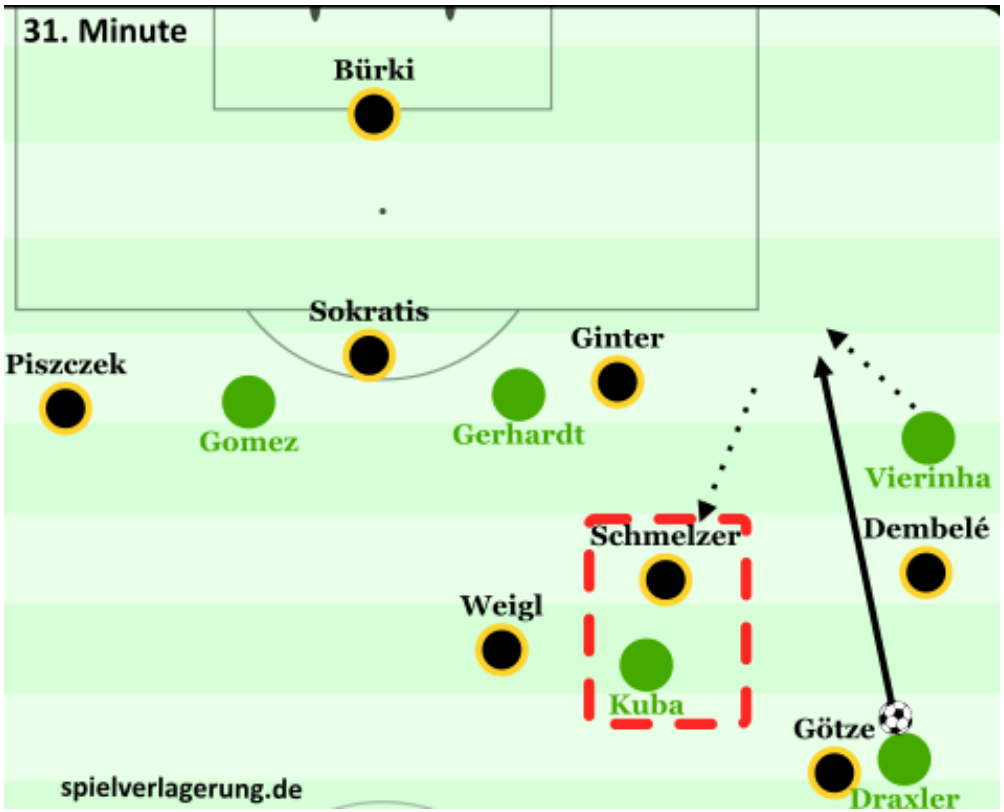
Universität Rostock Traditio et Innovatio

Spatiotemporal Pattern Matching in RoboCup

Tom Warnke **Adelinde Uhrmacher**

University of Rostock, Institute of Computer Science
Modeling and Simulation Group

Spatiotemporal Patterns in Football Tactics



Source: spielverlagerung.de

Spatiotemporal Pattern Matching in RoboCup

Contributions

Agents moving simultaneously give rise to spatiotemporal patterns. How to describe and find such patterns?

We

- model the movement data as a data graph
- describe spatiotemporal patterns as graph patterns
- find these graph patterns in the data graph

Starting Point

Raw data:

- RoboCup[1]: coordinates of each player and the ball are saved every 100 ms
- Real football: similar data is gathered (but largely inaccessible)

