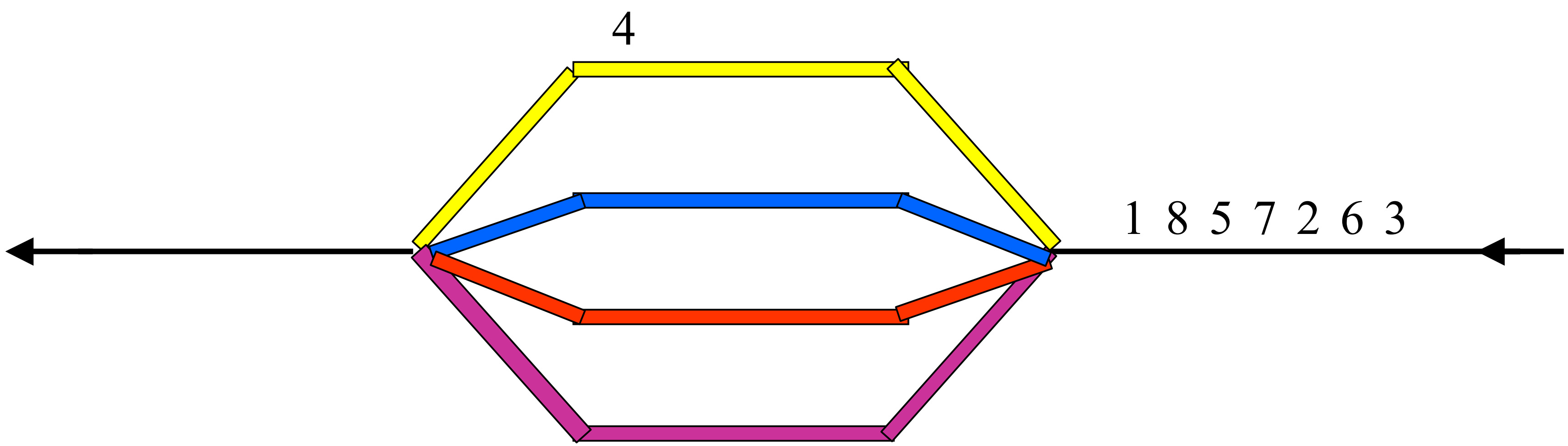
Train depot algorithms

- Train assignment
  - Marshalling problem



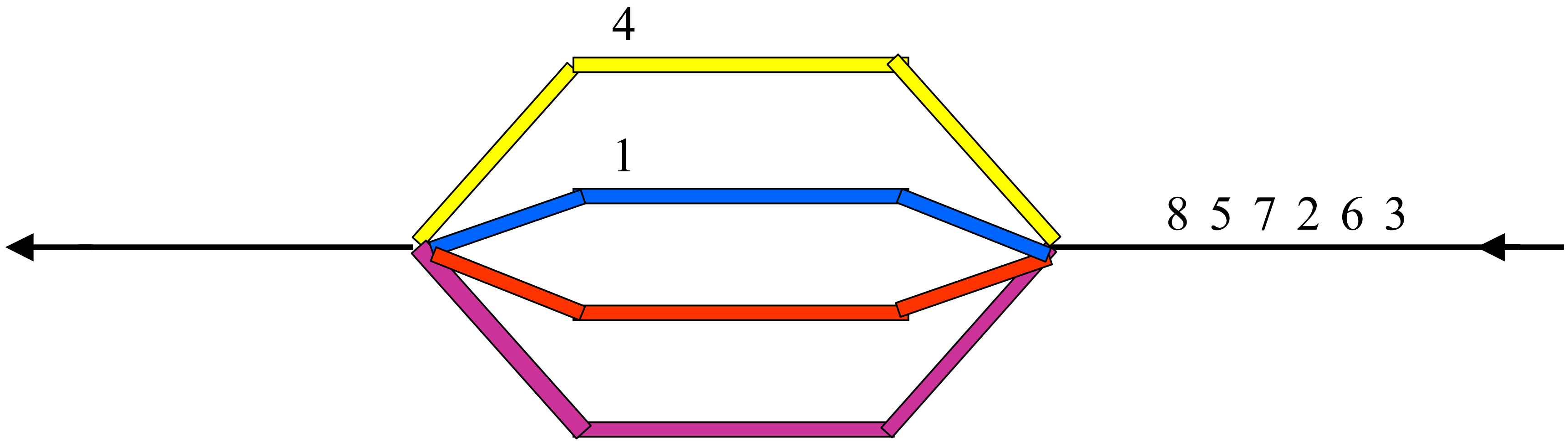
Train depot algorithms

- Train assignment
  - Marshalling problem



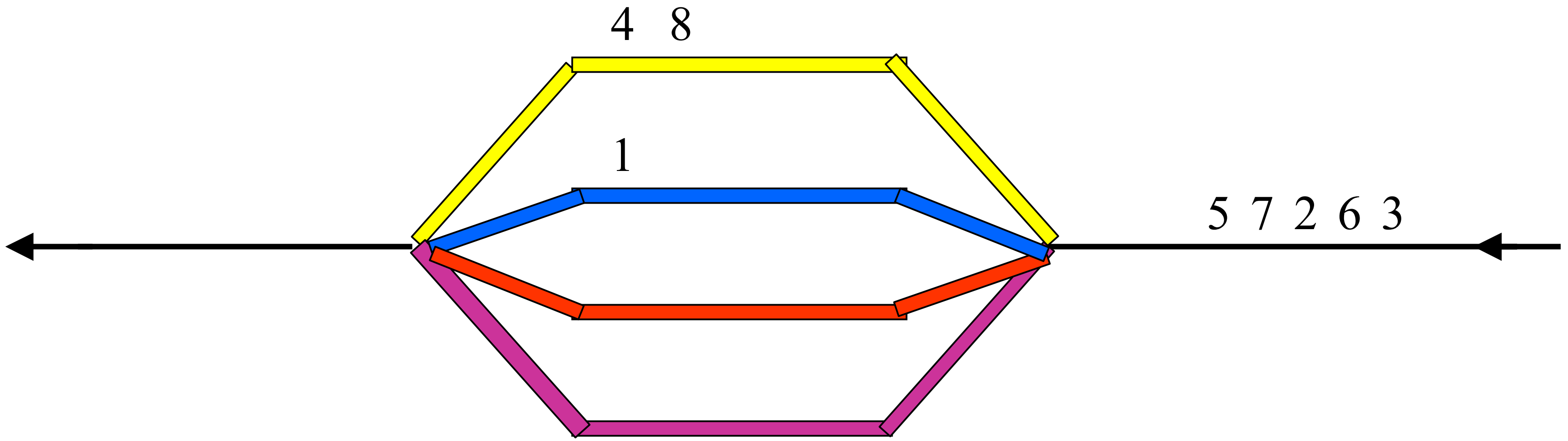
Train depot algorithms

- Train assignment
  - Marshalling problem



Train depot algorithms

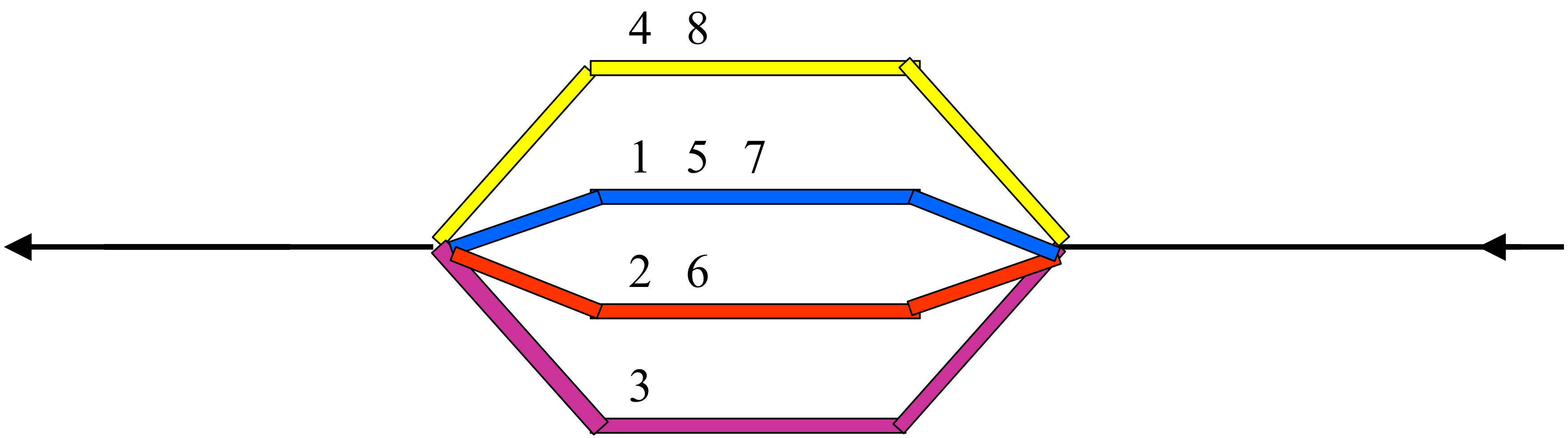
- Train assignment
  - Marshalling problem



