MERLIN

A polynomial solution for the Traveling Salesman Problem

Dr. Joachim Mertz, 2005

Scope

- This presentation provides a concise overview about the MERLIN algorithm:
  - Introduction
  - Idea and basic features
  - Description of the optimization model
  - Detailed definition of variables and constraints
  - Conclusion

- For more detailed information please refer to [1]

The Traveling Salesman Problem

Given:

- n cities/stations {0,…,n-1}
- Distances $c_{ab} \geq 0$ for each disjoint pair a,b of stations

Task:

Find a roundtrip of minimal total length visiting each of the n stations exactly once

The P=NP Question

- The Travelling Salesman Problem (TSP) is NP-complete
- NP-complete problems are known to require an exponential number of computational steps (e.g. 2^n or n!) on a deterministic machine.
- So far there is no algorithm which is able to solve a NP-complete problem by a polynomial number of steps (e.g. n^3 or n^5)
- NP-complete problems are considered as the ‘hardest’ problems within the class NP: If an algorithm would be able to solve any NP-complete problem by a polynomial number of steps, the class NP would collapse and would be part of the problem class P of problems solvable in polynomial time (P=NP)

MERLIN: Basic features

- MERLIN is based on linear programming
- A set of suitable variables and linear constraints defines an optimization model transforming the TSP into a linear programming problem
- Thus, the model parameters are used like real values though TSP is an integer optimisation problem
- The model requires only a polynomial number of variables and constraints

Description Linear Optimization

$$Min \sum_{i=1}^{p} c_i x_i \quad c_{i,}\ x_i \in R,\ \ x_i \geq 0$$

Considering a set of linear constraints:

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1p}x_p \geq (\leq, =)\ b_1$$

$$\ldots\ldots\ldots$$

$$a_{m1x1} + a_{m2x2} + \ldots + a_{mp}x_p \geq (\leq, =)\ b_m$$

with $a_{ij,}\ b_i \in R$

MERLIN Model: First definitions

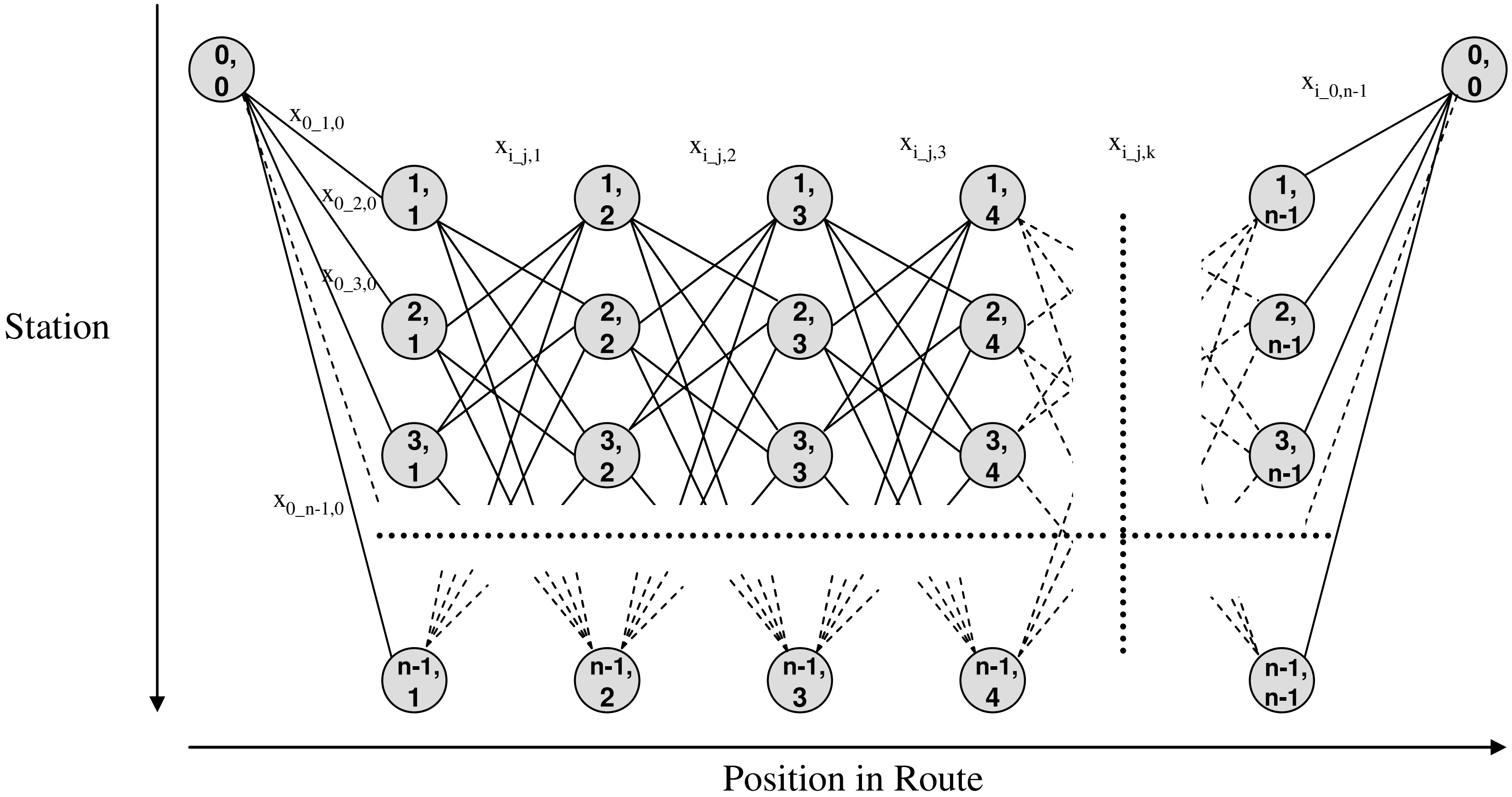
- Input parameters:
  - *n* stations (0…*n*-1)
  - cost matrix *C* with the distances $c_{ij} \geq 0$
    of the stations *i,j* ($0 \leq i,j \leq n-1$, $i \neq j$).
- Graph Model (see figure next page):
  - **Nodes** labelled as (<column>, <row>) = *(i,k)*
    → station *i* is at the *k*th position of the salesman roundtrip.
  - **Edges** labelled as (<from_node>, <to_node>, <row>) = *(i,j,k)*
    → The edges are represented by optimisation variables

