FFI Forsvarets forskningsinstitutt
Norwegian Defence Research Establishment

# Hierarchical path planning for walking (almost) anywhere

FOSS4G 2018, Dar es Salaam
Knut Landmark, Espen Messel

# Introduction and overview

- Problem and goal
- Prospective applications
- Open data (vs. restricted)
- Graph partitions, hierarchical method
- Accuracy
- Cost of walking

Detailed National Elevation Model, http://kartverket.no/ (Norwegian Mapping Authority)

# Problem and goal:

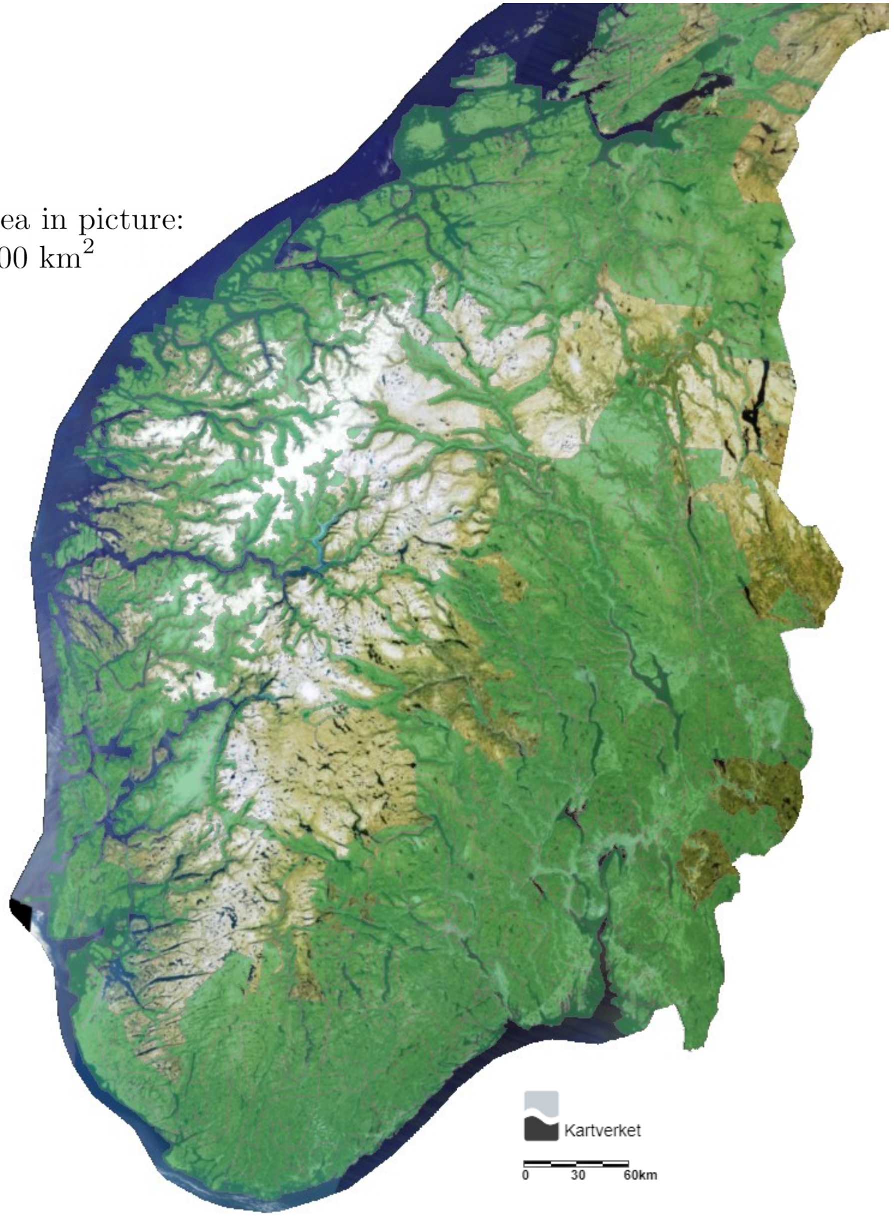
- Find route between two arbitrary land area positions.
- Take into account
    - transport networks
    - topography
    - terrain type, land cover/use
    - infrastructure.
- Exploit open-access LiDAR data.
- Arbitrary distance and resolution.

# Open-access LiDAR data

- National initiatives
- All-covering aerial LiDAR survey projects
- Open access policy
- «Transparency, efficiency, innovation»
- UK:
    - UK Environmental Agency
    - Open Government Licence
    - https://data.gov.uk/
- Norway:
    - Norwegian Mapping Authority
    - National Detailed Elevation Model
    - Creative Commons 4.0
    - https://hoydedata.no/LaserInnsyn/

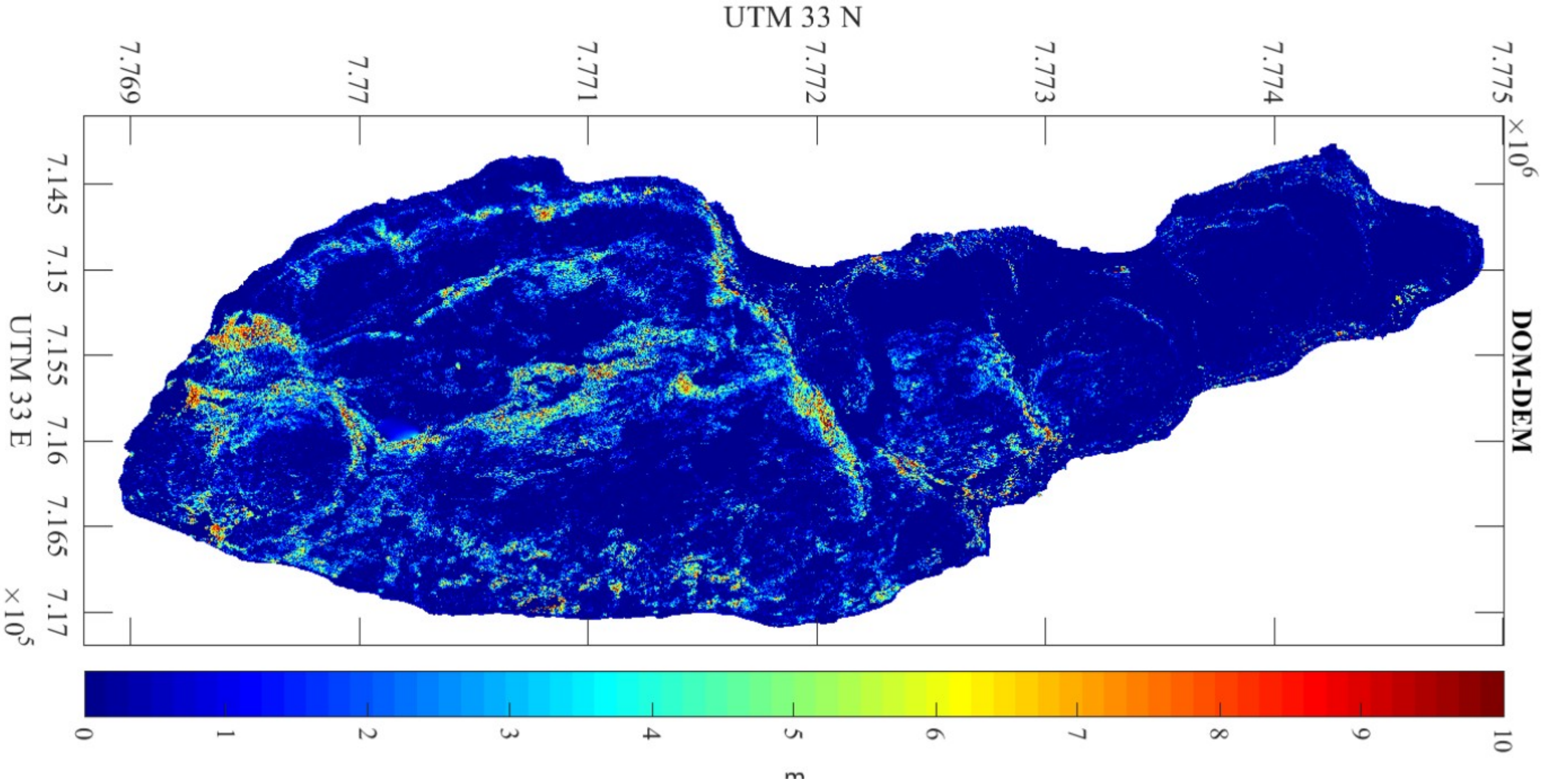
DATA.GOV.UK Beta
Opening up Government

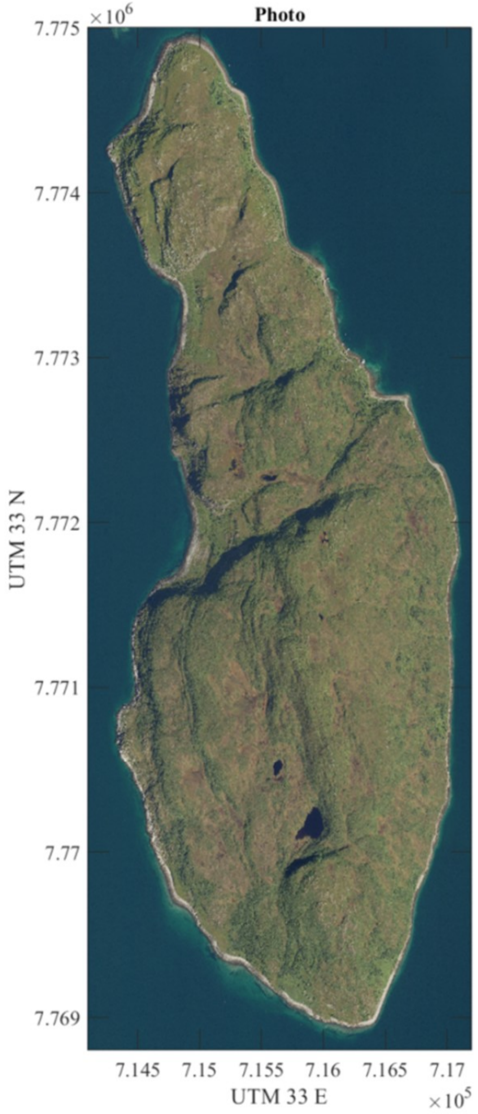
Kartverket



Land area in picture:
$\sim$ 200.000 km$^2$

# LiDAR-based DSMs and DEM enable realistic path planning





# Applications and prospects

- Archaeology, ancient networks
- Search and rescue
- Public transit planning
- Forestry
- Military operations
- Hiking, exercise, recreation
- Robotics, autonomous systems
- Animal migration patterns
- etc.

