Strict Theta*

Shorter Motion Planning Using Taut Paths

Shunhao Oh
ohoh@u.nus.edu

Hon Wai Leong
leonghw@comp.nus.edu.sg

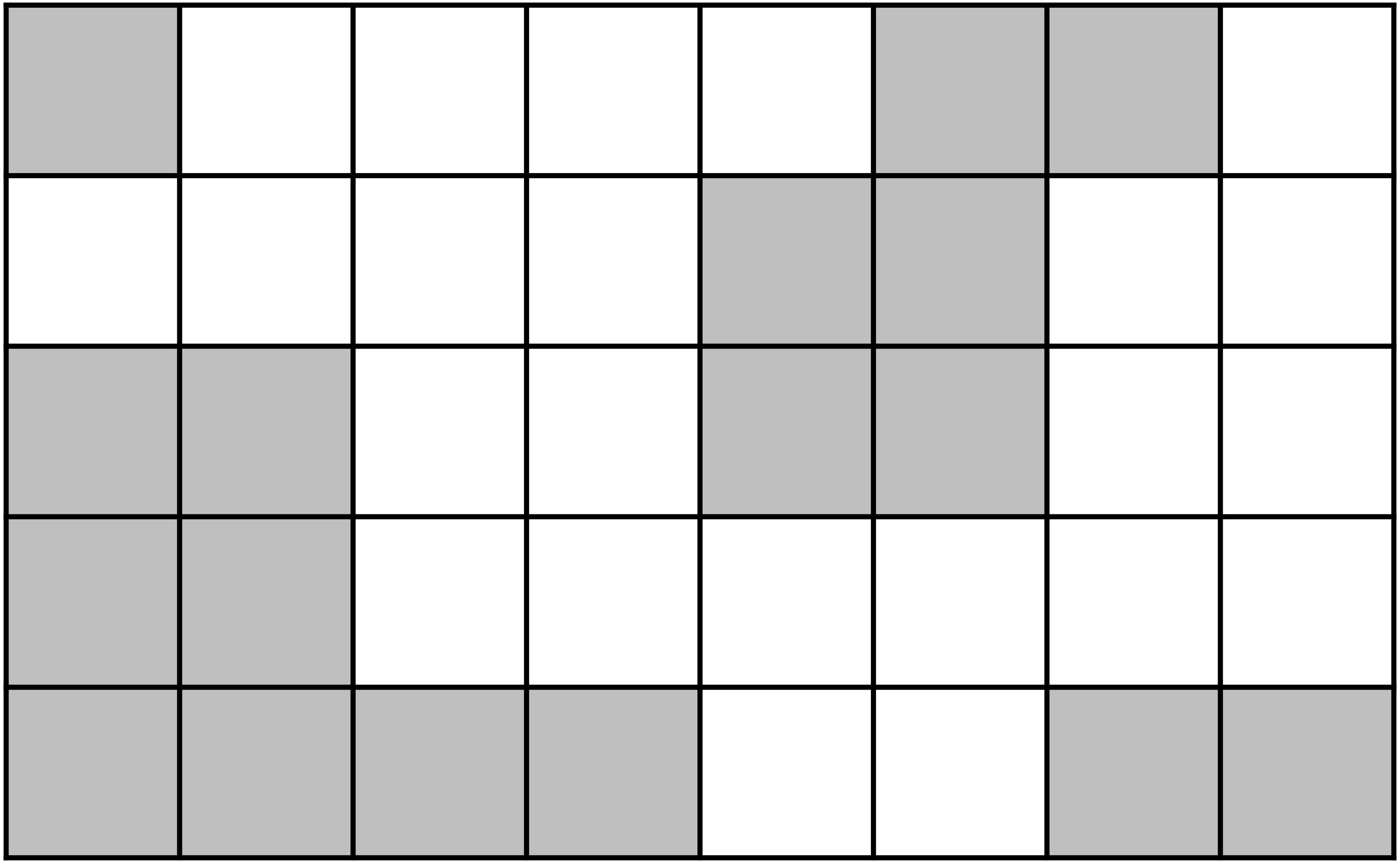
Department of Computer Science
National University of Singapore