    $$x_{i\_j,k} \quad (0 \leq x_{i\_j,k} \leq 1)$$

    signalling the presence of a directed edge with start station *i* and end station *j* at position *k* in the graph. E.g. if variable $x_{i\_j,k}=1$ then the edge from station *i* to station *j* is at the *k*th position of the salesman's roundtrip .
- W.l.o.g. we take station 0 as start and end station of the roundtrip (the salesman home office)

$$\sum_{j=1}^{n-1} x_{0\_j,0} = 1$$

Graph model



Station
Position in Route
0, 0
$x_{0\_1,0}$
$x_{0\_2,0}$
$x_{0\_3,0}$
$x_{0\_n-1,0}$
$x_{i\_j,1}$
$x_{i\_j,2}$
$x_{i\_j,3}$
$x_{i\_j,k}$
$x_{i\_0,n-1}$
1, 1
2, 1
3, 1
n-1, 1
1, 2
2, 2
3, 2
n-1, 2
1, 3
2, 3
3, 3
n-1, 3
1, 4
2, 4
3, 4
n-1, 4
1, n-1
2, n-1
3, n-1
n-1, n-1
0, 0

Some other definitions

- a **route** is a consecutive sequence of *n* edges from start-node (0,0) to end-node (0,*n*), in which the *n* edges are represented by *n* variables
  $x_{i0\_j0,0} \dots x_{in-1\_jn-1,n-1}$
- a route is **consecutive** when it is using exactly one edge per position (column) and the end-station of an edge at position *k* is identical with the start station of the following edge at position *k*+1
- a route is called **symmetric** if it is Hamiltonian, i.e. including all stations 0…*n*-1, so each station is an end node of exactly one edge of the route

We will have a general solution for the TSP if we are able to find always a unique symmetric and consecutive route with minimal cost from node (0,0) to node (0,*n*)

Linear Optimisation: Cost function

Min $\sum_{i=0}^{n-1} \sum_{j=0, i \neq j}^{n-1} \sum_{k=0}^{n-2} c_{ij} x_{i\_j,k}$