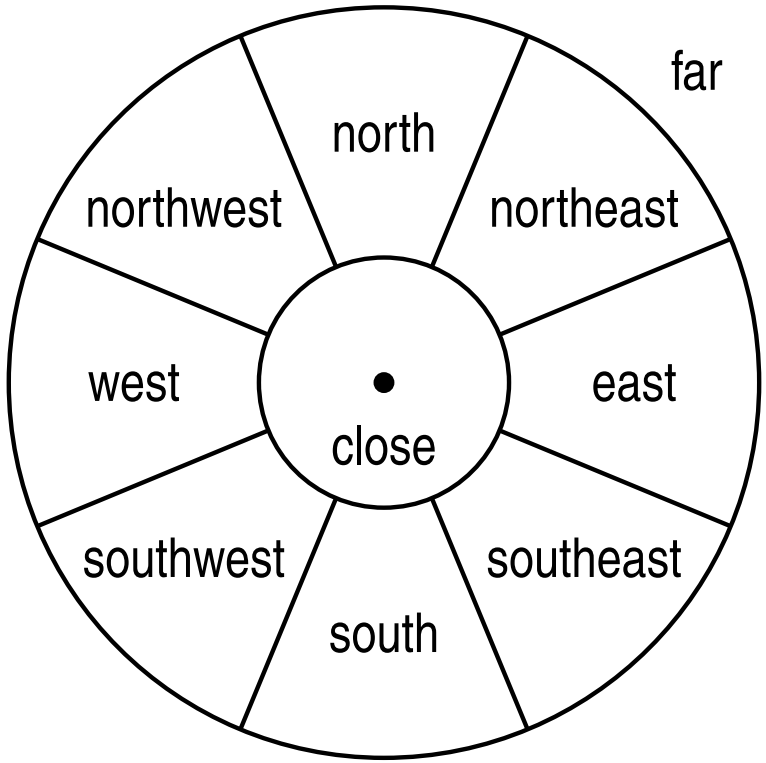
Patterns involve:

- Players of both teams and the ball
- Development during a (typically short) time span
- Relative position rather than absolute

[1] RoboCup 2D Soccer Simulation League

Modeling Relative Positions

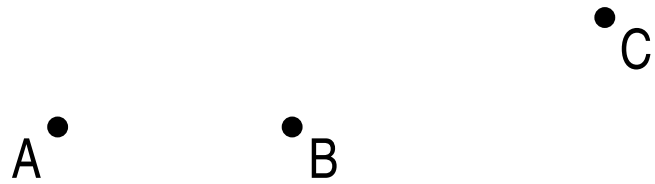
Partitioning of the Space Surrounding an Object



Frank, A.U.: Qualitative spatial reasoning about distances and directions in geographic space. Journal of Visual Languages & Computing 3(4), 343–371 (1992)