Any-Angle Path Planning

Problem Definition

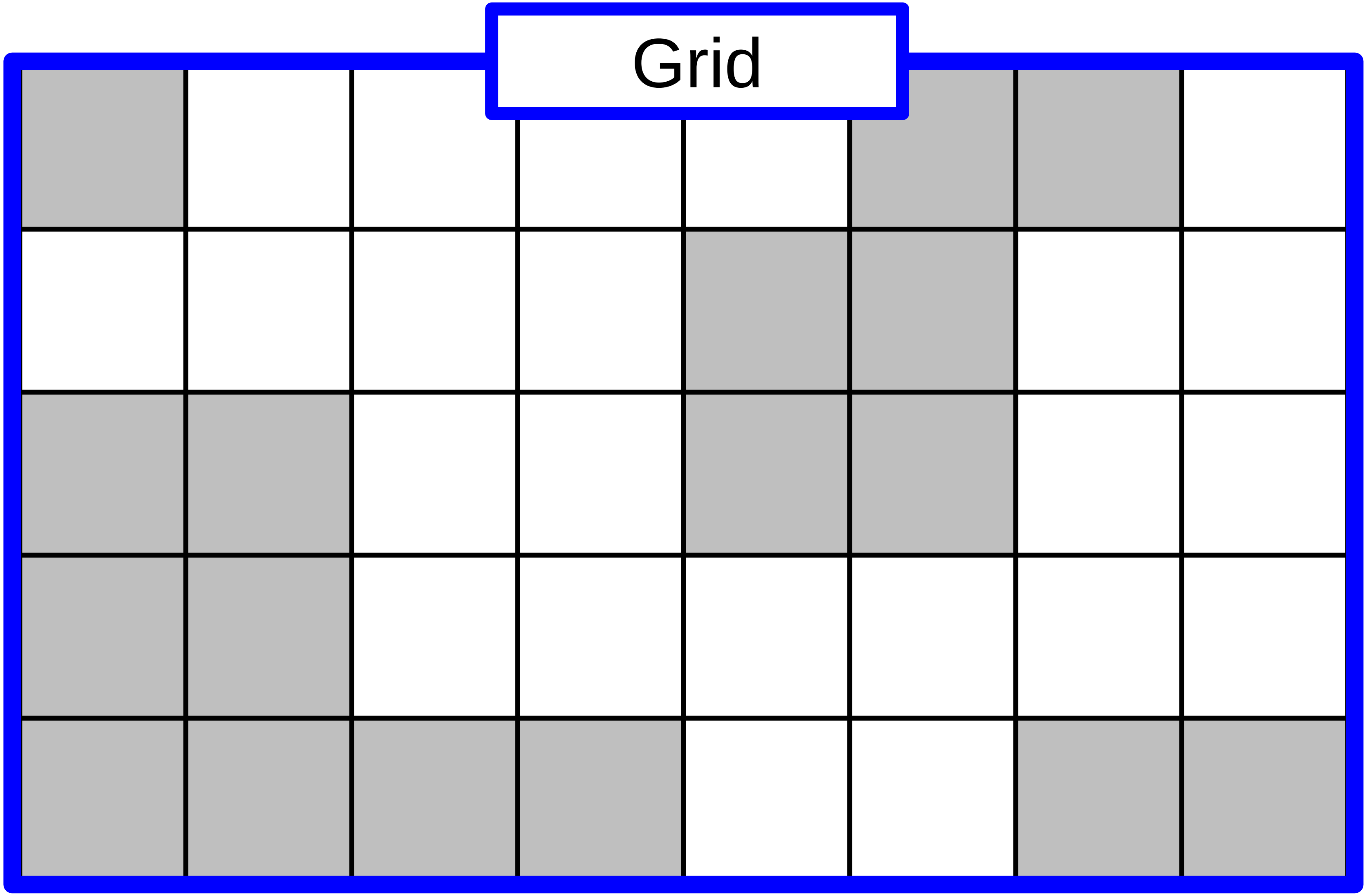
Any-Angle Path Planning

Problem Definition



Any-Angle Path Planning

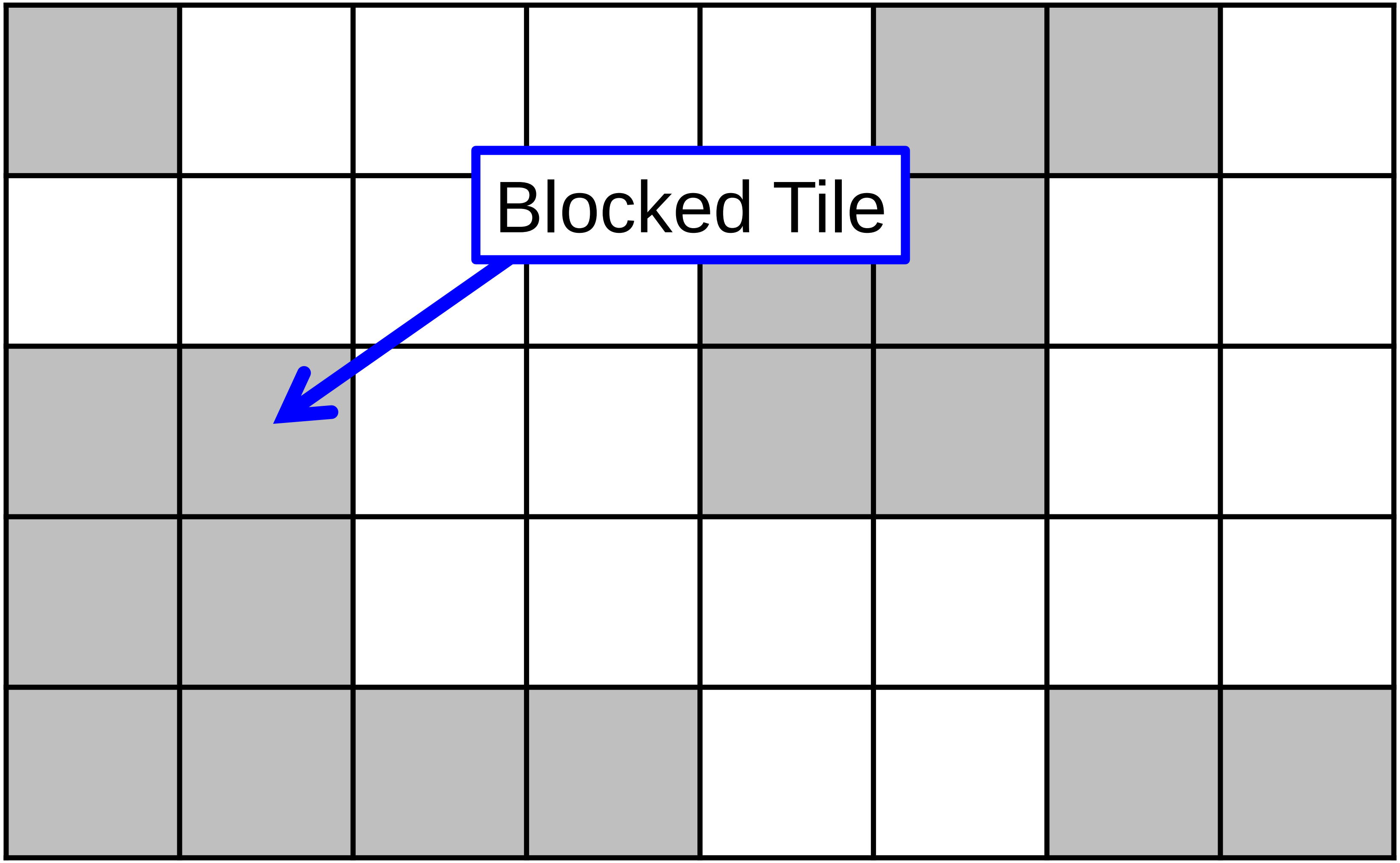
Problem Definition



Grid

Any-Angle Path Planning

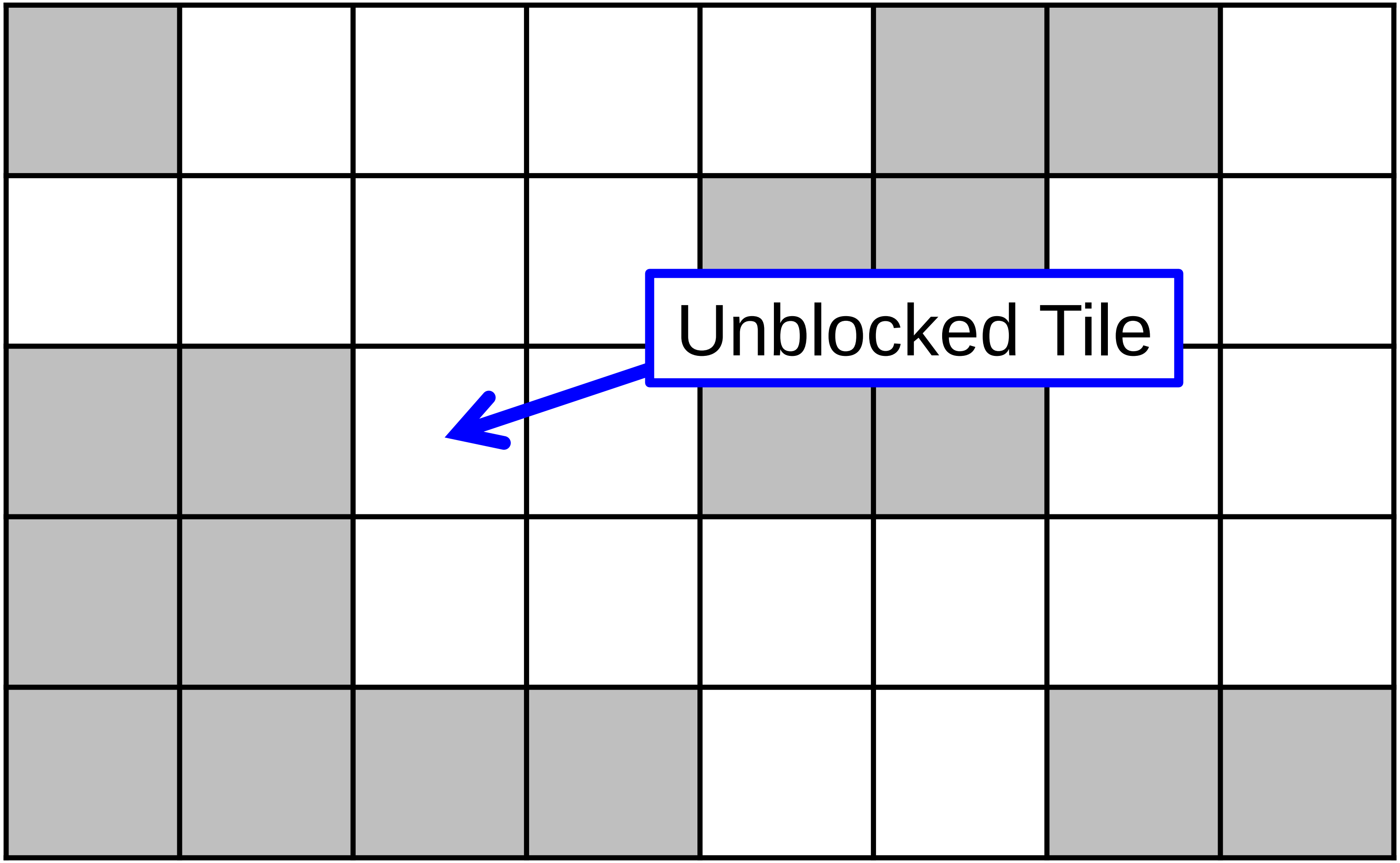
Problem Definition



Blocked Tile

Any-Angle Path Planning

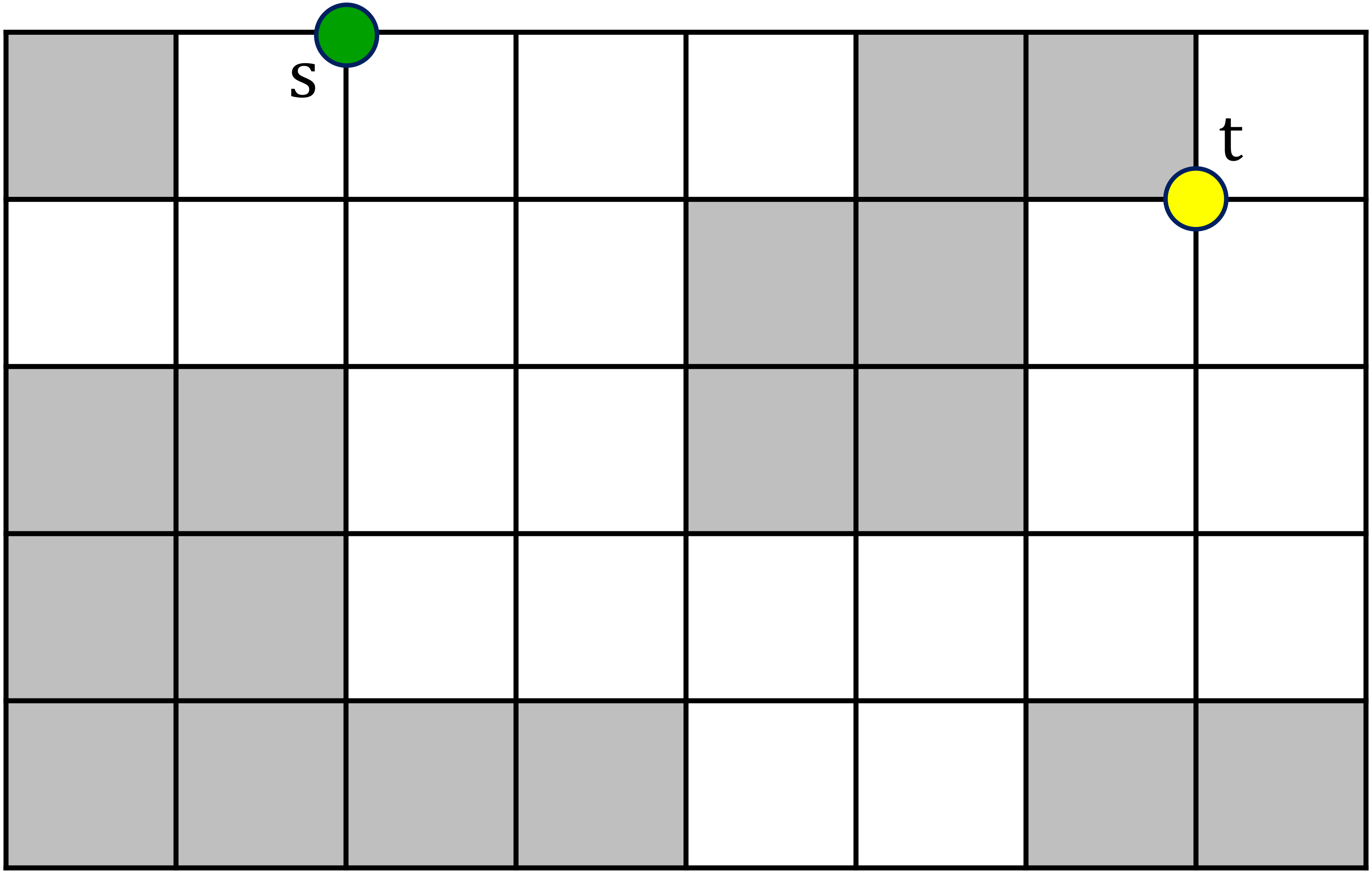
Problem Definition



Unblocked Tile

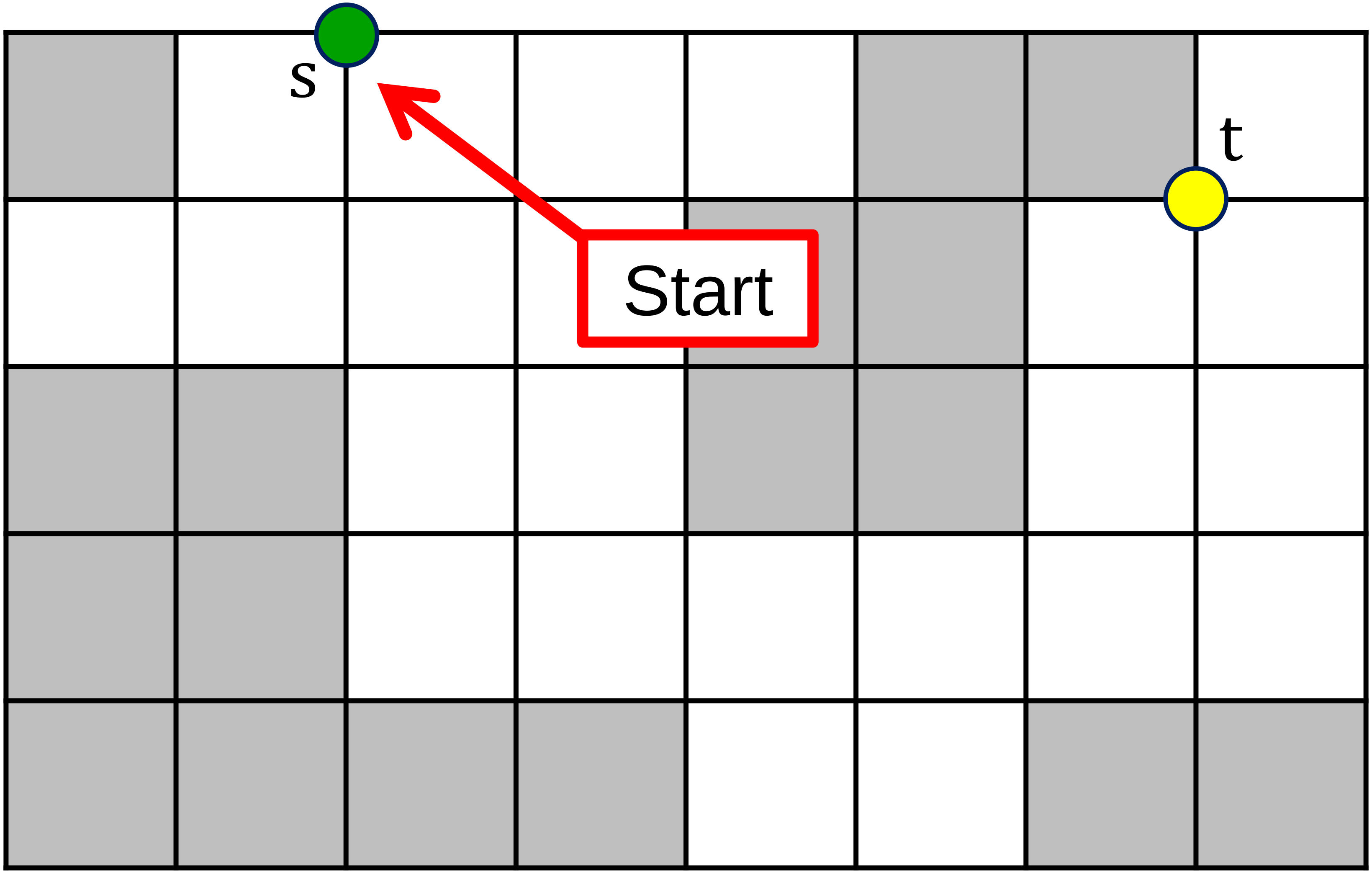
Any-Angle Path Planning

Problem Definition



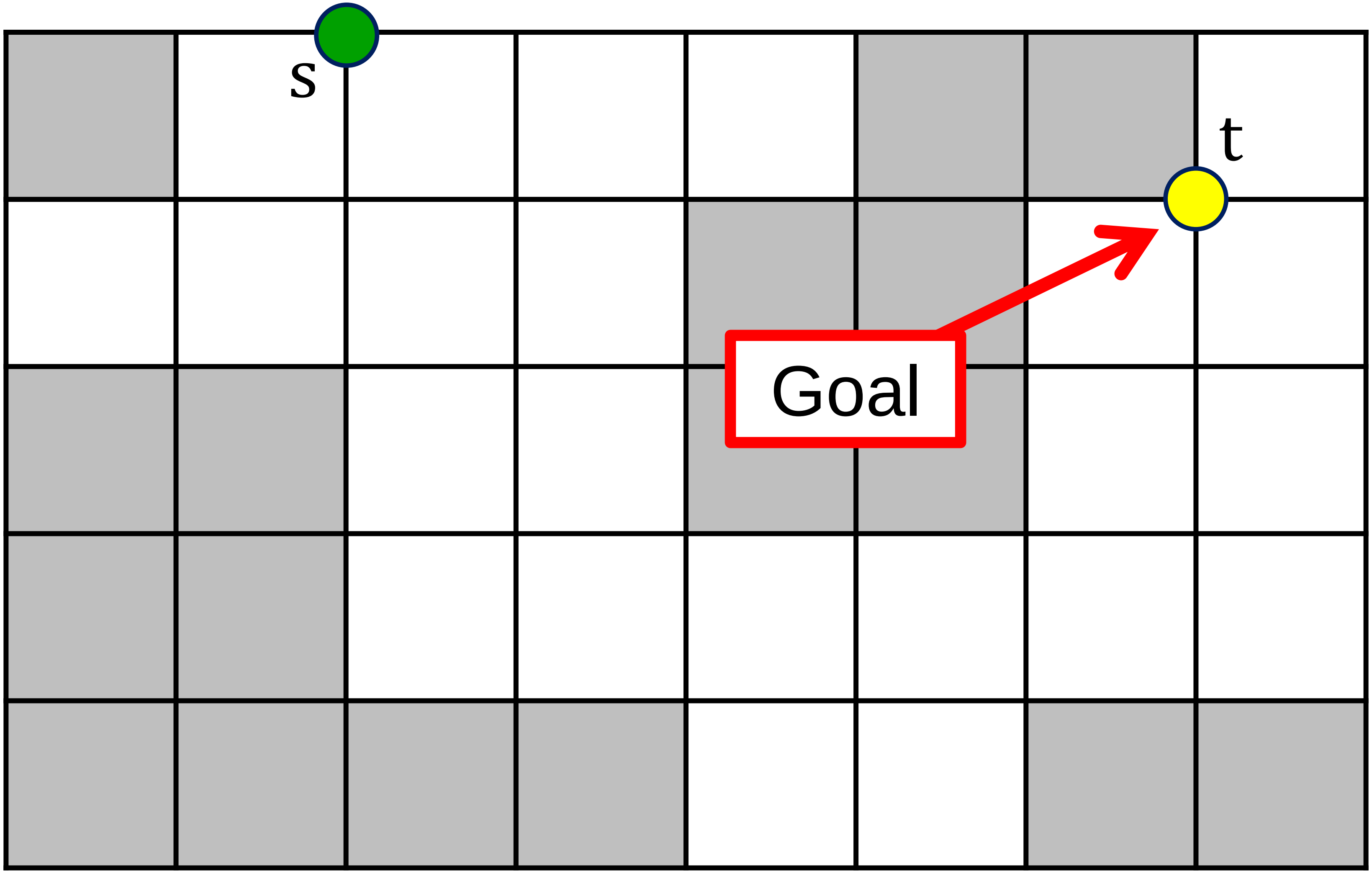
Any-Angle Path Planning

Problem Definition

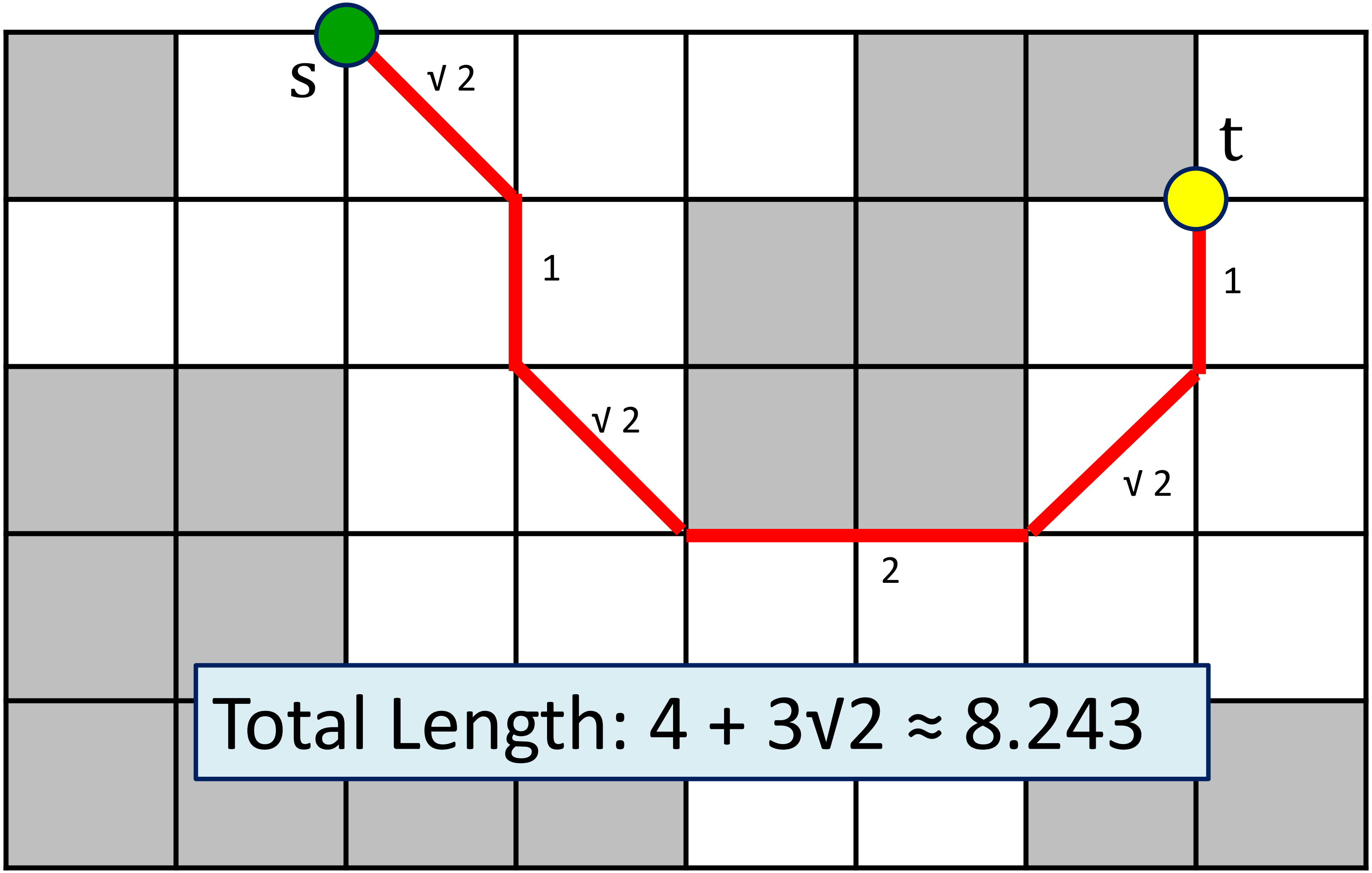


Any-Angle Path Planning

Problem Definition

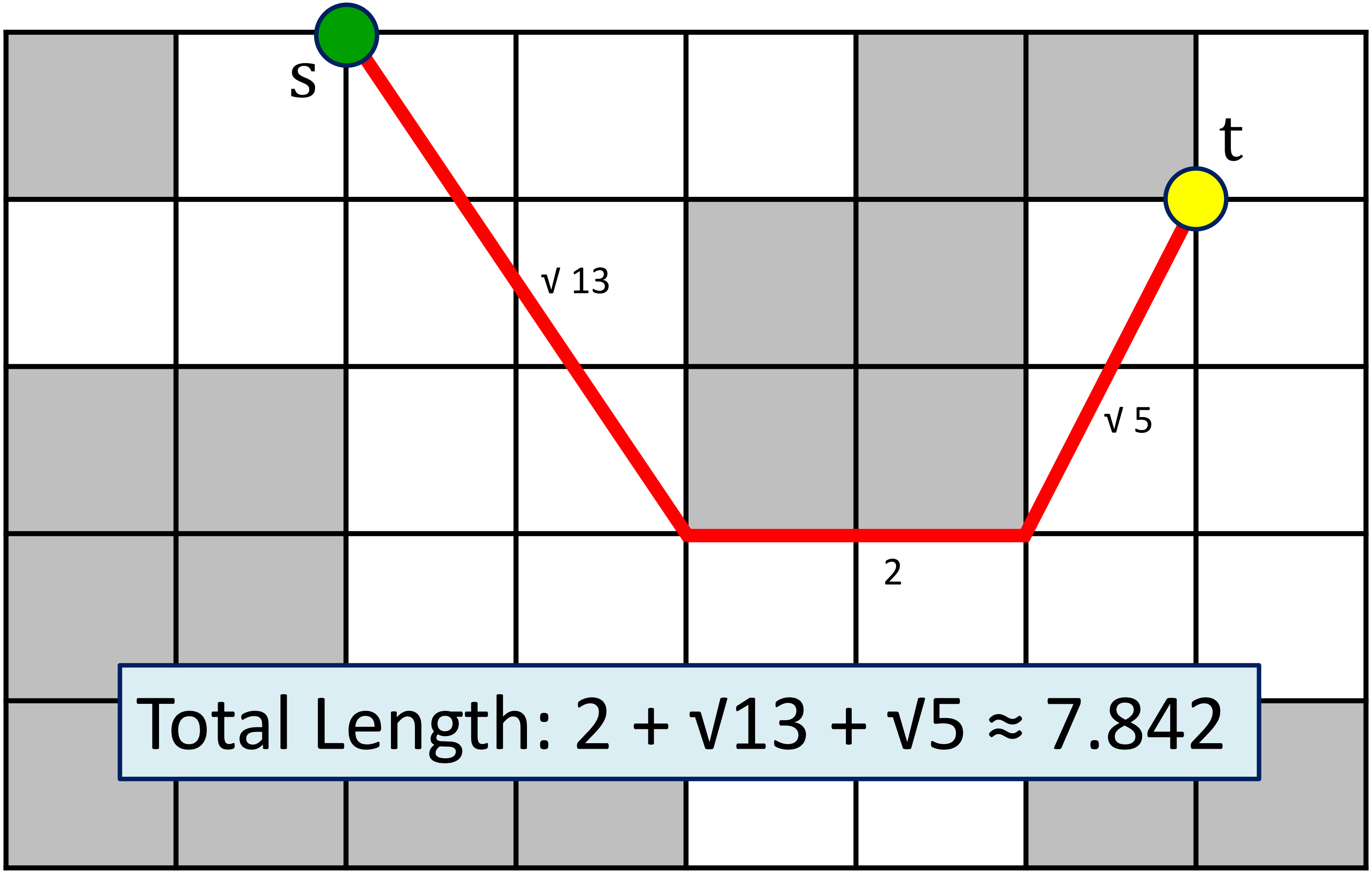


8-Directional Path

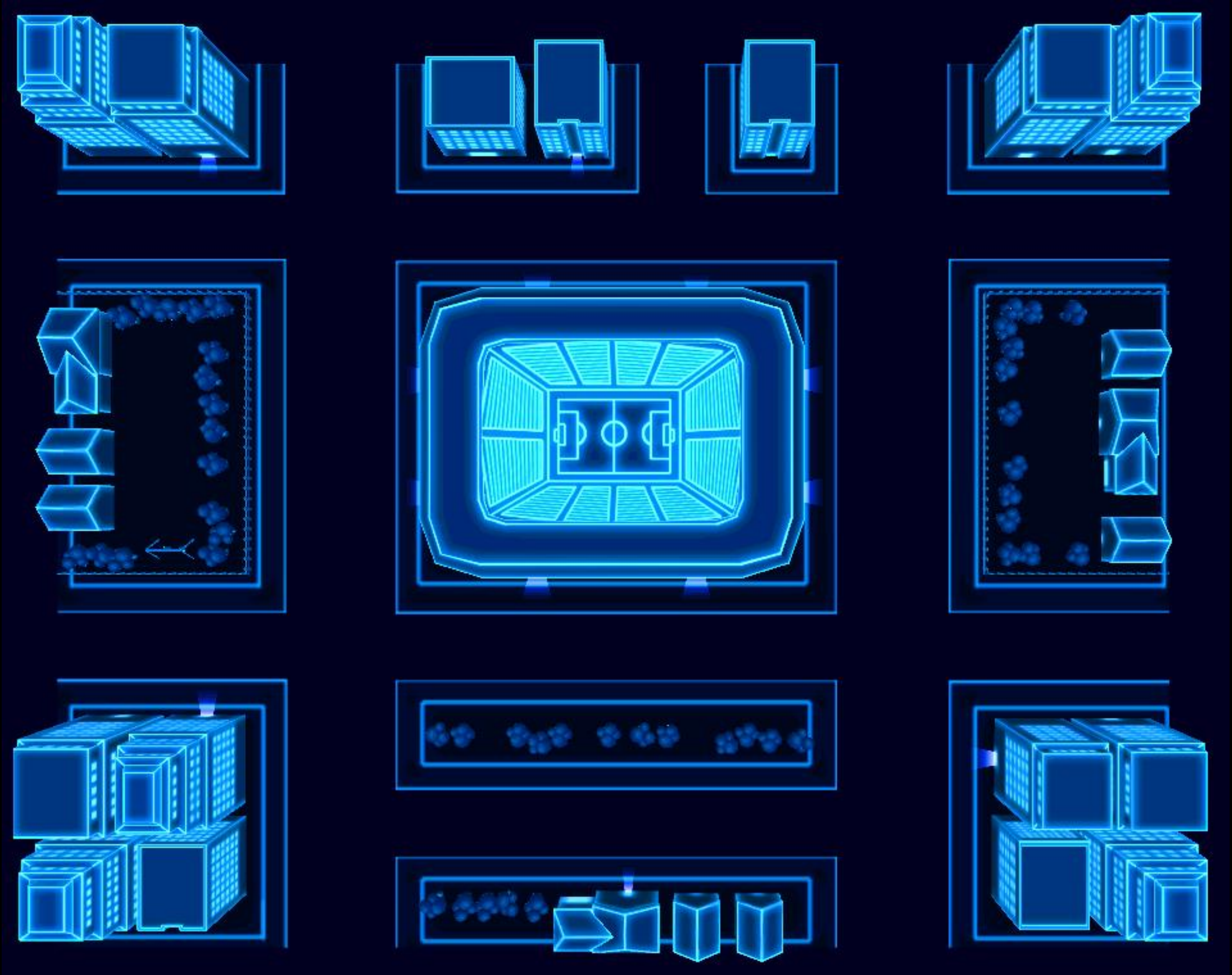


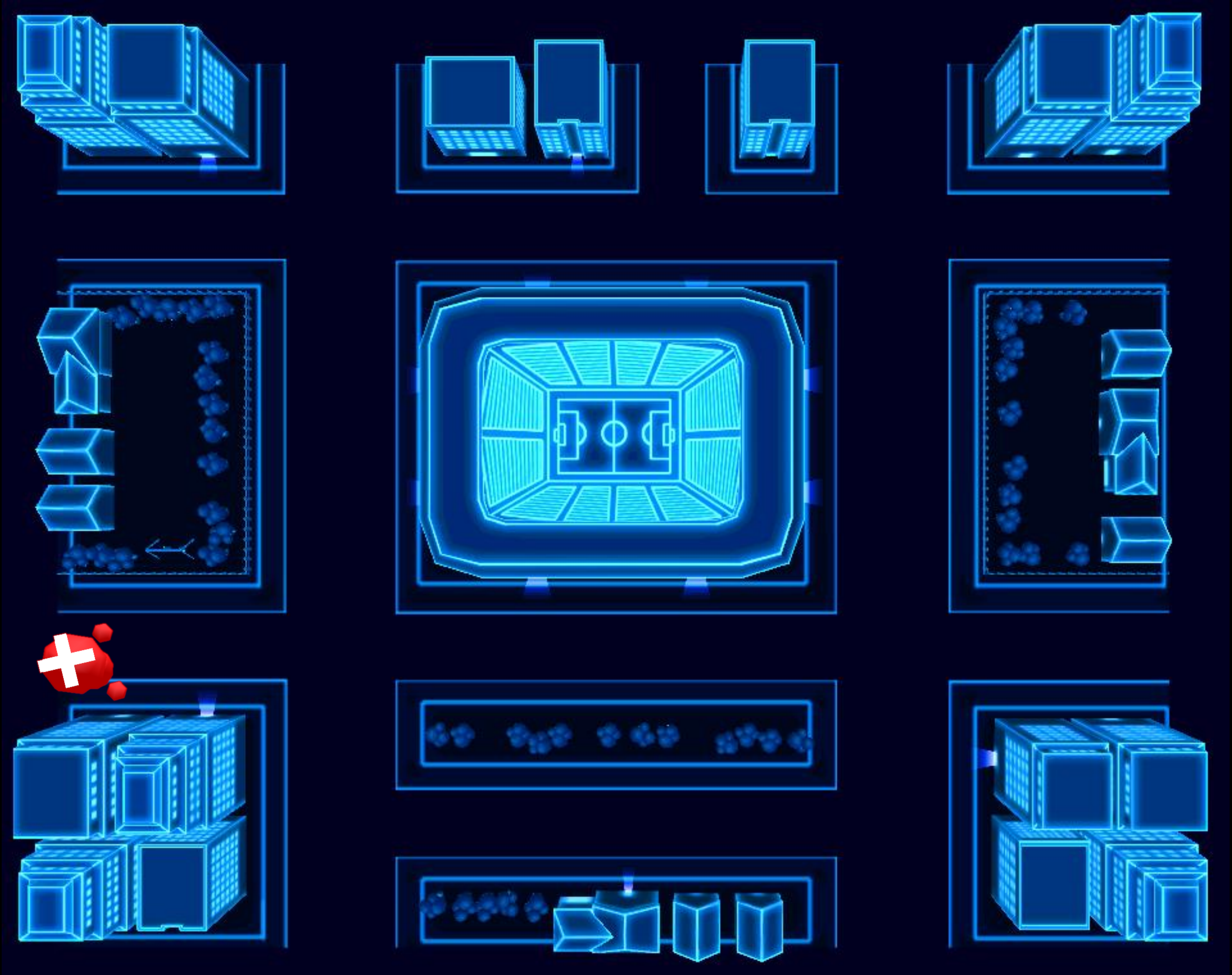