The Telegraph

News | Science

**Lost Roman roads could be found as Environment Agency laser scans whole of England from air**

S. Knapton, The Telegraph, 30. Dec. 2017

# Two challenges in cross-country path planning

- Realism, how to adapt data
  - Accurate weight (cost) function
  - Is a stream traversible; if so, at what cost?
  - What about a fence? Etc.
  - Answer not always found in data.
- Computational:
  - Manage very large graphs
  - Find optimum solutions efficiently

Image: US National Park Service, nps.gov

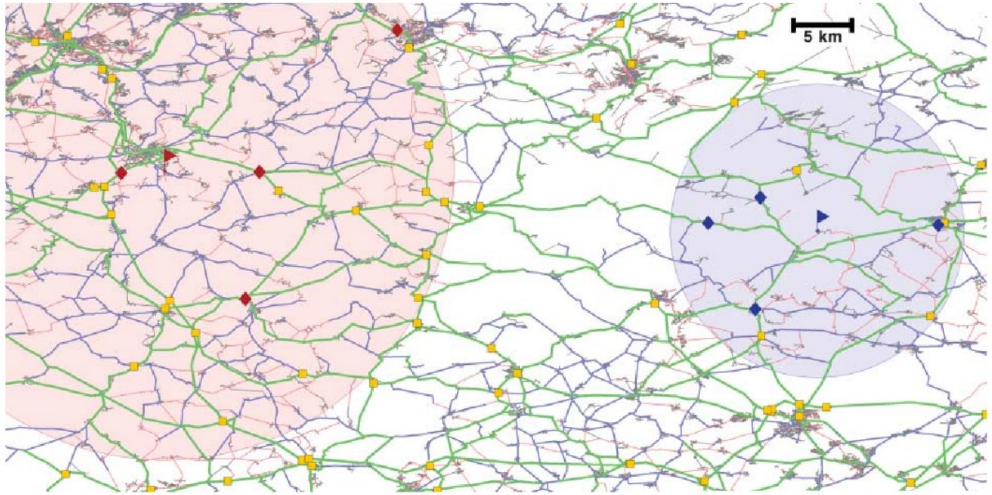
# Two challenges in cross-country path planning

- Realism, how to adapt data
  - Accurate weight (cost) function
  - Is a stream traversible; if so, at what cost?
  - What about a fence? Etc.
  - Answer not always found in data.
- Computational:
  - Manage very large graphs
  - Find optimum solutions efficiently

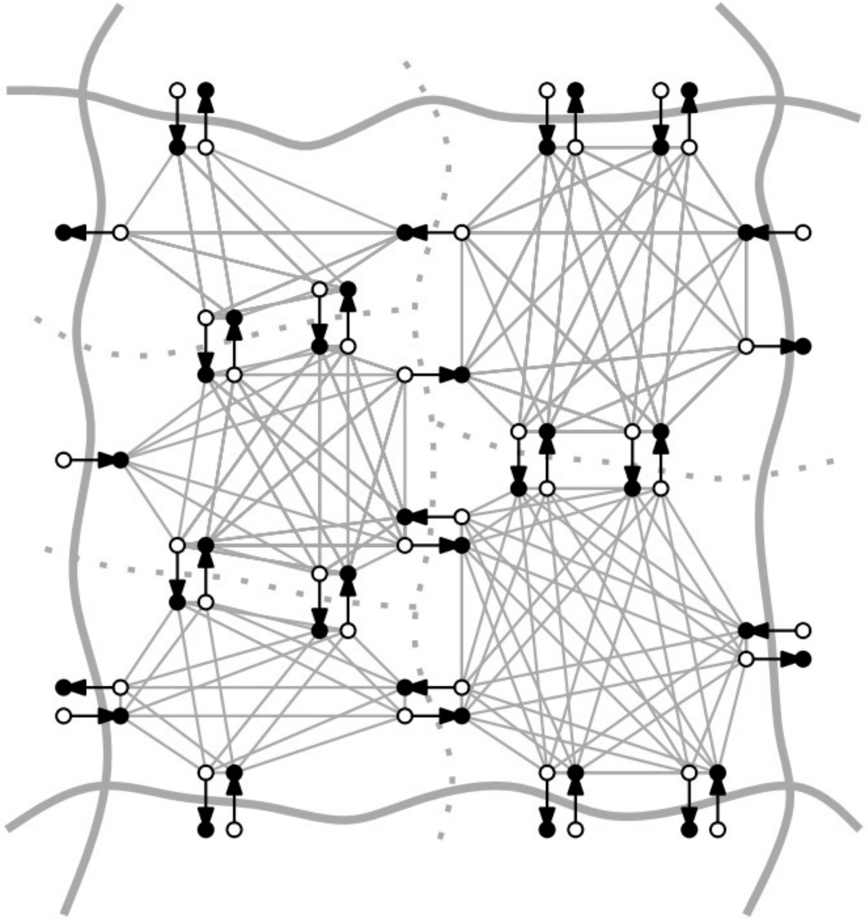
Image: www.flickr.com

# Road network routing and graphs

- Natural hierarchical structure
- Overlays, multiple levels
- Transit nodes
- Table look-ups
- Contractions
- Orders of magnitude faster than Dijkstra



Bast et al. (2007)

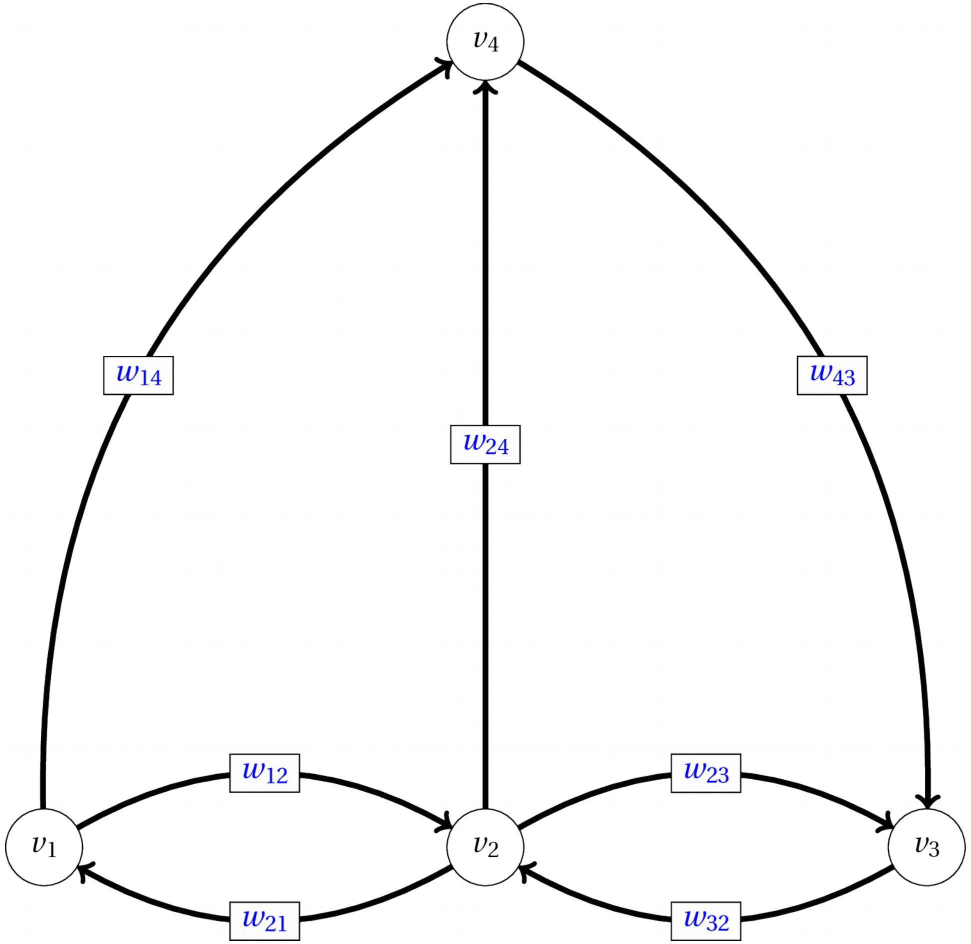


Delling and Werneck (2013)