Train depot algorithms

- Train assignment
  - Marshalling problem

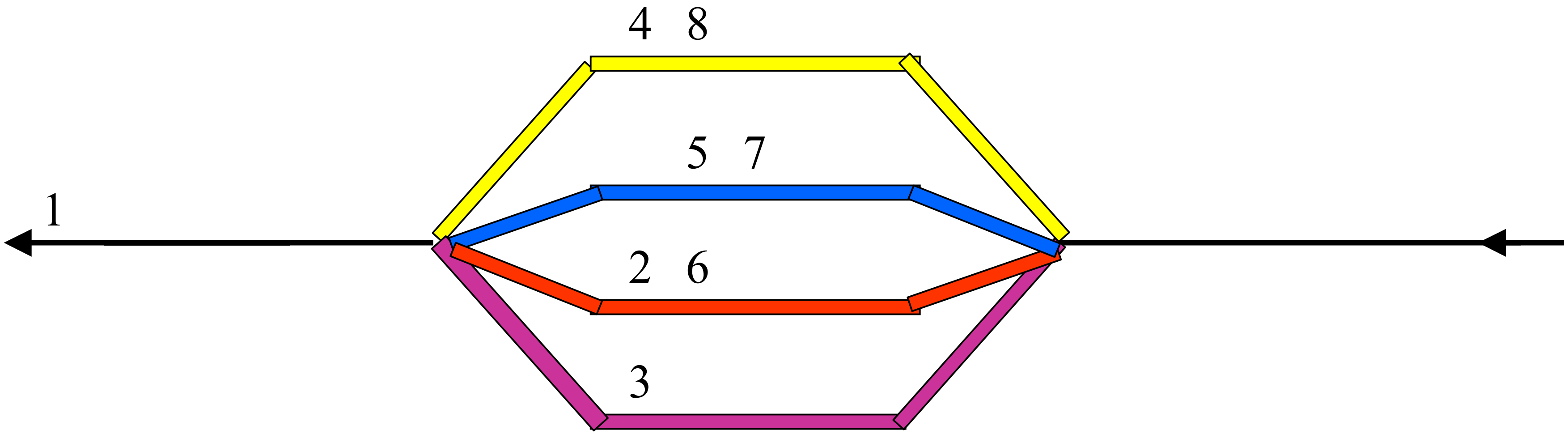
Night



Train depot algorithms

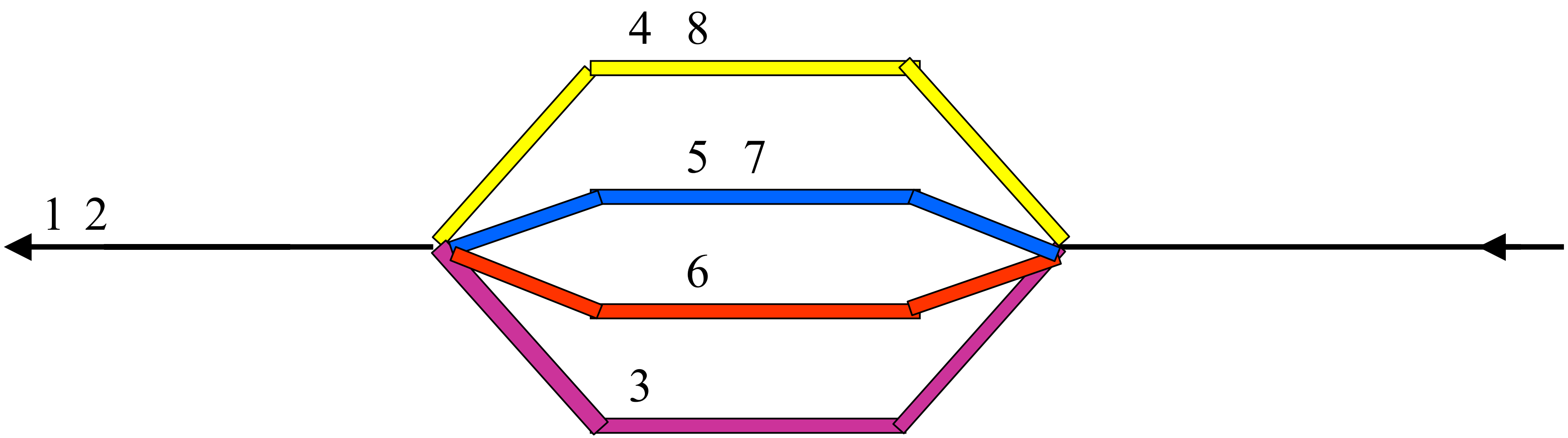
- Train assignment
  - Marshalling problem

Morning



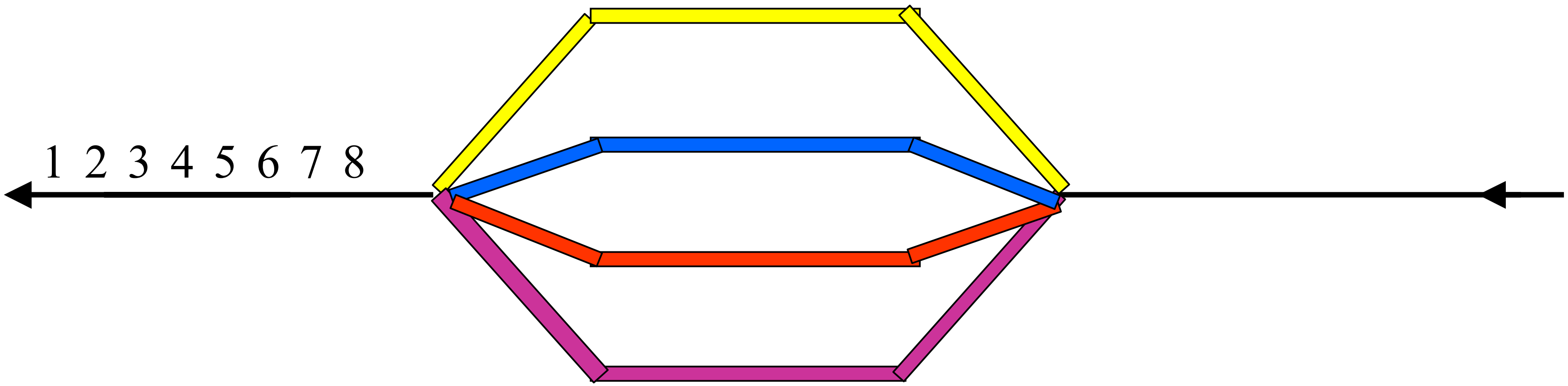
Train depot algorithms

- Train assignment
  - Marshalling problem



Train depot algorithms

- Train assignment
  - Marshalling problem

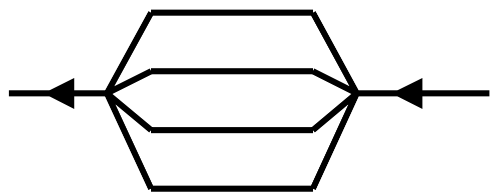


Ordering problem (1)

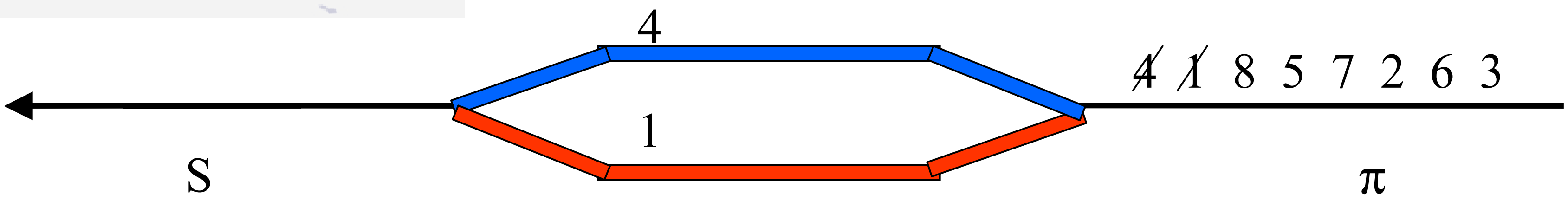
The *storage* of *N* trains in a *marshalling depot*
using the minimum number of tracks

is equivalent to

The *ordering* of a sequence of N numbers
using the minimun number of *queues*



Ordering problem (1)



- Train assignment

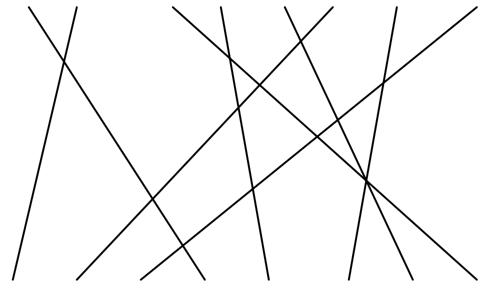
$$\pi = [\,4 \quad 1 \quad 8 \quad 5 \quad 7 \quad 2 \quad 6 \quad 3\,]$$