Where at the first/last position only edges from/to station 0 are relevant, so all other edge-variables can be set to zero:

$$\sum_{i=1}^{n-1} \sum_{j=0, i \neq j}^{n-1} x_{i\_j,0} = \sum_{i=0}^{n-1} \sum_{j=1, i \neq j}^{n-1} x_{i\_j,n-1} = 0$$

Linear Optimisation: Basic constraints

- Route has to be symmetric
  → each station is reached exactly once

$$\sum_{k=0}^{n-1}\sum_{i=0}^{n-1} x_{i\_j,k} = 1 \qquad \forall j\ (0 \ldots n-1),\ i \neq j \qquad (1)$$

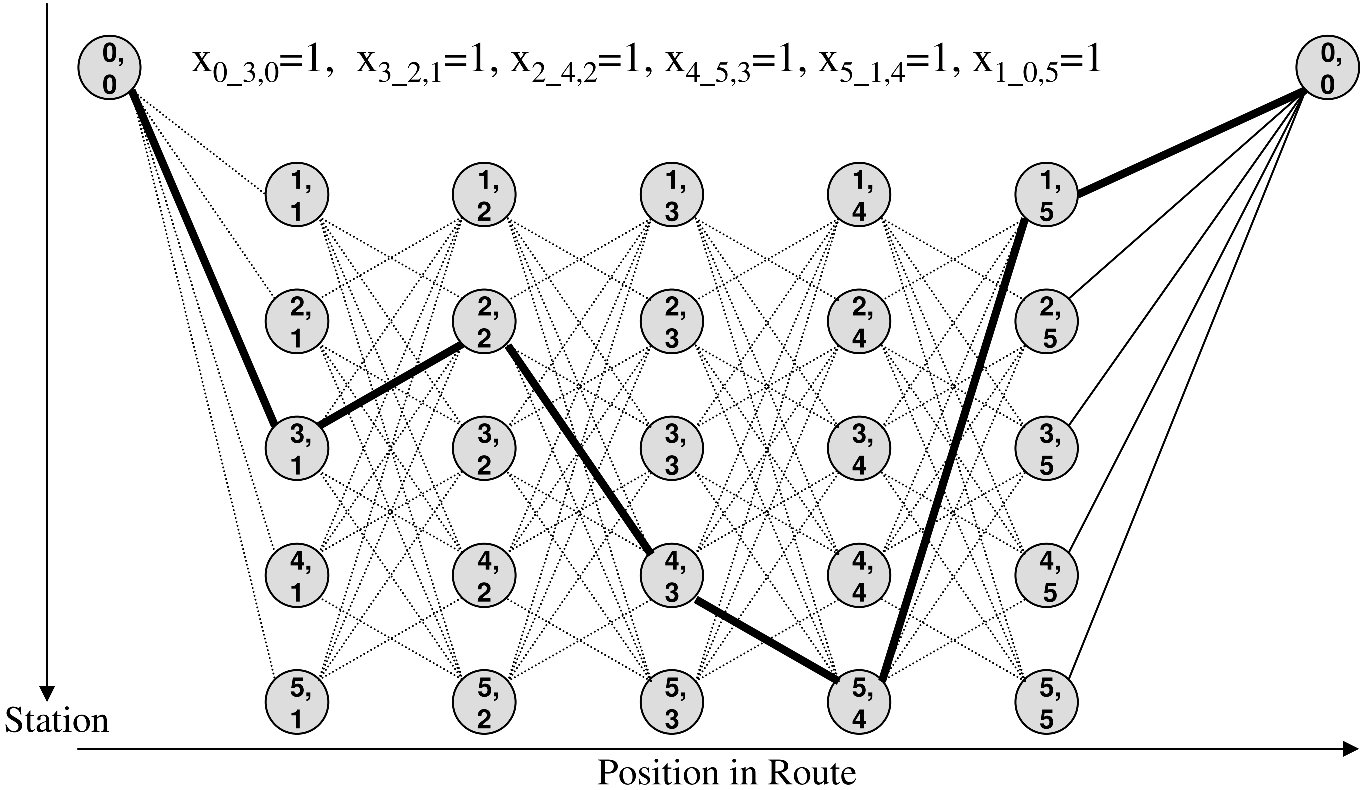
- Route has to be consecutive
  → Sum of entry variables at each node (*l,k*) equals to the sum of the exit variables