Total Length: 4 + 3√2 ≈ 8.243

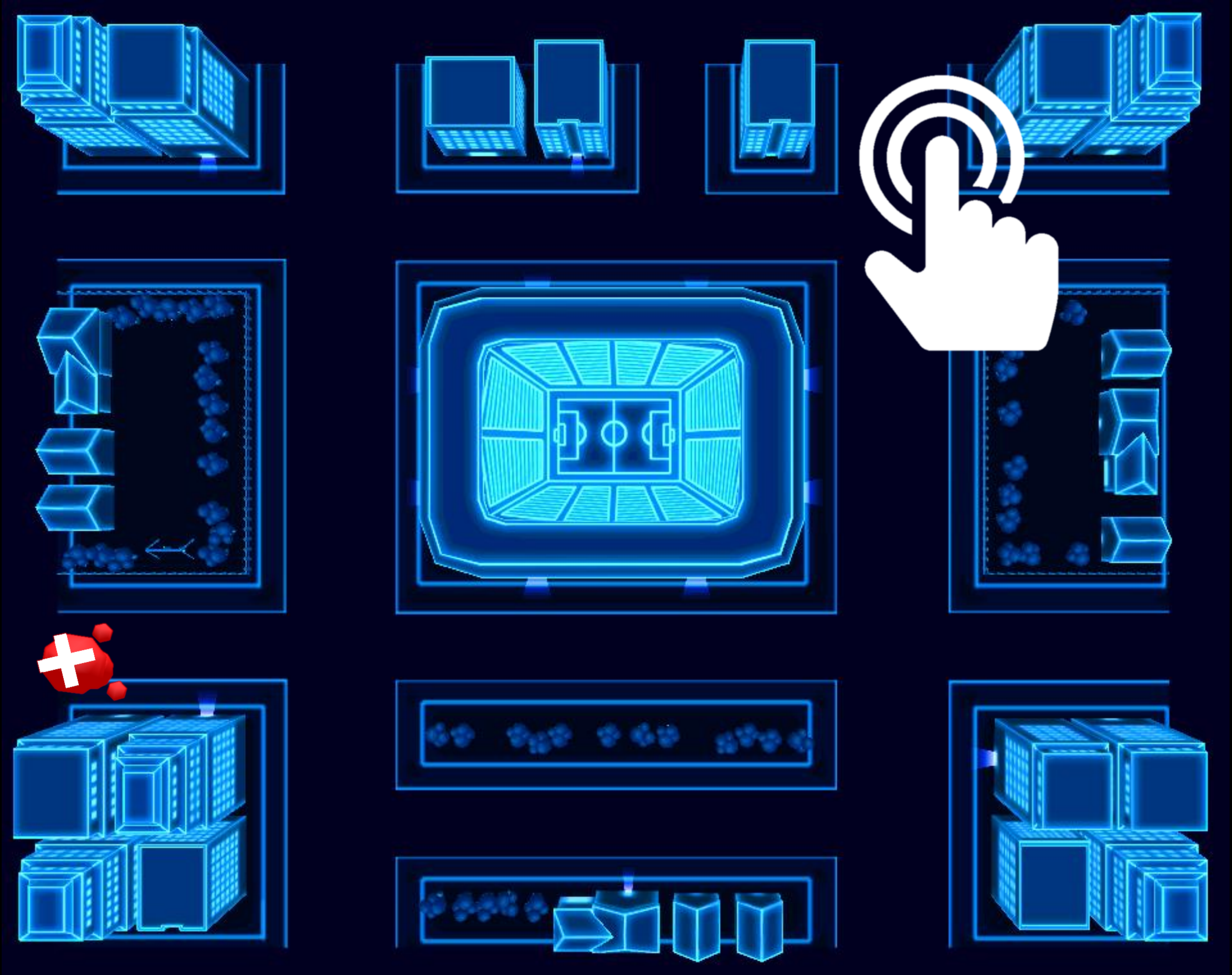
Any-Angle Path

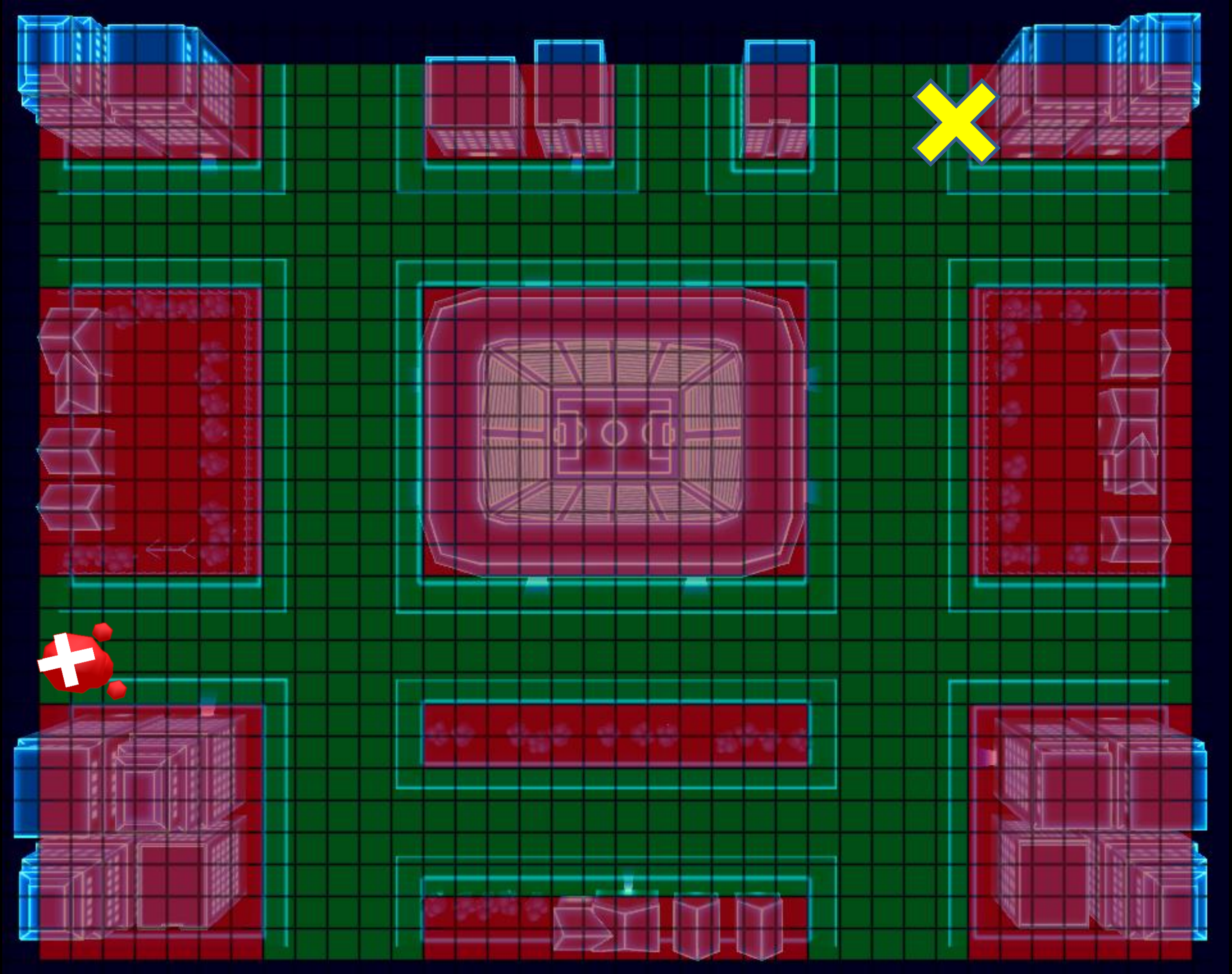


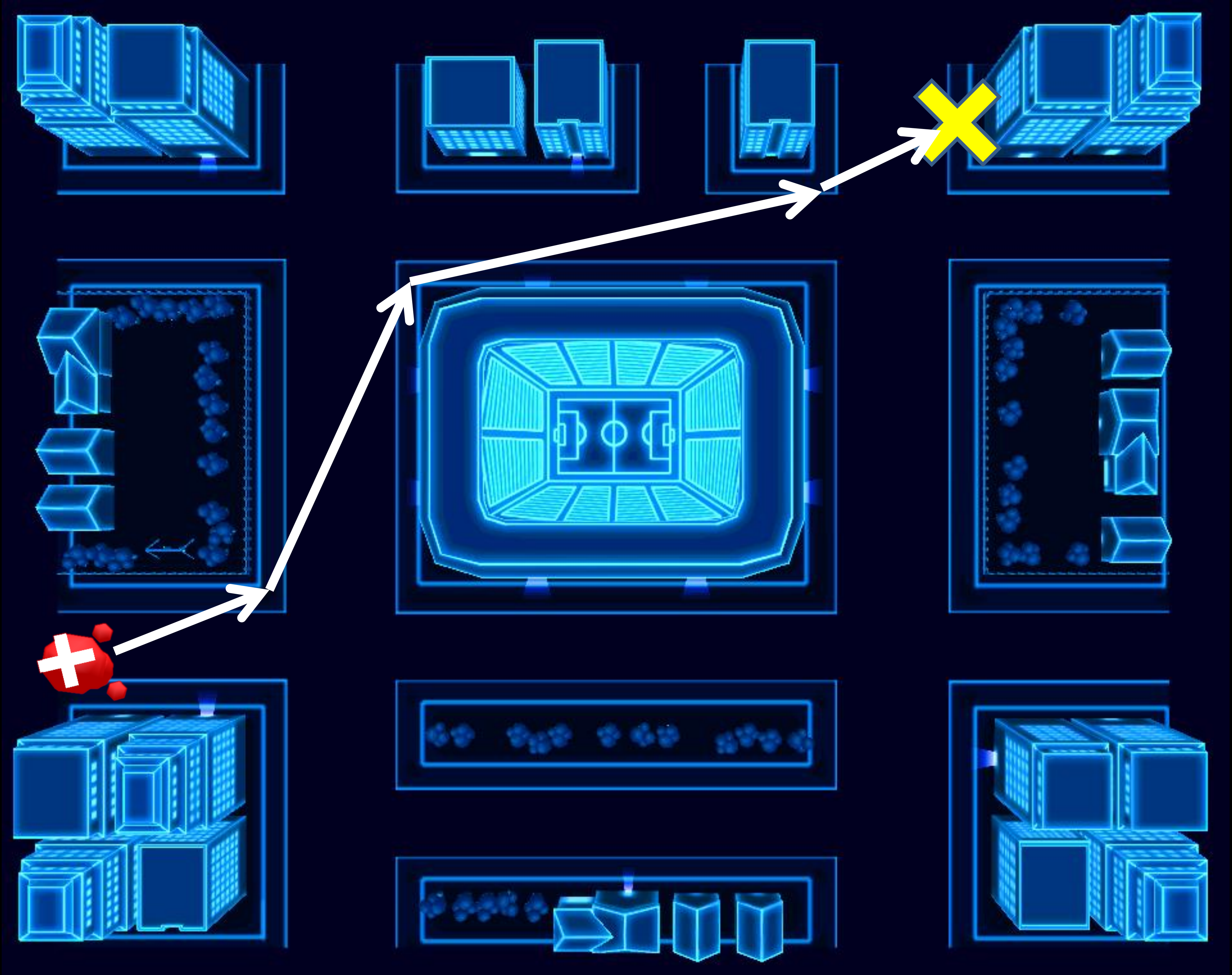
Total Length: 2 + √13 + √5 ≈ 7.842

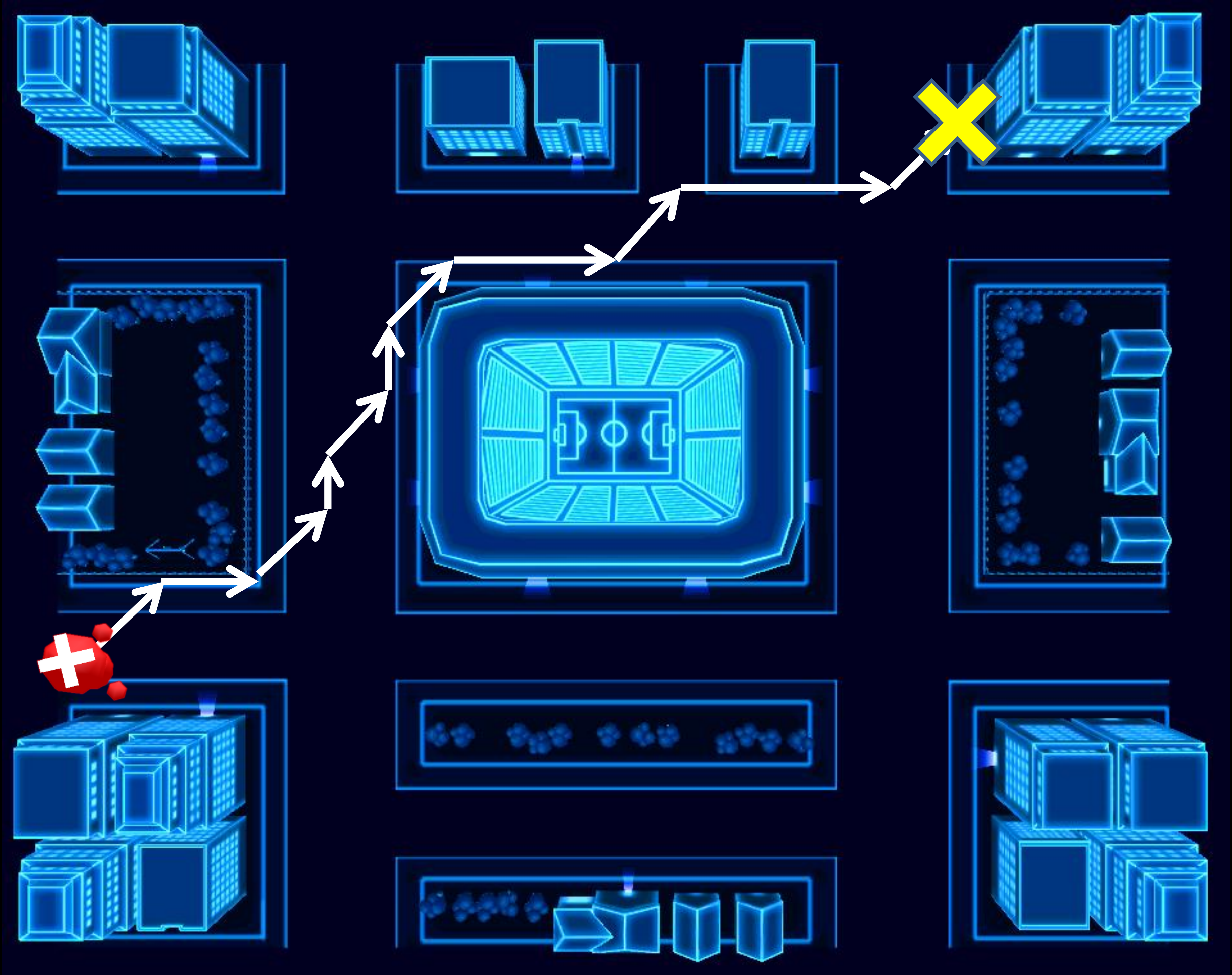




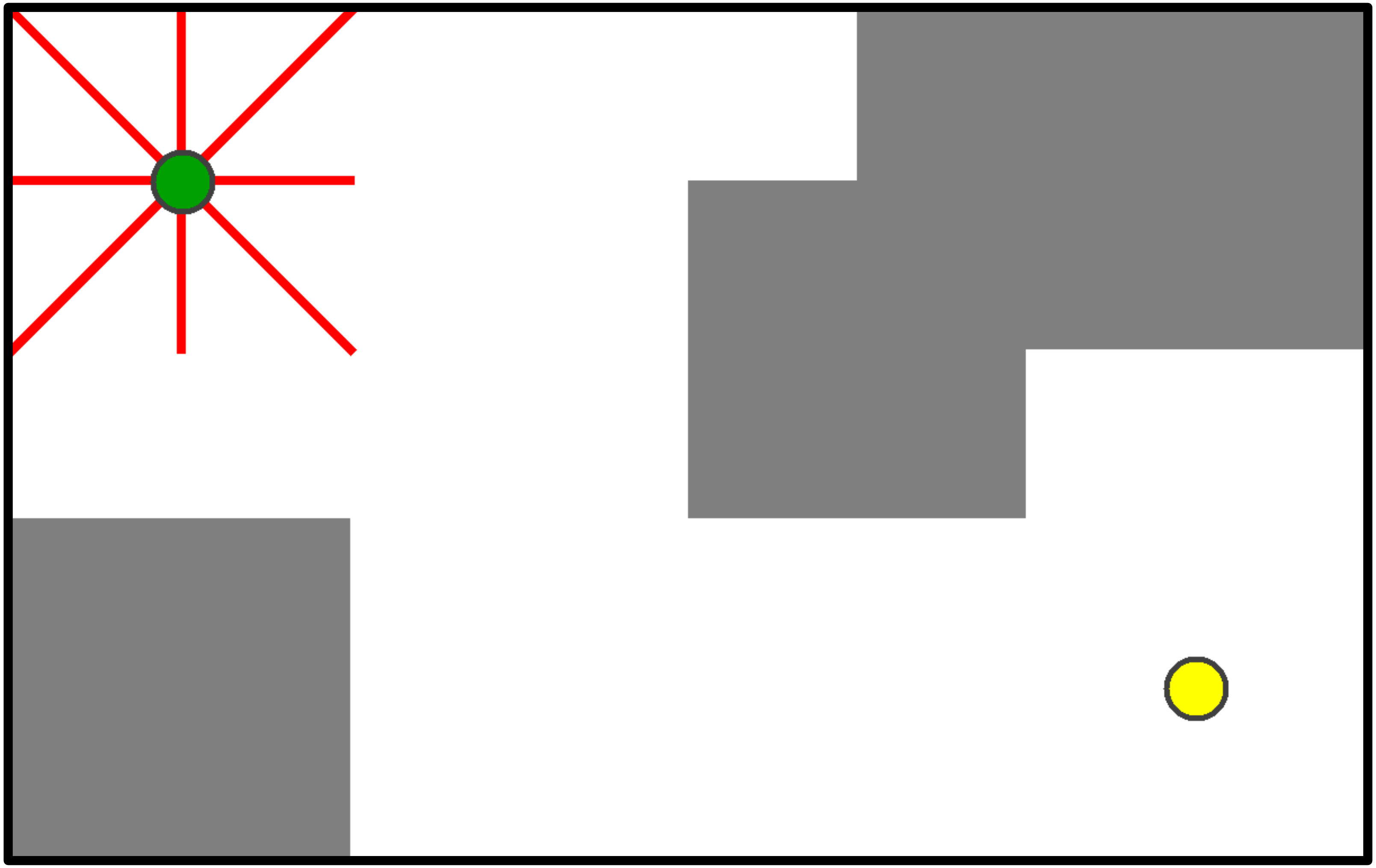




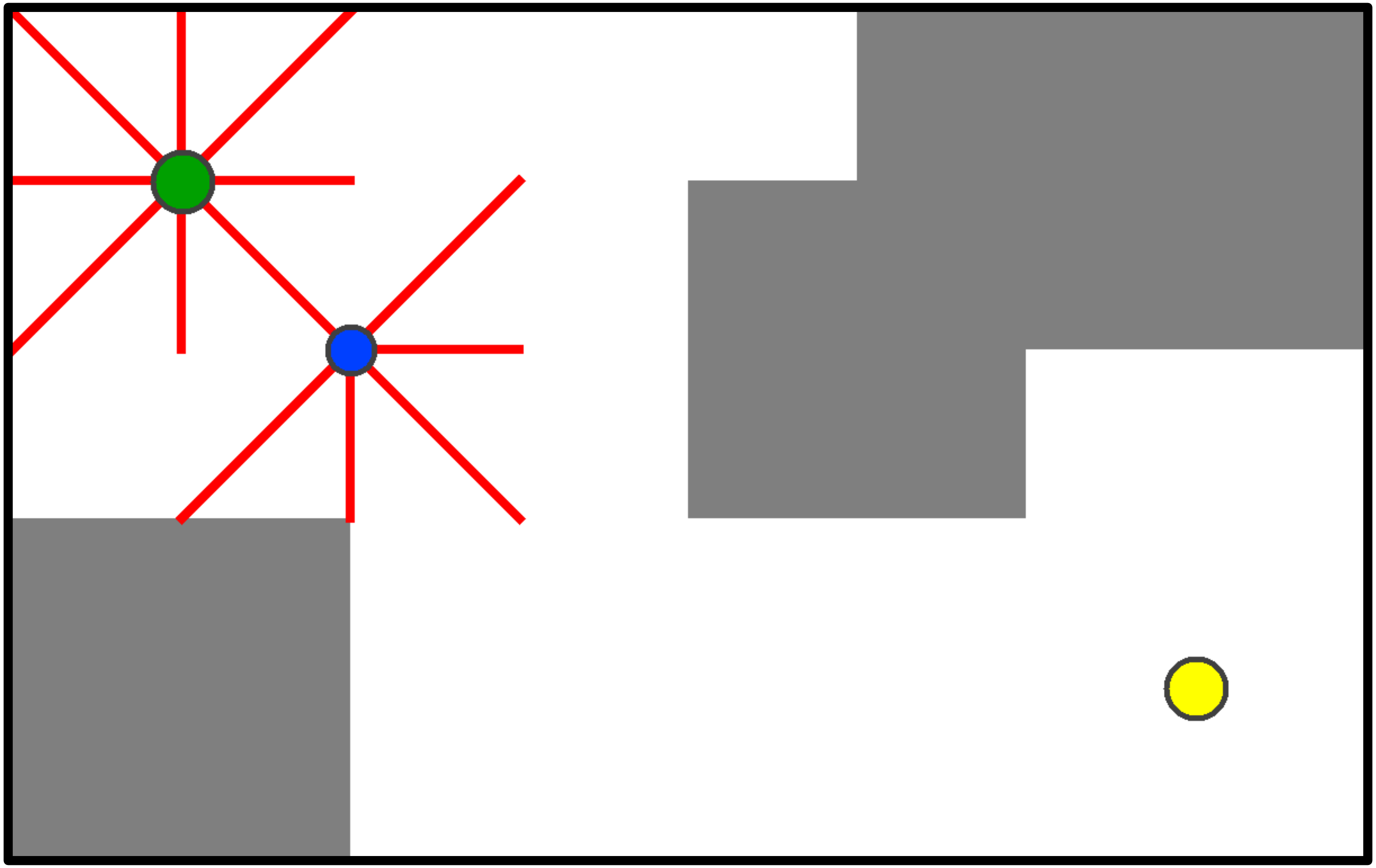




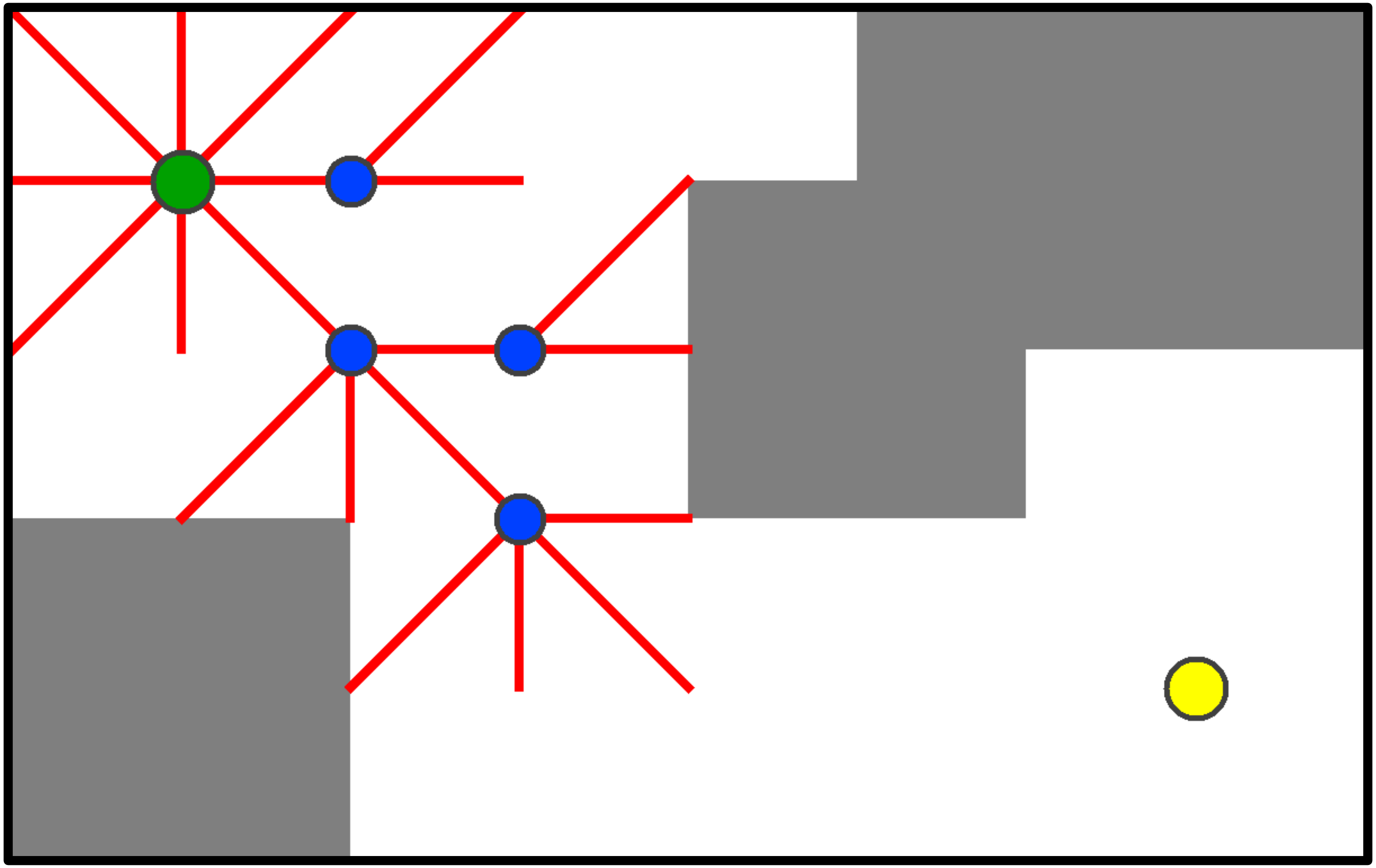
A* on 8-directional Grids



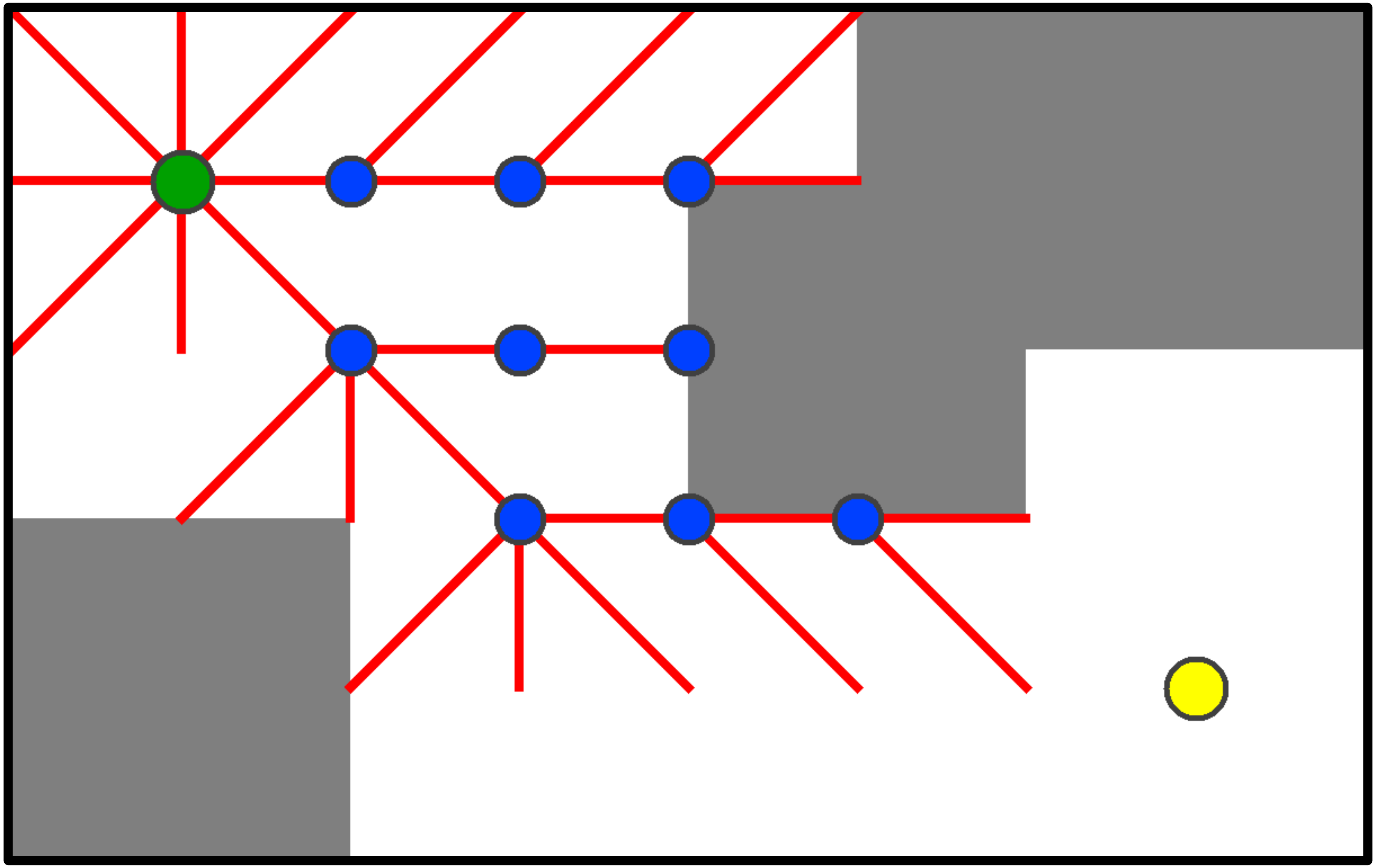
A* on 8-directional Grids



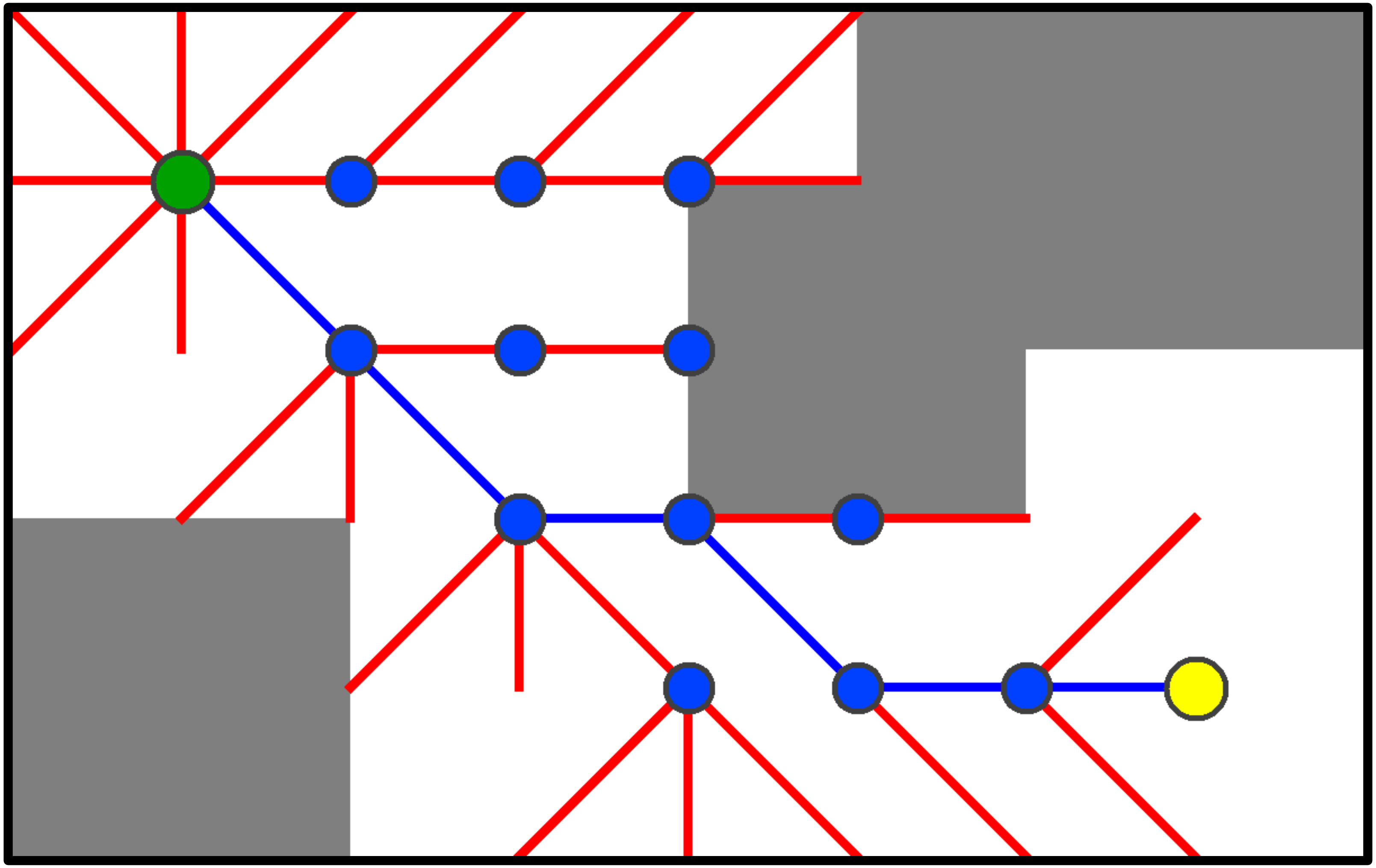
A* on 8-directional Grids



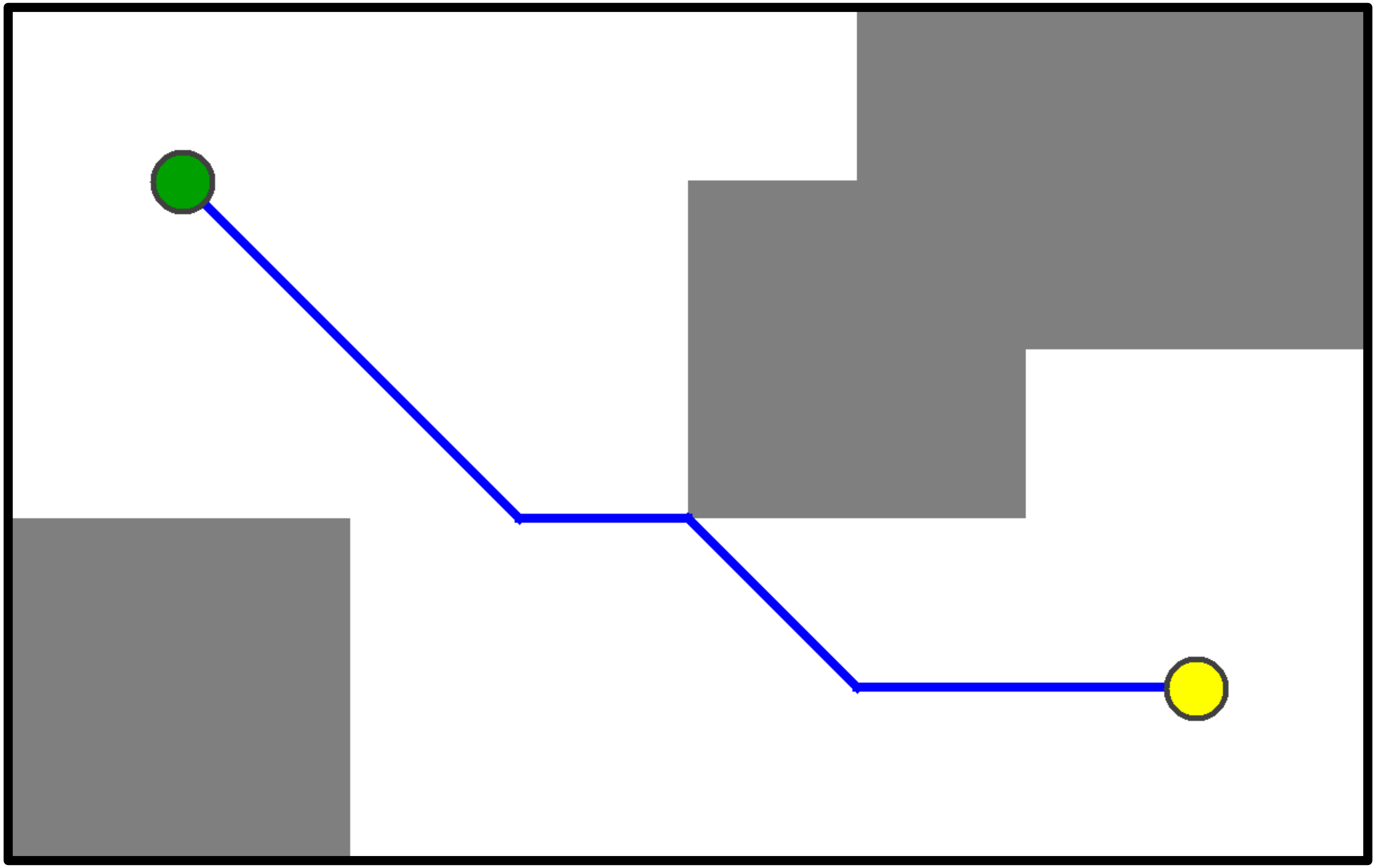
A* on 8-directional Grids



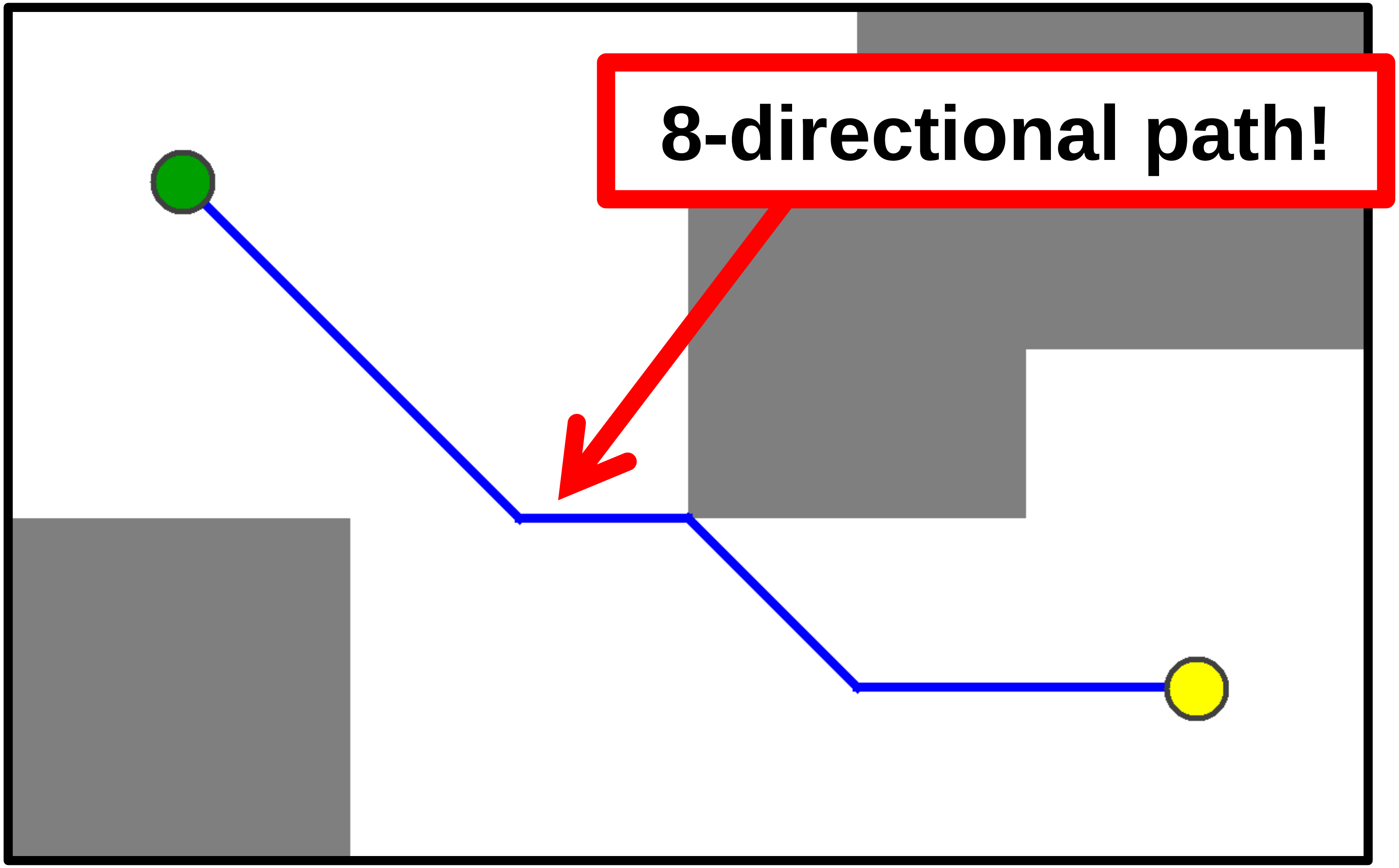
A* on 8-directional Grids



A* on 8-directional Grids



A* on 8-directional Grids

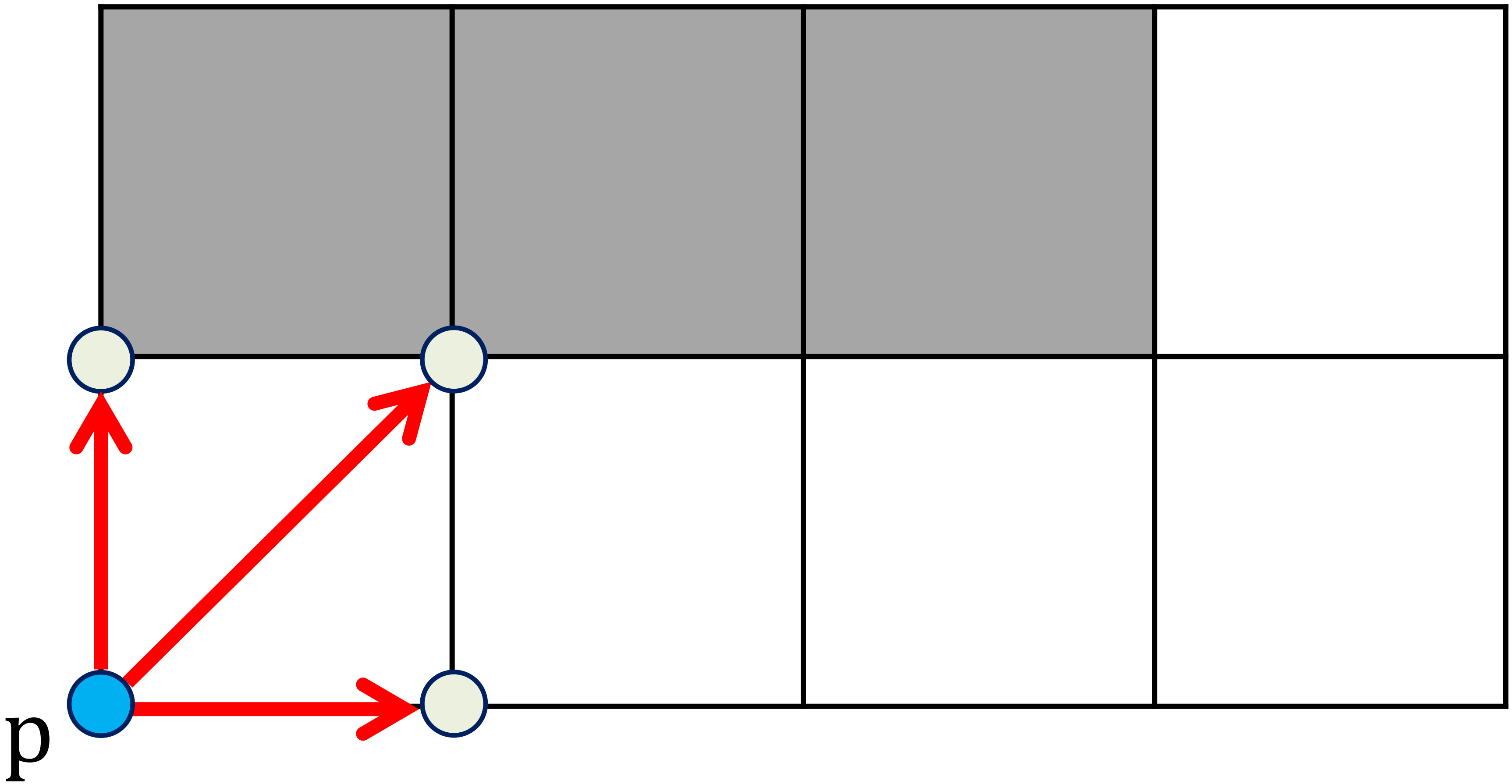


8-directional path!