$$S = [\,1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8\,]$$

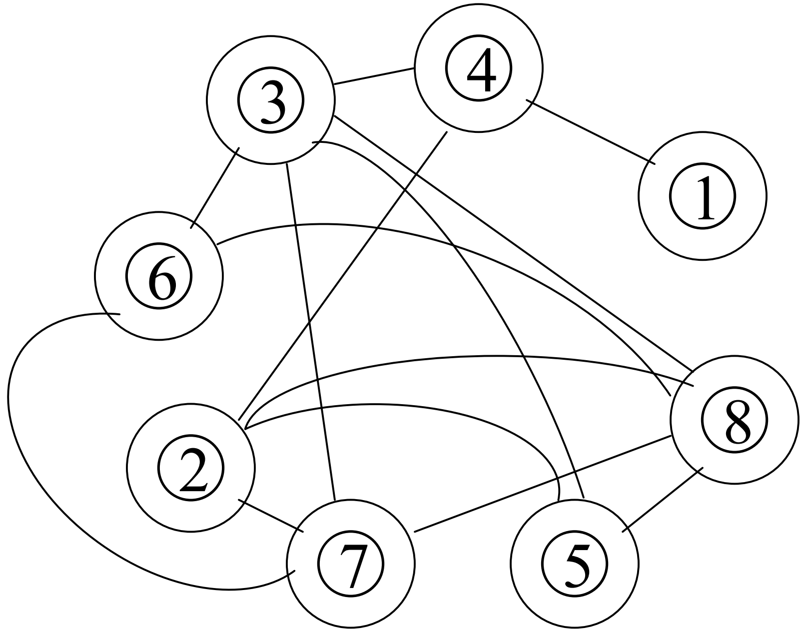
Graph equivalence

- Train assignment

$\pi = [\,4\ \ 1\ \ 8\ \ 5\ \ 7\ \ 2\ \ 6\ \ 3\,]$



$S = [\,1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 6\ \ 7\ \ 8\,]$



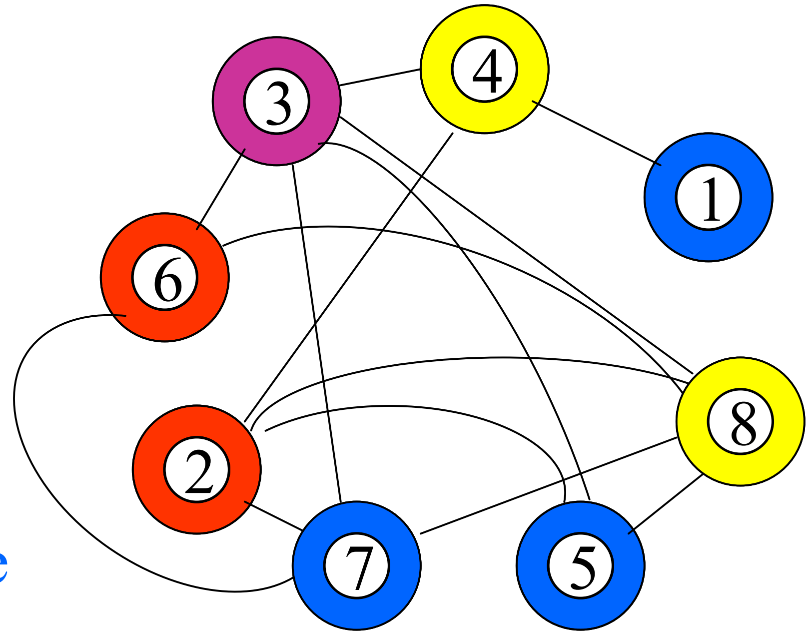
Permutation graph

Ordering problem $\equiv$ Permutation graph coloring

Colouring solution

$\pi = [\ 4,\ 1,\ 8,\ 5,\ 7,\ 2,\ 6,\ 3\ ]$

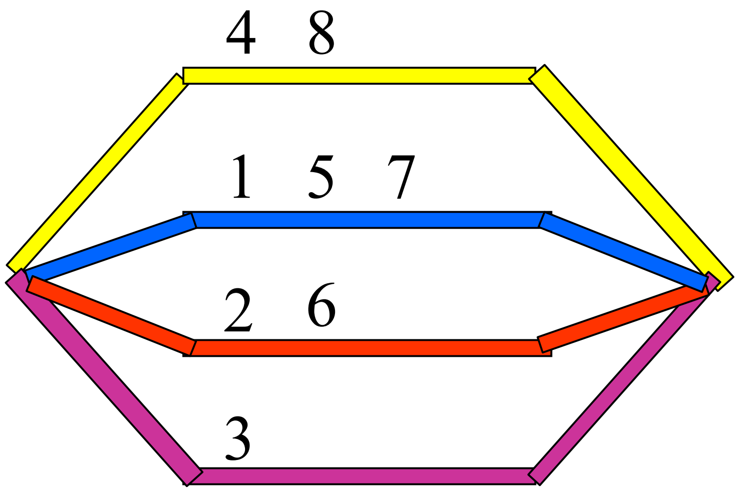
- Minimum colouring (colour = track) of a general graph is NP-complete
- Minimum colouring of a *permutation graph* is solved in $O(n \lg n)$ time



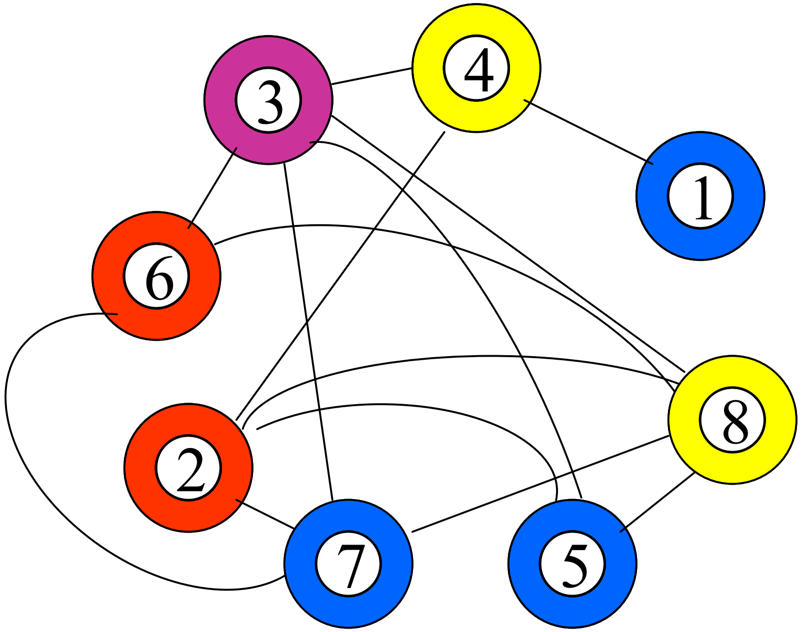
Colouring solution

$\pi = [\ 4,\ 1,\ 8,\ 5,\ 7,\ 2,\ 6,\ 3\ ]$

- Train assignment



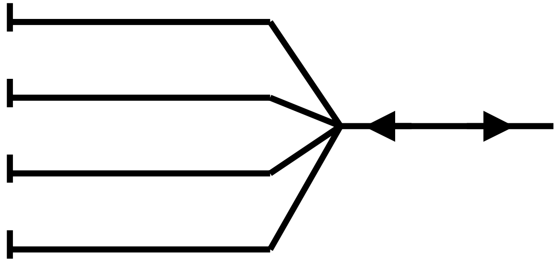
Depot



Permutation graph

Ordering problem (2)

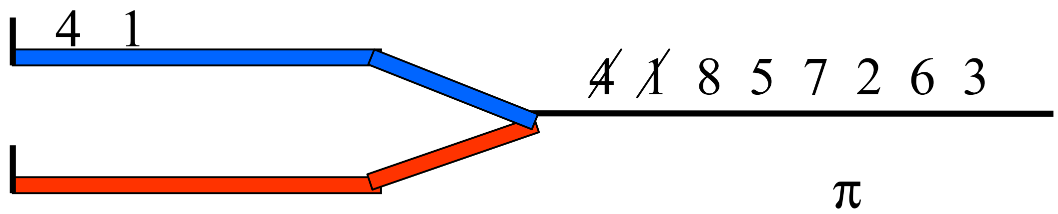
The storage of *N* trains in a *shunting depot* using the minimum number of tracks



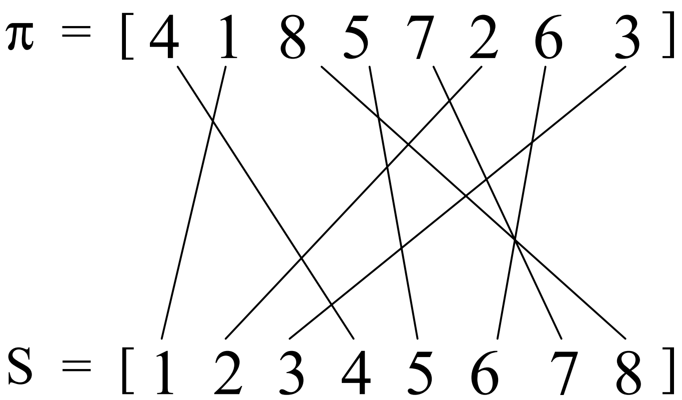
is equivalent to

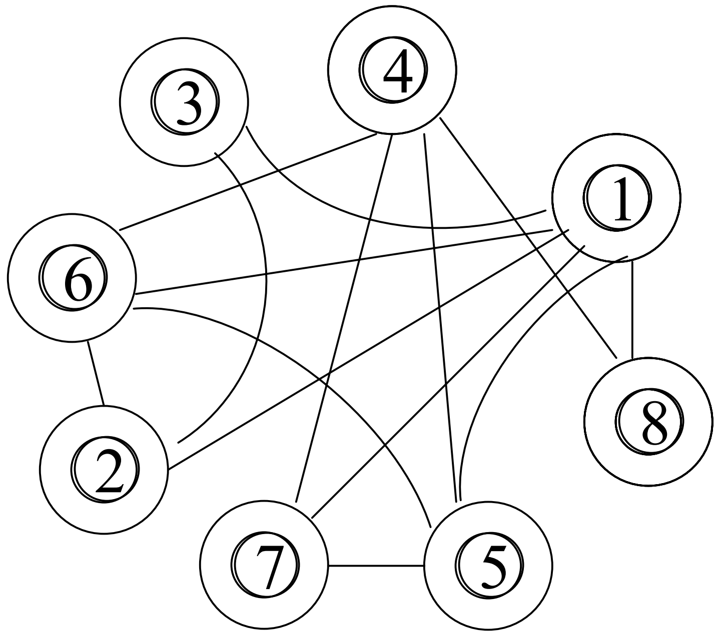
The ordering of a sequence of N numbers using the minimun number of *stacks*

Ordering problem (2)