$$\sum_{i=0, i\neq l}^{n-1} x_{i\_l,k} = \sum_{j=0, j\neq l}^{n-1} x_{l\_j,k+1} \qquad \forall l,k\ \ (0 \ldots n-1),\ k\ \ (0 \ldots n-2)$$

or

$$\sum_{i=0, i\neq l}^{n-1} x_{i\_l,k} - \sum_{j=0, j\neq l}^{n-1} x_{l\_j,k+1} = 0 \qquad (2)$$

Example for a valid solution



- Route is consecutive
- Route is symmetric
- Variables $x_{i\_j,k}$ are integer values (0 or 1)

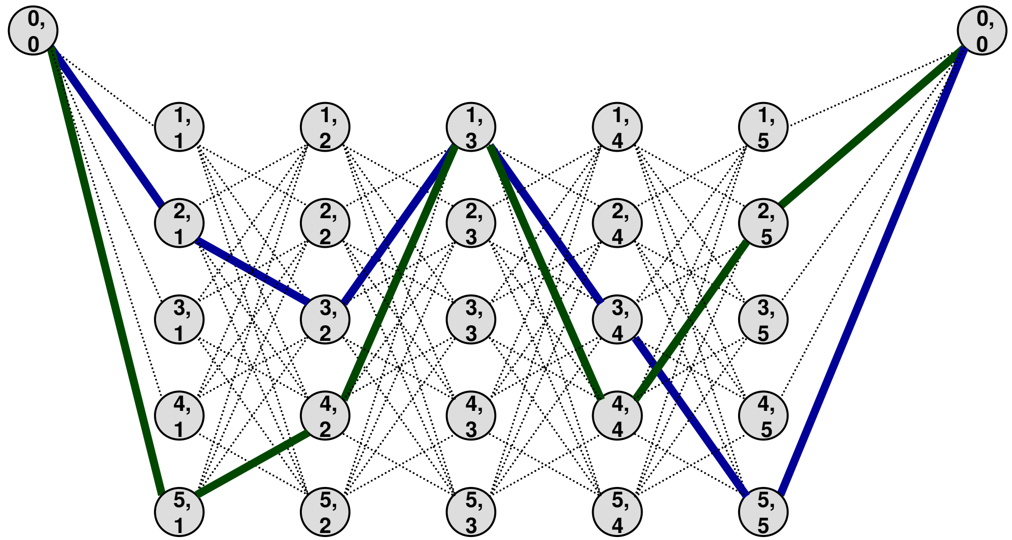
Problem:

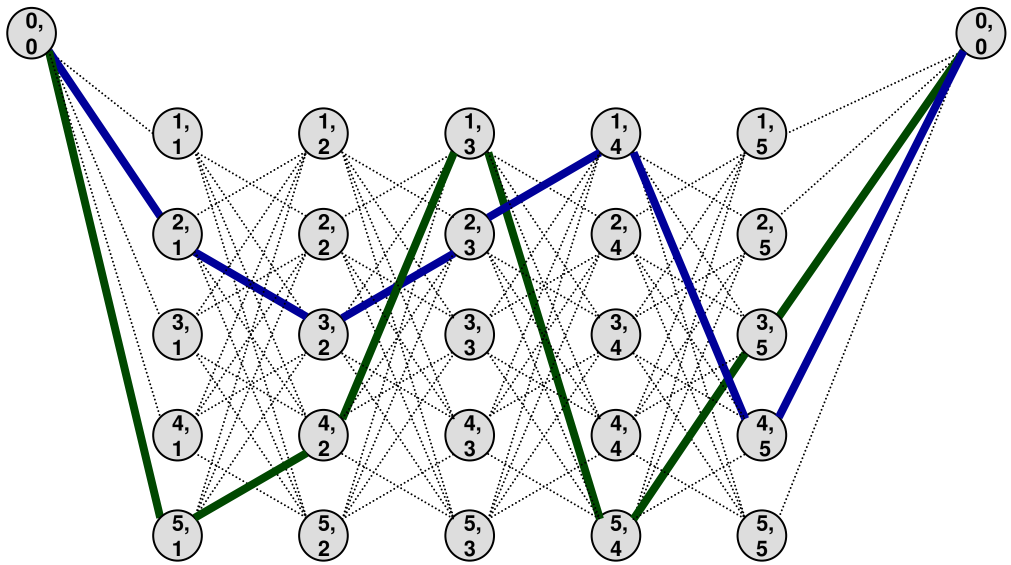
How can we enforce the 0/1 settings of the variables in a 'real value environment' like LP?