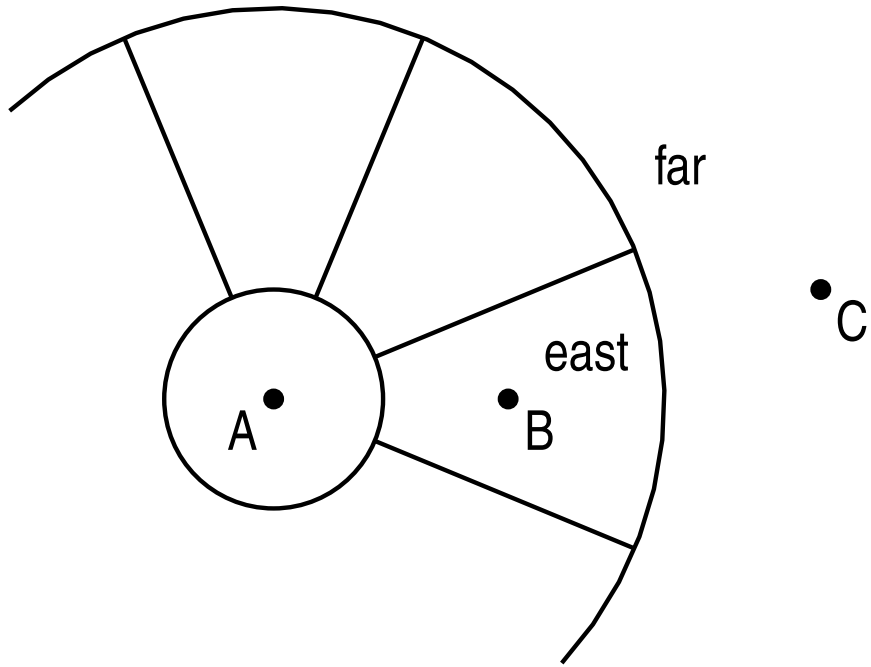
Creating the Graph

Three Object Positions in Space



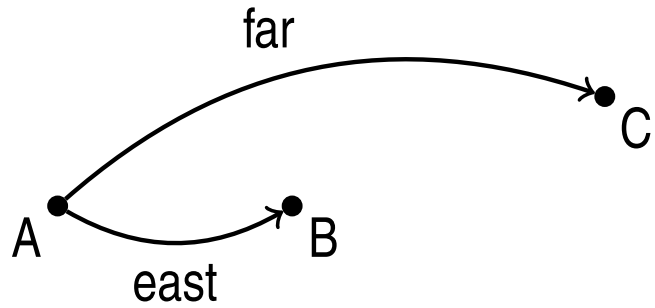
Creating the Graph

Three Object Positions in Space



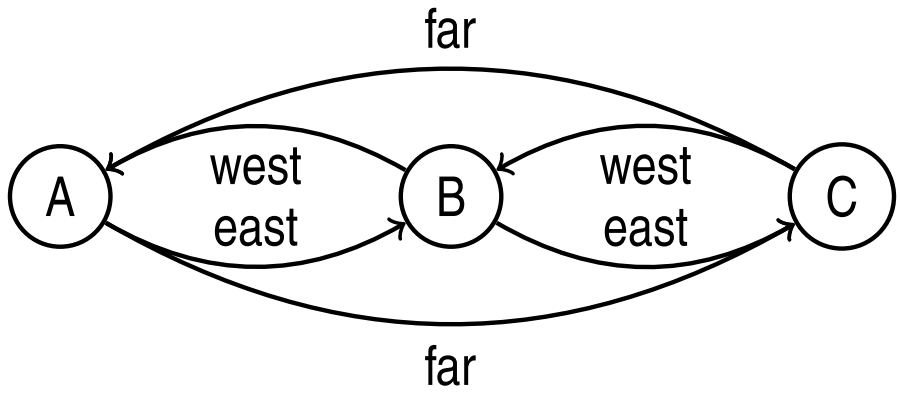
Creating the Graph

Relation Graph



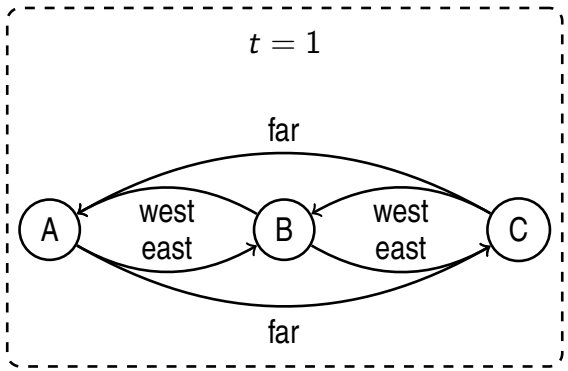
Creating the Graph

Complete Relation Graph



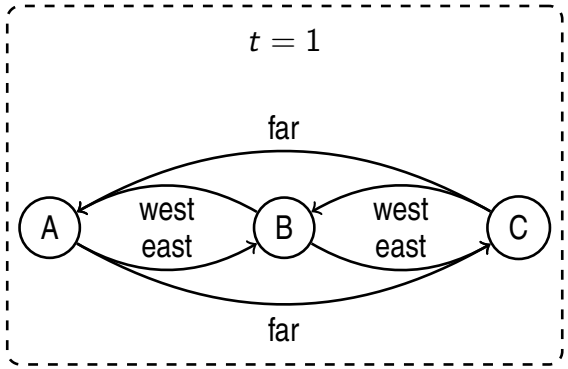
Creating the Graph

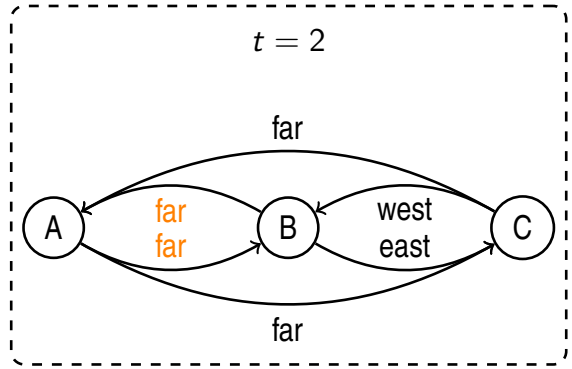