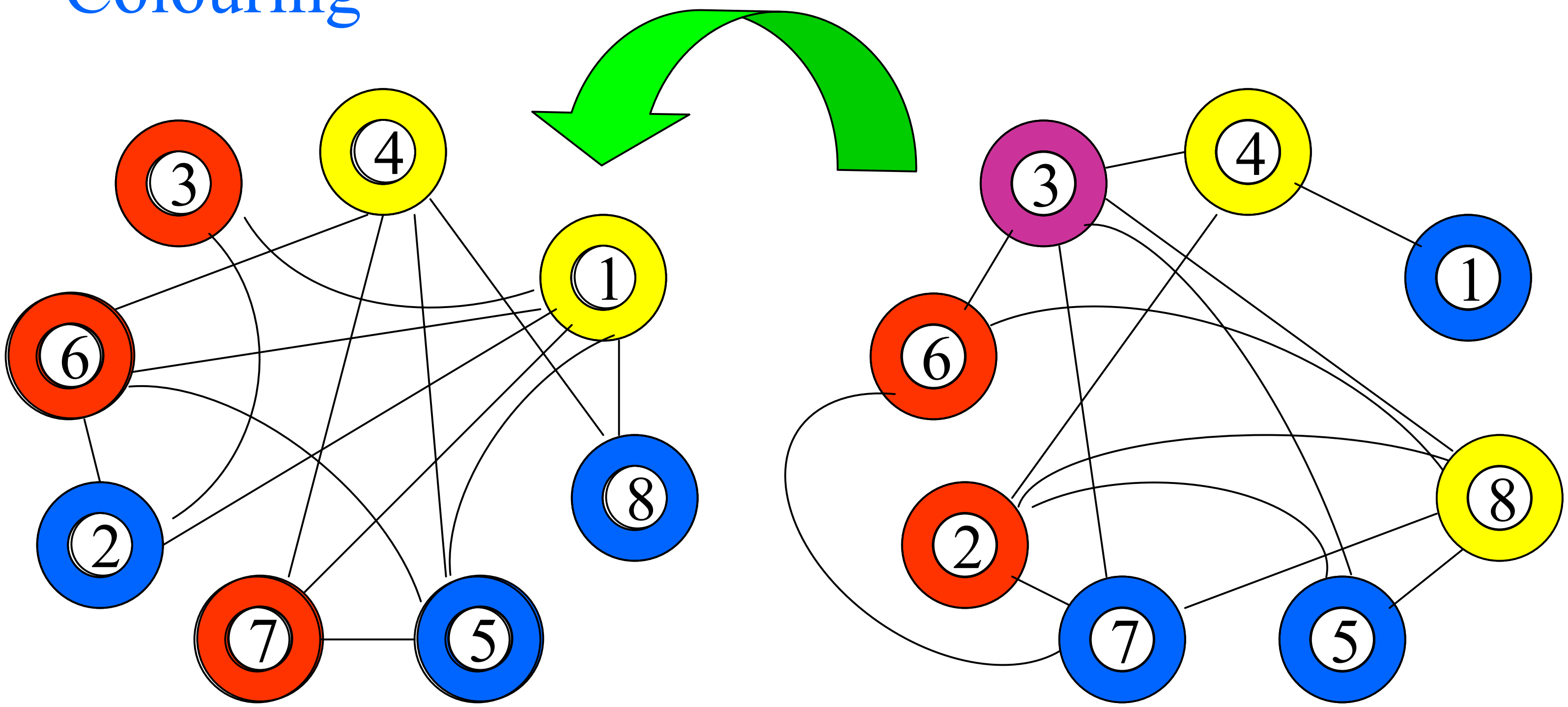
- Train assignment





Complement graph

- Colouring



Complement permutation graph *Permutation graph*

Coloring complexity

- What is the complement graph of a permutation graph?

Complement permutation graph

(Shunting area)

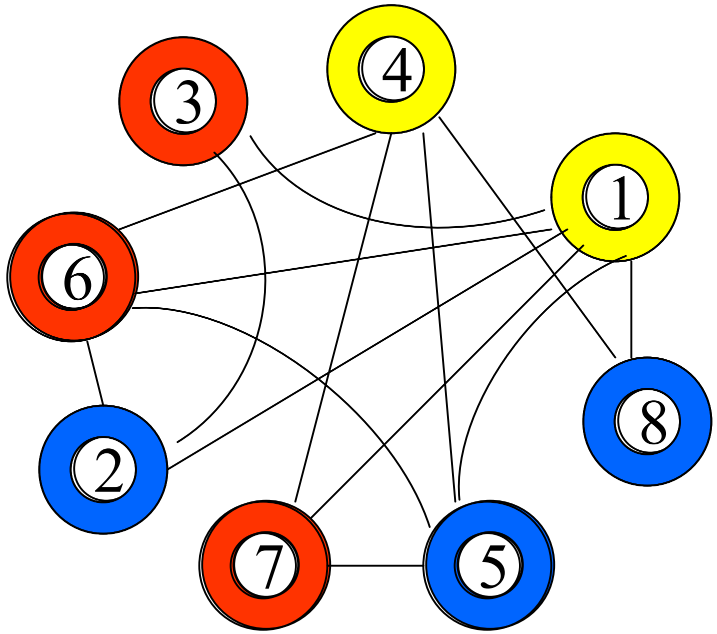
Permutation graph

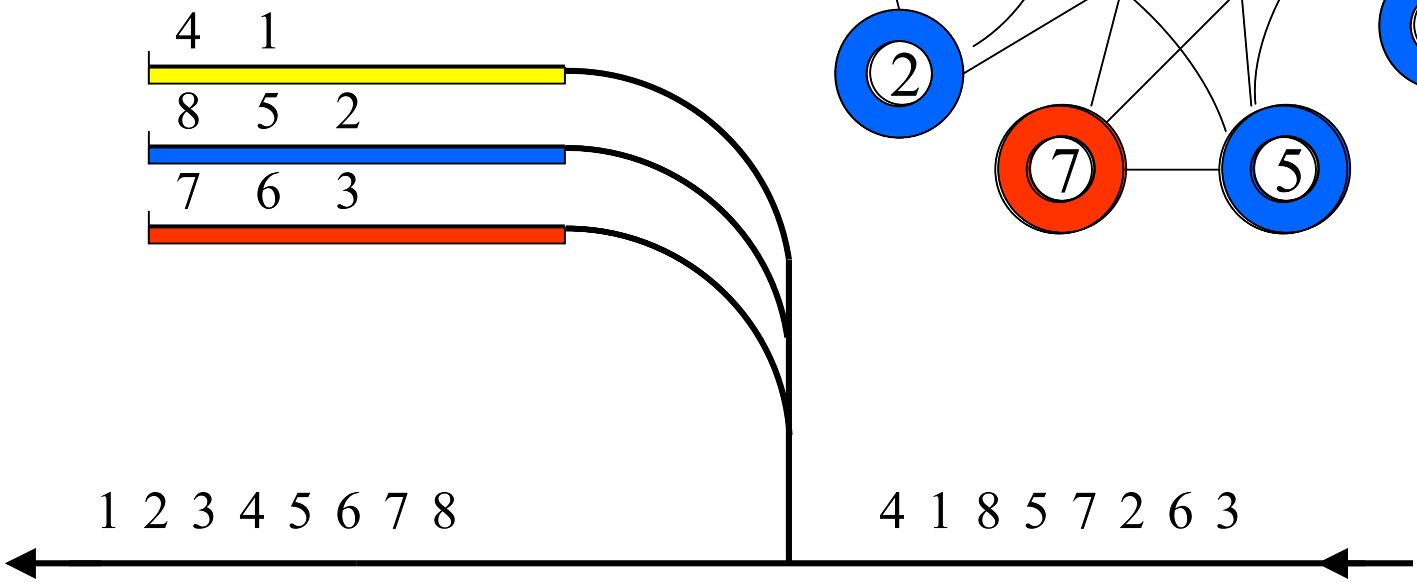
(Marshalling area)

Colouring in O (n lg n) time

Train depot algorithms

- Train assignment
  - Shunting area





Online Problem (3)

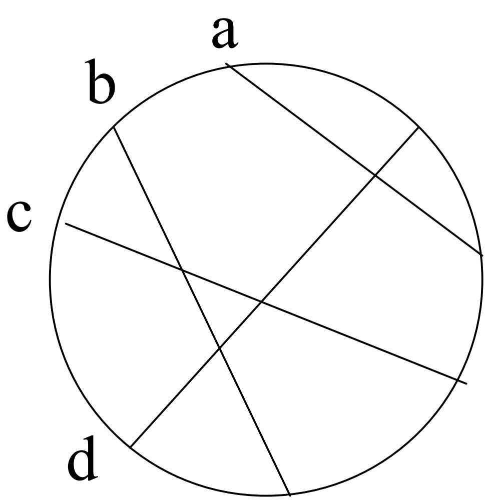
- Train assignment
  - Offline
    - The algorithm is given the entire sequence of trains to store in the depot
      - *Coloring of permutation graph*
  - Online
    - When assigning a train to a depot track, there is no knowledge of the remaining incoming trains
      - *Greedy assignment to tracks*

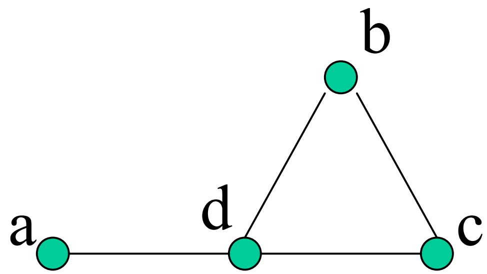
Train depot algorithms

- Train assignment
  - conclusions
    - These train storage problems are ordering problems
    - The problems are equiv. to coloring of permutation graphs
    - The coloring is solvable in $O(n \lg n)$ [Pnueli et al., '71]
    - Offline solution ≡ Online solution

Circle Graphs

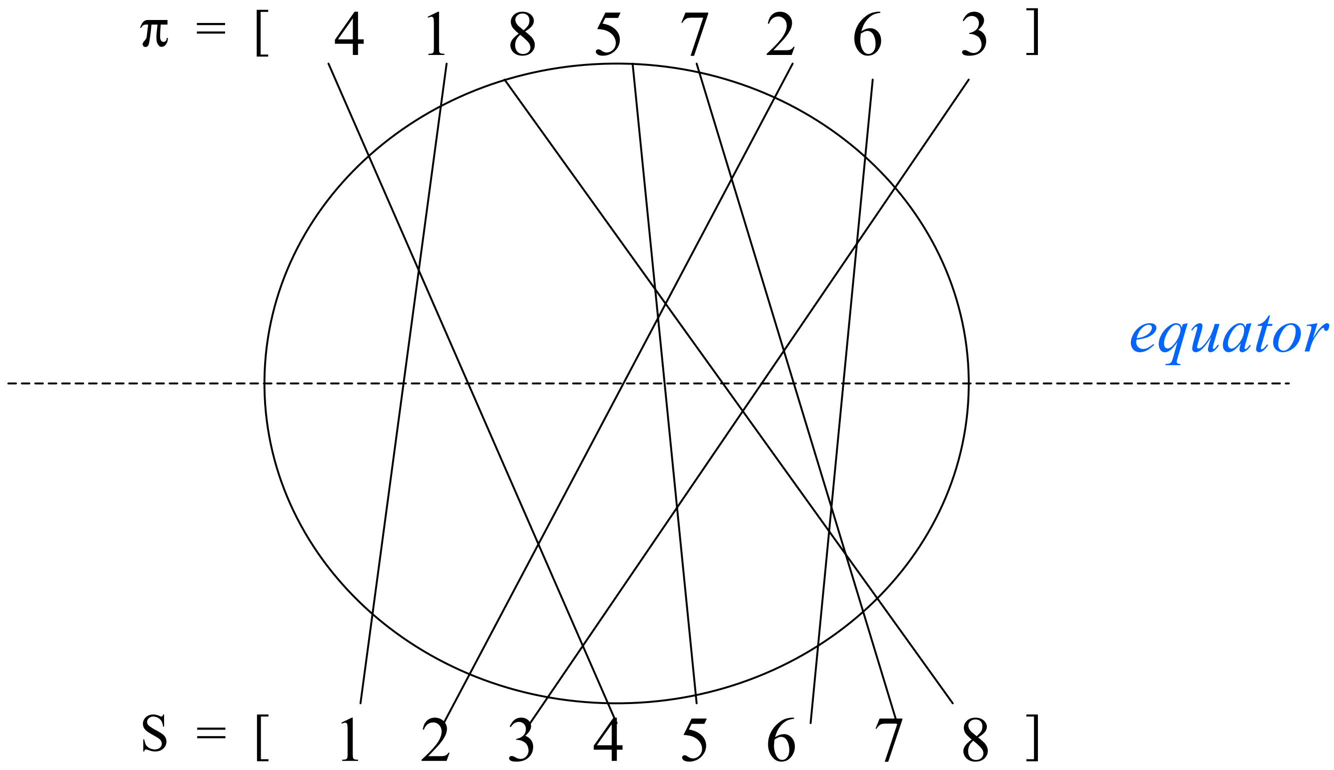
- Definition: Intersection graphs of chords in a circle.