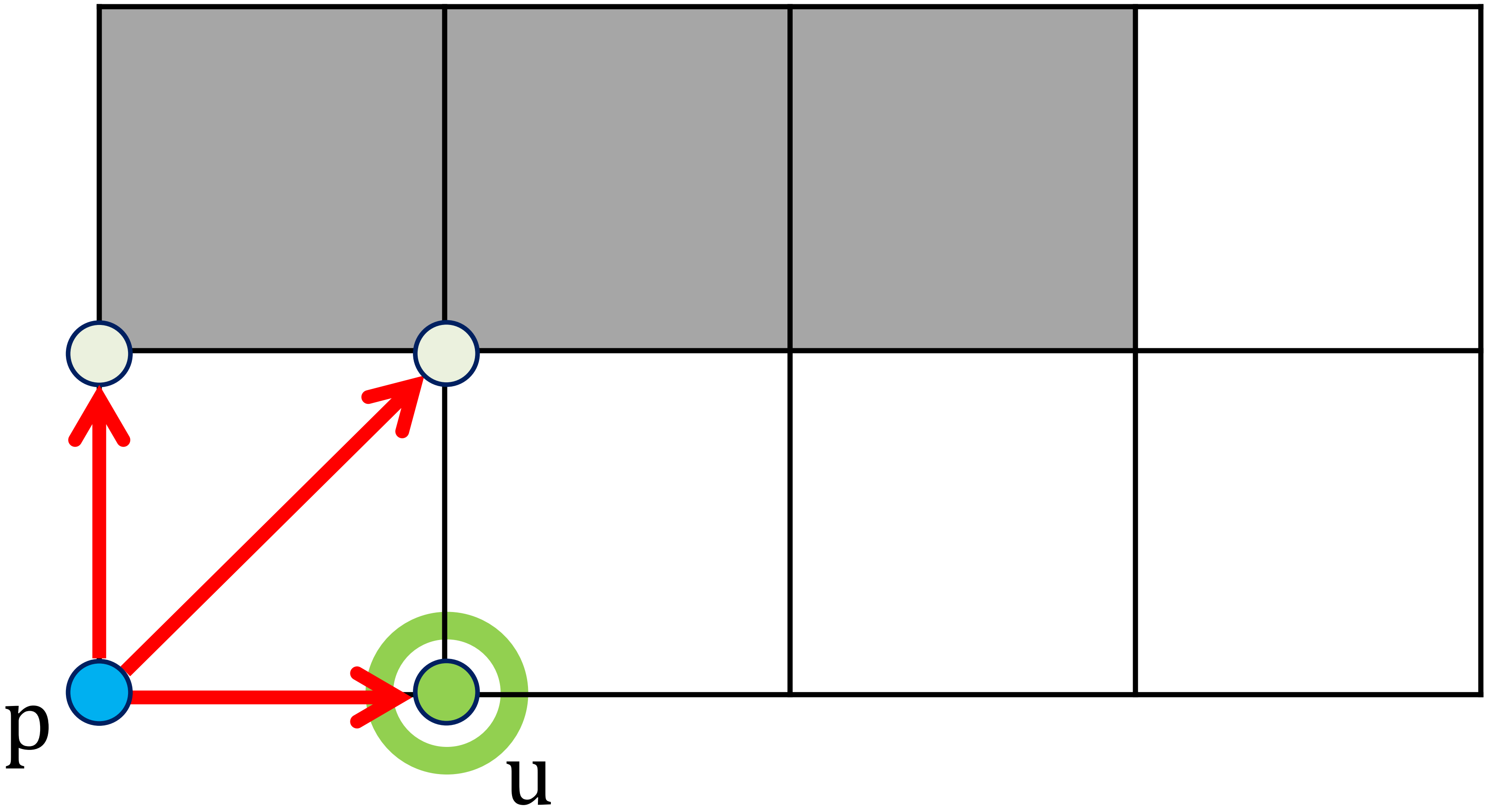
Basic Theta*

(Daniel, Nash, Koenig, Felner, 2007)