# Matrix-graph duality

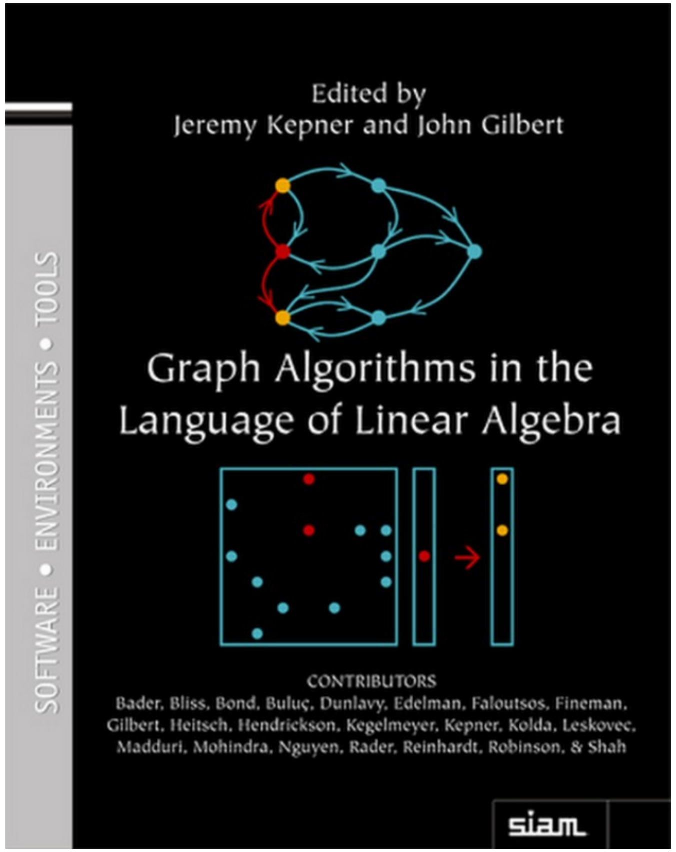
$$G = \begin{array}{cl} \overset{\text{Node index } j}{\xrightarrow{\hspace{3cm}}} & \\ \begin{bmatrix} 0 & w_{12} & 0 & w_{14} \\ w_{21} & 0 & w_{23} & w_{24} \\ 0 & w_{32} & 0 & 0 \\ 0 & 0 & w_{43} & 0 \end{bmatrix} & \Big\downarrow \text{Node index } i \end{array}$$

$$v_i = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \Big\downarrow \text{row } i$$



# Matrix-graph duality (no parallel arcs)

- Breadth-first search ↔ matrix multiplication
- Sparse matrix representation
- Adjacency matrix
- Vectorization
- Array-based software



SOFTWARE • ENVIRONMENTS • TOOLS

Edited by
Jeremy Kepner and John Gilbert

Graph Algorithms in the
Language of Linear Algebra

CONTRIBUTORS
Bader, Bliss, Bond, Buluç, Dunlavy, Edelman, Faloutsos, Fineman,
Gilbert, Heitsch, Hendrickson, Kegelmeyer, Kepner, Kolda, Leskovec,
Madduri, Mohindra, Nguyen, Rader, Reinhardt, Robinson, & Shah

siam

NumPy

SciPy

julia

GNU Octave

MathWorks®

# Assumptions

- Graph $G = (V, E, w)$, $V = \{ v_1, v_2, \ldots, v_N \}$
- Directed
- Positive weight
- No self-loops
- Connected (consider each component by itself)
- (Simple)
- Spatial position $x(v)$

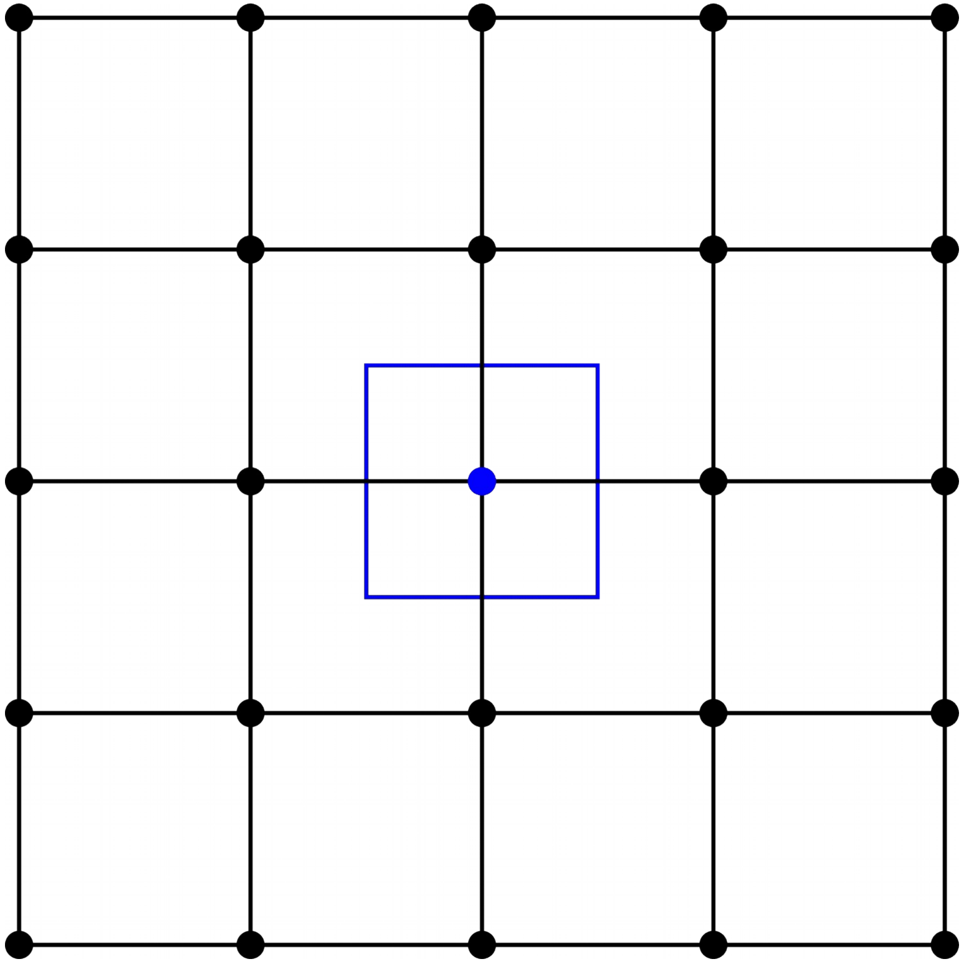
# Generalized path planning with graphs

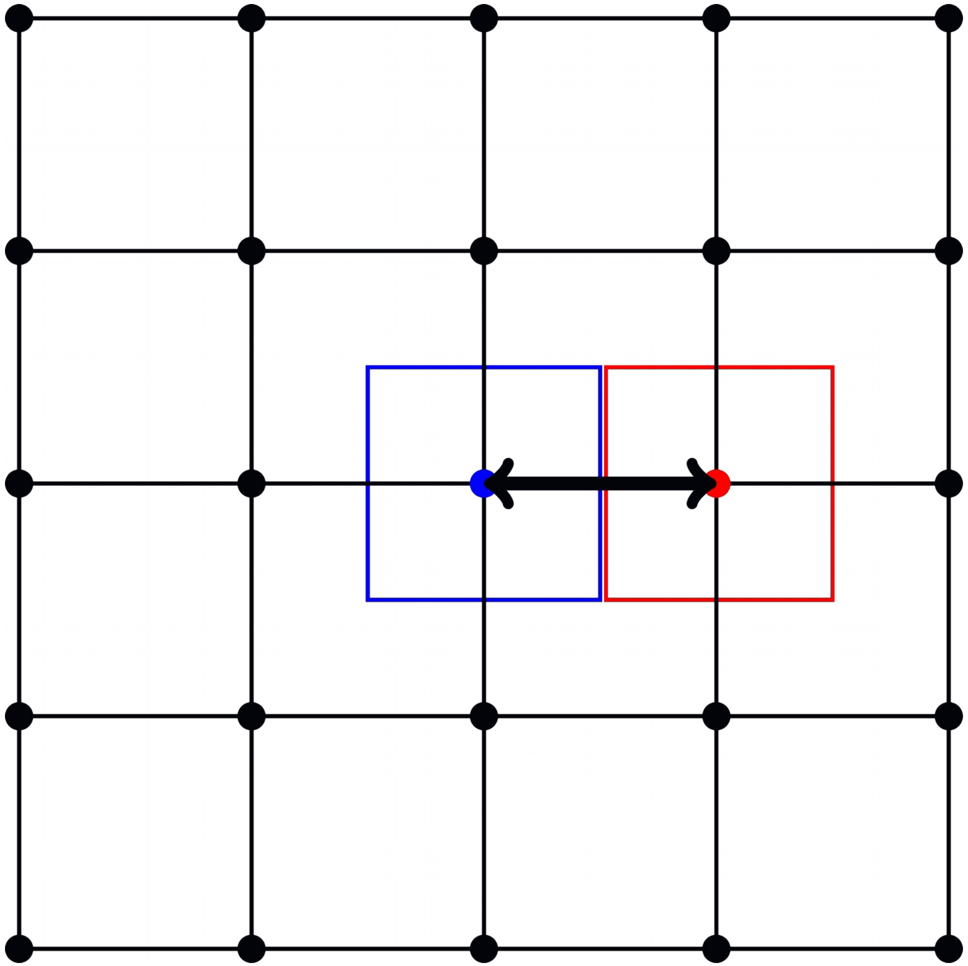
- No natural hierarchical structure
- Additional degree(s) of freedom
- High node density (almost) everywhere
- Finite position accuracy