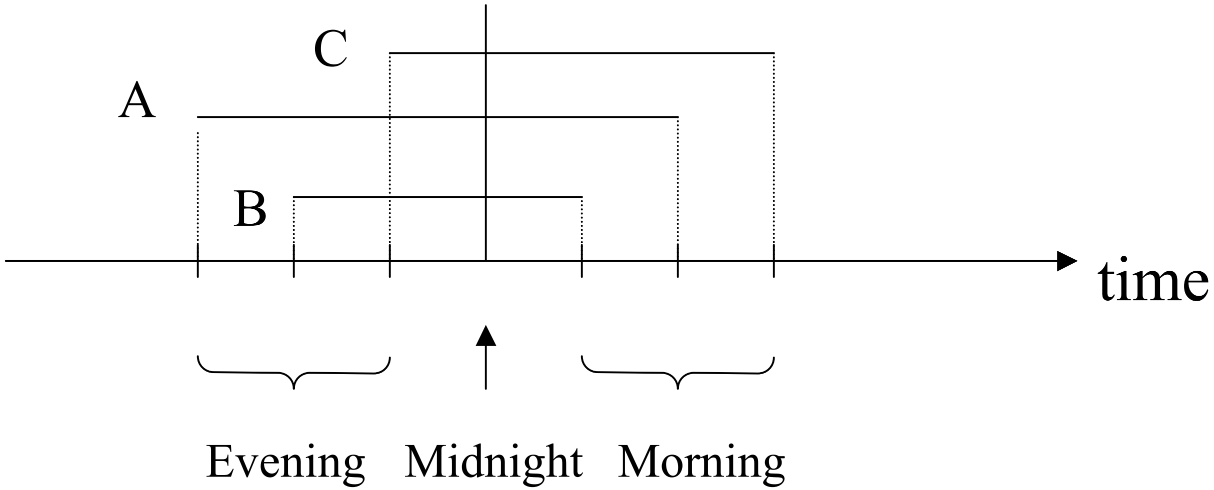


Circle Graphs

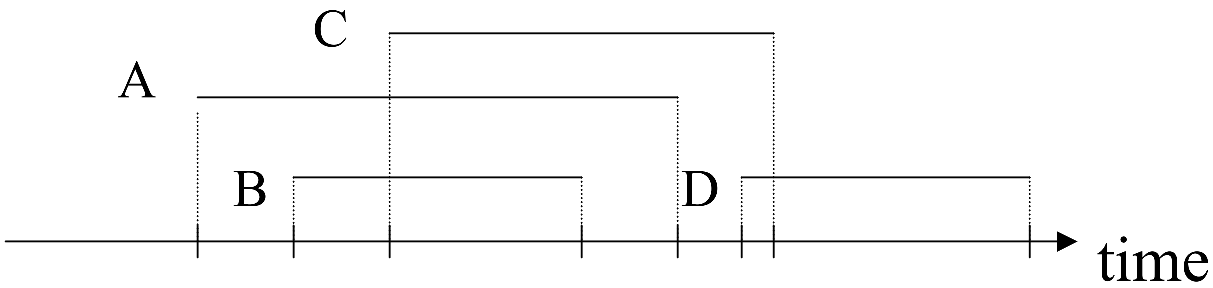
- Permutation Graphs are Circle Graphs with an equator



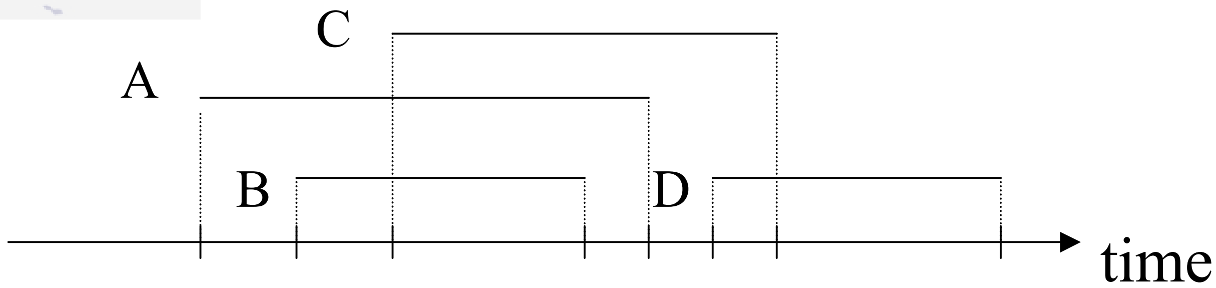
Removing the “night” in a shunting area



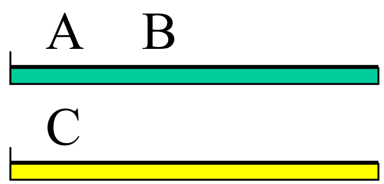
Removing the “night” in a shunting area



Removing the “night” in a shunting area



Let X and Y be two trains, and
let I_X and I_Y be the relative intervals

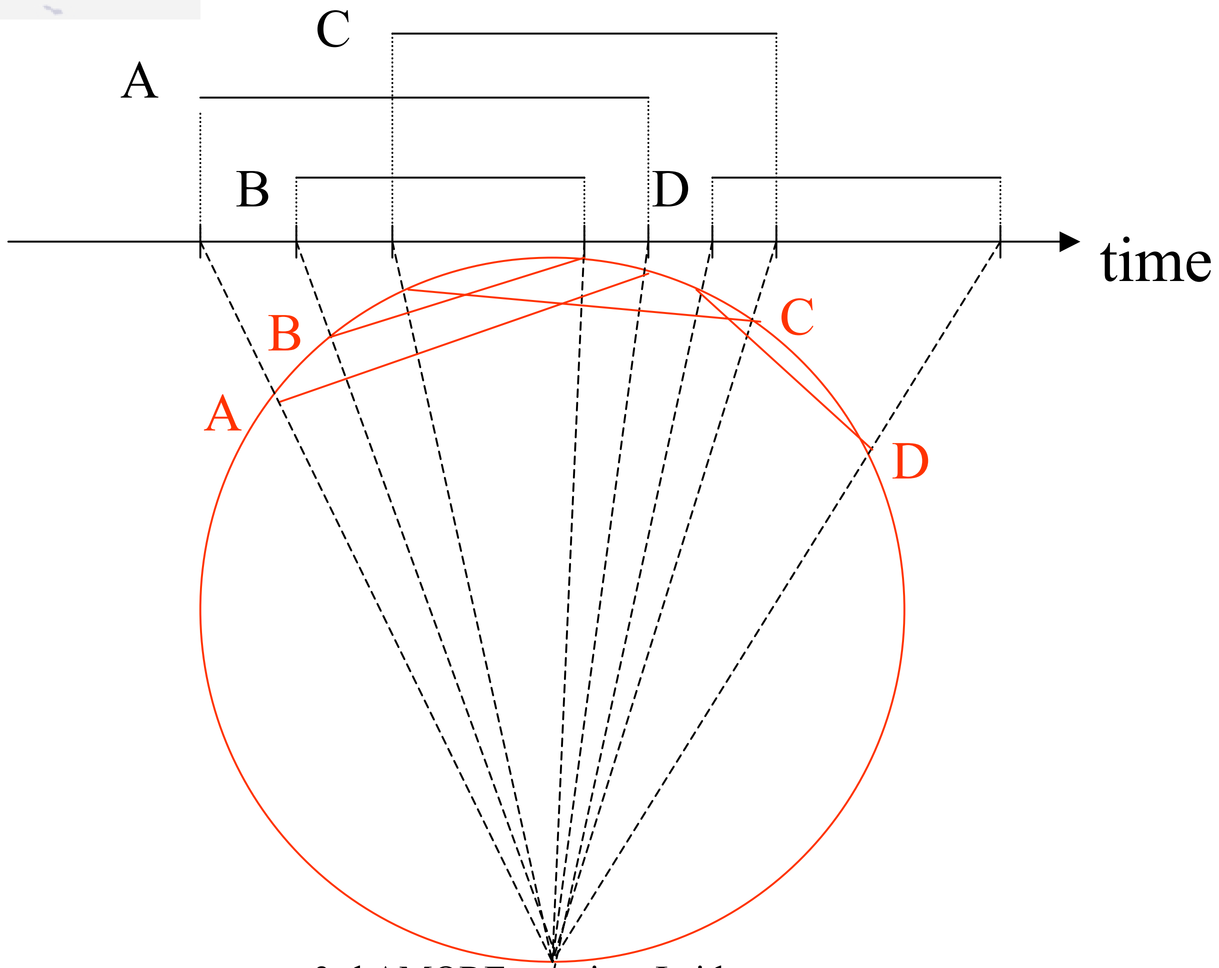


If I_X and I_Y overlap

(i.e. $I_X \cap I_Y \neq \phi$ but neither $I_X \subseteq I_Y$ nor $I_Y \subseteq I_X$)