One Relation Graph for Each Time Point



Creating the Graph

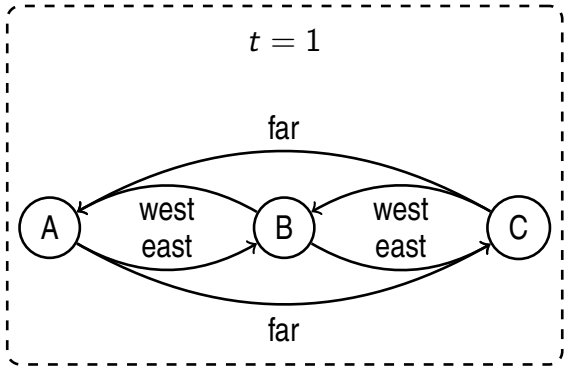
One Relation Graph for Each Time Point

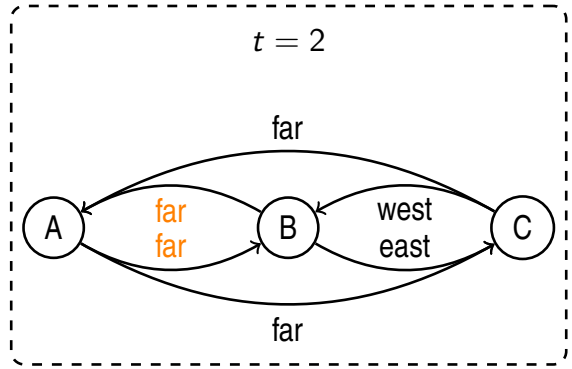


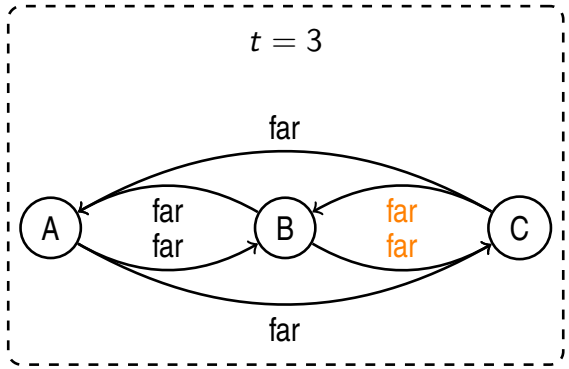


Creating the Graph

One Relation Graph for Each Time Point





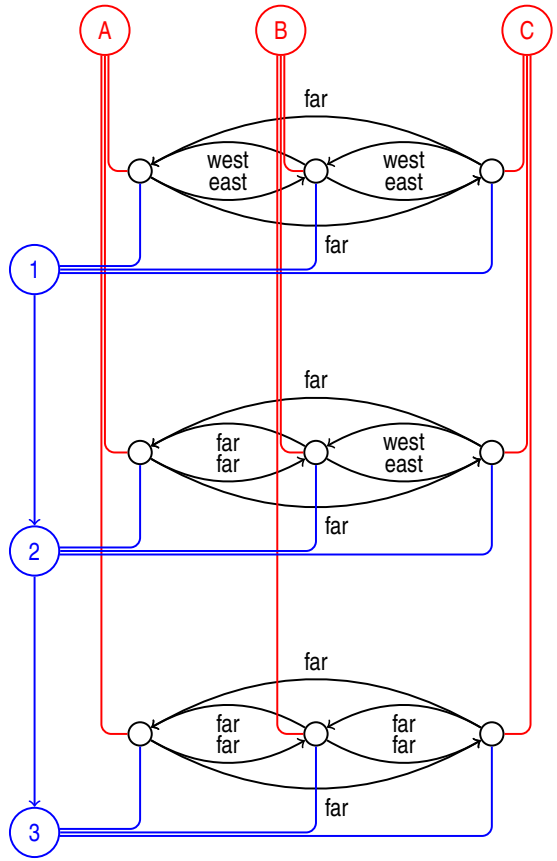


Intermission

Simple patterns (without change over time) are already findable in these graphs (e.g., three objects in a straight line from east to west).