## What is accuracy?

1. Node position close to true position.
2. Length of graph solution ≈ length of real (actual) optimum path.

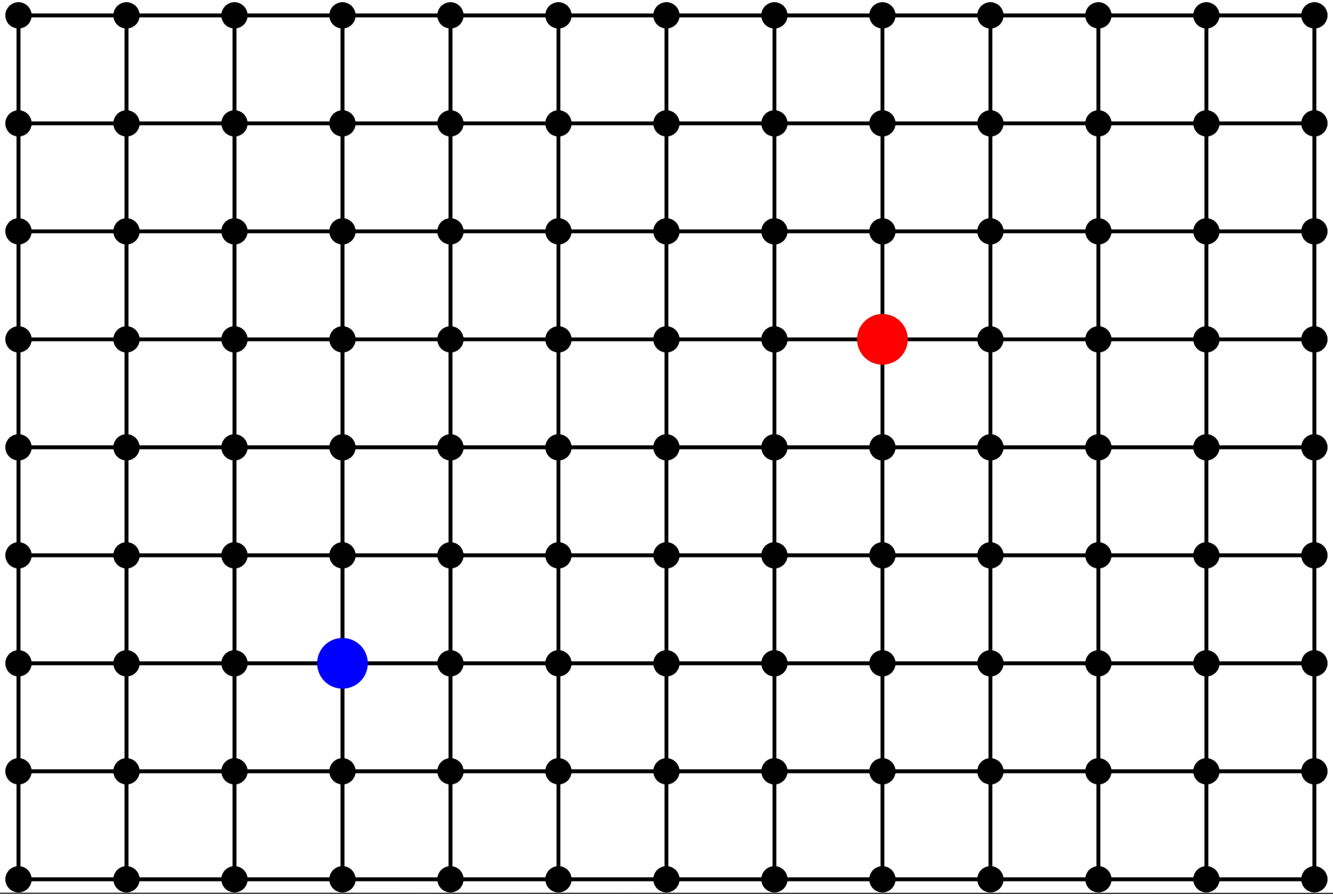
- Accuracy is an issue in all graph-based approaches!
- Discrete model allows finite set of positions, movements