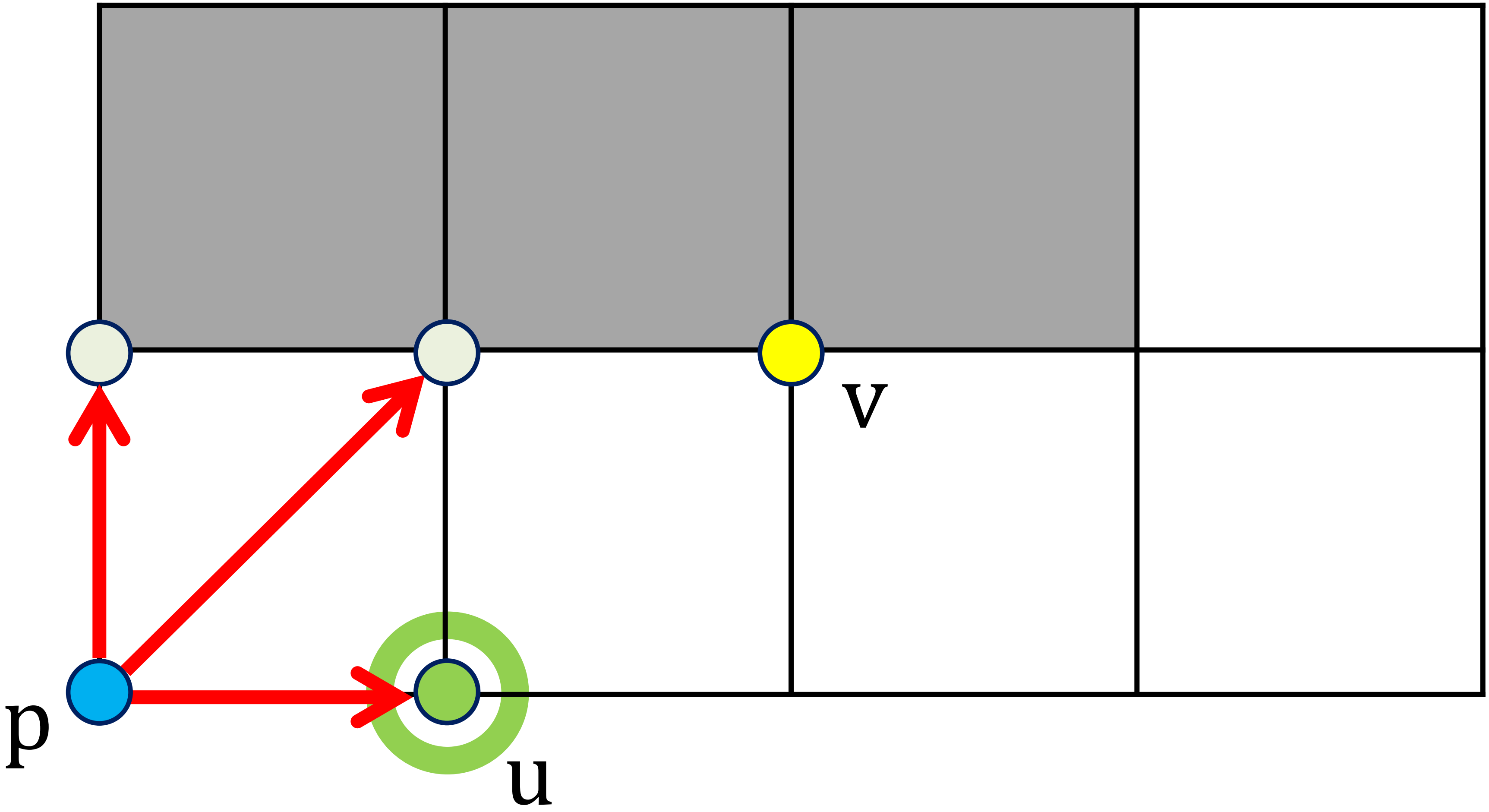
Theta* Algorithm



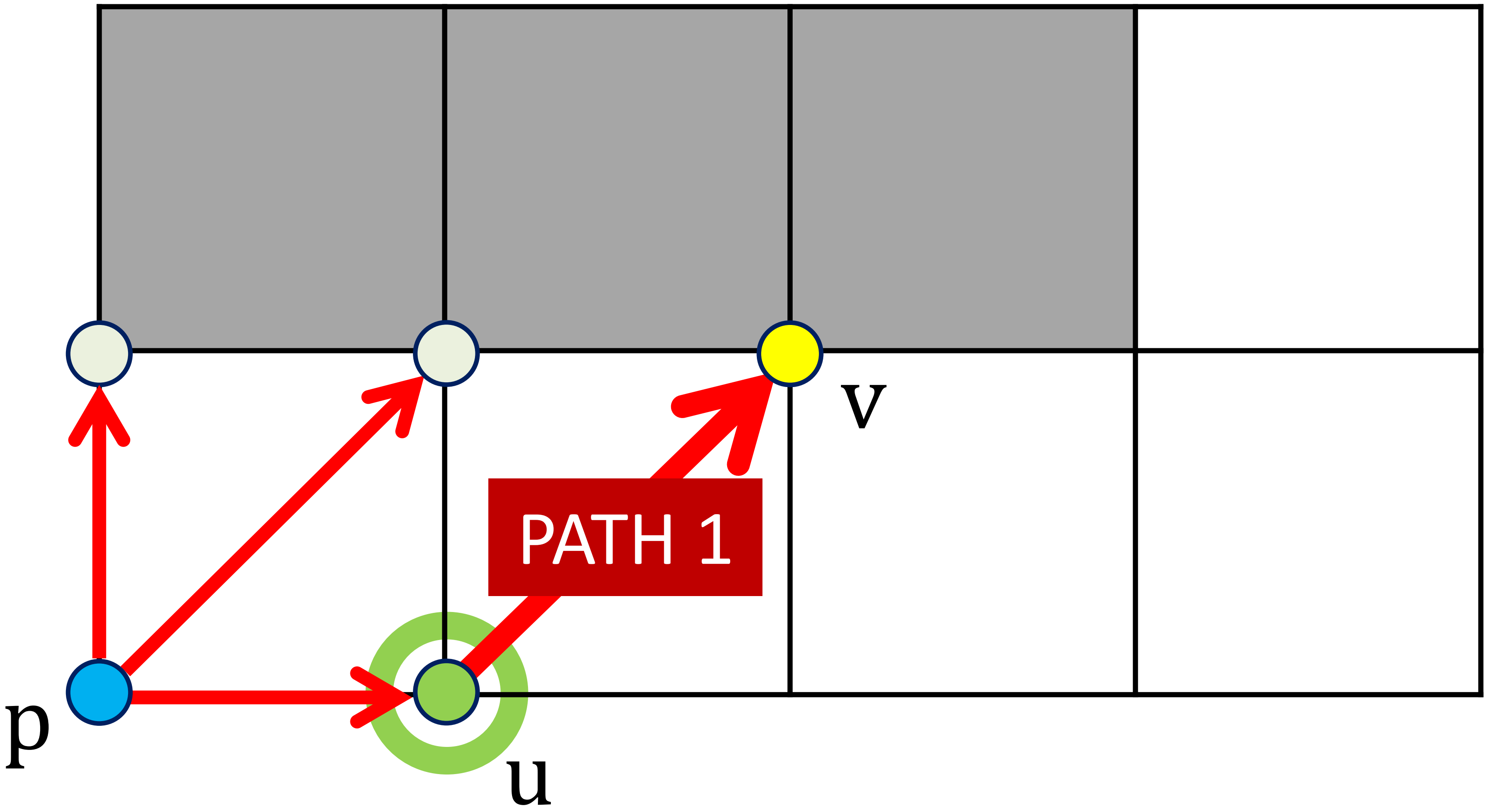
Theta* Algorithm



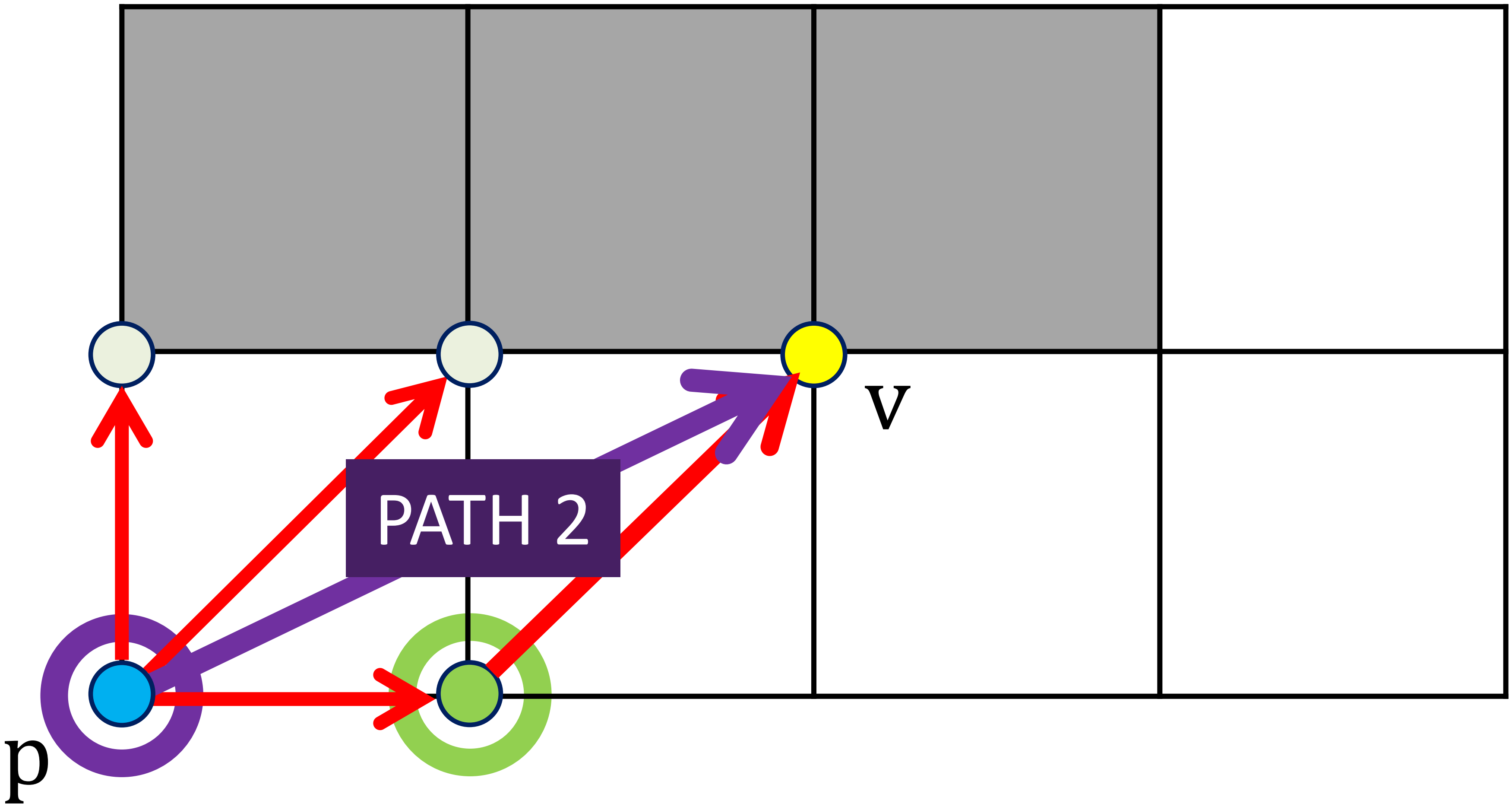
Theta* Algorithm



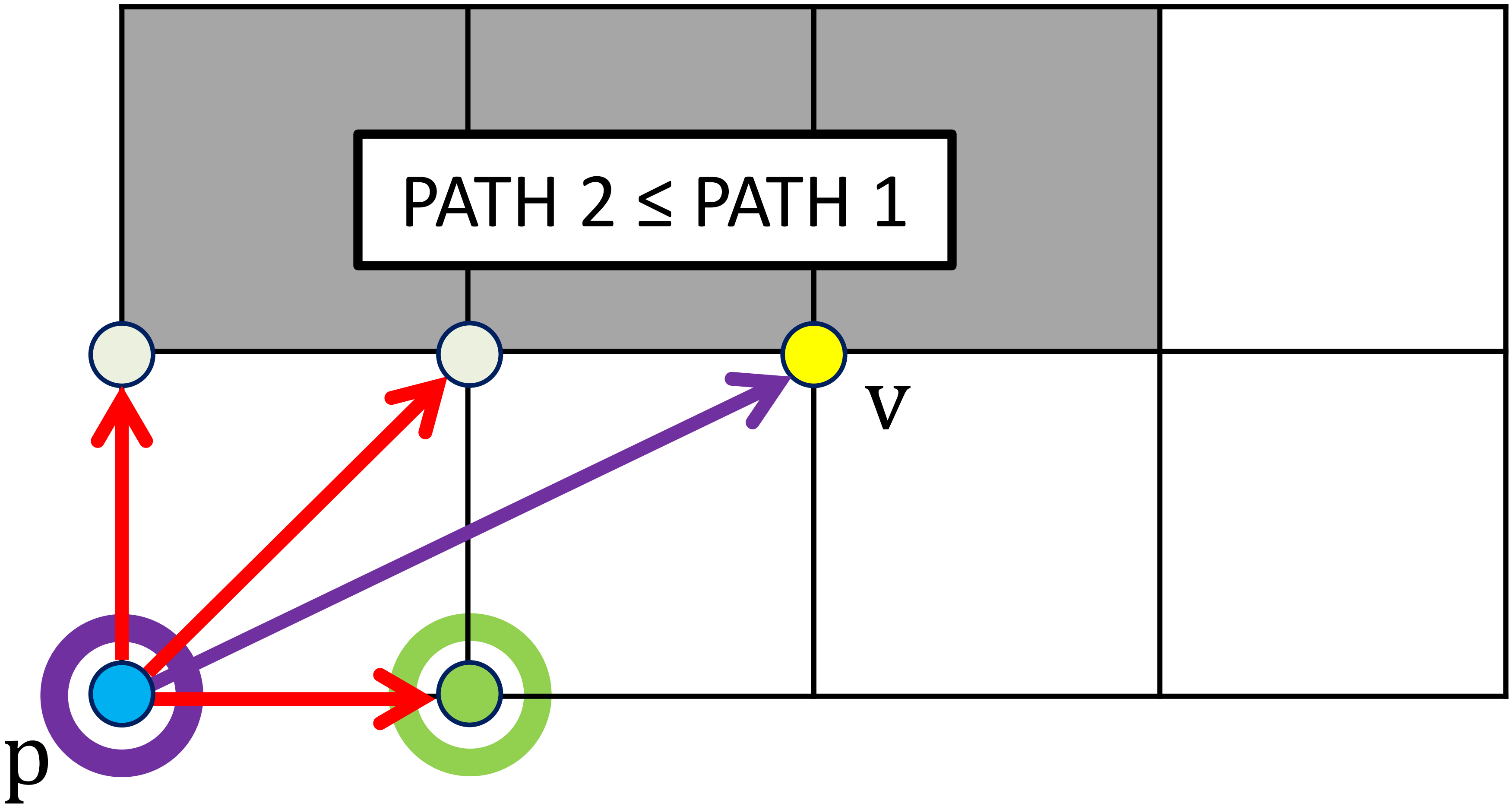
Theta* Algorithm



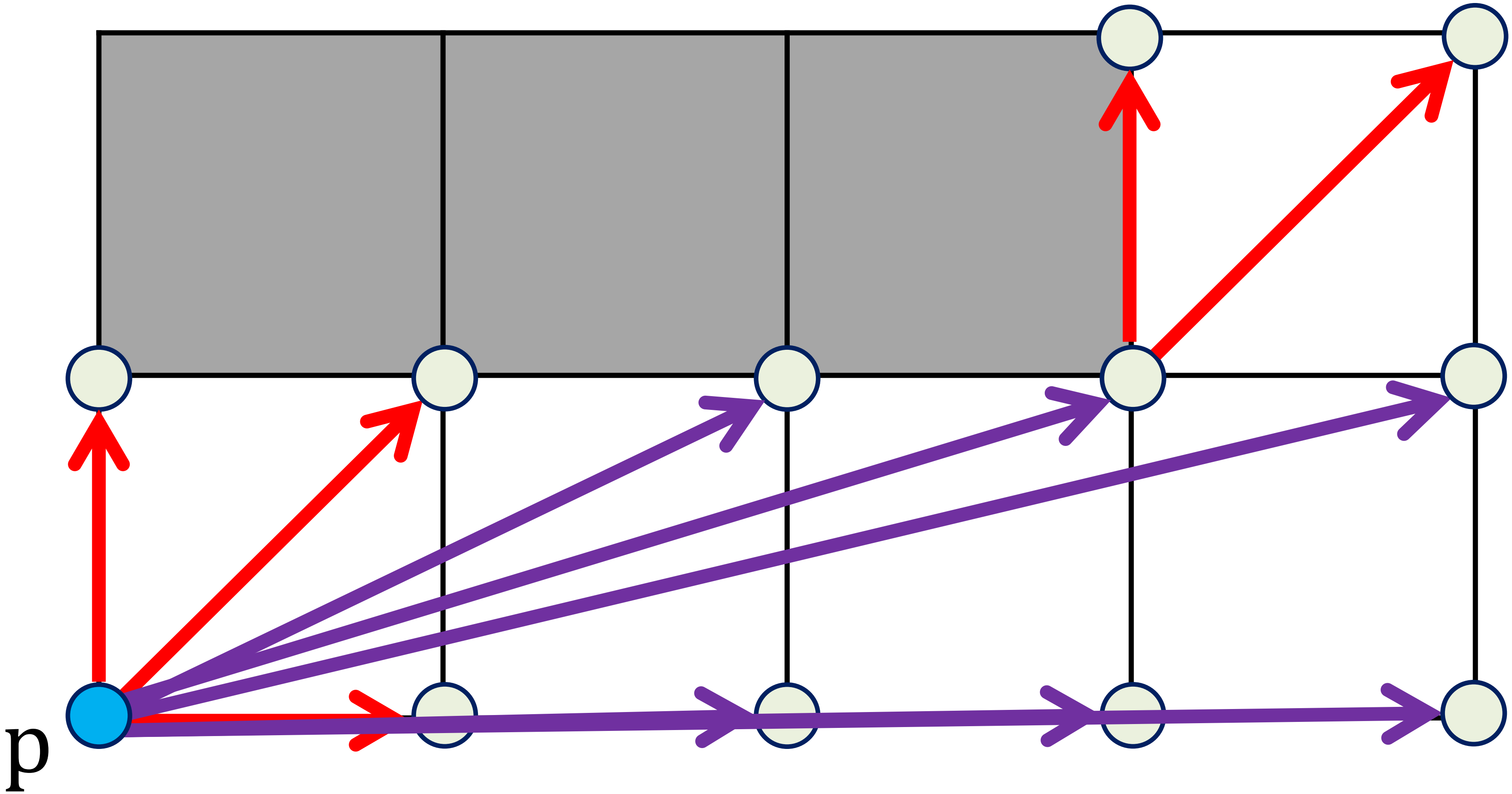
Theta* Algorithm



Theta* Algorithm



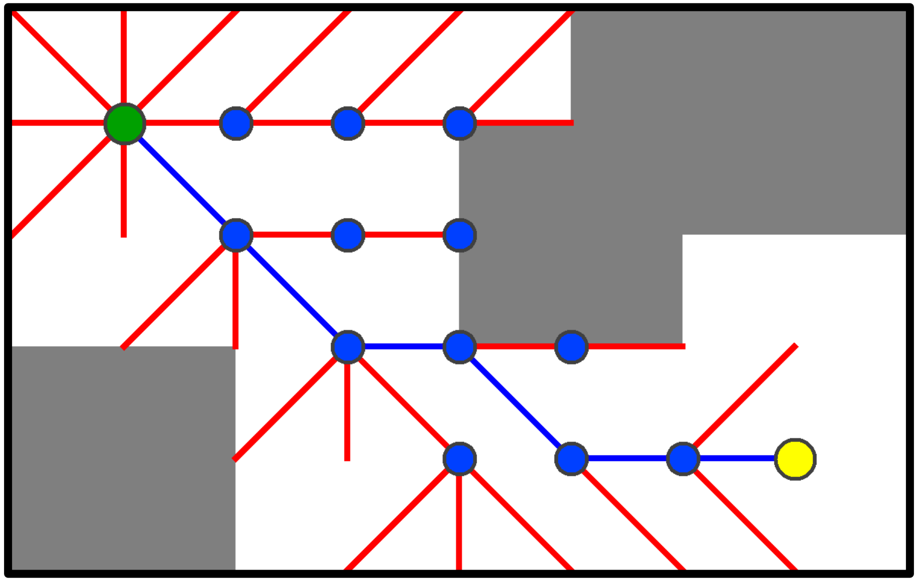
Theta* Algorithm

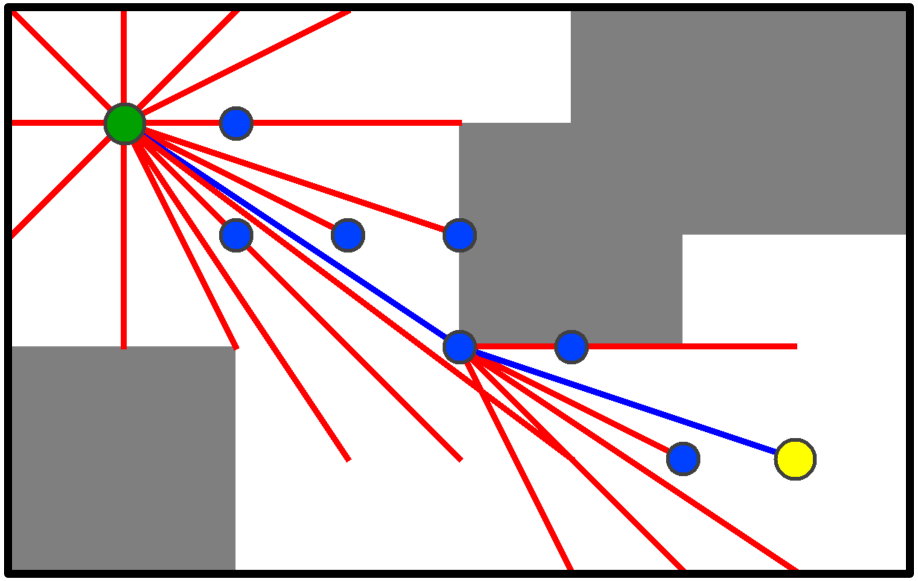


Basic Theta*

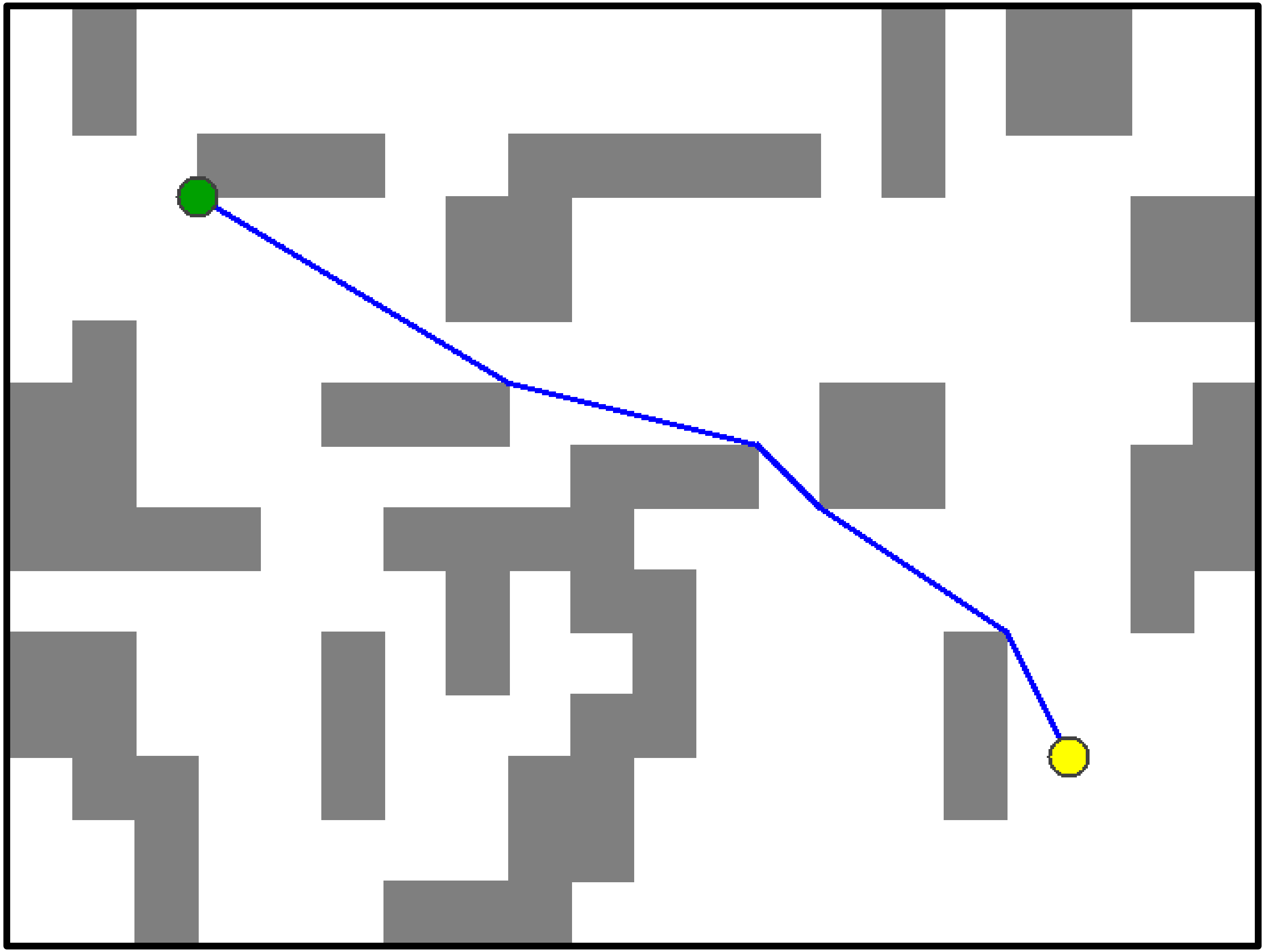
8-directional A*

Theta*

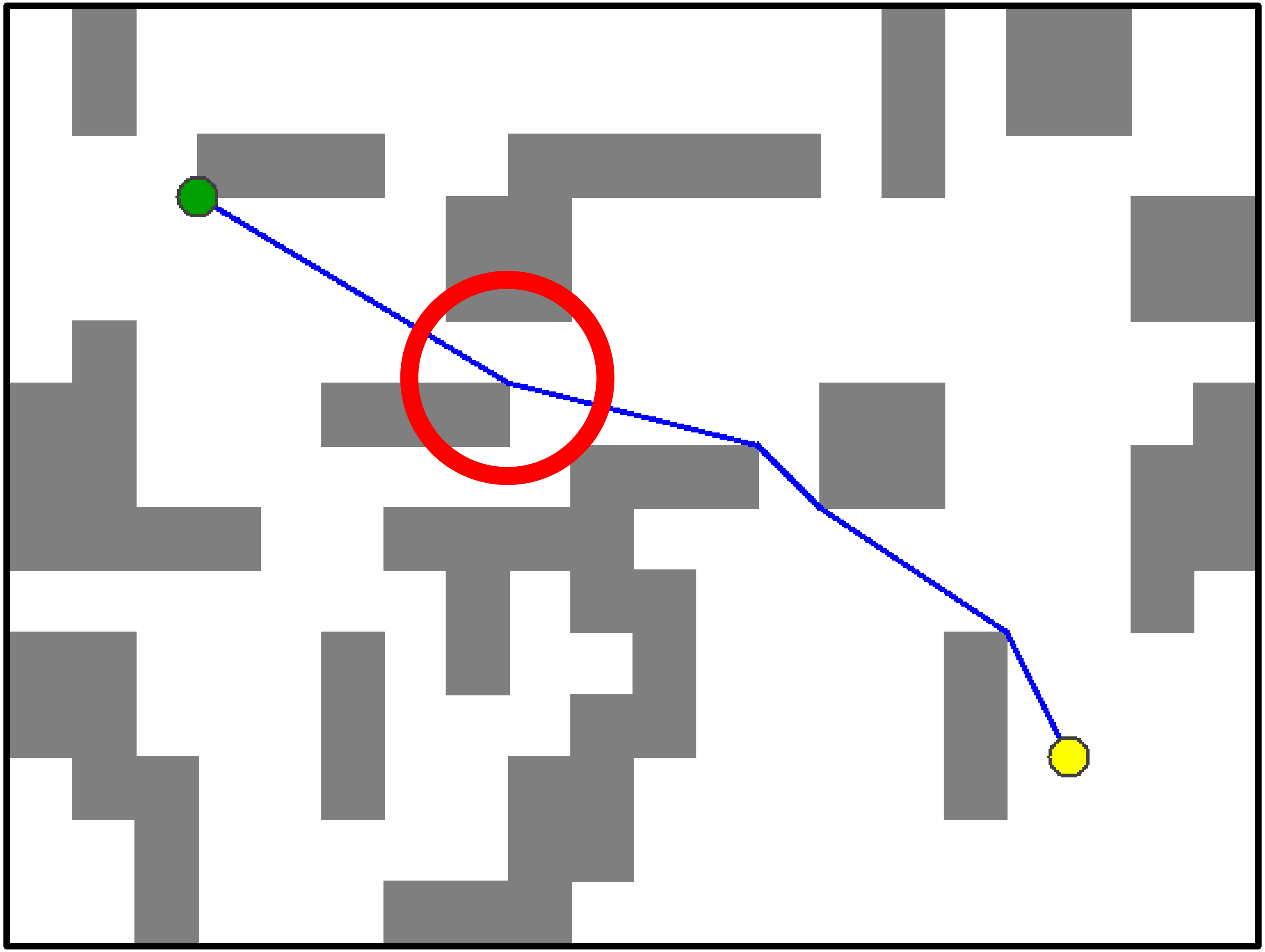




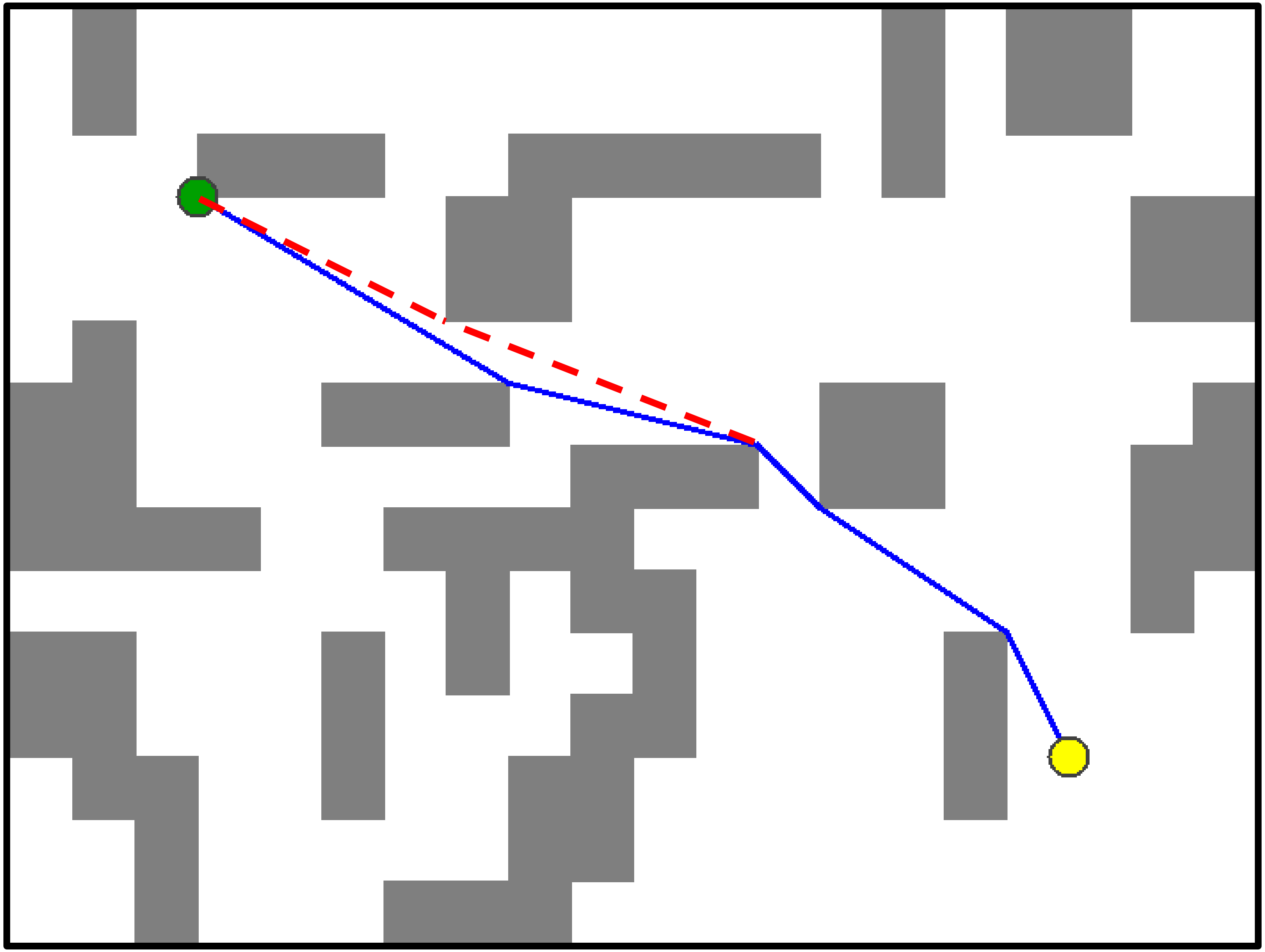
Basic Theta*



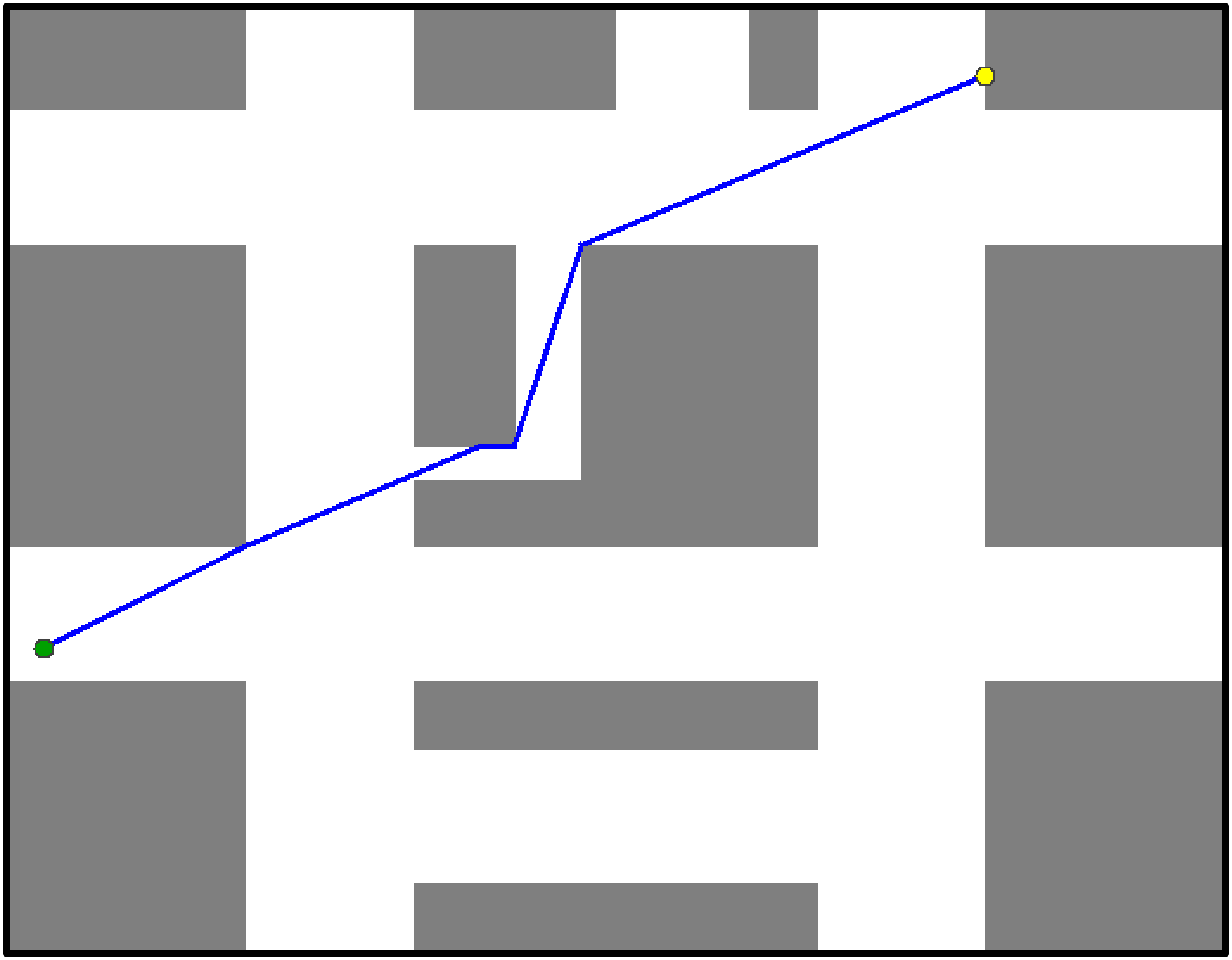
Basic Theta*



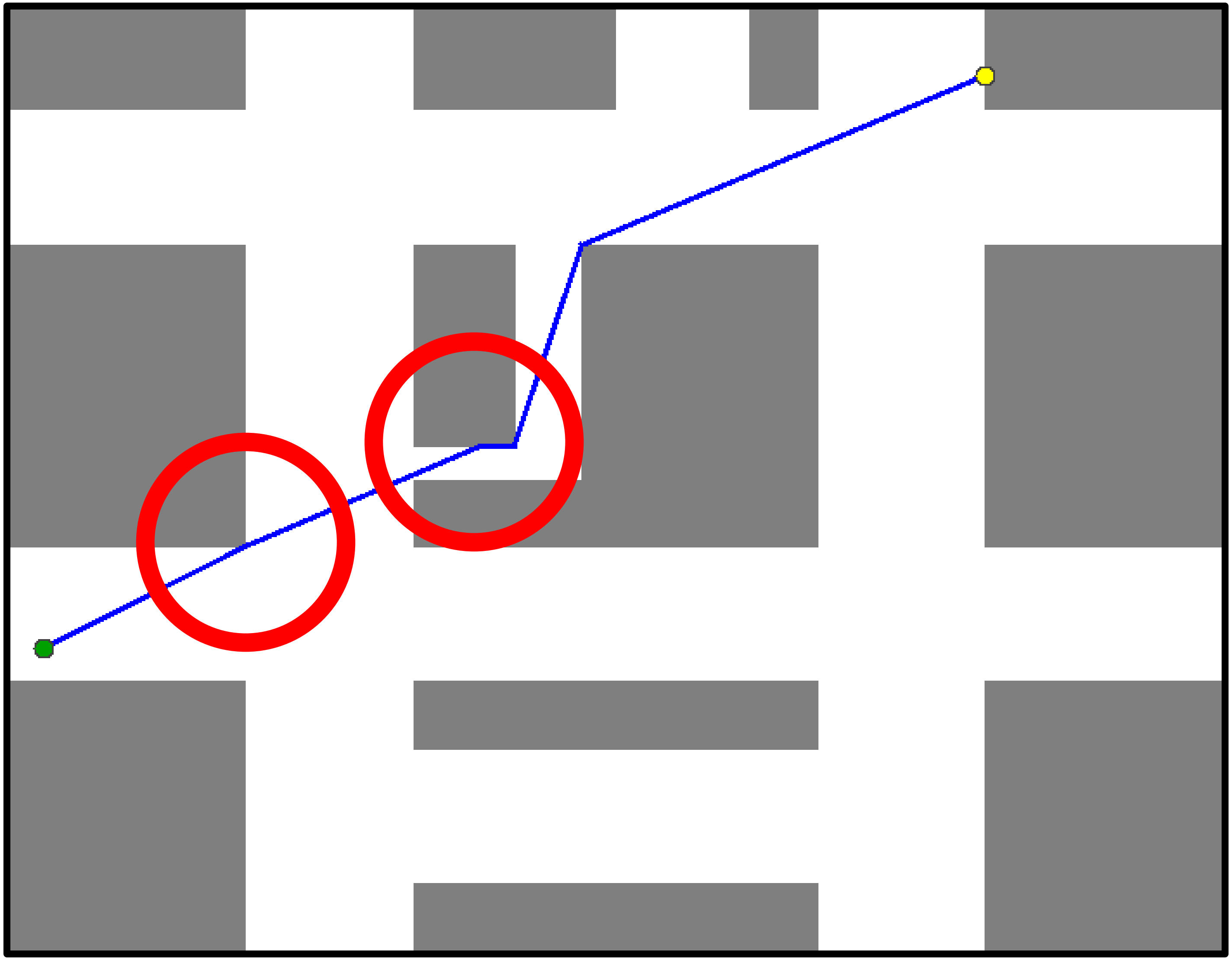