→ Seems not to be not possible but we have to ensure it, otherwise …

Examples for degenerated solutions

… bad things are happening
→ non-integer solutions

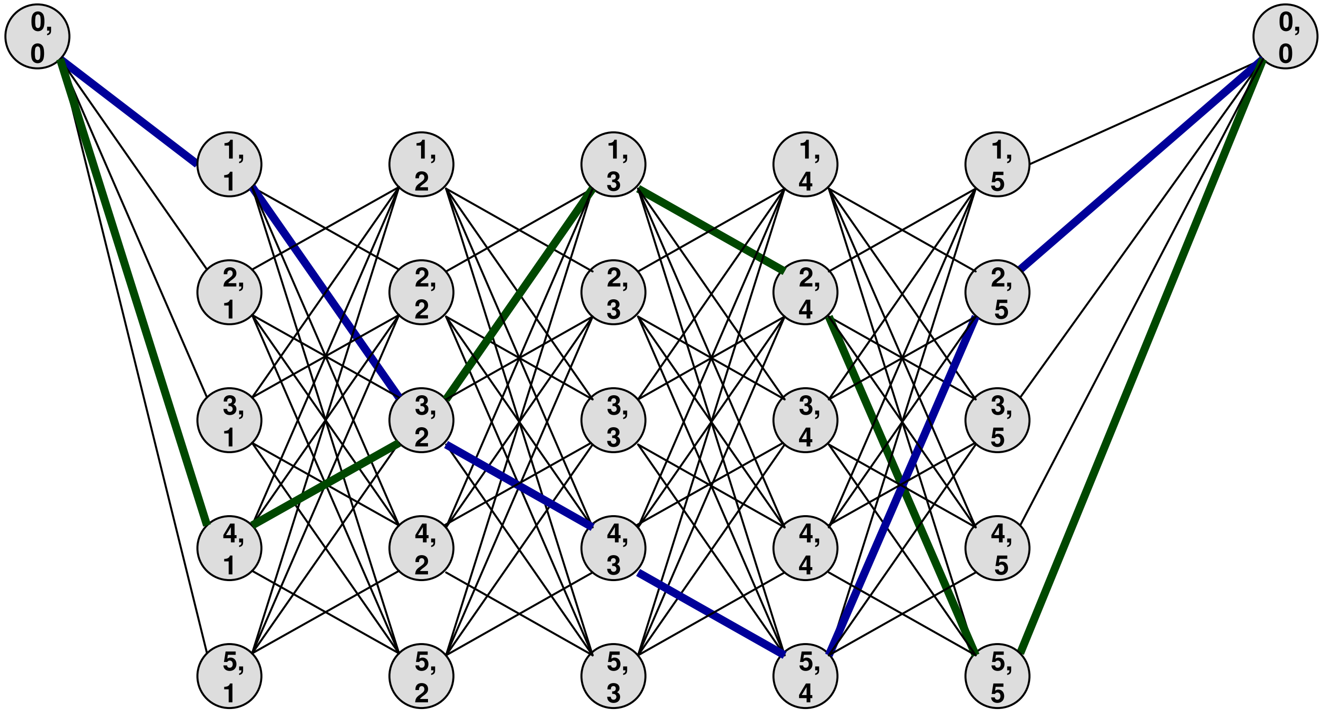




Considerations non-integer solutions

- Non-integer solutions combine several sub-routes from start-node (0,0) to end-node (0,n), each with a weight below 1 but altogether with an overall weight of 1
- There are two classes of non-integer solutions:
  - **The bad ones:**
    Combinations of sub-routes where at least two of them are 'asymmetric' i.e. having particular stations more than once in their sequence and leaving others instead. The particular sub-routes have to be complementary such that a skipped station of a sub-route is covered by the sequence of another sub-route
    → we will call this asymmetric solutions in the following
  - **The good ones:**
    Combinations of sub-routes of complete and valid solutions such that each sub-route is symmetric, i.e. including all stations 0…n-1 once
    → we will call this symmetric solutions in the following
- The first ones have to be avoided, but we can live comfortably with the latter (see next page)