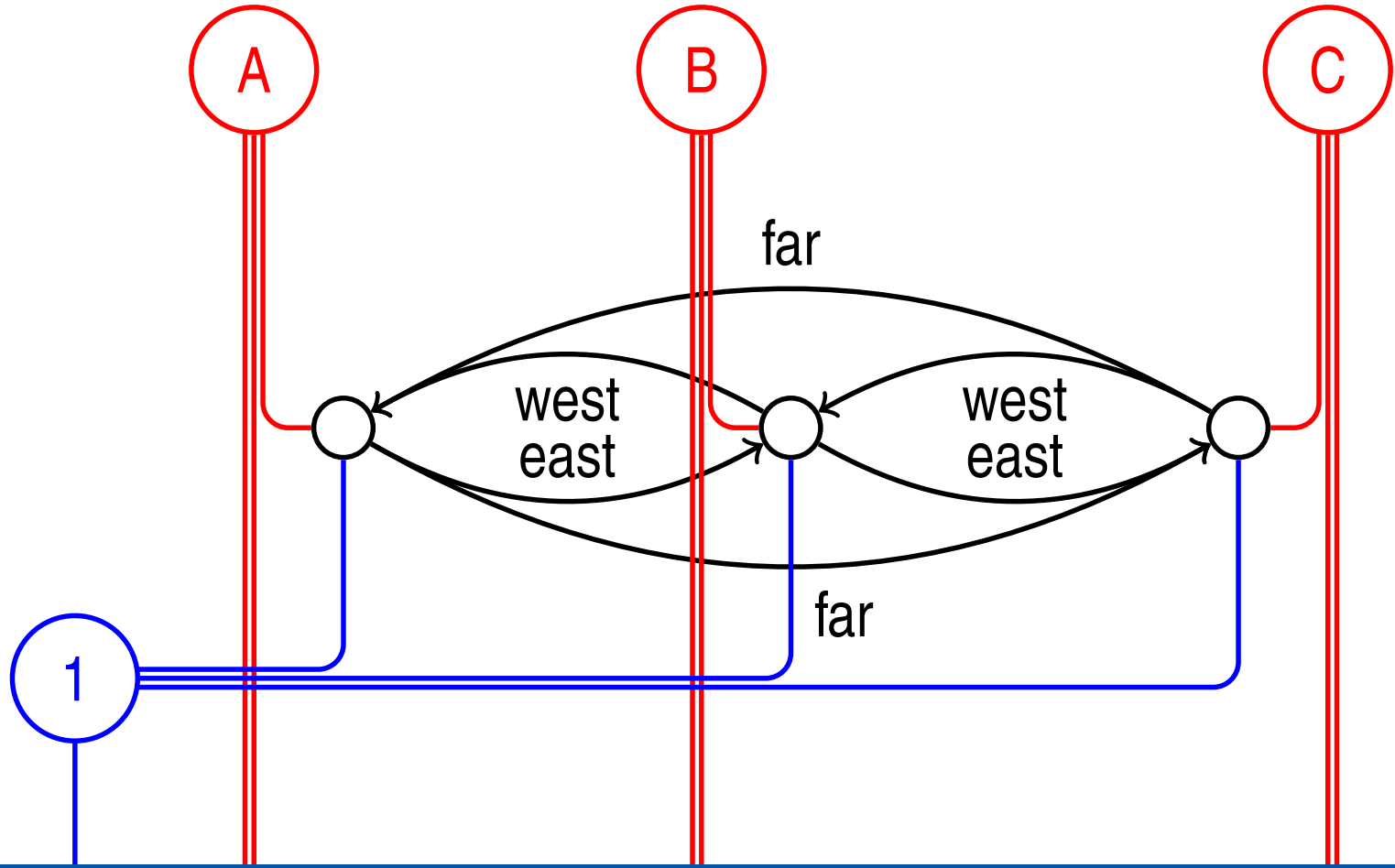
To include change over time, we have to connect the single graphs.