Basic Theta*



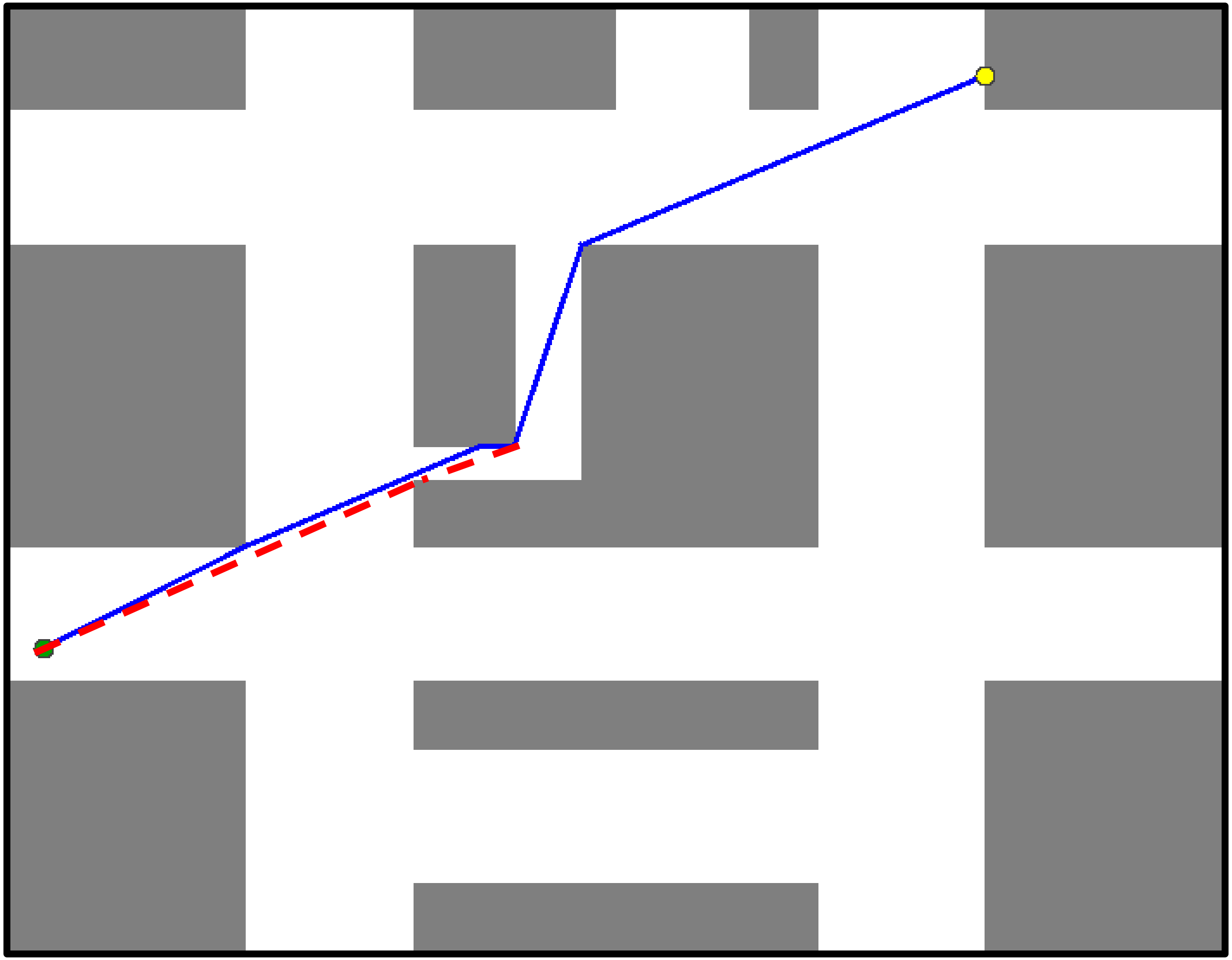
Basic Theta*



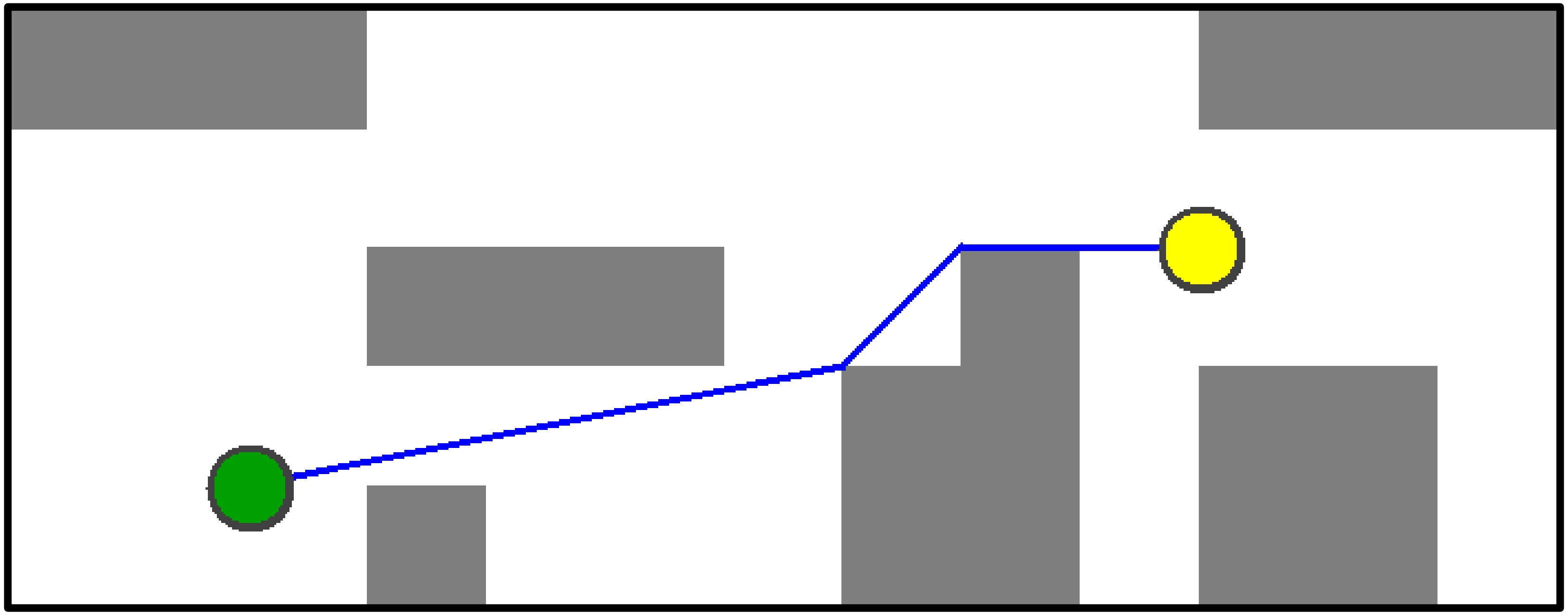
Basic Theta*



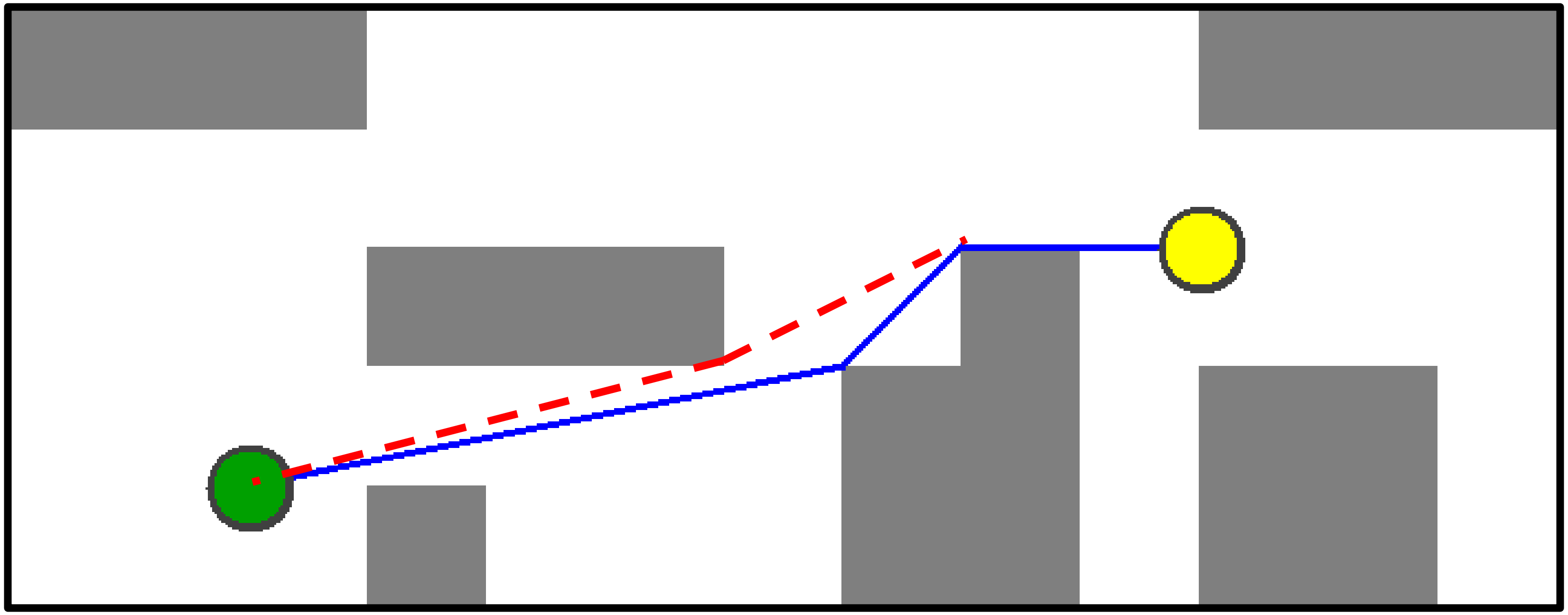
Basic Theta*



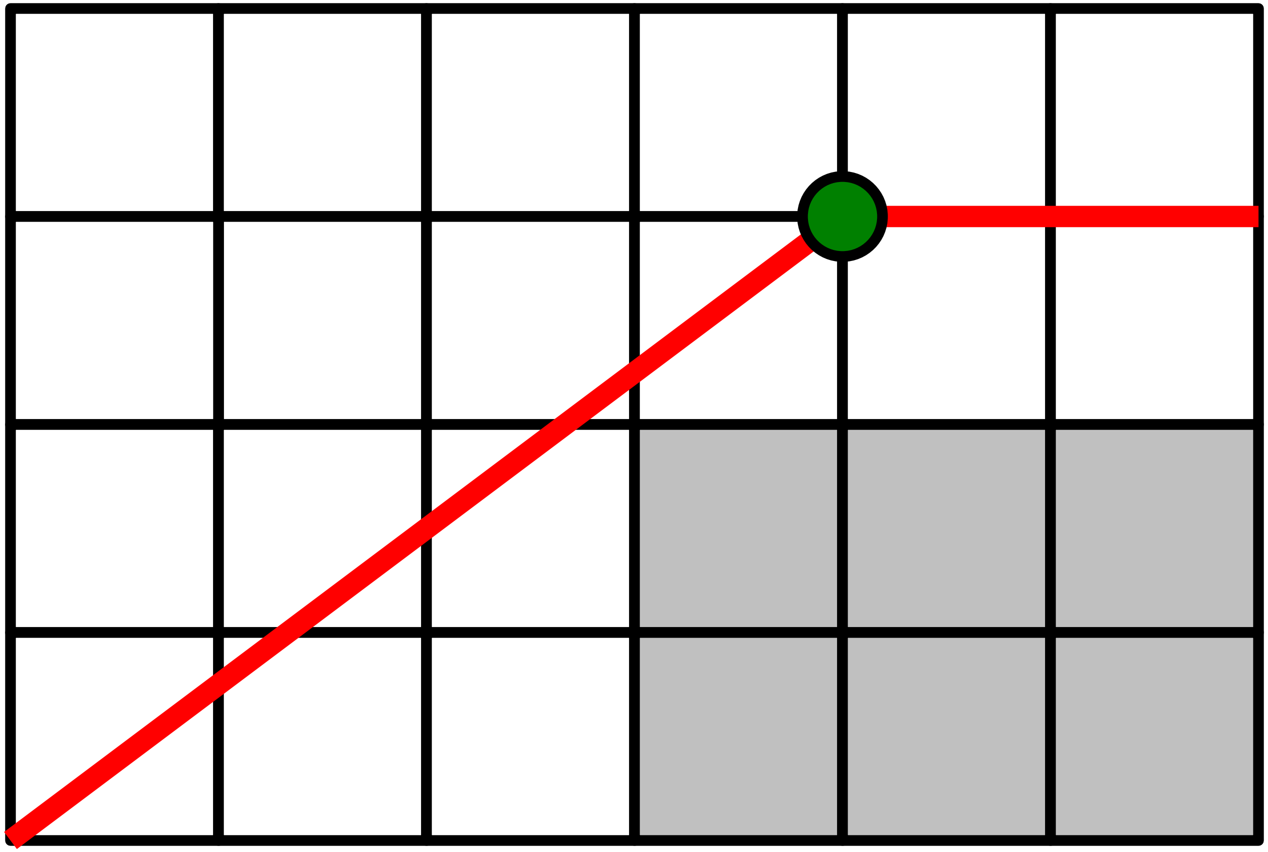
Basic Theta*



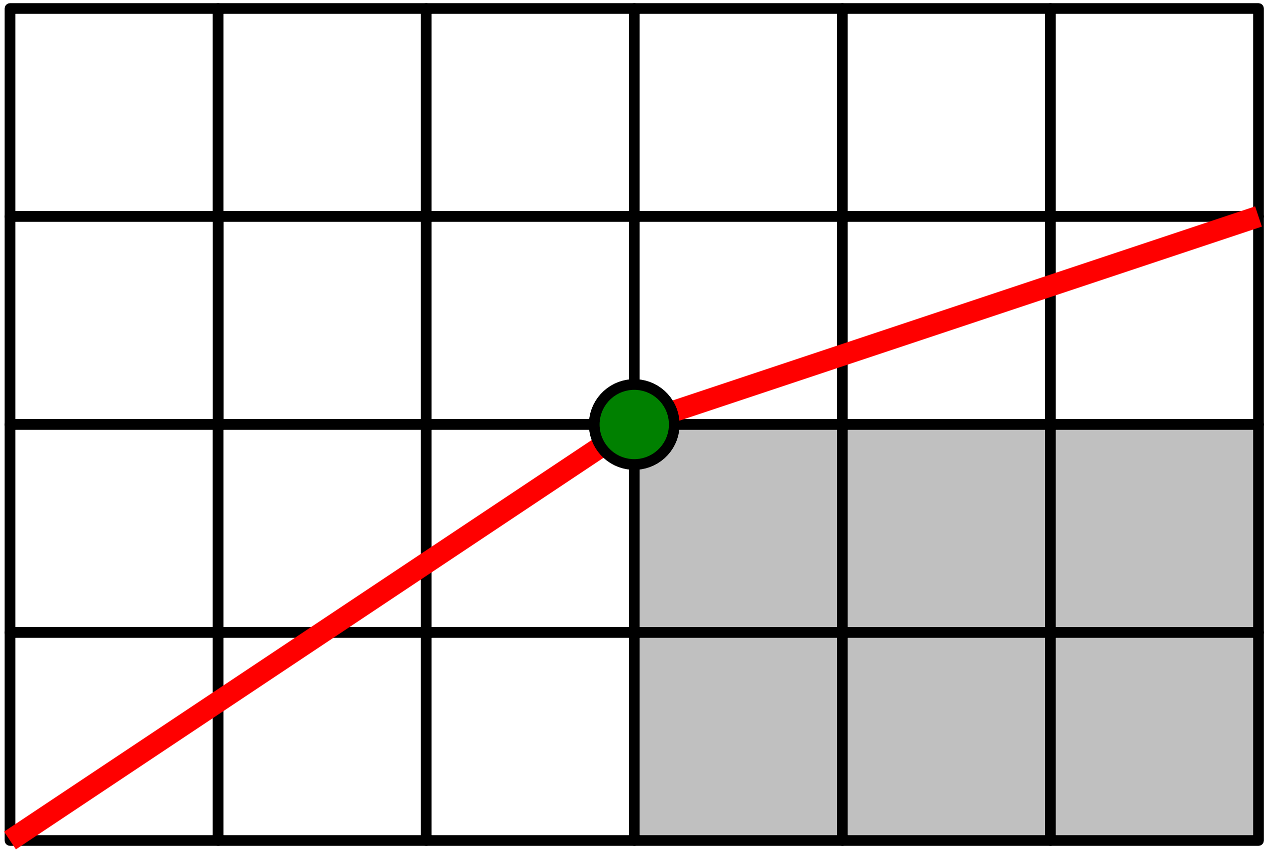
Basic Theta*



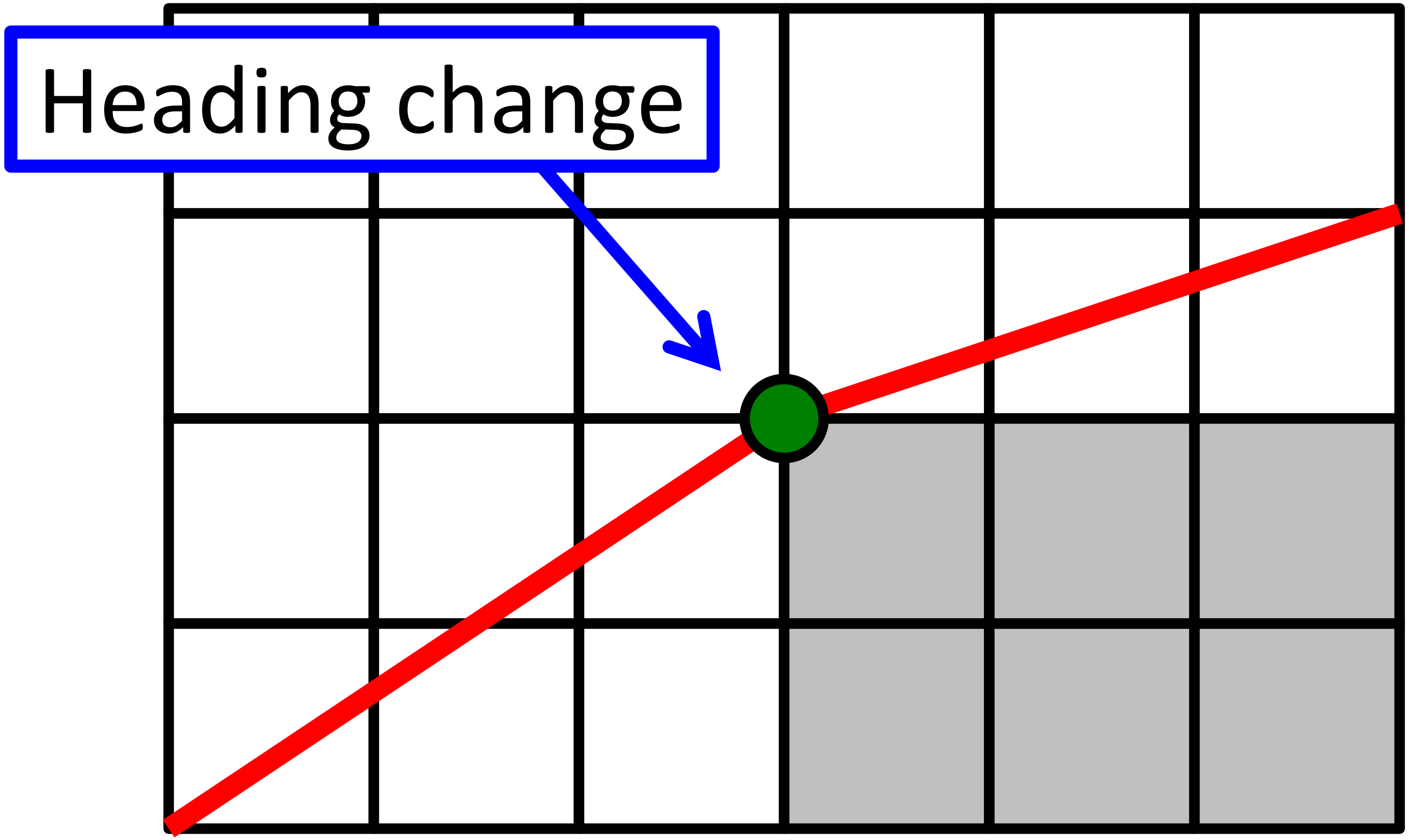
Non-Taut Path



Taut Path

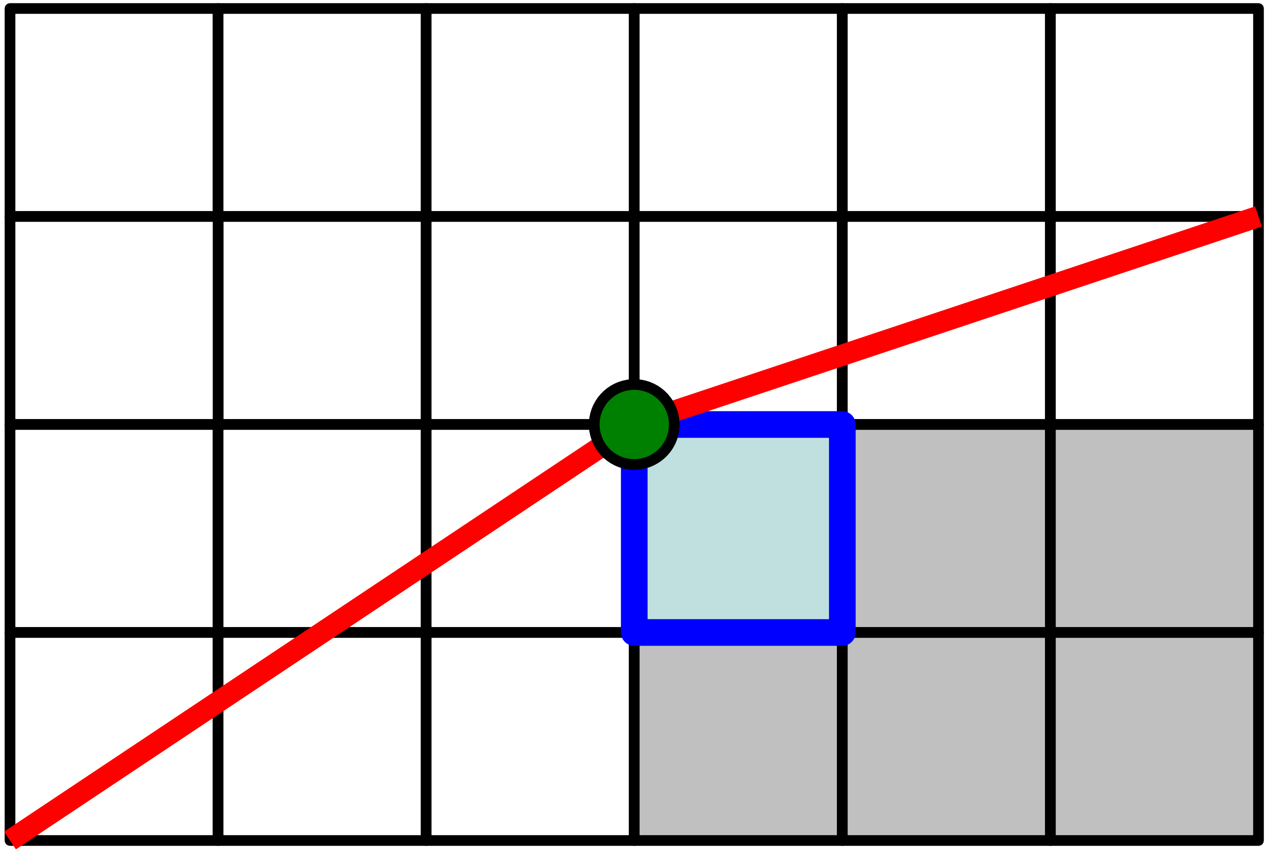


Taut Path



Heading change

Taut Path



Strict Theta*

Shorter Motion Planning Using Taut Paths

Strict Theta*

Strict Theta*

Easy to implement

Much shorter paths

Low runtime overhead

Idea:

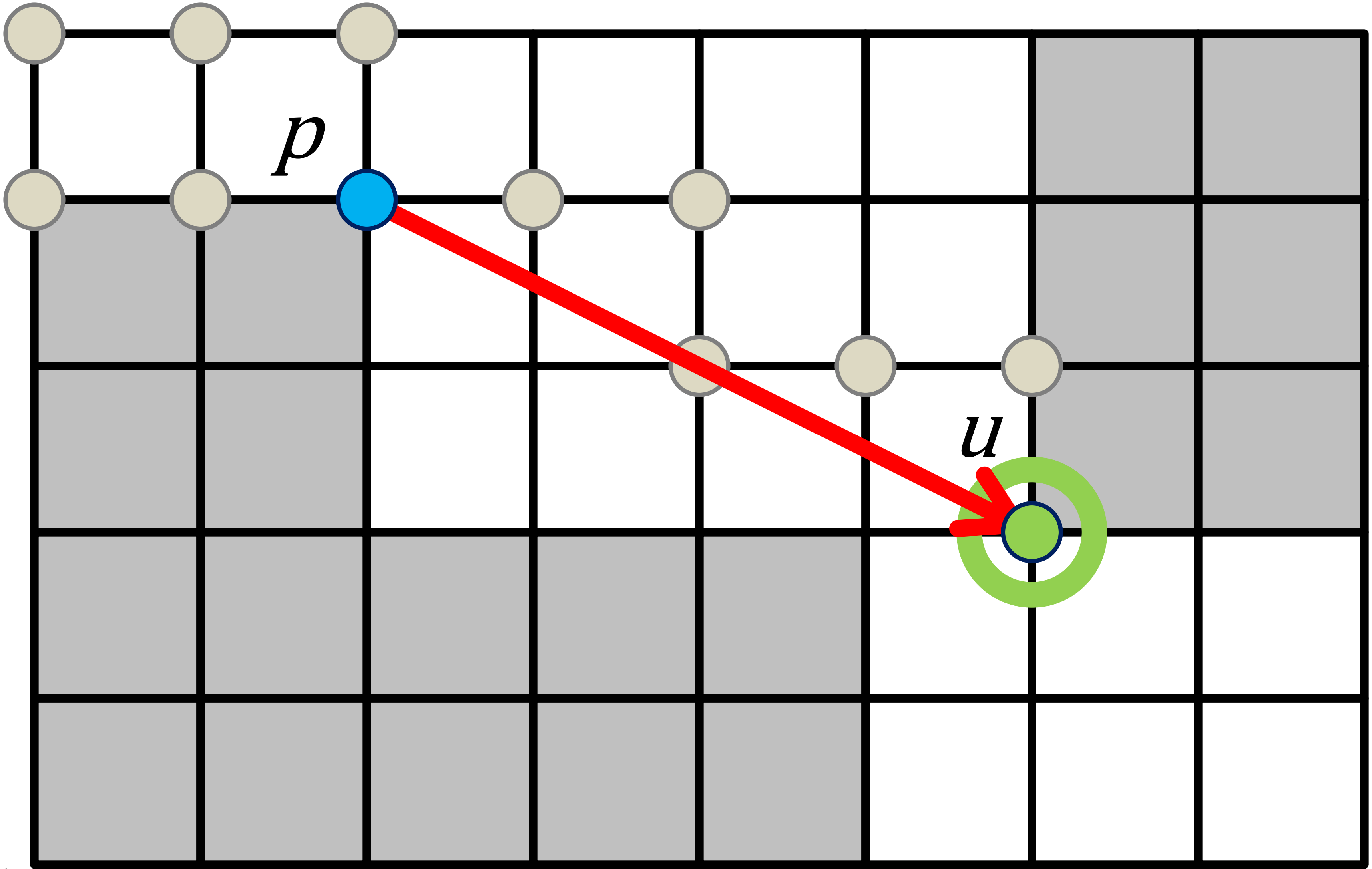
Restrict the search to Taut Paths

Idea:

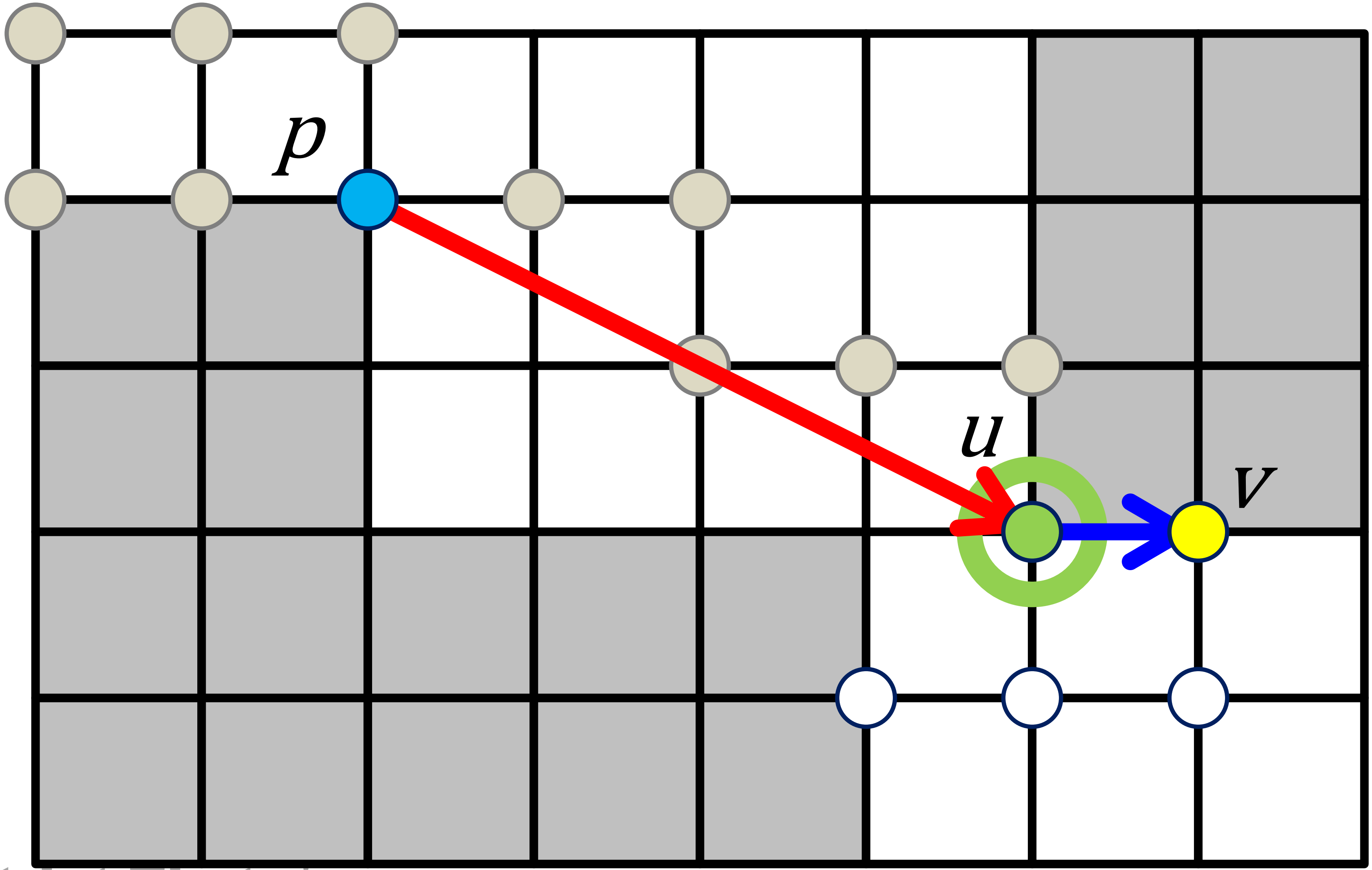
“Restrict” the search to Taut Paths

Penalise non-taut paths

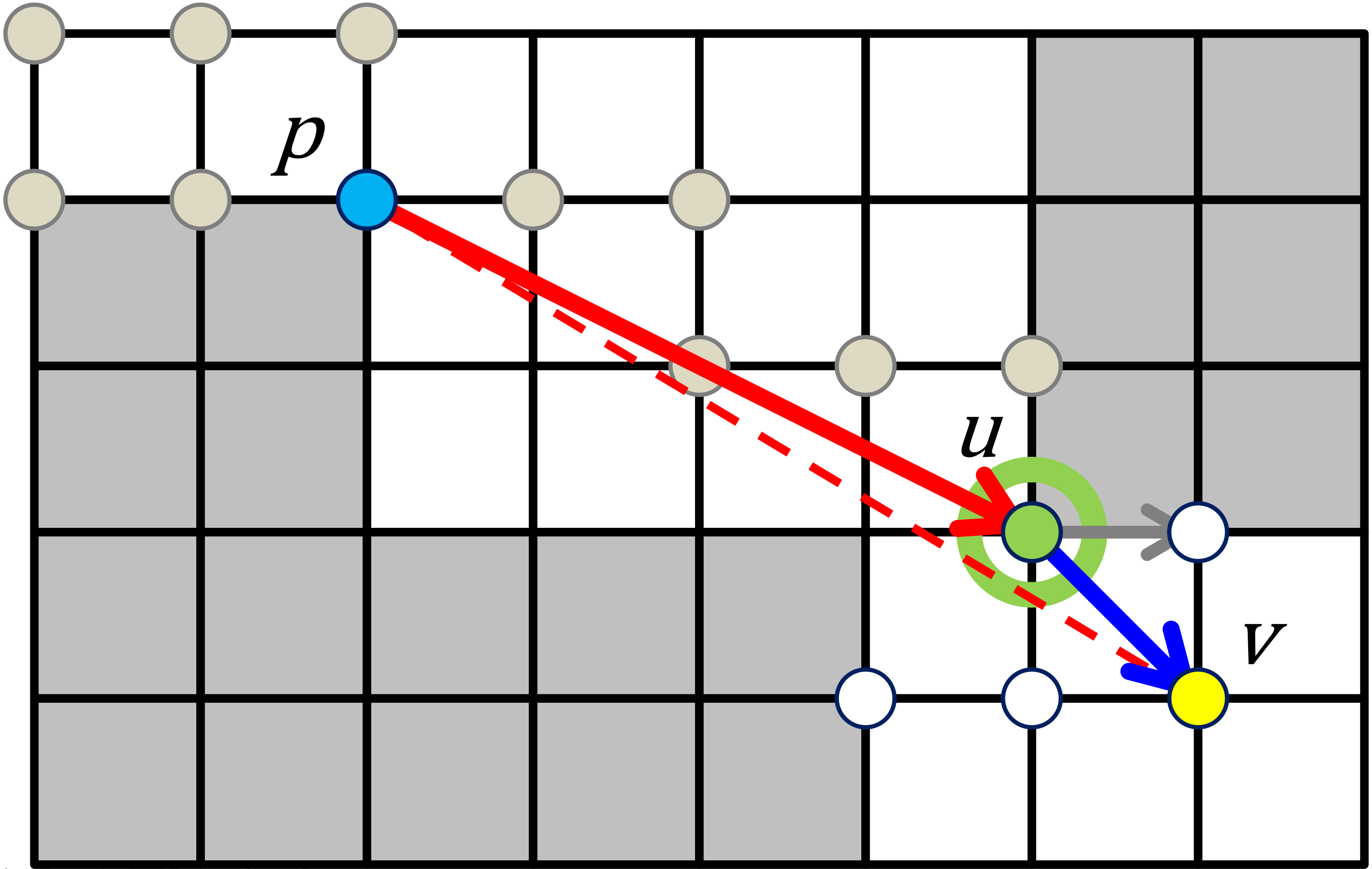
Strict Theta*



Is Taut

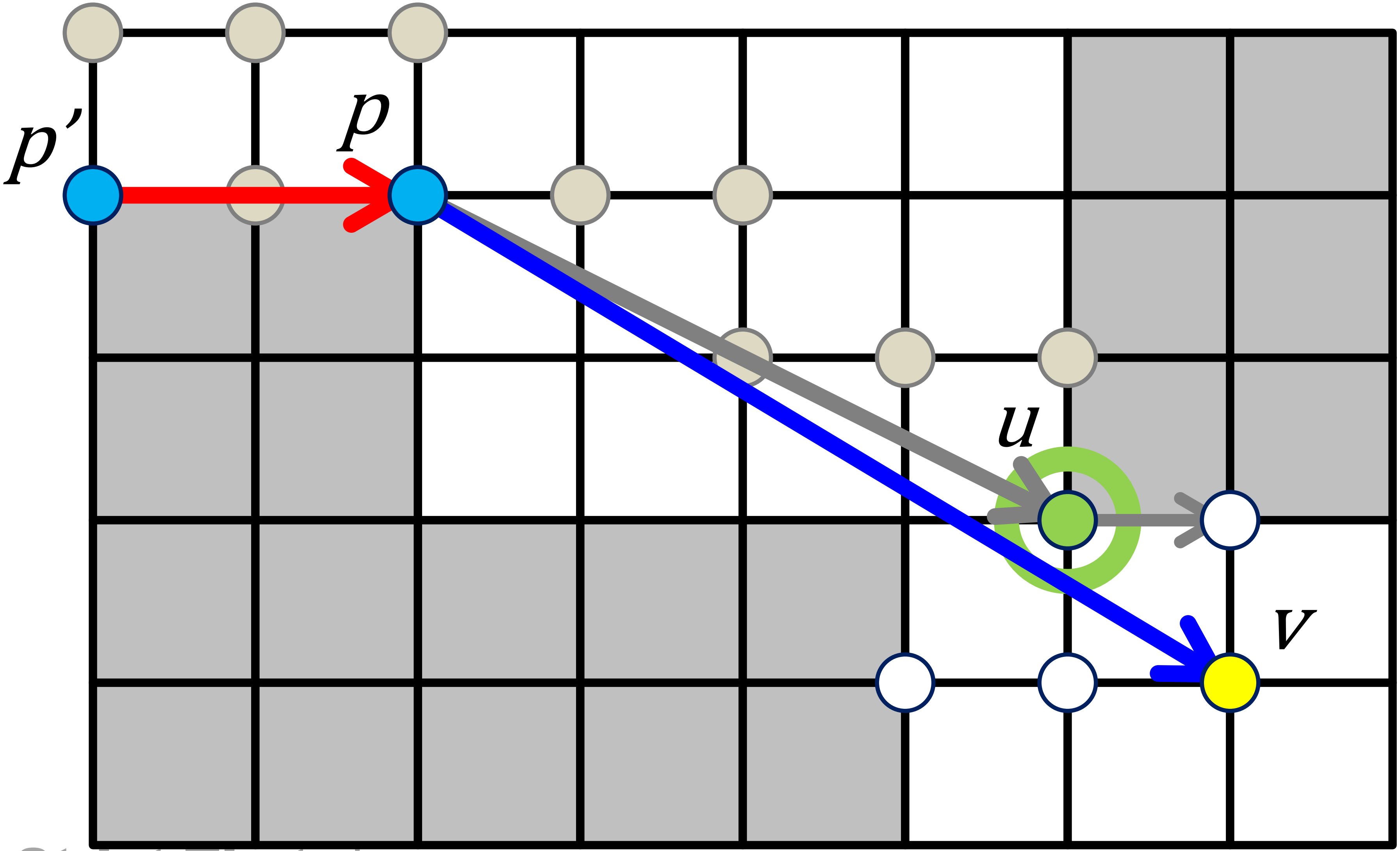


Strict Theta*



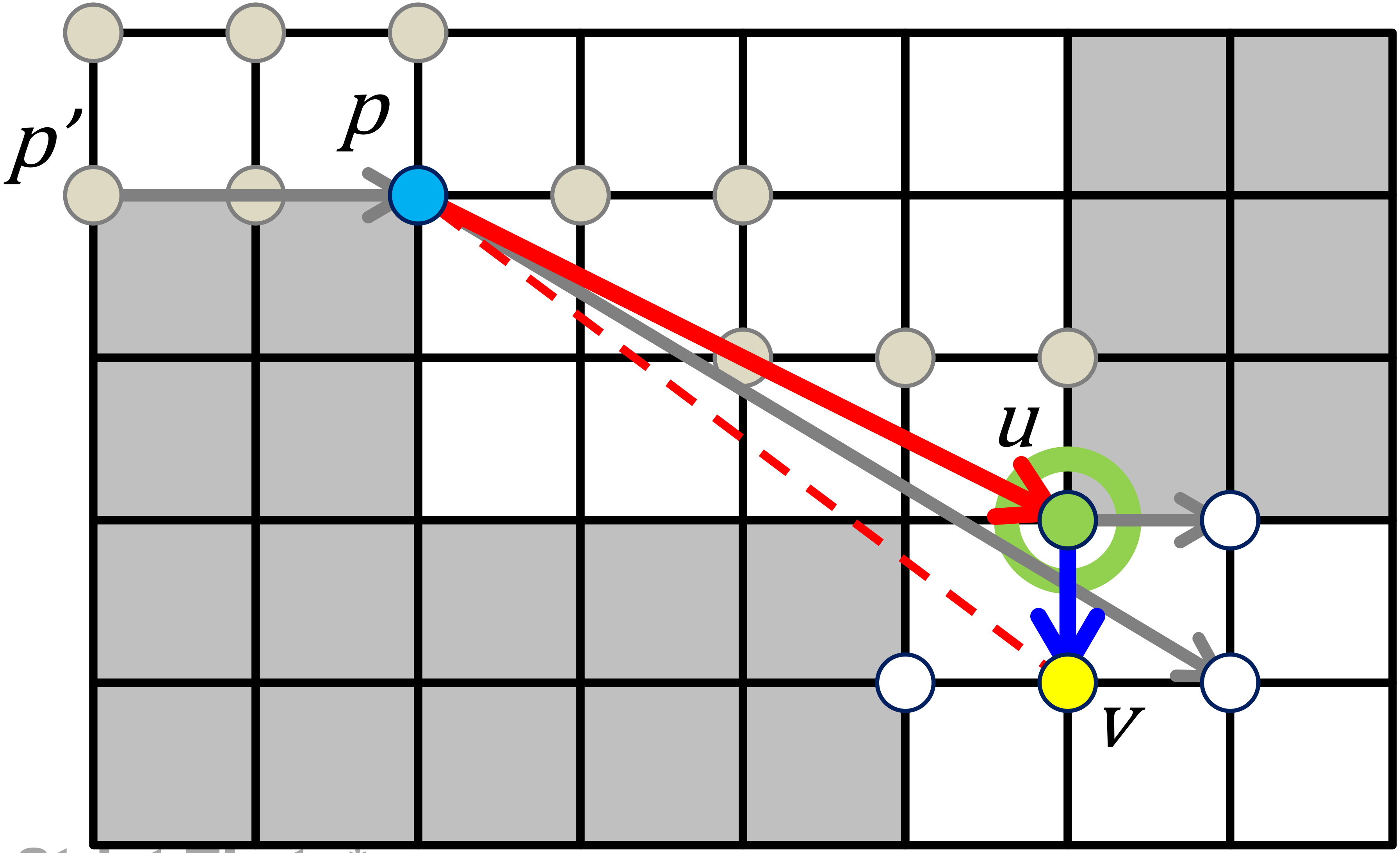
Strict Theta*

Is Taut



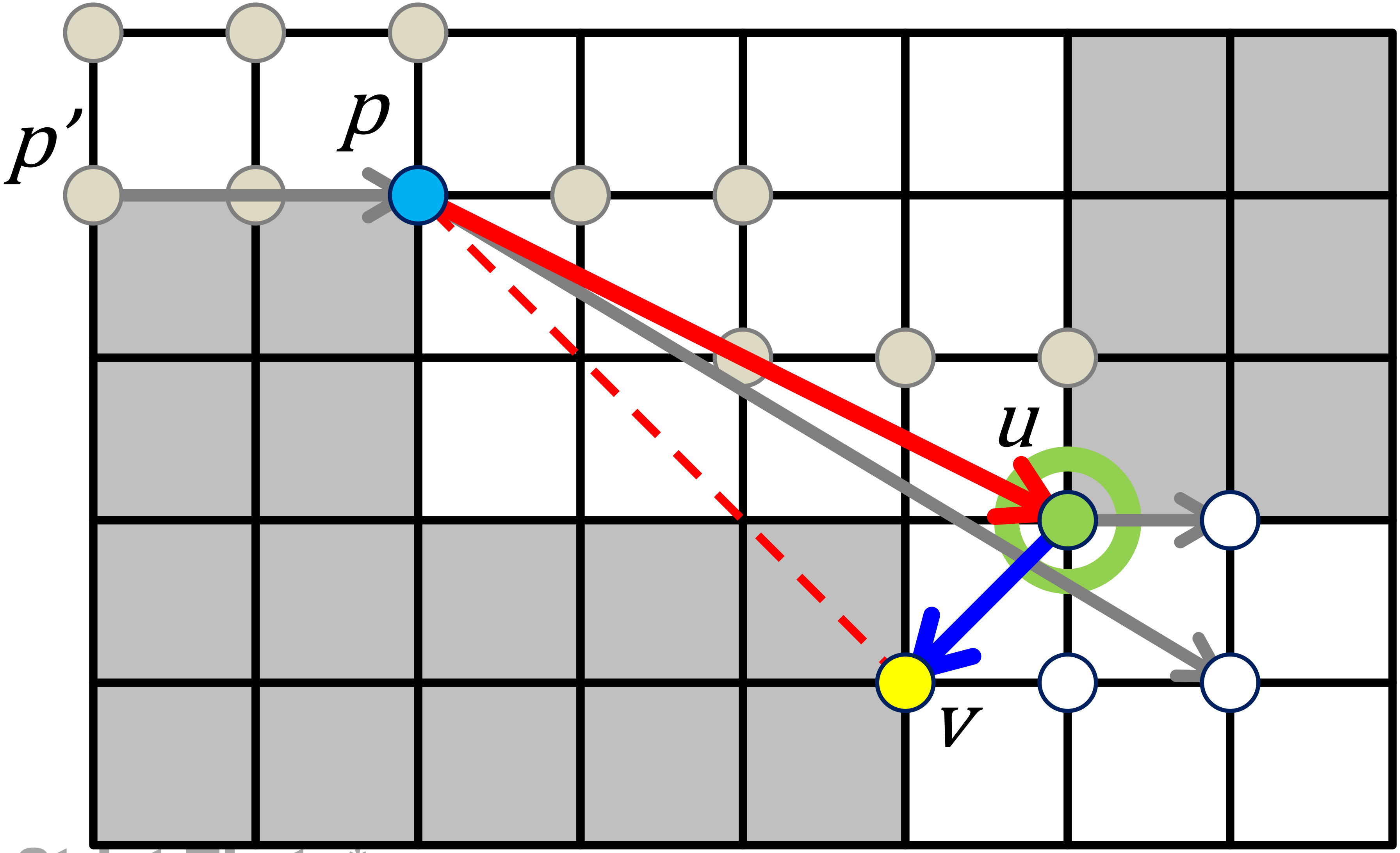
Strict Theta*

Not Taut



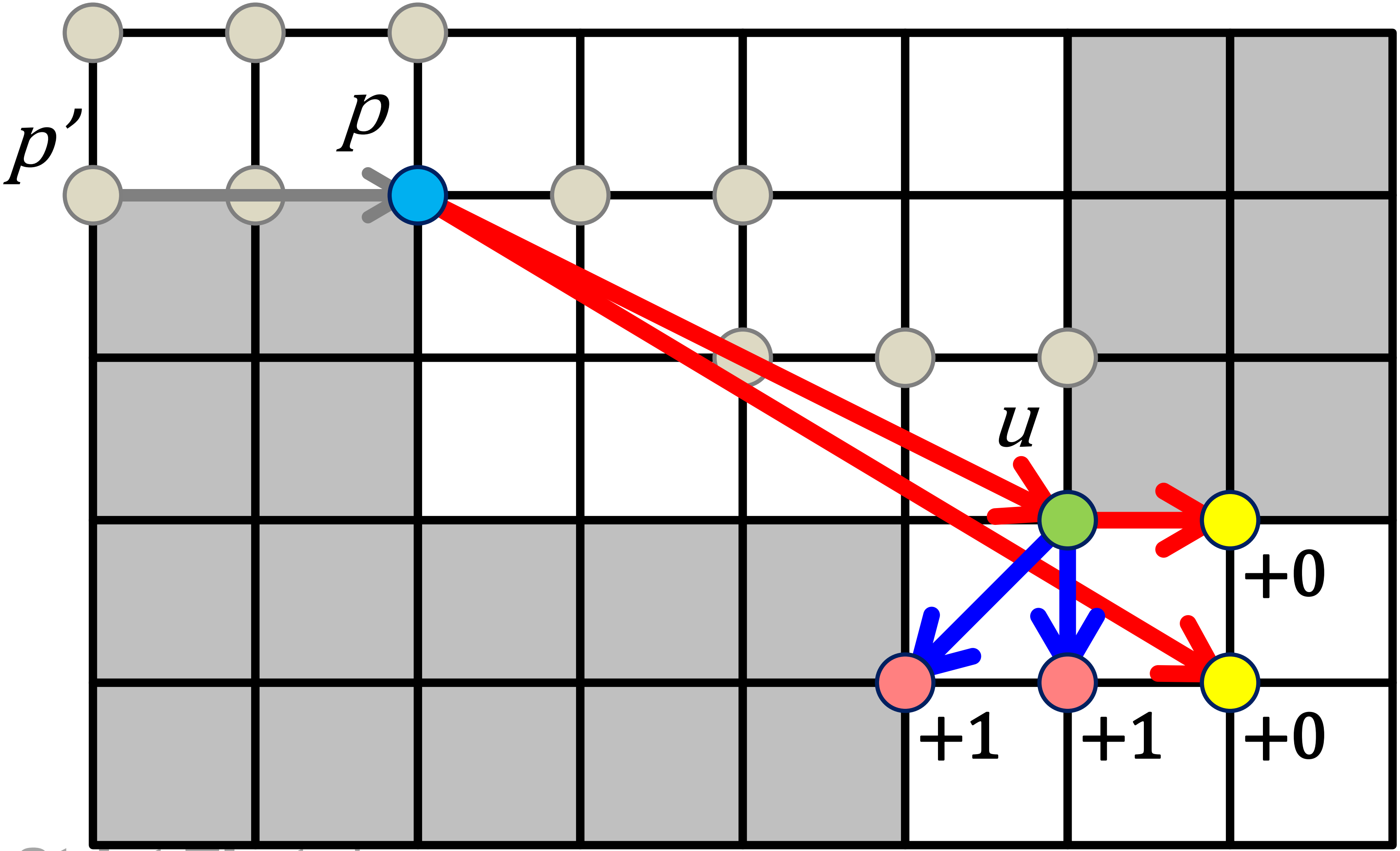
Strict Theta*

Not Taut

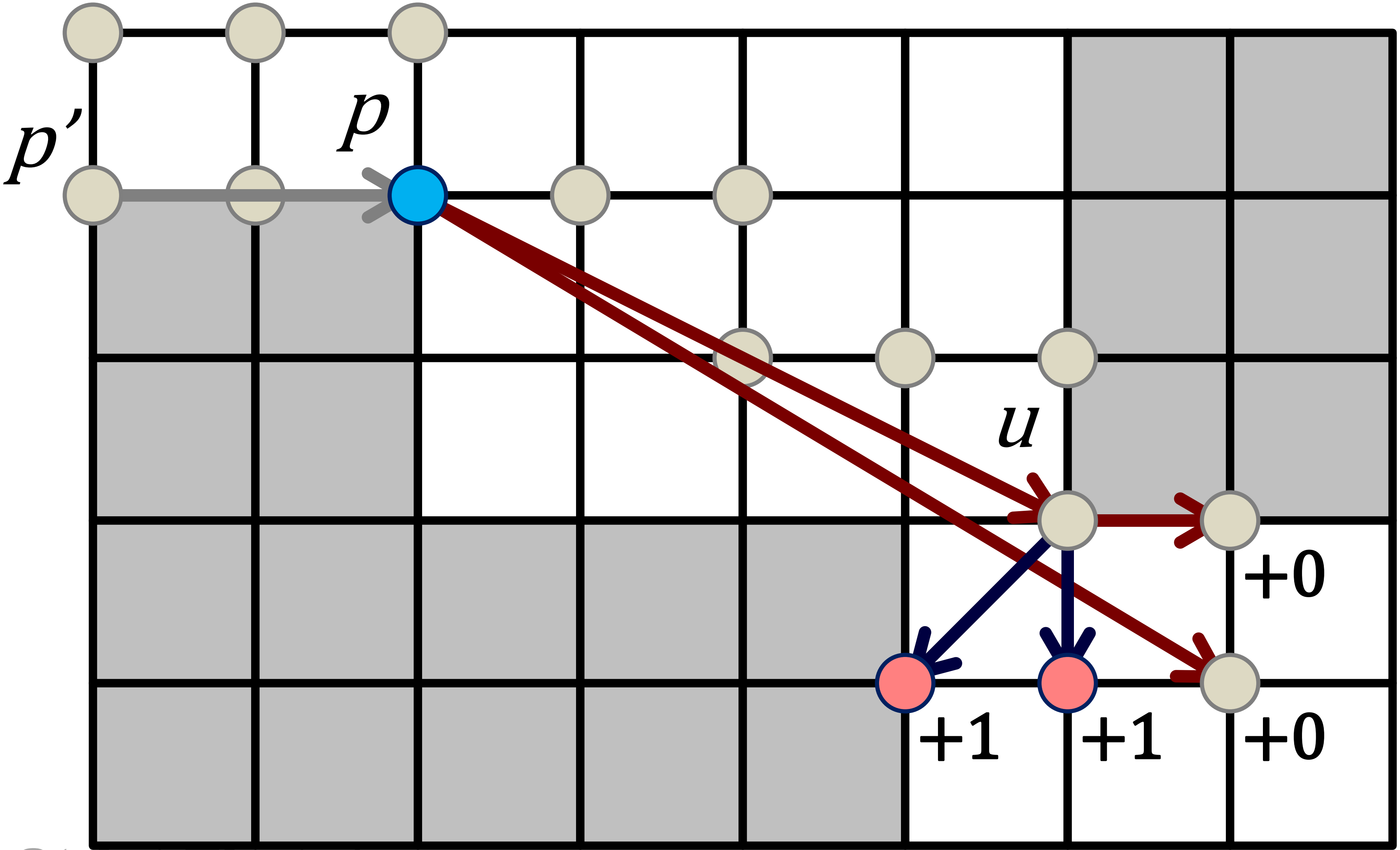


Strict Theta*

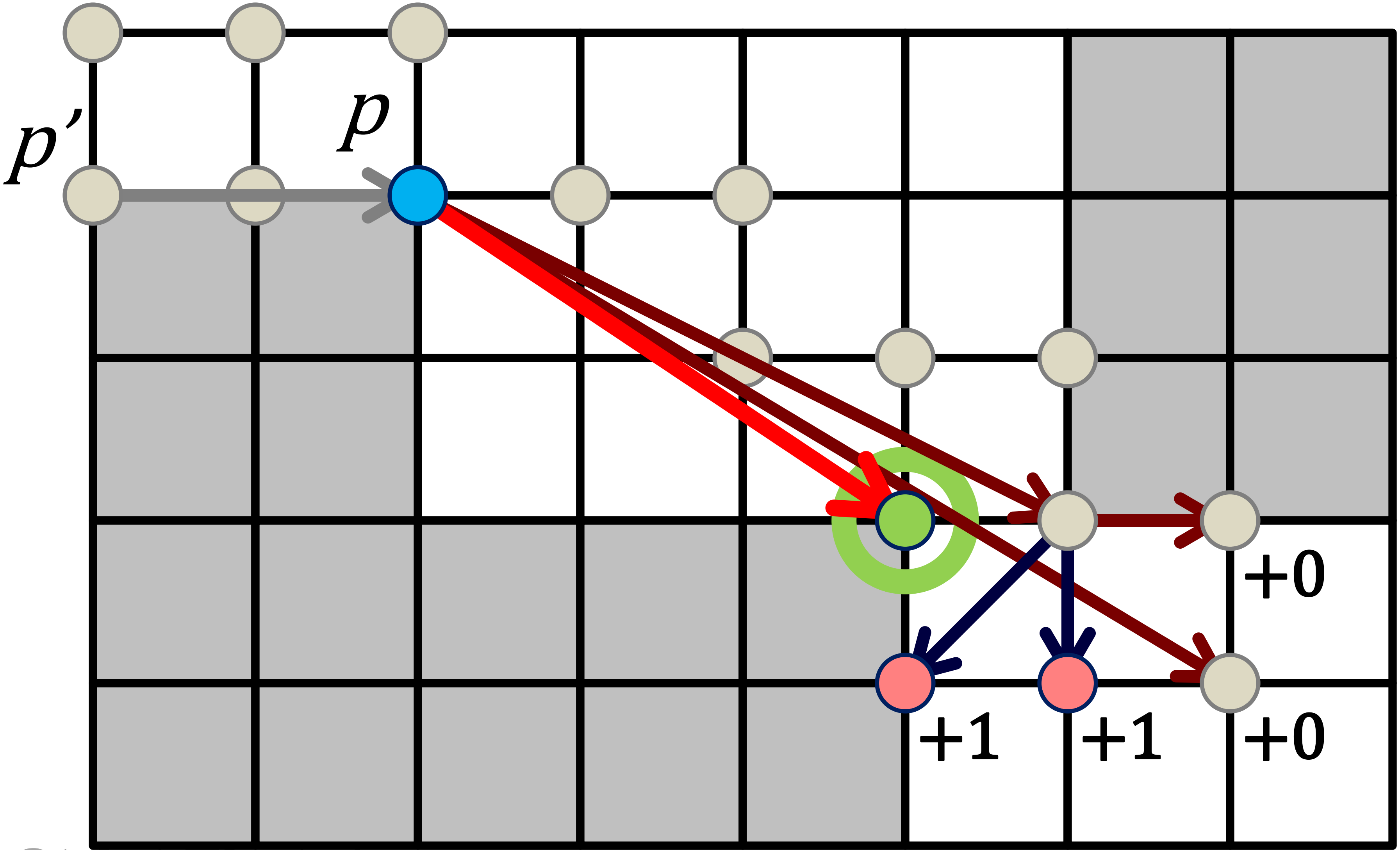
Add Penalties



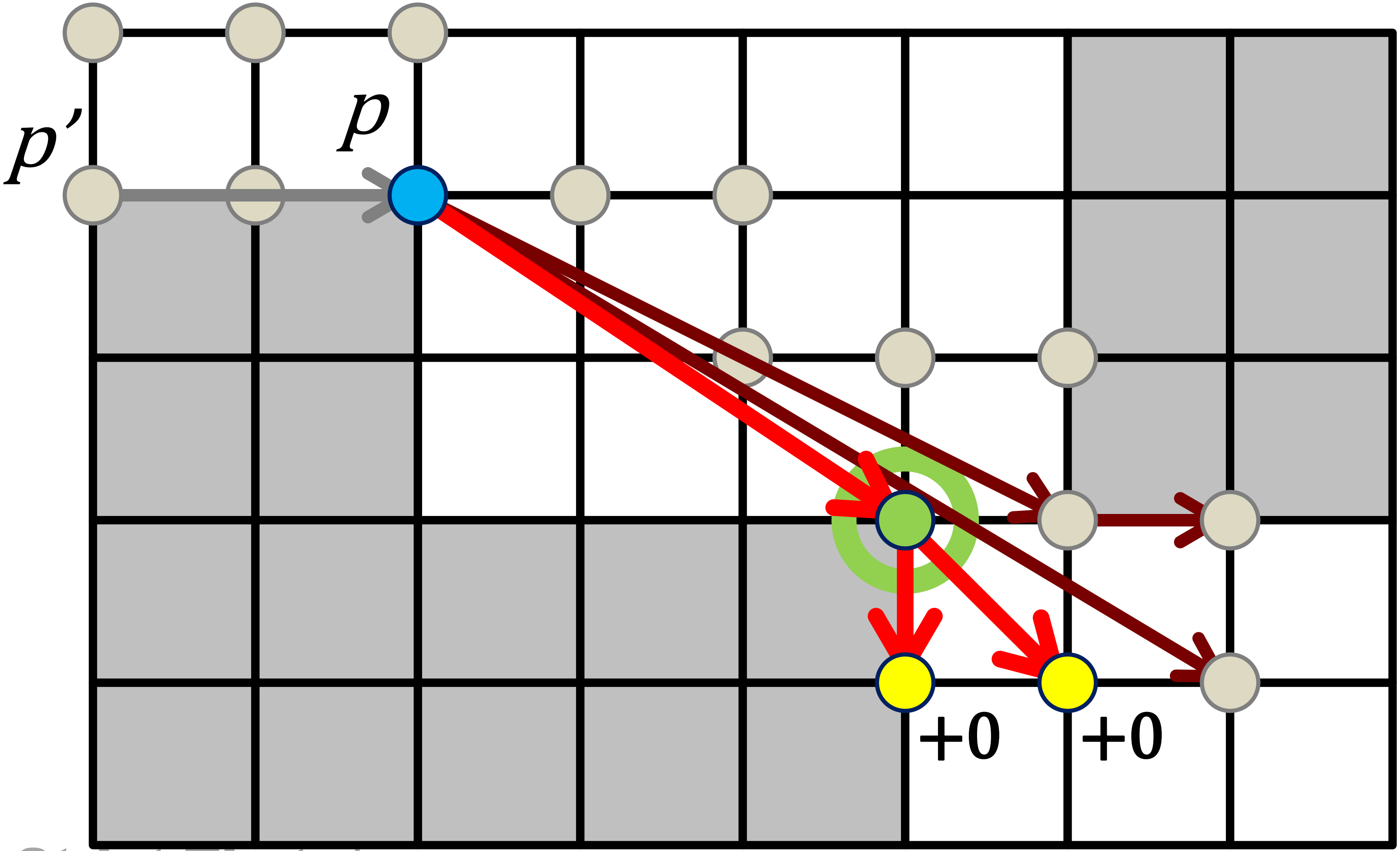
Strict Theta*



Strict Theta*



Strict Theta*

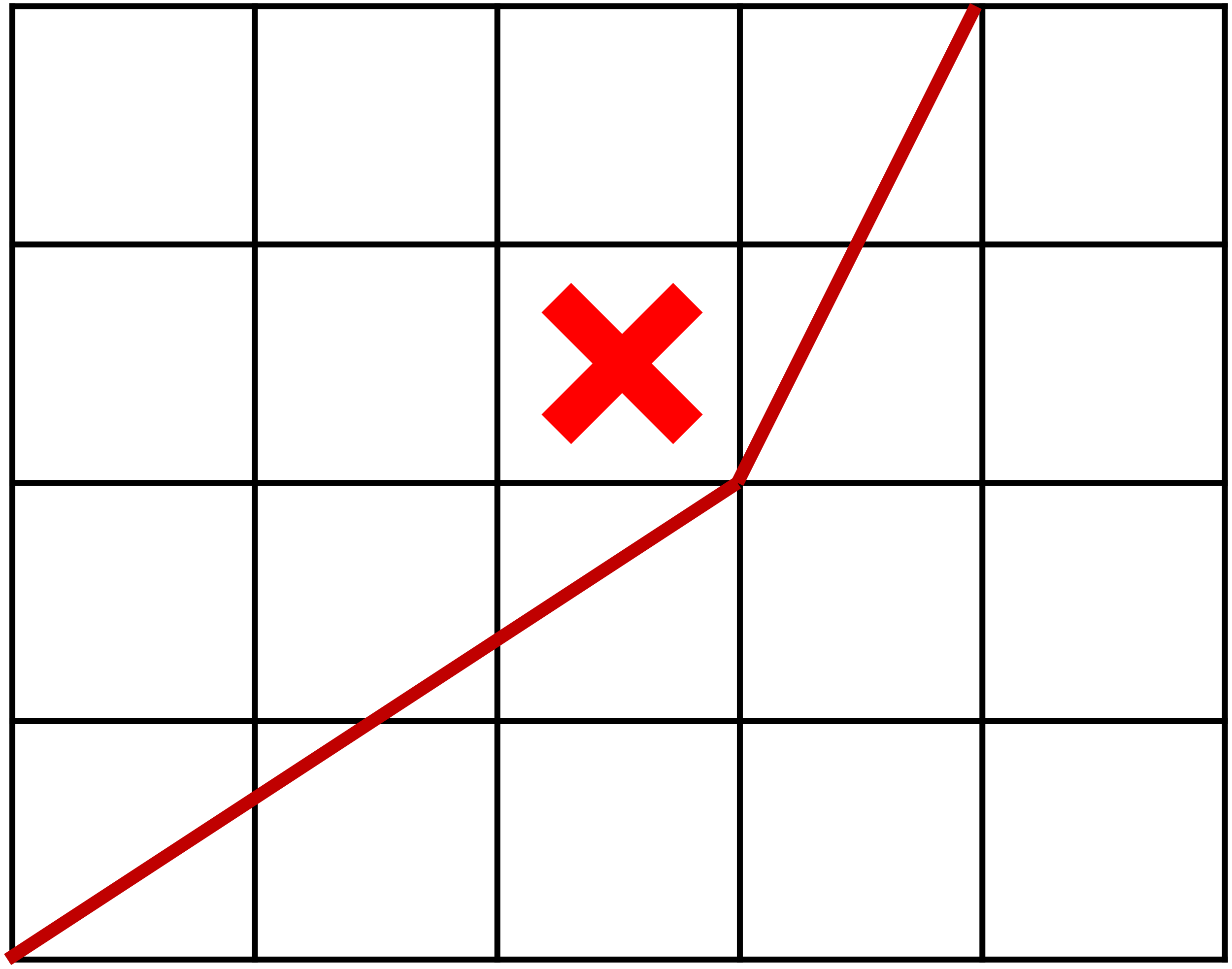


Strict Theta*

Tautness checks

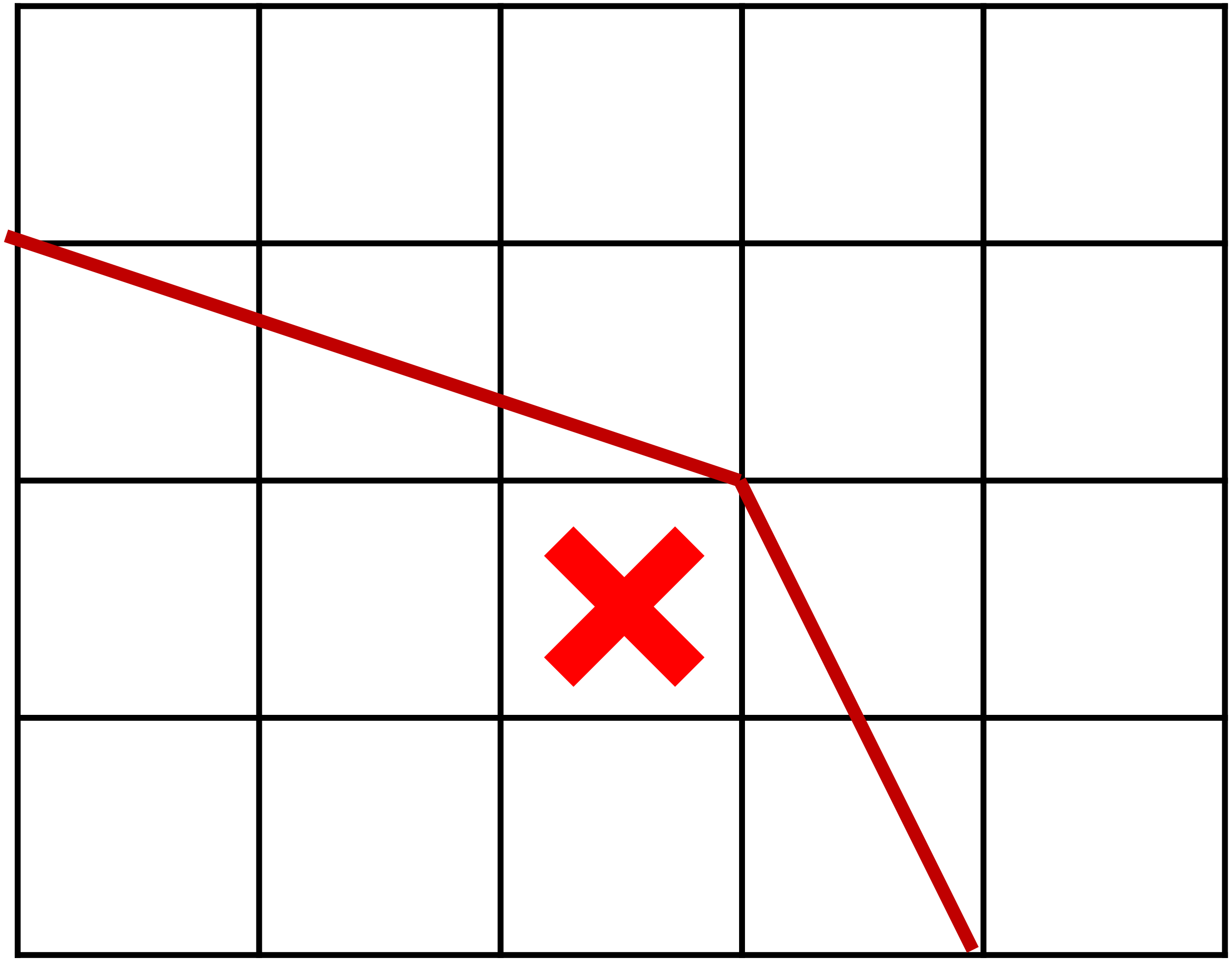
Tautness Checks

We need only check one tile.