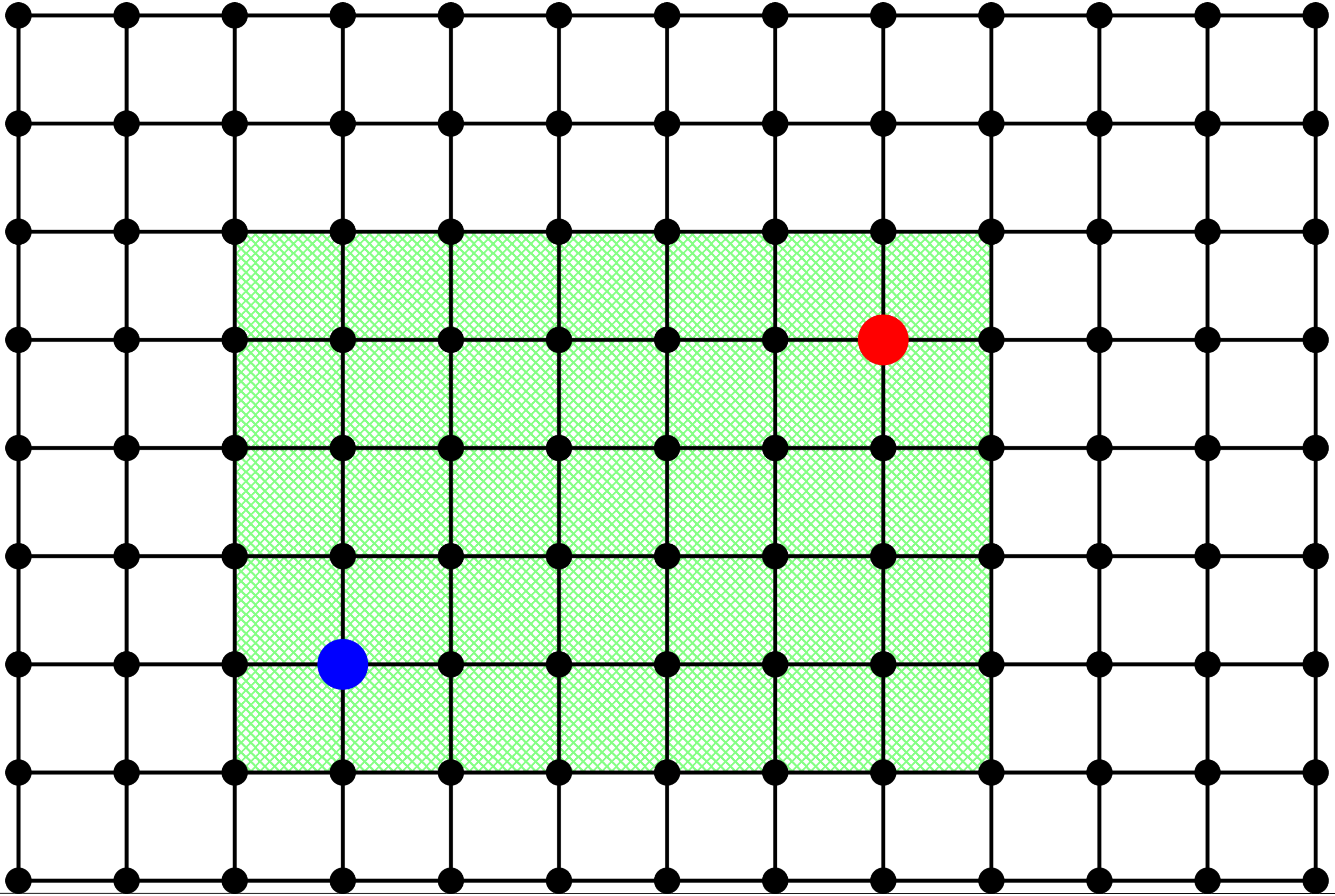


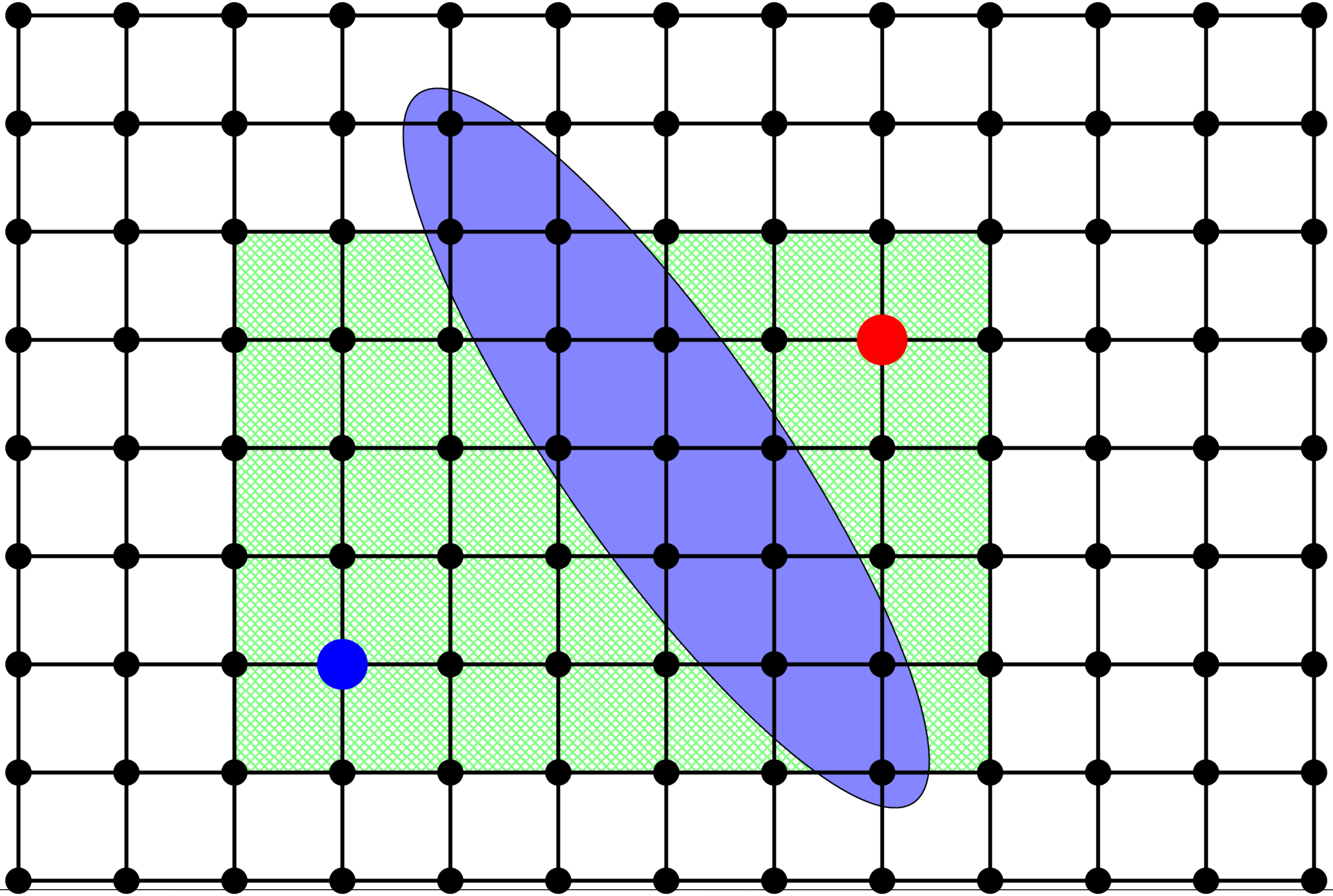


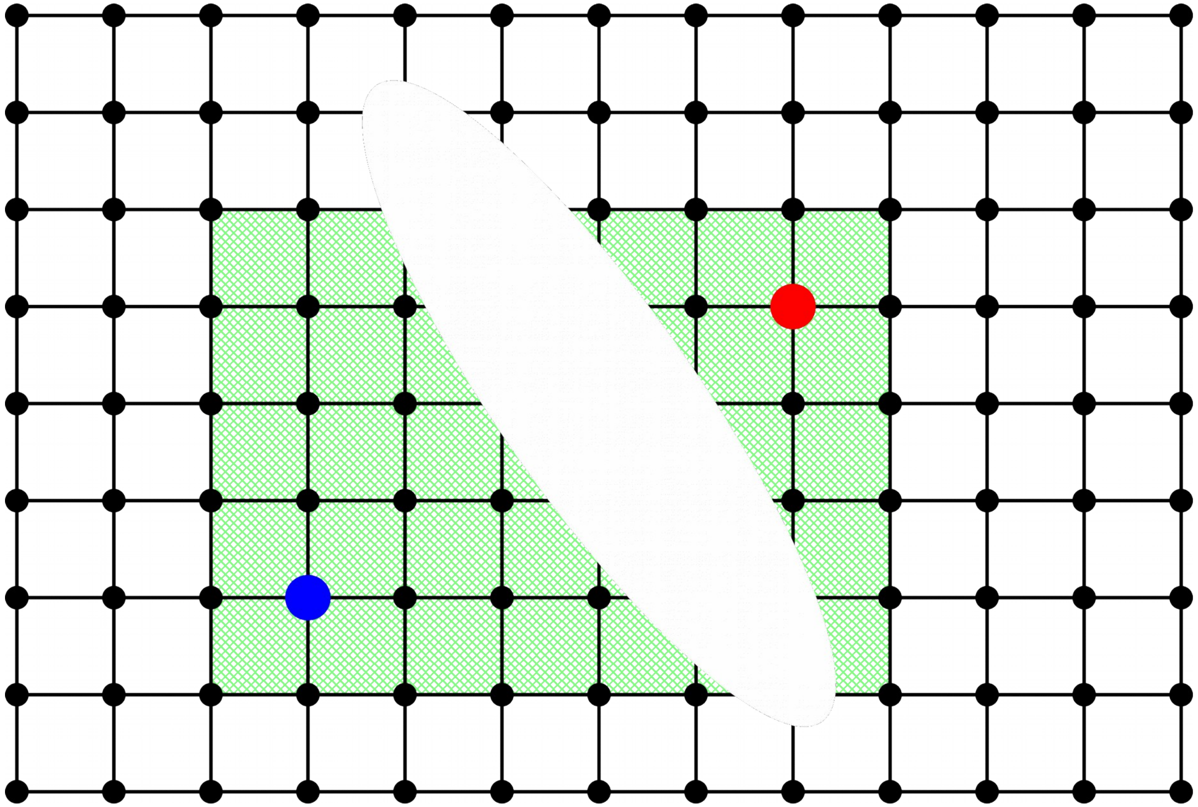
# Simple strategy

- Very detailed basal graph (eliminates position uncertainty)
- Restrict search to small part of basal graph



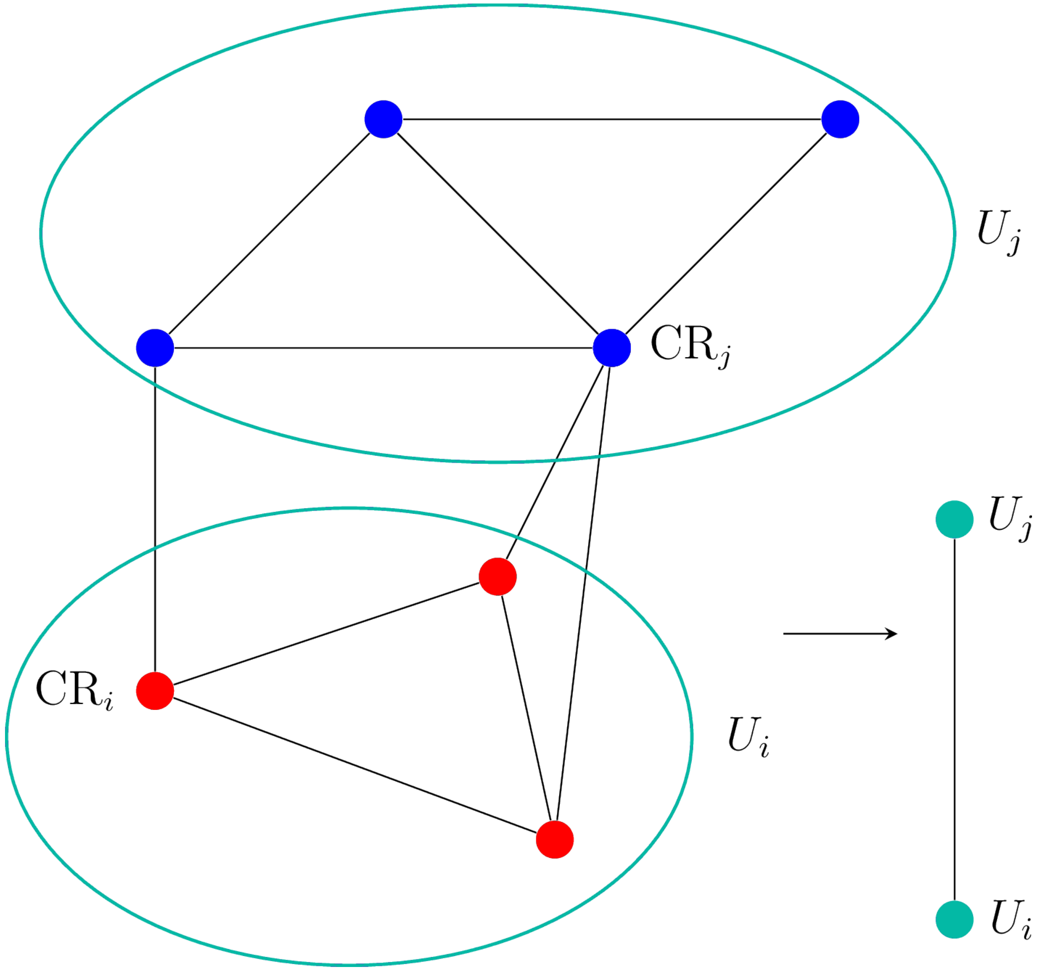






- Must reduce search space.
- Subgraphs may become *disconnected*.
- Consequently, no solution is found.

# Aggregate nodes



# Modified strategy

- For a graph $G_0$ with $N$ nodes, partition $G$ into $K < N$ connected components $U_1, U_2, \ldots U_K$.
- Form a simplified graph $G$ with nodes $V(G_1) = \{ U_1, U_2, \ldots, U_K \}$.
- Control the "size" (graph-theoretical radius) of components (accuracy!).
- Expand a solution in $G_1$ to obtain a reduced search space (subgraph) in $G_0$.
- Can be iterated several times to form hierarchy of simplified graphs, $\mathcal{G} = \{ G_0, G_1, \ldots G_h \}$.
- A solution in $G_0$ is found (if it exists).

# Complexity-accuracy trade-off

- As small radius ($R$) as possible
- As few components ($K$) as possible

# *k*-center problems

- Find partition $V(G) = \cup_{k=1}^{K} U_k$, each $G\big|_{U_k}$ is (strongly) connected, and

  C1: Minimum $K$ such that $\max_{1 \leq k \leq K} r(U_k) \leq R$ for fixed radius $R$.

  C2: Smallest radius $R$ such that $\max_{1 \leq k \leq K} r(U_k) \leq R$ for a fixed $K$.