Then two different tracks for X and Y

Transf. into a circle graph

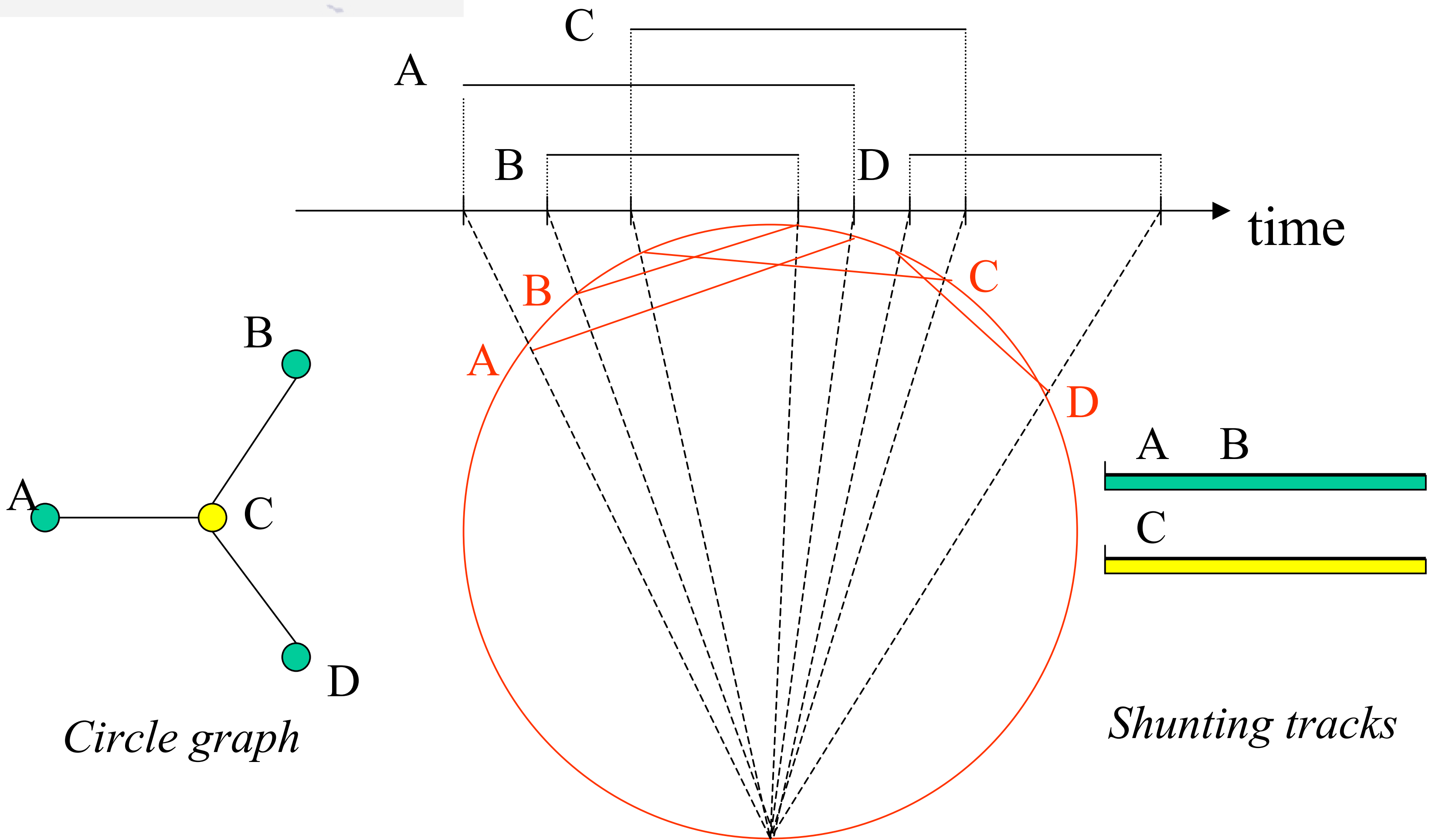


No equator

that is

No night

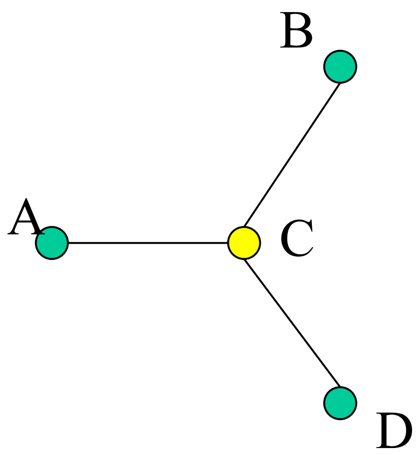
Transf. into a circle graph



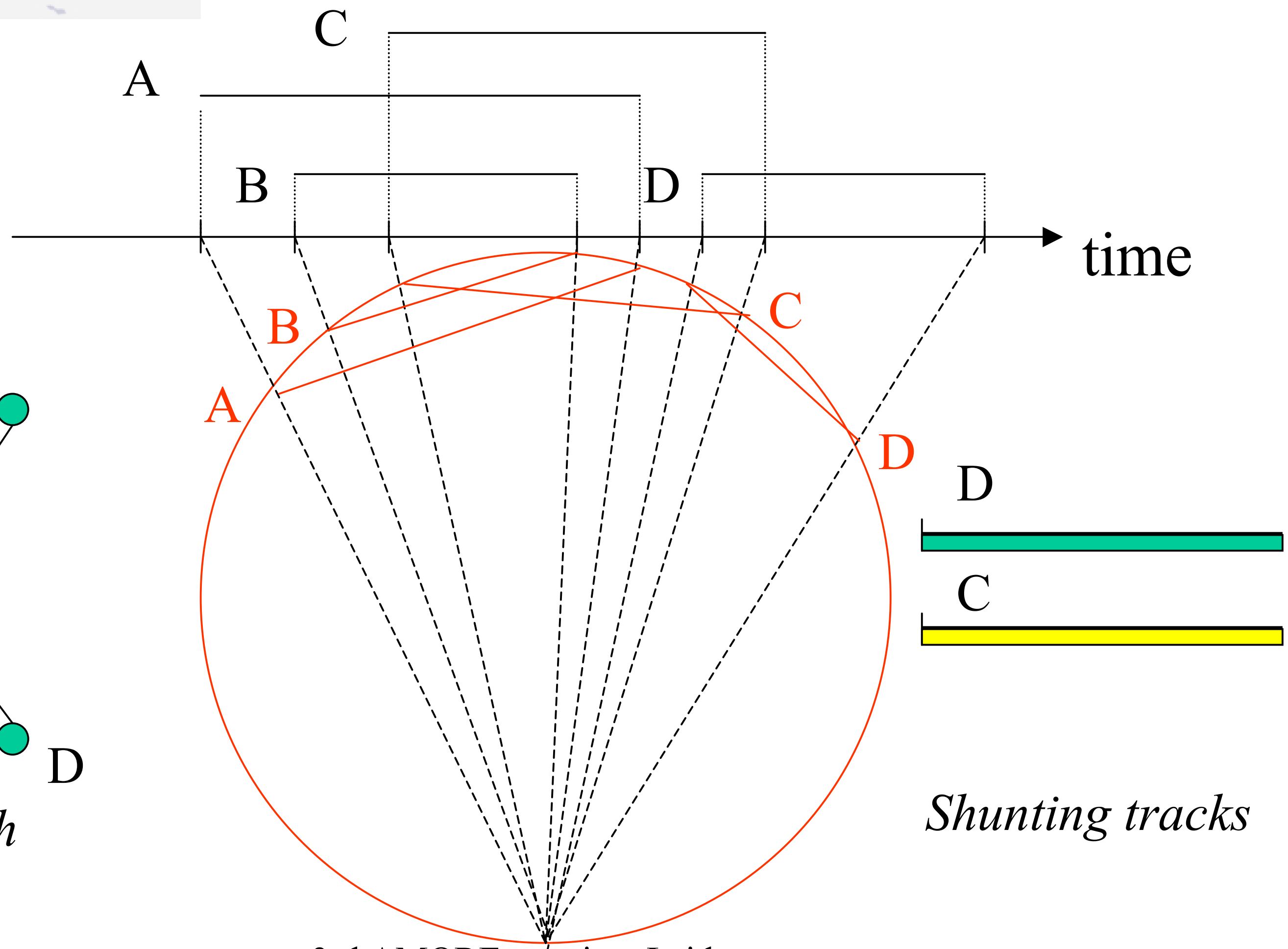
Circle graph

Shunting tracks

Transf. into a circle graph



Circle graph



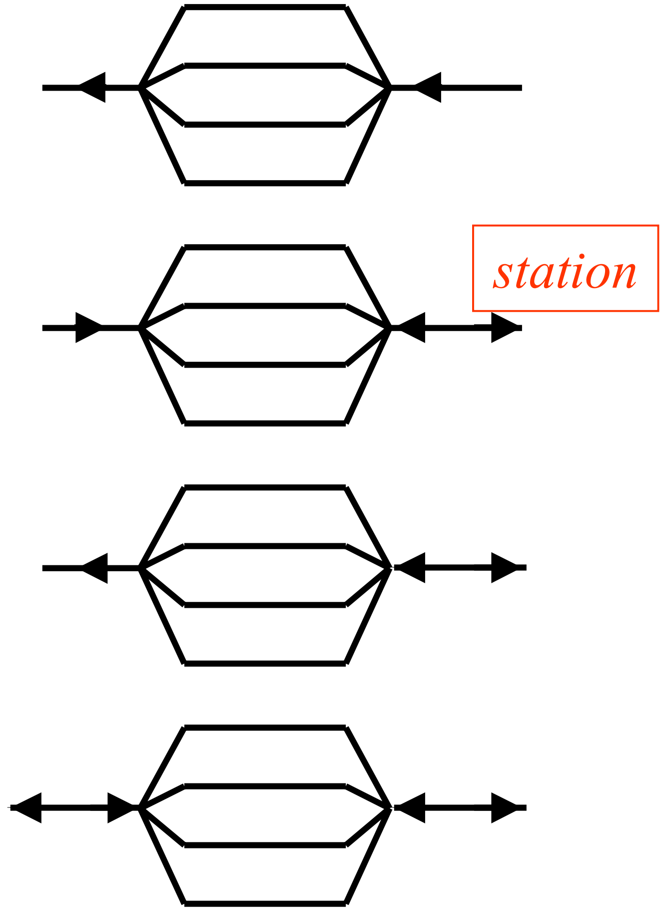
Shunting tracks

Train depot algorithms

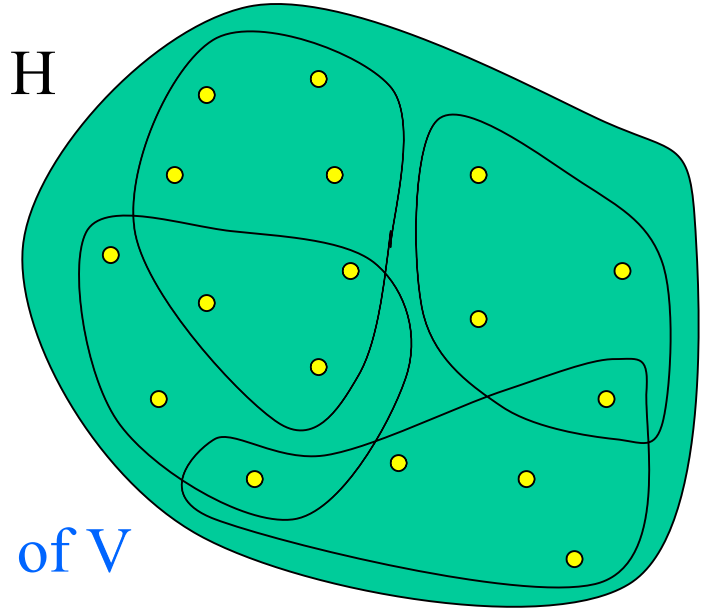
- Assignament on a shunting area without “night”
  - conclusions
    - This train storage problem is equiv. coloring of circle graphs
    - Coloring of circle graphs is NP-complete [GT4]
    - Is 2-Approx. but not (3/2)-Approx.
    - 3-coloring is in P; 4-coloring is NP-complete [Unger, 88]
    - The same problem for a marshalling area is solvable in $O(n \lg n)$