The Data Graph



The Data Graph

Zoom

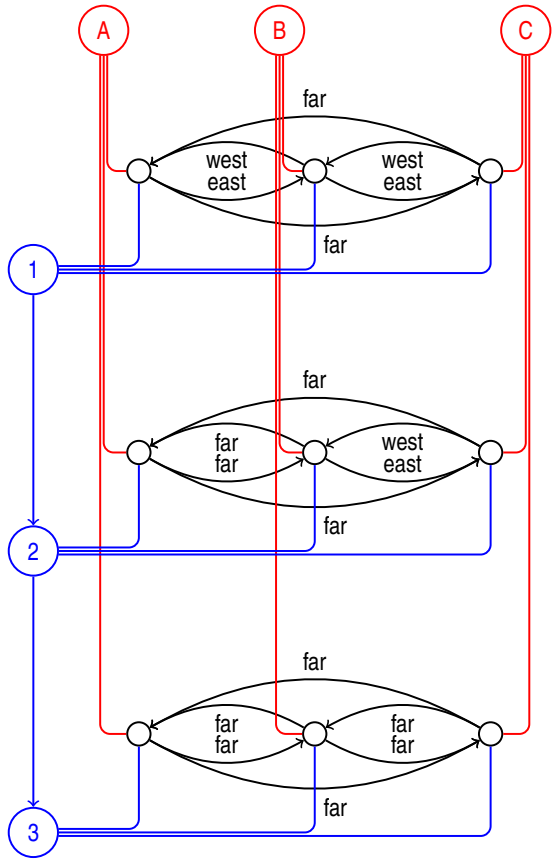


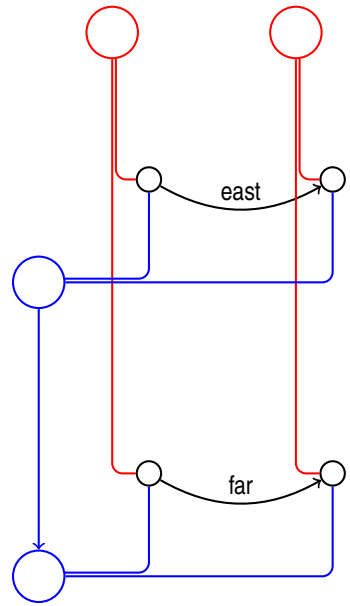
Patterns in the Data Graph

All relevant movement data is captured in the graph.

Movement patterns can be defined as graph patterns.

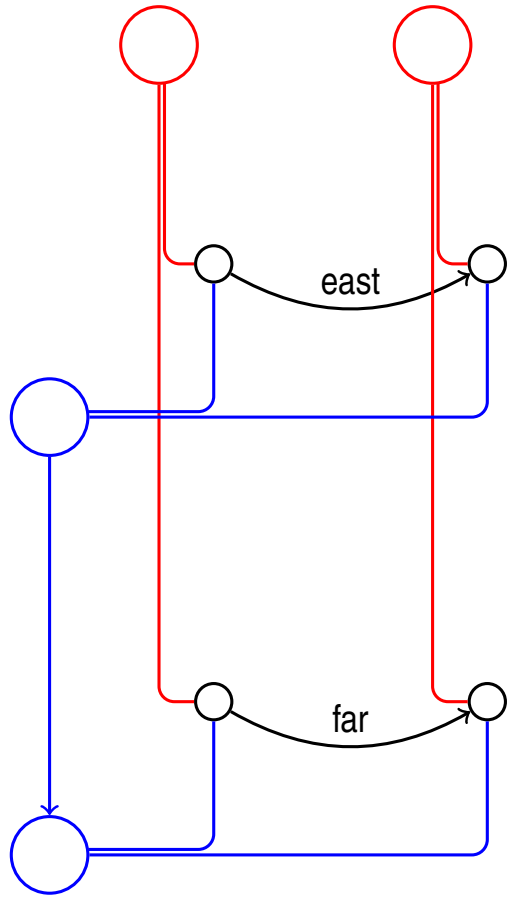
Patterns in the Data Graph





Patterns in the Data Graph

Zoom



Finding Patterns in the Data

- Ullmann's algorithm[2], a classic subgraph isomorphism algorithm
  - Custom Java implementation
  - Match nodes one by one, use backtracking
  - Heavy optimization for the data graph structure
- Neo4j[3], a graph database system with a query language
  - Well-established and actively developed open-source software
  - Powerful query language Cypher allows for succinct pattern definitions
  - Advanced features (e.g., joins, where clauses, paths of variable length)

[2] Ullmann, J.R.: An Algorithm for Subgraph Isomorphism. Journal of the ACM 23(1), 31–42 (1976)
[3] neo4j.com

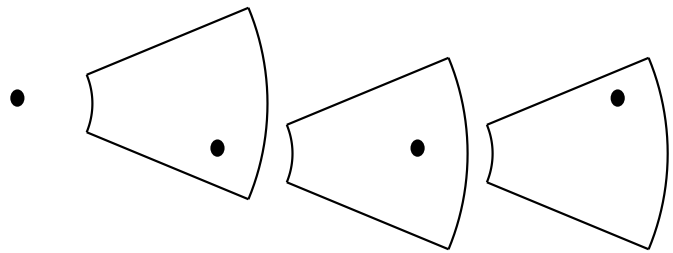
Proof of concept

Evaluation on RoboCup Simulation Data

- Simulate and monitor RoboCup matches of the team WrightEagle[4] against itself
- Construct data graph from gathered data
- Find patterns in the data
  - „Back four“
  - „Getting past a defender“
- Source code available at `git.informatik.uni-rostock.de/mosi/RobocupAnalysis`