- NP-hard optimization: $k$-center problems

# *k*-centre algorithm

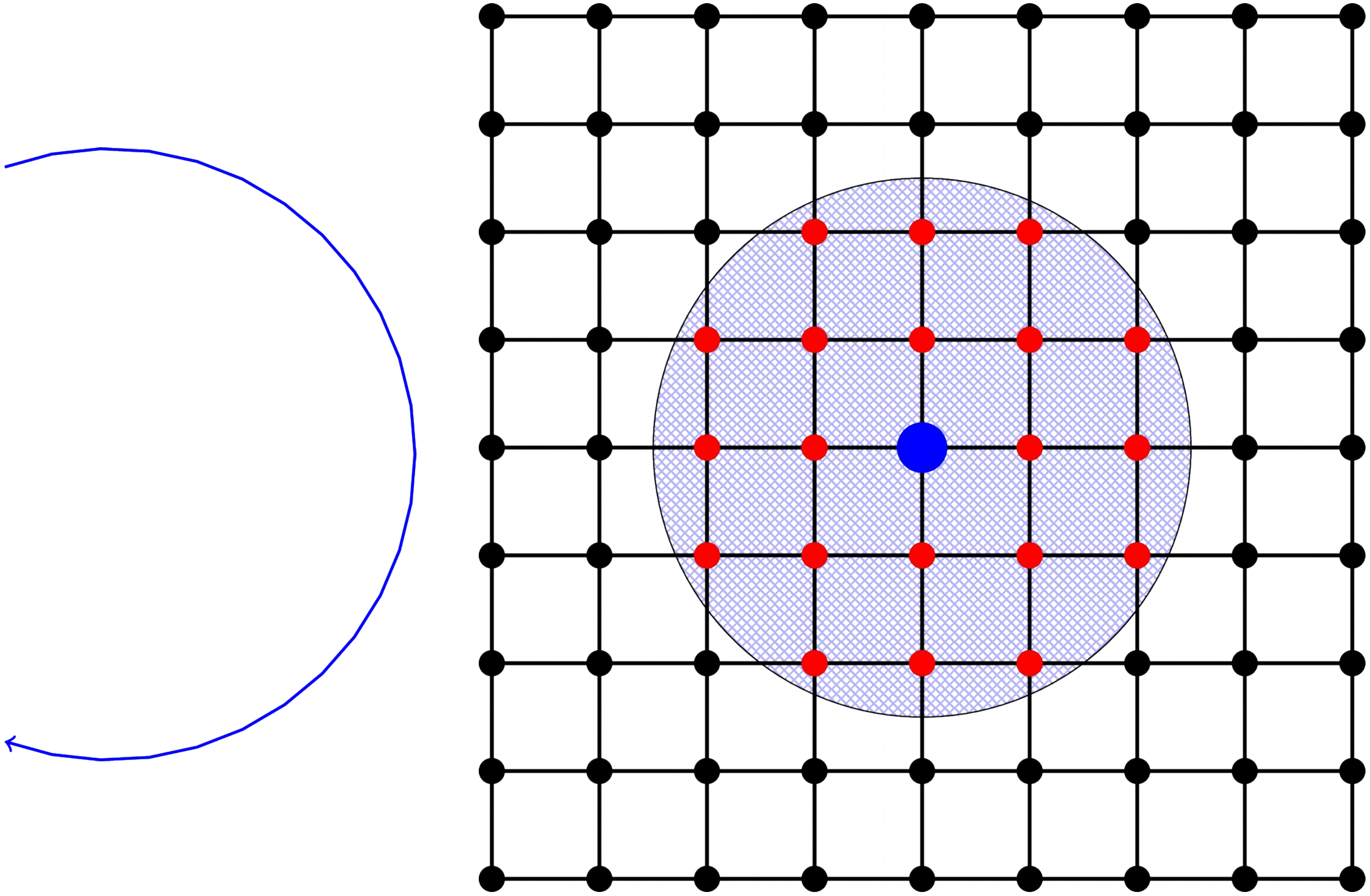
Find $K$ cluster representatives (CRs) $H = \{ h_1, \ldots, h_K \}$ and assign nodes to their nearest CR.

- Start with one random CR, $H = \{ h_1 \}$.
- Compute distance from $h_1$ to all nodes.
- Add the $H$ the node farthest from present clusters
- Recompute cluster assignmments for all nodes.

# *k*-center algorithm

- Requires at most $K-\max$ single-source all-shortest-path computations.
- In each iteration, can store disance from new CR to its vicinity (in $K-\max \times N$ sparse matrix).
- From this we obtain edge weights in simplified graph.

# The Dijkstra wavefront



https://timrwalls.wordpress.com/

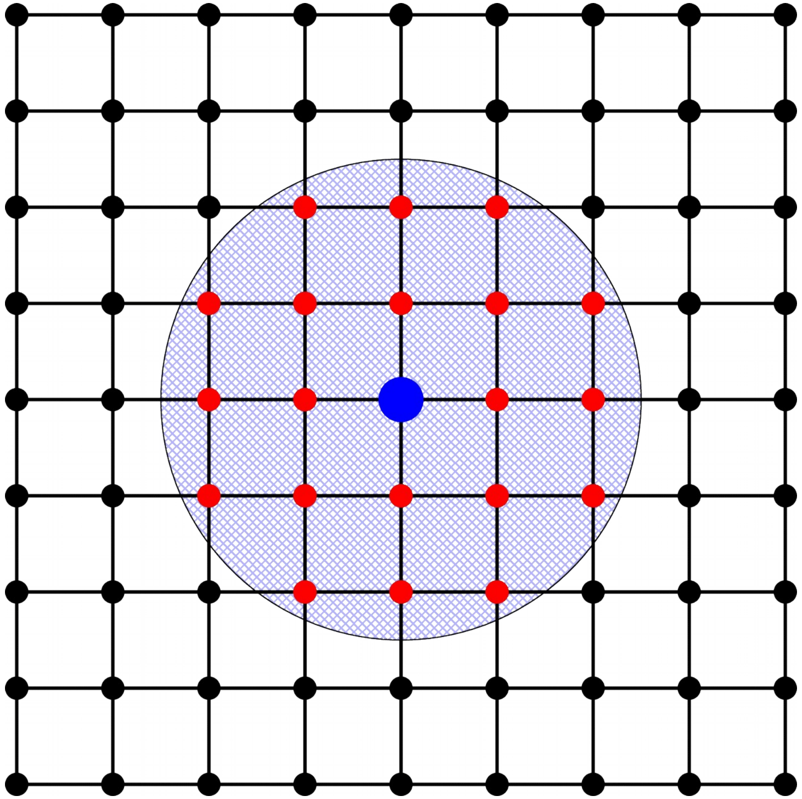
**1** Modified Dijkstra loop

```
$s \leftarrow \{$ source node $\}$
$N \leftarrow$ number of nodes
$\Lambda \leftarrow$ cut-off distance
$c \leftarrow 0$
$U \leftarrow \{$ all nodes $\}$
**while** $U$ not empty **do**
    $u \leftarrow$ node $v \in U$ with minimum distance $d(v)$
    $U \leftarrow U \setminus \{ u \}$

    // Additional work:
    $c \leftarrow \frac{N}{N-1} c + \frac{1}{N} d(u) / d_{\mathrm{EUCL}}(u)$
    **if** $d(u) > \Lambda$ **then**
        **break**

    // Update distances:
    **for all** $v \in \{$ active neighbors of u $\}$ **do**
        $d(v) = \min(d(u) + \mathrm{weight}(u, v), d(v))$

**for all** $v \in U$ **do**
    $d(v) = \max(\Lambda, c^{-1} d_{\mathrm{EUCL}}(v))$
```



# Two extreme cases

- For a hierarchical graph $\mathcal{G} = \{ G_0, G_1, \ldots G_h \}$, accuracy of solution depends on choice of search levels $0 \leq h_1 \leq \ldots \leq h$.
- If $h = 0$, exact solution is found on complete basal graph (at high cost).
- If start and end nodes are adjacent in $G_h$, the distance *uncertainty* at level $h$ has the same magnitude as the true distance.