Generalized problems

- Track access constraints
  - Single Input Single Output (SISO)
  - Double Input Single Output (DISO)
  - Single Input Double Output (SIDO)
  - Double Input Double Output (DIDO)



Hypergraphs



H = (V , E)

V is a set of *vertices*

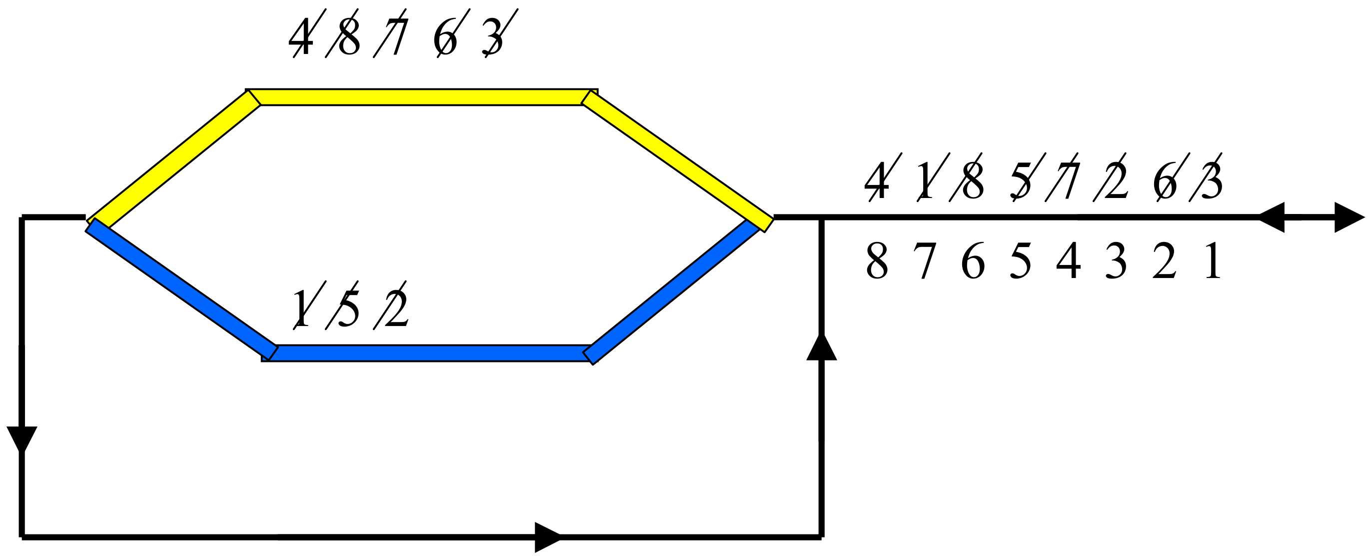
E is a set of subsets (*hyperedges*) of V

If all hyperedges have size *k*, H is called *k*-uniform

2-uniform hypergraphs are normal graphs

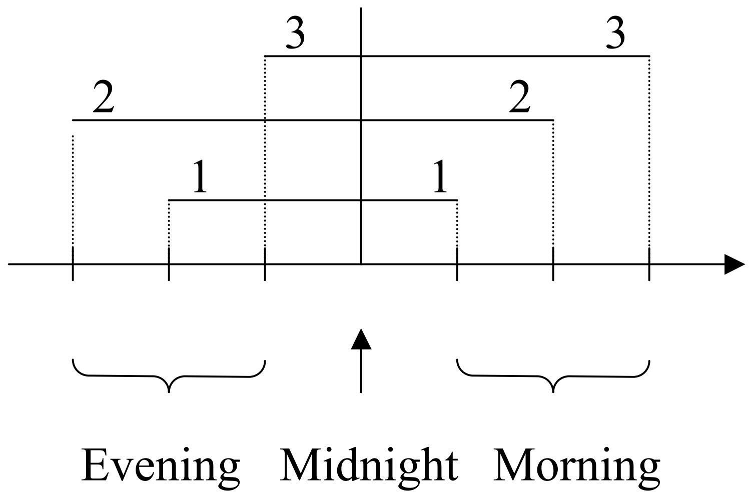
Single Input Double Output

- Train assignment
  - Two tracks are enough



SIDO constraint

- Train assignment
  - SIDO triple constraint



Can we use a single track?

2 1 3

No

Single Input Double Output

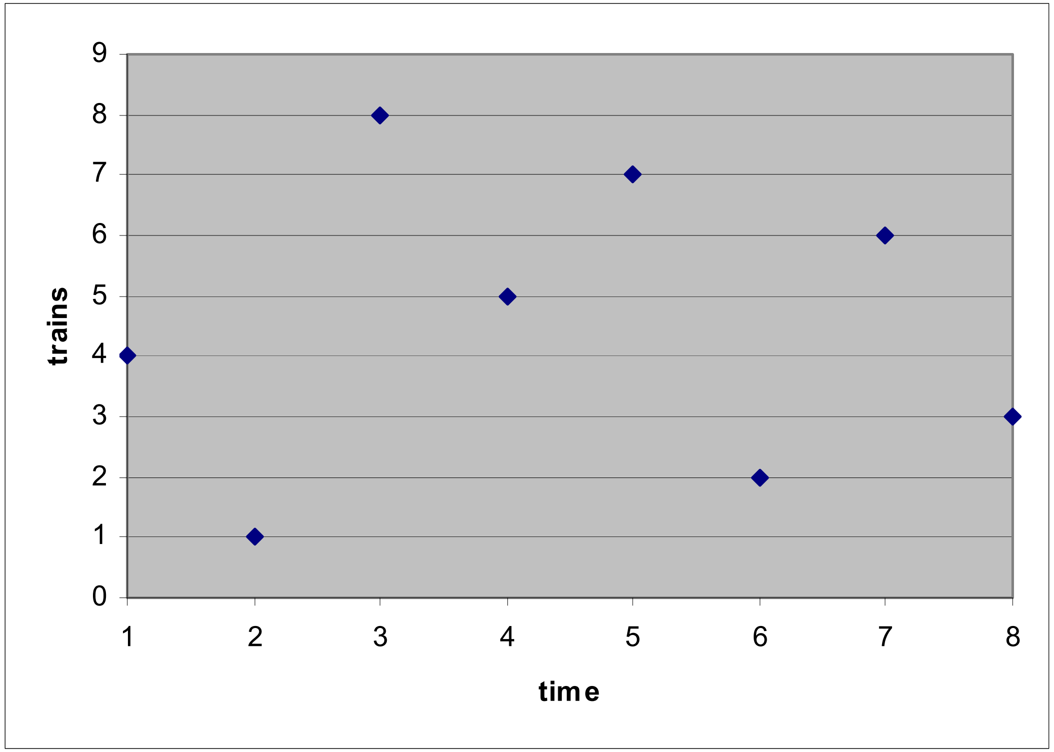
- *Why?*
  - triple constraint: three trains in the input sequence form a valley.