Example symmetric solution



The example above causes no trouble because each sub-route is a valid and optimal solution for the particular TSP problem

→ pick one out e.g. by the following procedure:

- set of one of the non-zero variables $x_{i\_j,0}$ at the first column to 1
- run the overall algorithm again
- if still variables between 0 and 1 occur, set one of them to 1 a.s.o.

→ This procedure terminates in less than n steps

How to deal with the bad ones?

- Symmetric solutions are welcome so we only have to avoid asymmetric sub-routes
  → This can be done by two steps:
    1. Separate the complementary sub-routes
    2. Enforce the symmetry of each sub-route
- For that, we introduce a new mechanism, the **mirror:**
  – For each graph node (*l,k*) the mirror *Y*(*l,k*) provides an exact representation of the sub-route(s) crossing this node
  – A mirror *Y*(*l,k*) consists of about n^3 variables $y^{(i,k)}{}_{i\_j,d}$ each representing a variable $x_{i\_j,k}$ of the original graph

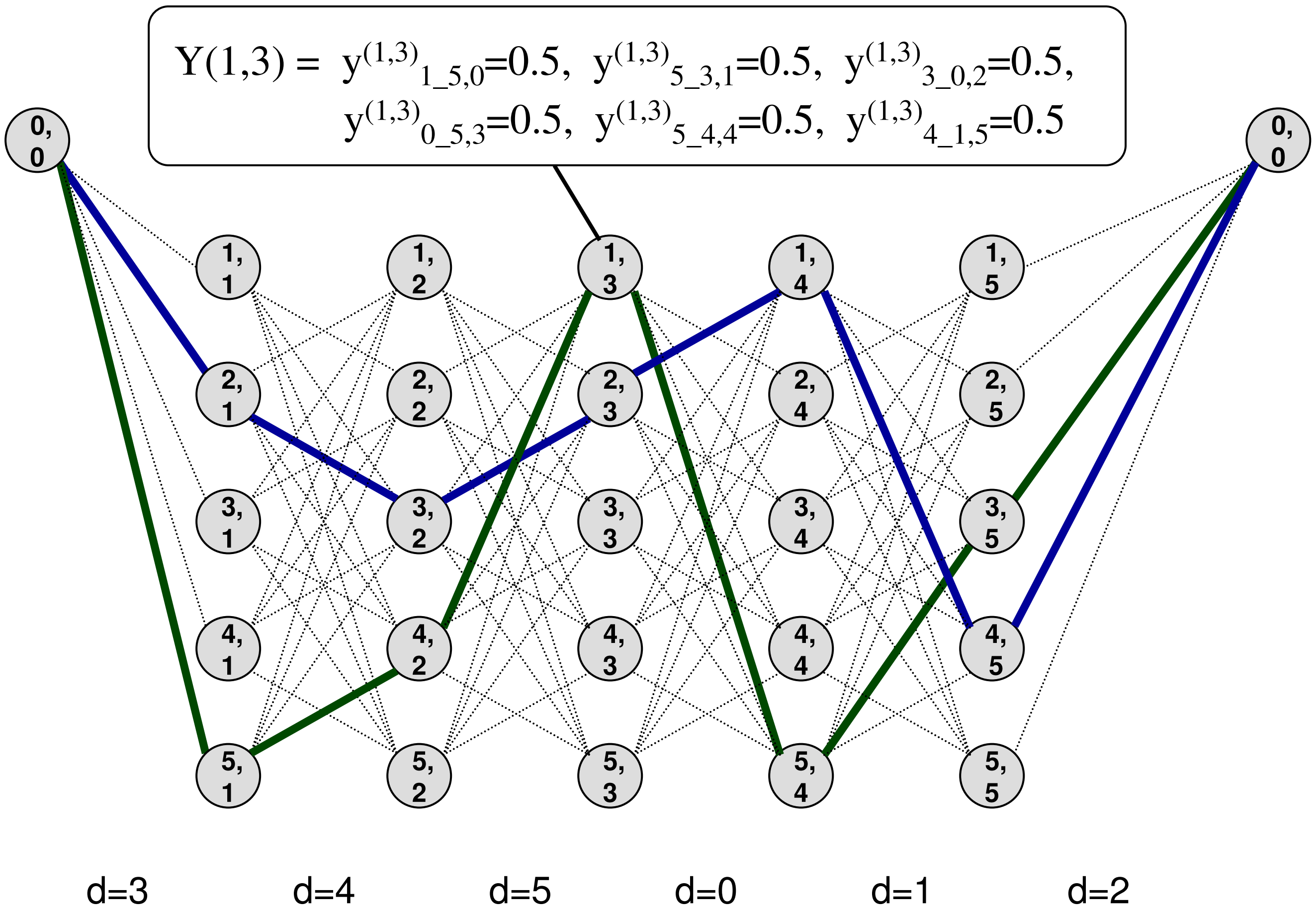
$$Y(l,k) = \{ y^{(l,k)}{}_{i\_j,d} \} \qquad \forall l,k,i,j,d \ \ (0 \ldots n-1)$$