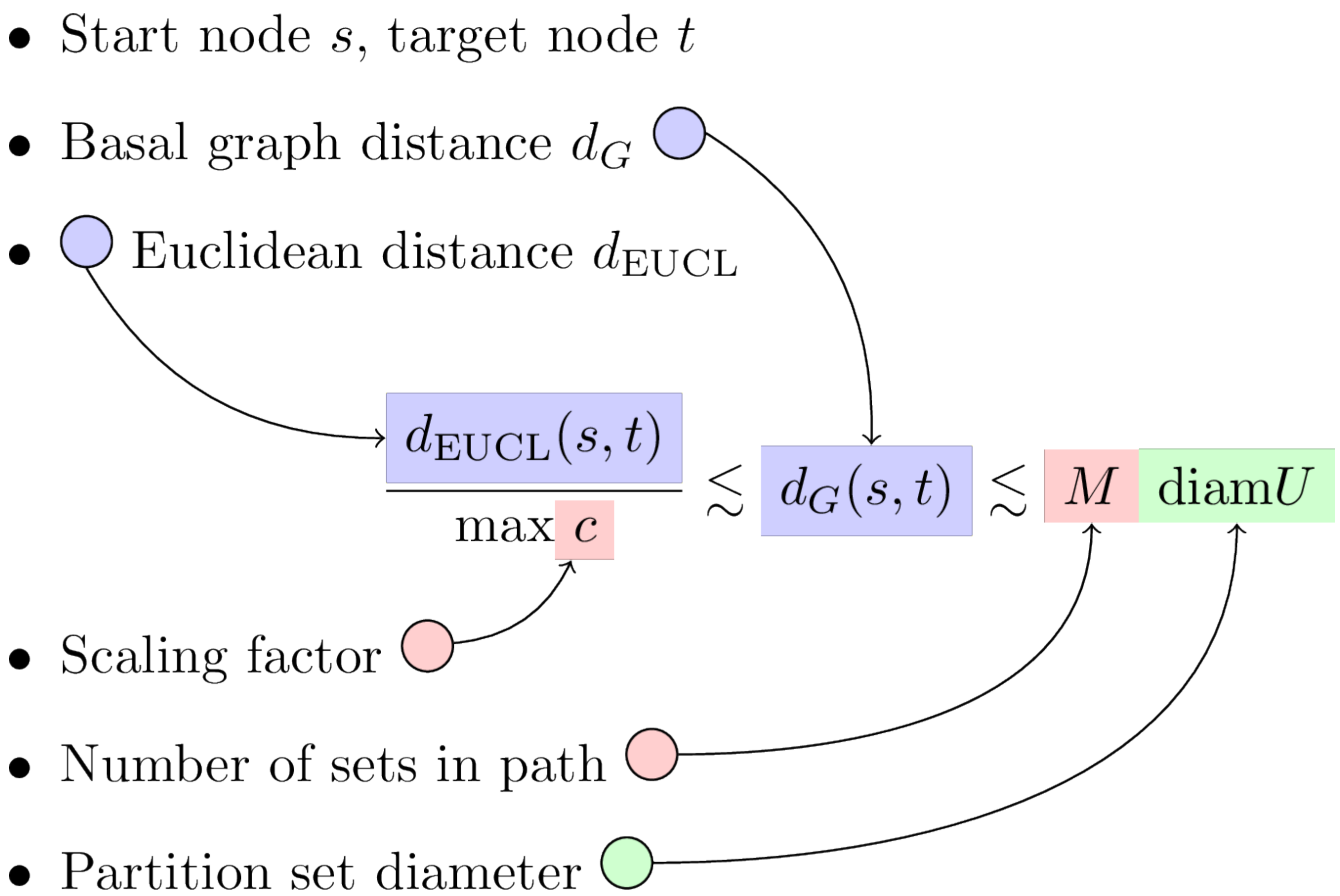
# Accuracy revisited

What is accuracy?

1. Length of hiearchical solution ≈ length of basal graph solution
2. Length of basal graph solution ≈ length of real (actual) optimum path

Accuracy is an issue in all graph-based approaches!

- Discrete model implies finite set of positions, movements
- Hierarchical approach allows arbitrarily fine basal mesh
- Requires careful selection of levels



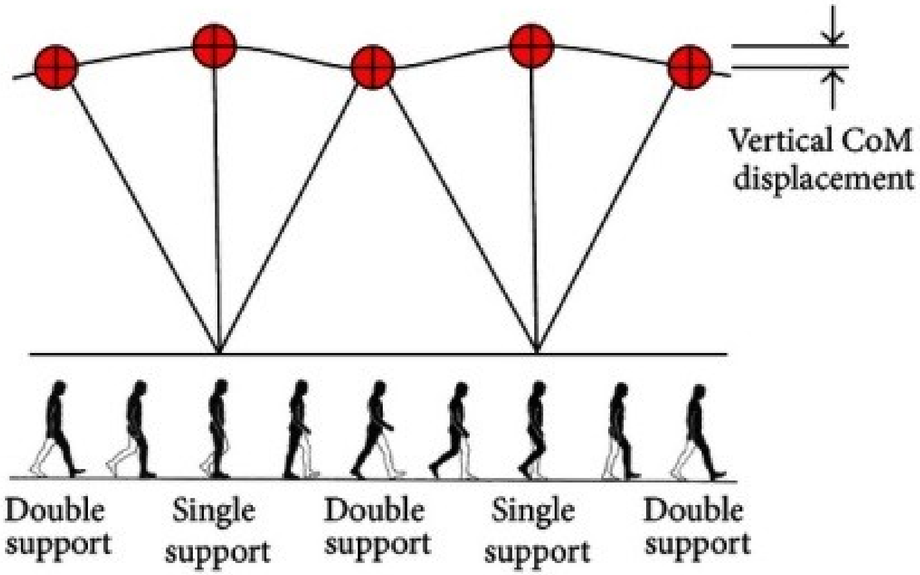
- Start node $s$, target node $t$
- Basal graph distance $d_G$ ●
- ● Euclidean distance $d_{\text{EUCL}}$

$$\frac{d_{\text{EUCL}}(s,t)}{\max c} \lesssim d_G(s,t) \lesssim M \operatorname{diam} U$$

- Scaling factor ●
- Number of sets in path ●
- Partition set diameter ●

# Walking patterns

- Uphill: extrinsic
- Downhill: concentric
- Level and moderate slopes: inverted pendulum



Lobet, Detrembleur & Hermans (2013)

# Energetics: Power balance

- Kinetic energy $E_{\mathrm{kin}}$
- Mass $m$
- Tangential speed $v$
- Inclination (slope) $\alpha$
- Locomotive power $P_{\mathrm{loc}}$ ●

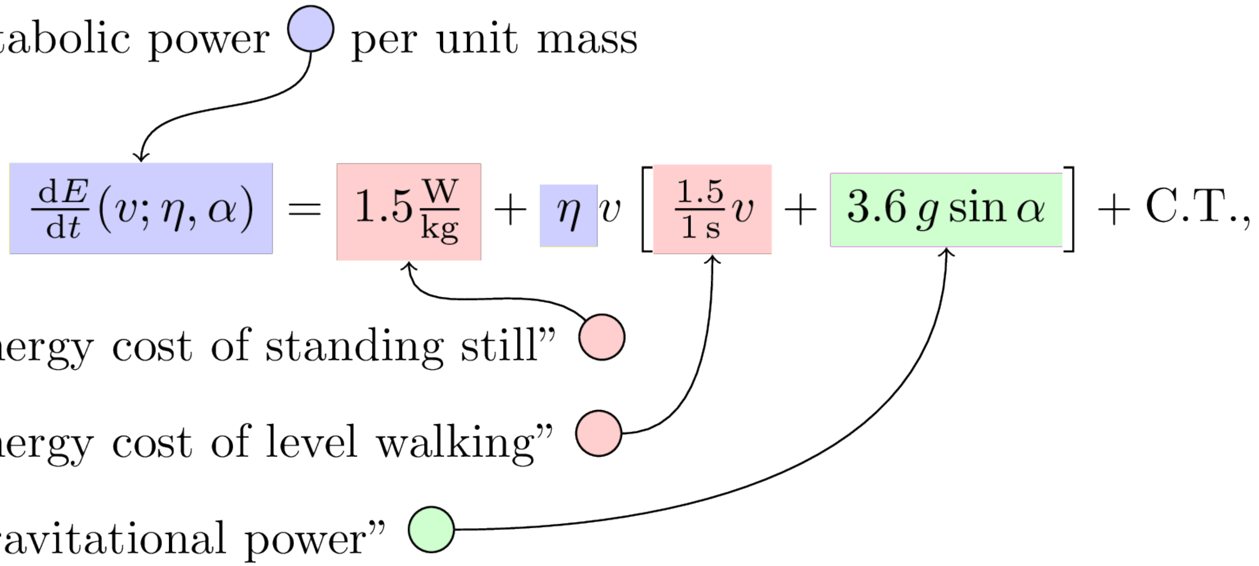
- Friction coefficient $\mu$
- Normal force $F_N$
- Gravitational $g = 9.81\ m/s^2$
- Cross sectional area $A$
- Air density $\rho$

$$\frac{\mathrm{d}E_{\mathrm{kin}}}{\mathrm{d}t} = mv\frac{\mathrm{d}v}{\mathrm{d}t} = \boxed{P_{\mathrm{loc}}} - \boxed{\mu F_N v} - \boxed{mgv\sin\alpha} - \overbrace{\boxed{\tfrac{1}{2}\rho C_d A v^3}}^{\approx 0}$$

- Friction power ●
- Gravitational power ●
- Air resistance (here: negligible) ●

# Metabolic power and Pandolf equation (rewritten)

- Metabolic energy per unit mass $E$
- Terrain coefficient $\eta$
- Correction term (C.T.) for downhill slopes (Santee et al., 2003)
- Metabolic power per unit mass