Forbidden sequences: [2 1 3] and [3 1 2]

admissible sequences : [1 2 3], [1 3 2], [2 3 1], and [3 2 1]

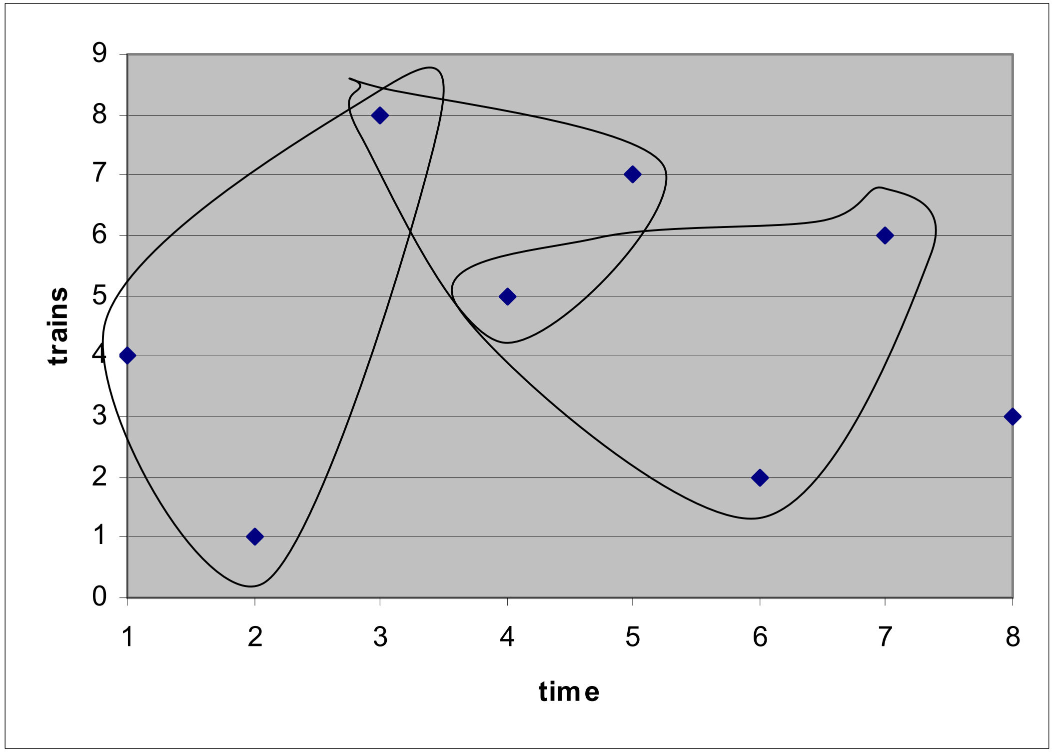
Single Input Double Output

- Input sequence representation: π =[4,1,8,5,7,2,6,3]



Single Input Double Output

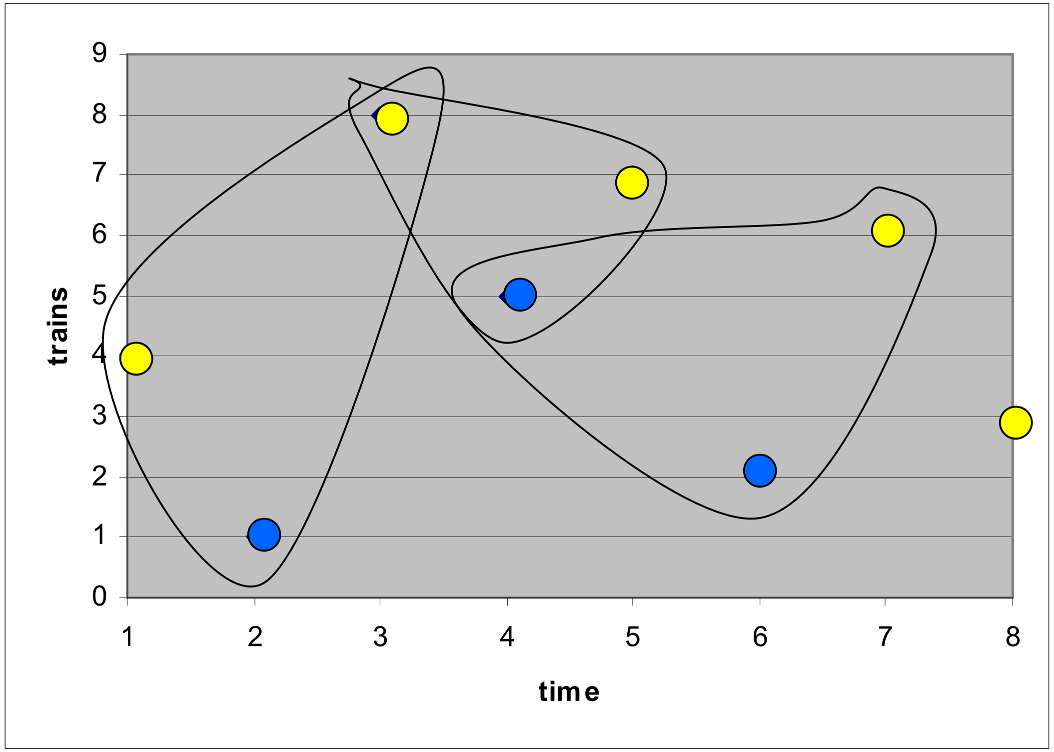
- Modelling as a 3-uniform hypergraph:



Valley hypergraph $H(\pi)$

Single Input Double Output

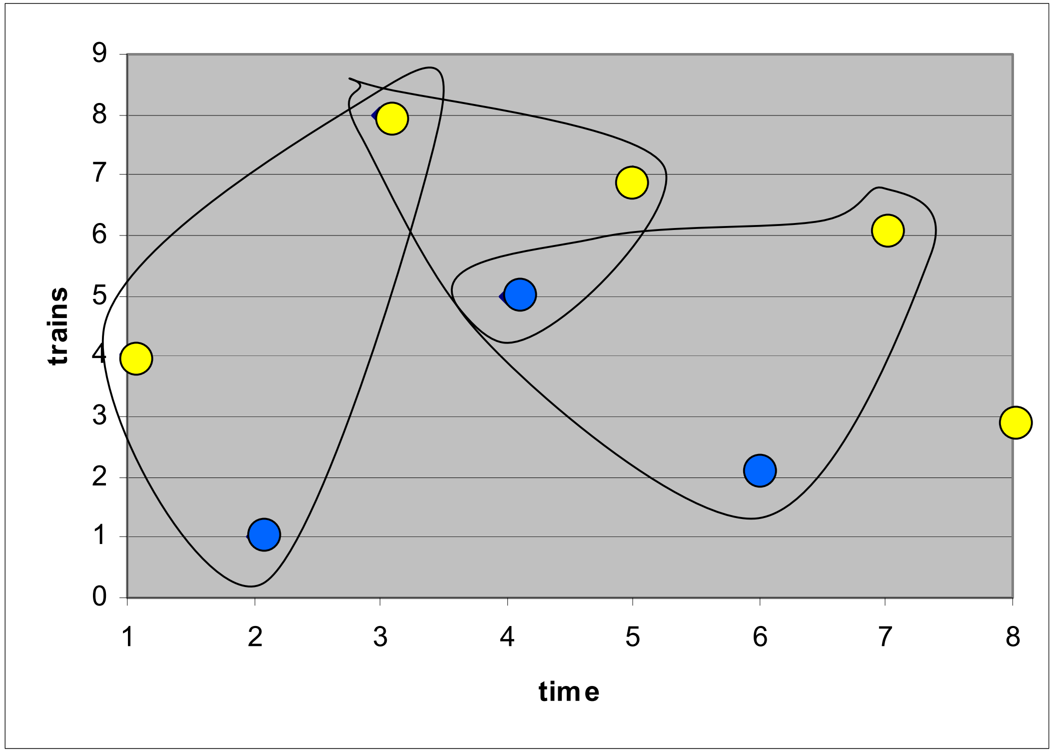
- Track assignament = coloring of H(π)

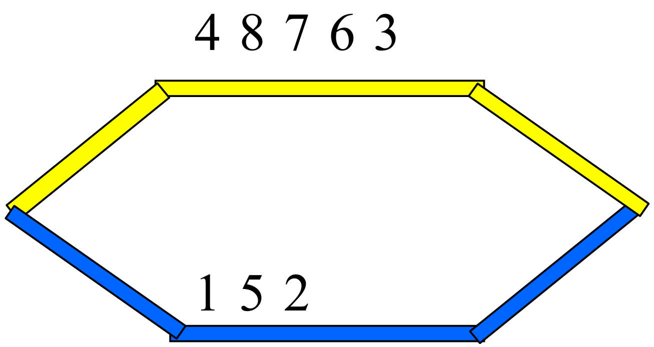


At least two nodes in a hyperedge have a different color.

Single Input Double Output

- SIDO track assignament $\equiv$ coloring of $H(\pi)$





SIDO vs. DISO

- These two problems are equivalent
  - Relation SIDO/DISO
    - Given an arbitrary train permutation $\pi = [\pi_1 , \pi_2 \dots \pi_N]$, the permutation index $\pi^{-1} = [\pi_1^{-1} , \pi_2^{-1} , \dots , \pi_N^{-1}]$, and the time reversing operator R, s.t. $(\pi^{-1})^R = [\pi_N^{-1} , \dots , \pi_2^{-1}, \pi_1^{-1}]$, then

$$\boxed{\mathrm{SIDO}\ (\pi) \equiv \mathrm{DISO}\ (\pi^{-1})^R}$$

SIDO/DISO conclusions

- Coloring 3-uniform hypergraphs:
  - NP-hard
  - *k*-coloring is approx. within $O(n/(\lg^{k-1} n)^2)$
    [Hofmaister, Lefmann, '98;]
  - 2-coloring is NP-complete for 3-uniform hypergraphs
    [Approx. results: Krivelevich et al., 2001]
  - 2-coloring is in P for *valley hypergraphs*
  - *k*-coloring of *valley hypergraphs* ? *Open!*

Double Input Double Output

- Train assignment
  - SIDO (DISO)
    - Model: coloring of valley hypergraphs
  - DIDO
    - Model: coloring of certain 4-uniform hypergraphs

Open!

Other generalizations

- Take care of train/tracks lengths
  - Equivalence with bin packing problems (?)
- Take care of types of trains
  - Subgraphs of permutation graphs

Input: [A B A C]

Output:[A C A B]

- Take care of specific depot topologies
  - Open

Other generalizations

- Take care of train/tracks lengths
  - Equivalence with bin packing problems. (?)

- Take care of types of trains
  - Subgraphs of permutation graphs

Input: [A B A C]

Output:[A C A B]

- Take care of specific depot topologies
  - ??


THE END