where

i,j = start/end stations of the corresponding graph edge

d = distance of the edge from node (*l,k*) in numbers of columns

Example for a mirror



Constraints to build up mirrors (1)

1. A mirror *Y*(*l,k*) has to represent all sub-routes crossing node (*l,k*) → edges with *d*=0, i.e. starting at node (*l,k*), can be assigned directly to the corresponding mirror variables

$$y^{(l,k)}{}_{l\_j,0} = x_{l\_j,k} \qquad \forall l, j, k \ (0 \ldots n-1),\ l \neq j \qquad (3)$$

2. There have to be edges within each row of mirror *Y*(*l,k*) with overall weight equal to the weight of node (*l,k*)

$$\sum_{i=0}^{n-1} \sum_{j=0, i \neq j}^{n-1} y^{(l,k)}{}_{i\_j,d} = w(l,k) = \sum_{j'=0, j' \neq l}^{n-1} x_{l\_j',k} \qquad (4)$$

$$\forall d \ (0 \ldots n-1),\ \forall l, k \ (0 \ldots n-1)$$

Constraints to build up mirrors (2)

3. The sub-routes in the mirror have to be consecutive

$$\sum_{i=0, i \neq h}^{n-1} y^{(l,k)}{}_{i\_h,d} = \sum_{j=0, j \neq h}^{n-1} y^{(l,k)}{}_{h\_j,d+1} \qquad (5)$$

$$\forall l,k,h,d \ \ (0 \ldots n-1)$$

4. Each graph edge has to be represented by the combined mirrors of a particular column

$$\sum_{l=0}^{n-1} y^{(l,k)}{}_{i\_j,d} = x_{i\_j,k+d} \qquad (6)$$

$$\forall i,j,k,d \ \ (0 \ldots n-1)$$

Annotation: Operator '+' applied on variable indices means the modulo sum with basis *n* such that the result is always 0...*n*-1

Constraints to build up mirrors (3)

5. The sub-routes in the mirror have to be symmetric
   → all stations have to be reached

$$\sum_{d=0}^{n-1} \sum_{i=0, i \neq j}^{n-1} y^{(l,k)}{}_{i\_j,d} = w(l,k) \qquad \forall j \ (0 \ldots n-1) \tag{7}$$

That's it!

Conclusion

- MERLINn uses
  - $O(N^5)$ variables and
  - $O(N^4)$ constraints

  to define a Linear Formulation of the Travelling Salesman Problem
- It is a general applicable approach for the TSP
- LP is known to be polynomial
- <u>TSP solvable in a polynomial number of steps as well</u>

P=NP!

References

[1] Mertz, J.: 'The Dragon War', Applied Mathematics and Computation, Vol. 186/1, 1 March 2007, pg. 907-914

http://www.sciencedirect.com/science/journal/00963003

[2] MERLIN-page with further information (validation results, LP-formulations, examples …)
http://www.merlins-world.de

[3] Download of LP solver QSOpt

http://www2.isye.gatech.edu/~wcook/qsopt/downloads/downloads.htm