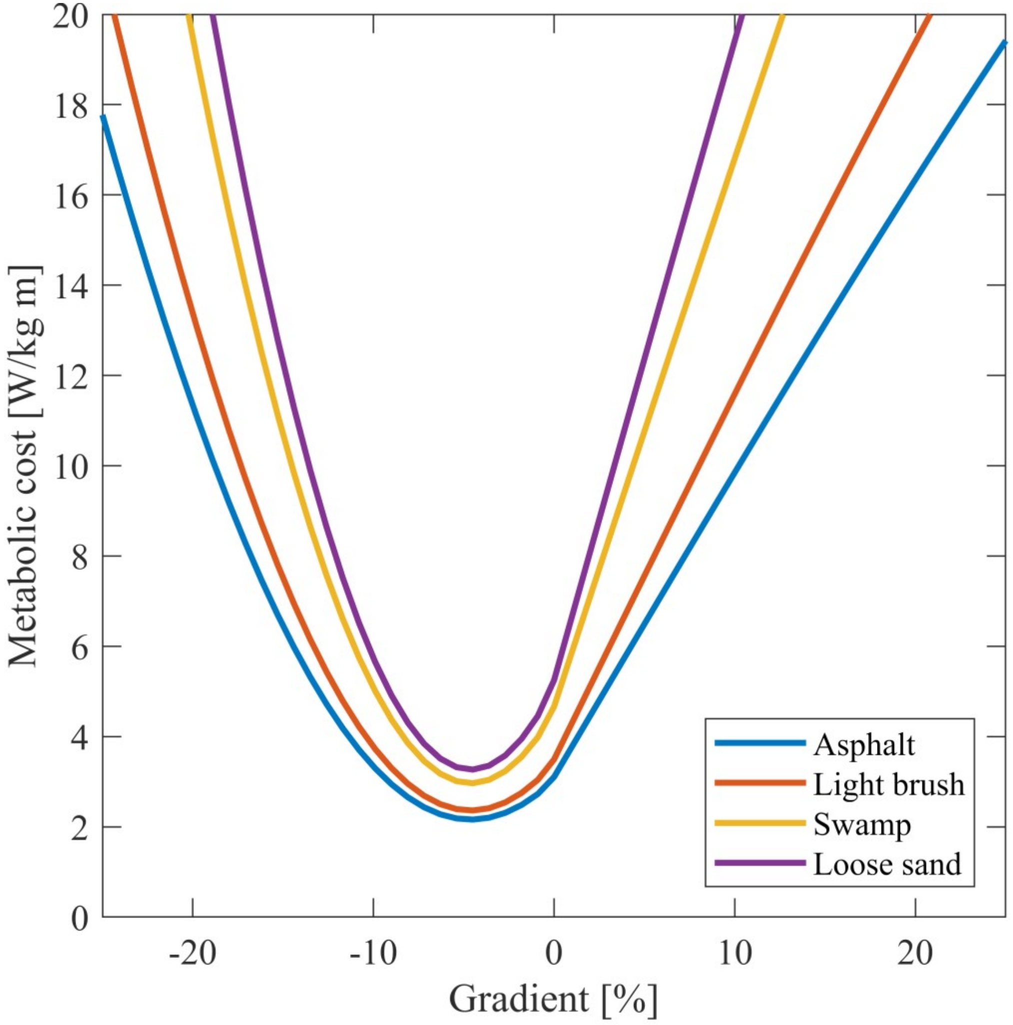


$$\frac{\mathrm{d}E}{\mathrm{d}t}(v;\eta,\alpha) = 1.5\,\frac{\mathrm{W}}{\mathrm{kg}} + \eta\, v \left[ \frac{1.5}{1\,\mathrm{s}} v + 3.6\, g \sin\alpha \right] + \text{C.T.},$$

- “Energy cost of standing still”
- “Energy cost of level walking”
- “Gravitational power”
- Gross energy cost per unit distance $C_g = \left[\frac{\mathrm{d}E}{\mathrm{d}t}\right]/v$.

# Weight function (example)

- Metabolic energy per unit mass and distance
- Any path is traversible in both directions
- Connected components are strongly connected
- Symmetric with respect to ground condition
- Assymmetric with respect to slope



# Data

Open data:

- LiDAR: Creative Commons 4.0 (No), Open Government (UK), etc.
- OpenStreetMap (ODbL)
- ElVeg (roads, free data)

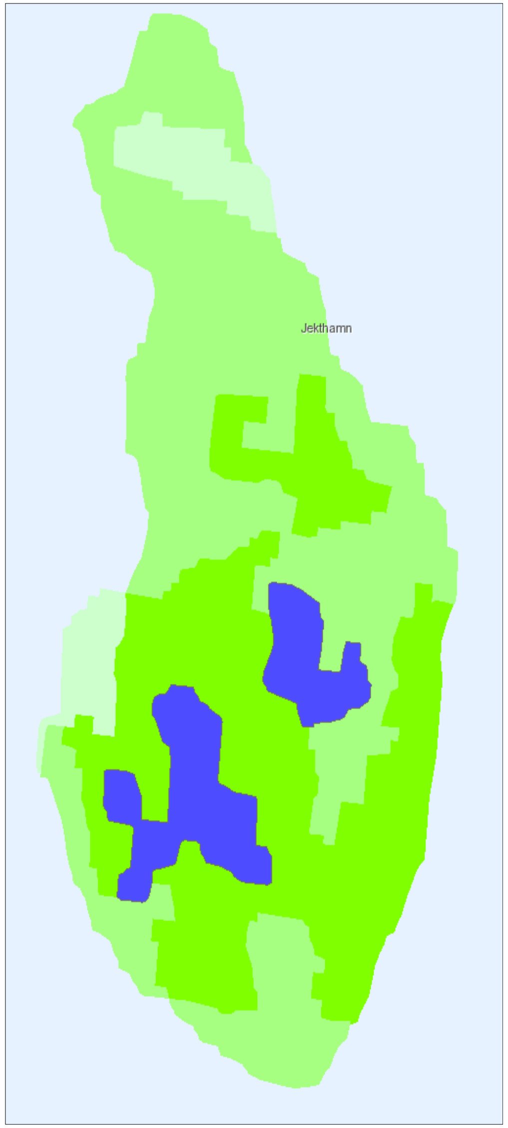
Restricted data:

- Land cover/land use from national competent authorities (attribution)
- 1:1000 – 1:5000 vector data (detailed infrastructure)

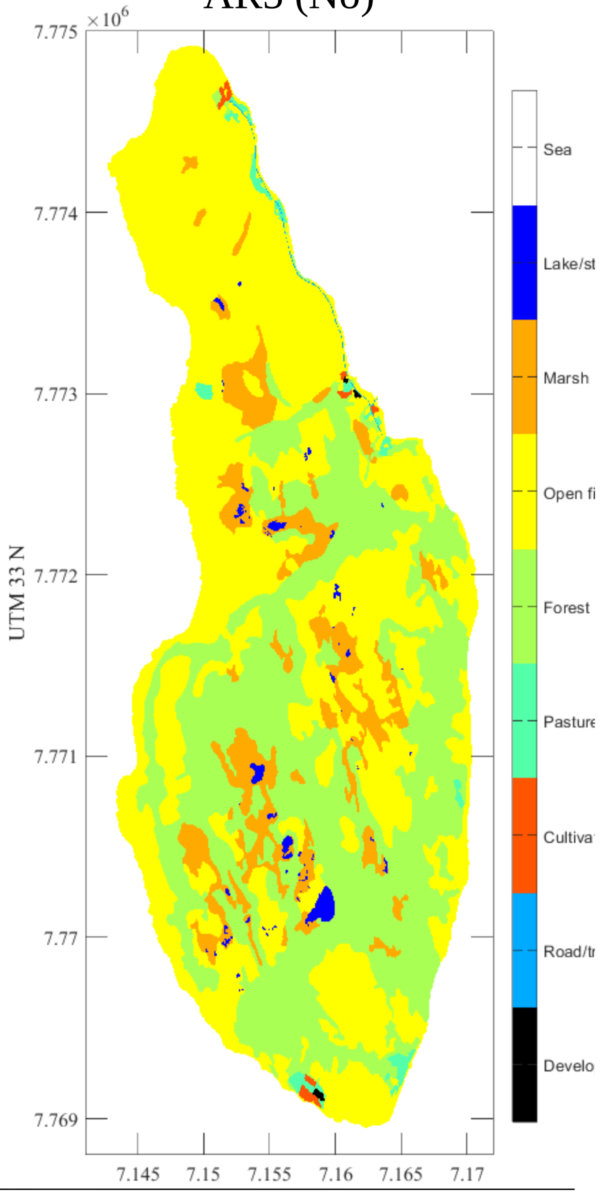
# Open vs restricted data

- OSM + open remote sensing data yield detailed land cover in some areas (Schultz et al., 2017).
- Dedicated public land cover datasets (e.g. CORINE) generally good but not sufficiently detailed.
- Currently detailed land cover from national competent authorities required.
- However, information from (open) LiDAR data may reduce need for hi-res vector data.
- OSM infrastructure sufficient in rural areas

Copernicus (EU)



AR5 (No)



# Going the right way:

- **LiDAR processing**

pdal

- **Raster/Vector pre-processing**

GDAL

PROJ.4

GEOS
Geometry
Engine
Open
Source

- **Data inspection**

QGIS

PostGIS

- **Graph representation**

# Going the right way:

- **Optimum paths** pgRouting
- **Processing service (WPS)** ZOO-Project
- **Web application** OpenLayers MapServer open source web mapping
- **Graph construction, partitioning**

# Conclusion

- Open LiDAR data enable realistic cross-country path planning.
- LiDAR reduces the need for detailed restricted data.
- The hierarchical method can eliminate position uncertainty at low computational cost.
- Requires careful selection of search levels.
- Efficient preprocessing is accomplished with $k$-center algorithm and heuristic Dijkstra.
- Current work to find cost function constrained by biomechanical principles, tuned to empirical data on walking.

# References

- Bast, Funke, Sanders and Schultes (2007). Fast routing in road networks with transit nodes, Science 316, pp. 566.
- Delling, Sanders, Schultes and Wagner (2009). Engineering route planning algorithms, in Lerner, Wagner and Zweig (eds.), *Algorithmics of Large and Complex Networks*, pp. 117-139, Springer.
- Kepner and Gilbert (2011). Graph Algorithms in the Language of Linear Algebra.
- Pandolf, Givoni and Goldman (1977). Predicting energy expenditure with loads while standing or walking very slowly, Journ. Appl. Physiol. Respir. Environ. Exerc. Physiol. 43, pp. 577-581.
- Har-Peled (2008), Geometric approximation algorithms, University of Illinois.
- Schultz, Voss, Auer, Carter and Zipf (2017). Open land cover from OpenStreetMap and remote sensing, Int. Journ. Appl. Earth Observation and Geoinformation 63, pp. 206-213.