[4] `http://ai.ustc.edu.cn/2d/`

Pattern I: The back four



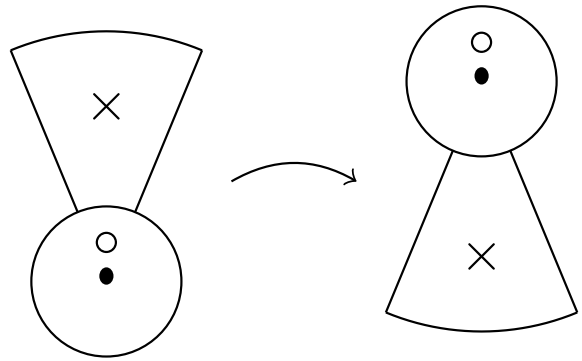
- Players must be in the same team
- Must hold for 5 (10) consecutive time points
- Neo4j took 2 (3) minutes, Ullmann's algorithm 8 (12) minutes
- Majority of pattern occurrences found in the defensive lines (~30% each)

Cypher query

Back four

```
MATCH (l1:LOCATION)-[:EAST]->(l2:LOCATION)-[:EAST]->
(l3:LOCATION)-[:EAST]->(l4:LOCATION),
(l1)--(one:OBJECT), (l2)--(two:OBJECT),
(l3)--(three:OBJECT), (l4)--(four:OBJECT),
(l1)--(t:TIME_POINT)
WHERE one.team = two.team AND two.team = three.team AND three.team
    = four.team
WITH one, two, three, four, collect(t) AS times
MATCH path=(start:TIME_POINT)-[:NEXT_TIME_POINT*4]->(end:TIME_POINT
    )
WHERE ALL (t in nodes(path) WHERE t in times)
RETURN one.id, two.id, three.id, four.id, start.time
```

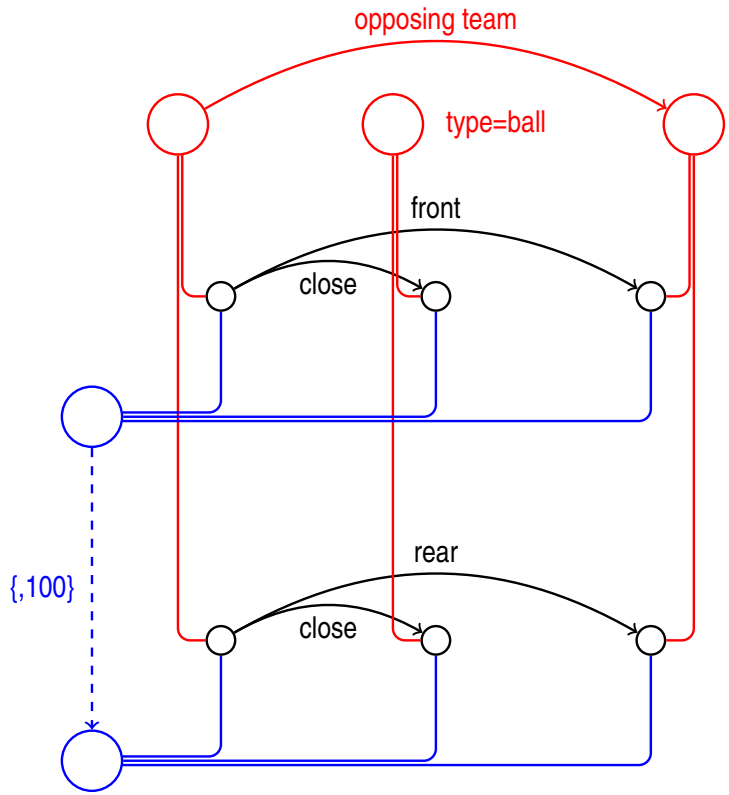
Pattern II: Getting past a defender



- Players in opposing teams, movement towards opponents goal
- Second state must be reached at most 100 steps ($= 10s$) after the first
- Neo4j took less than one minute, Ullmann's algorithm does not support paths of variable length
- Most occurrences involve the central forwards

Pattern Graph

Getting past a defender



Conclusions

- With an appropriate partition of space, spatiotemporal data and patterns can be transformed to graphs and graph patterns.
- Graph patterns are declarative.
- Non-spatial information can easily be integrated.
- Many methods for pattern definition and finding exist and can be exploited.
- Neo4j is a useful tool, but its correct use is not so easy.
- Our approach is useful for *coordinated* simultaneous movement of agents.