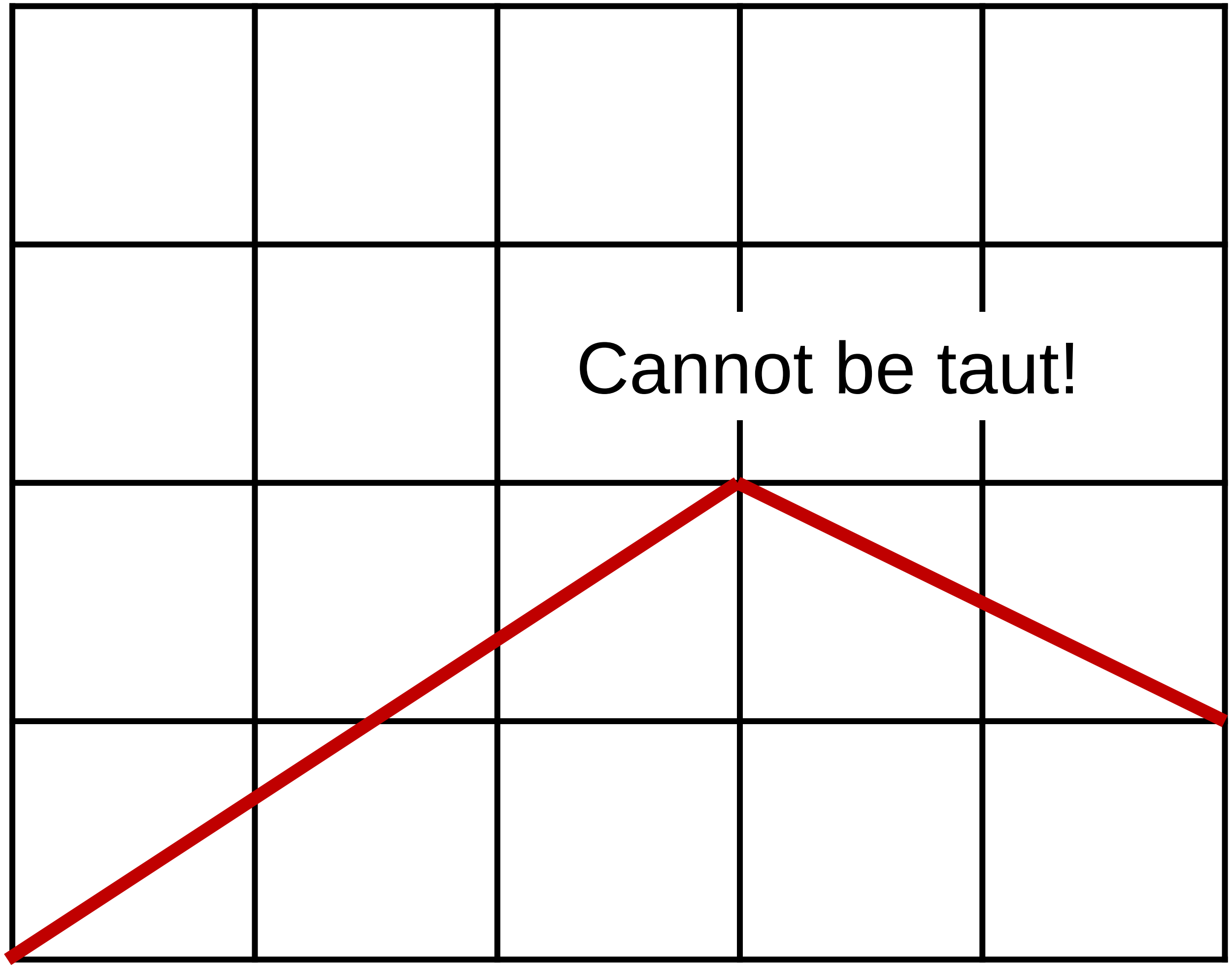
Tautness Checks

We need only check one tile.



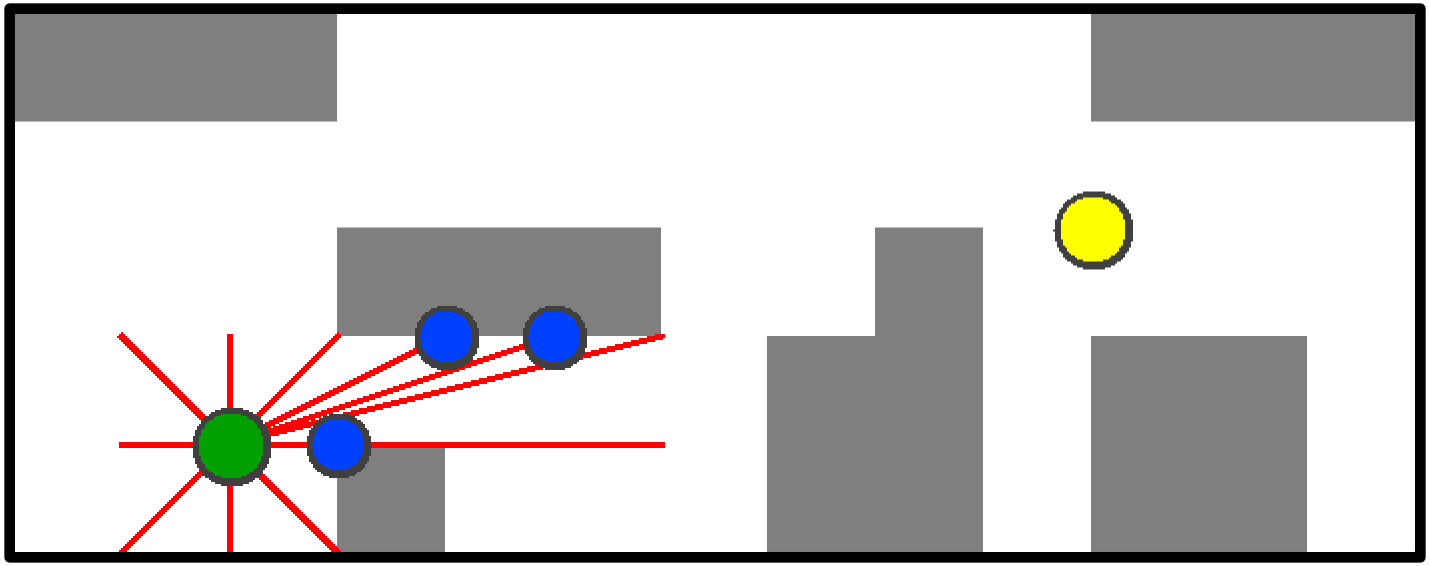
Tautness Checks

We need only check one tile.

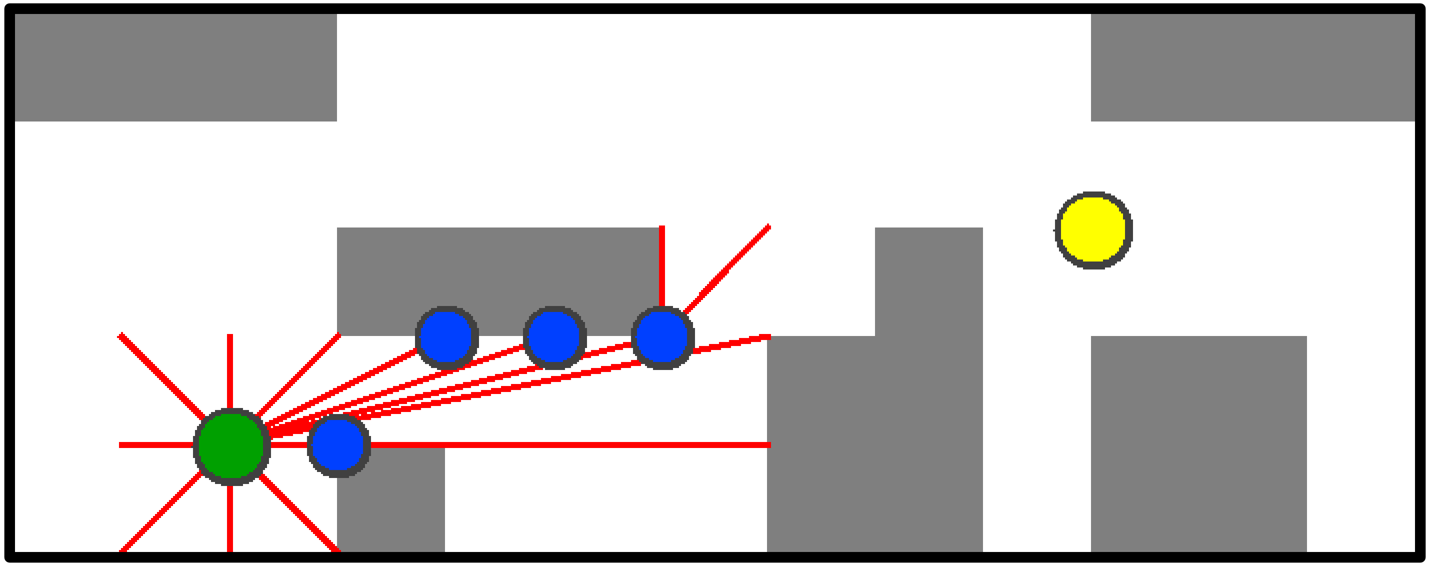


The Advantage of Taut Path Restriction

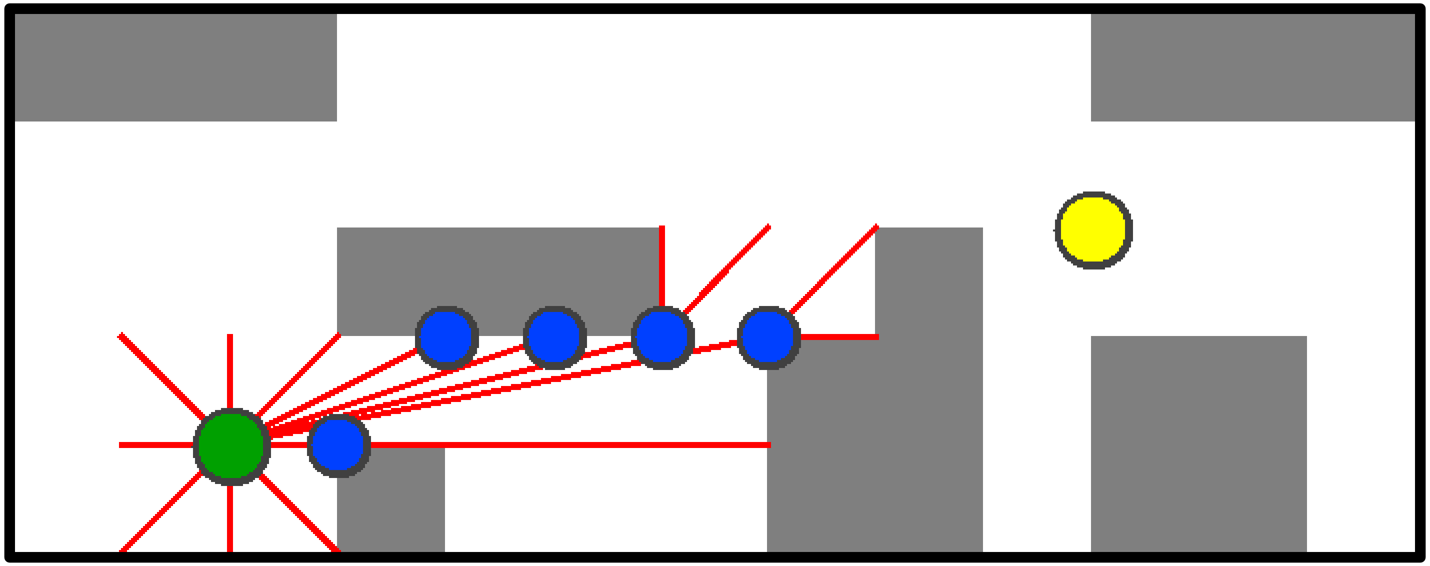
Basic Theta*



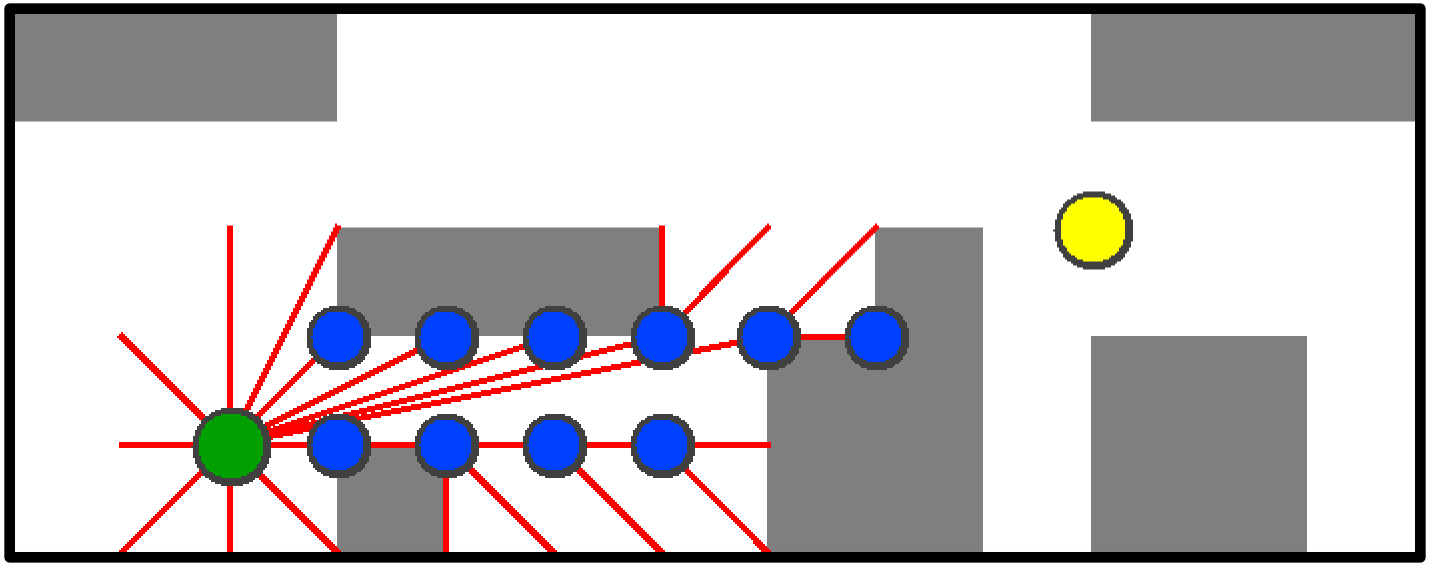
Basic Theta*



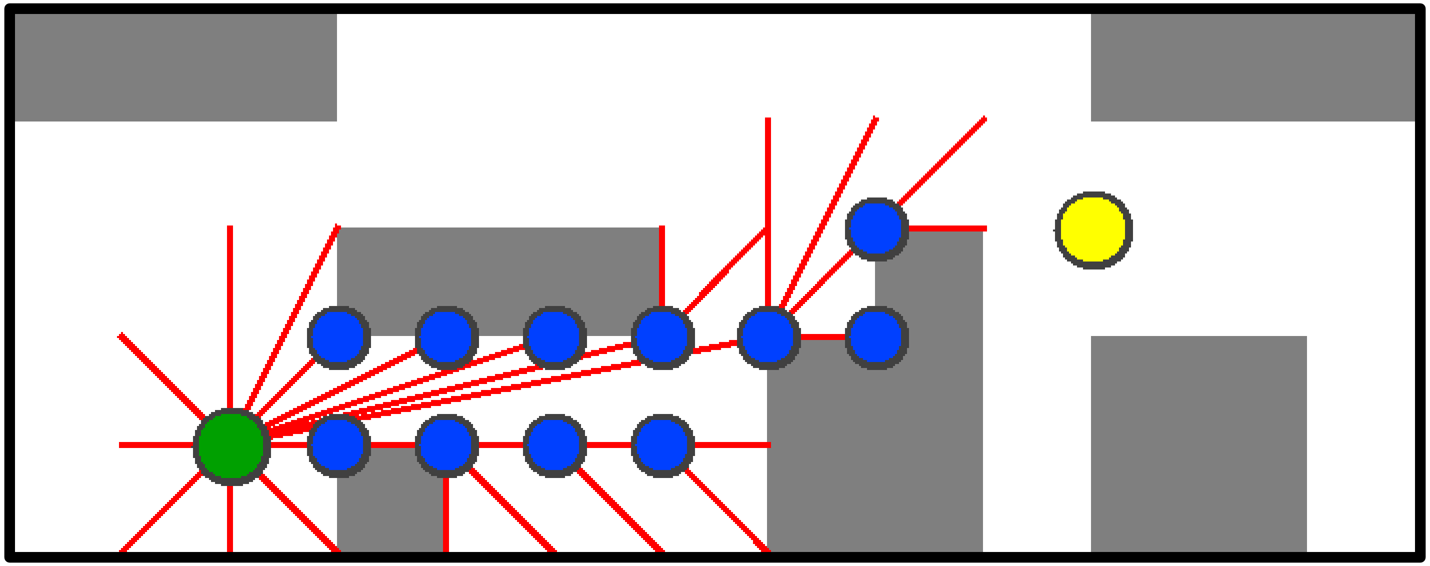
Basic Theta*



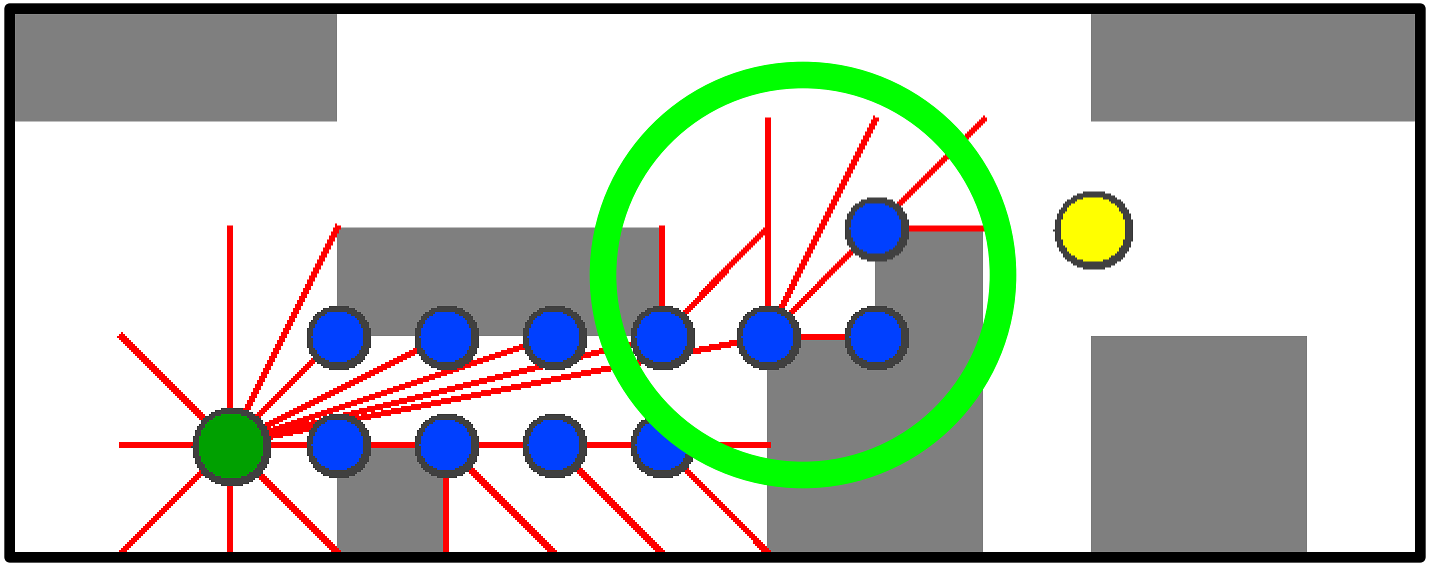
Basic Theta*



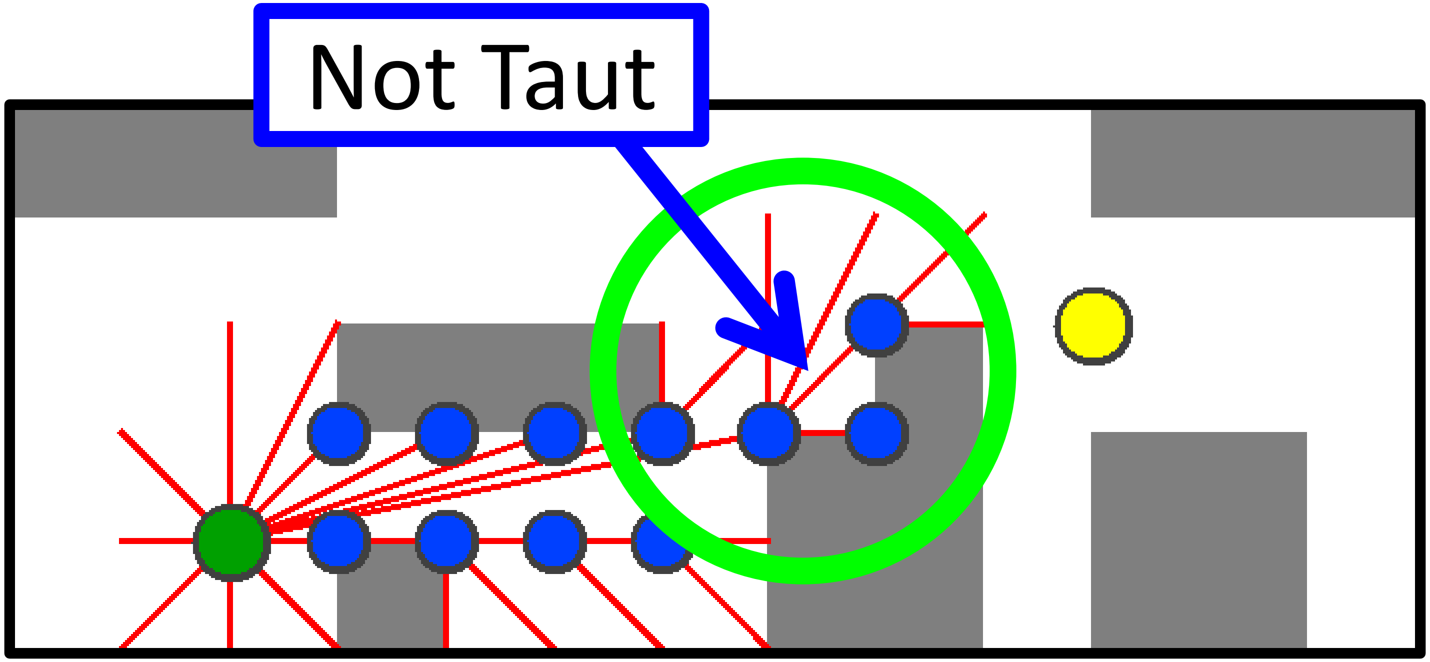
Basic Theta*



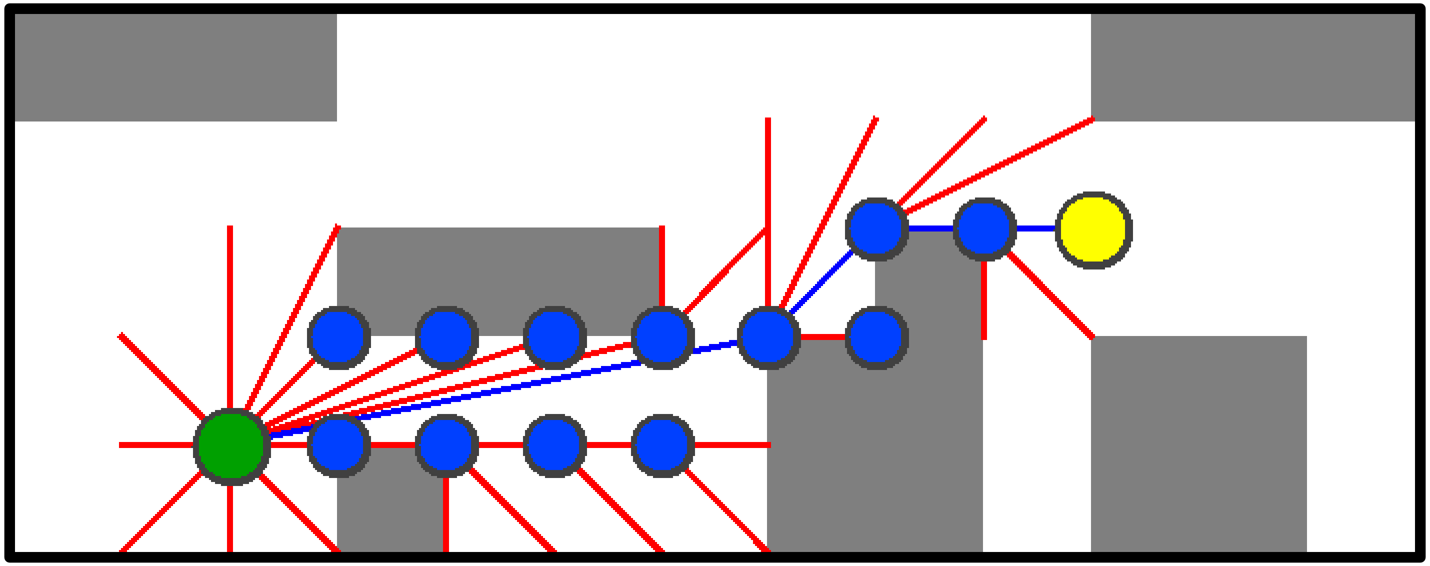
Basic Theta*



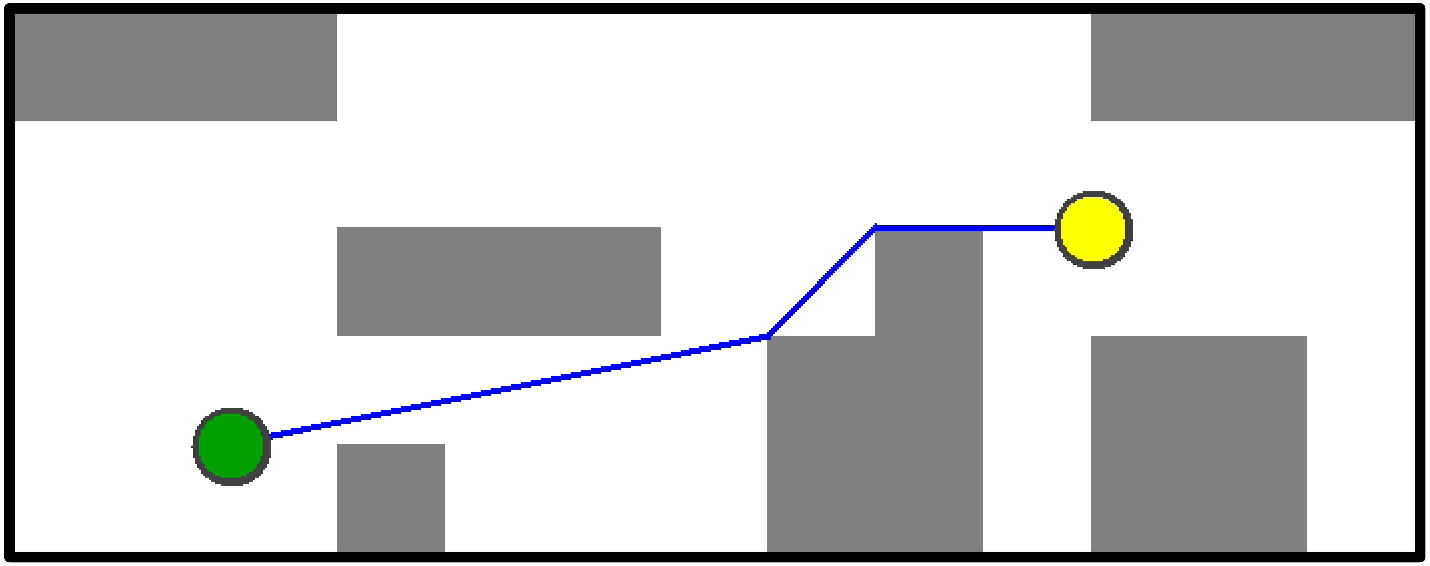
Basic Theta*



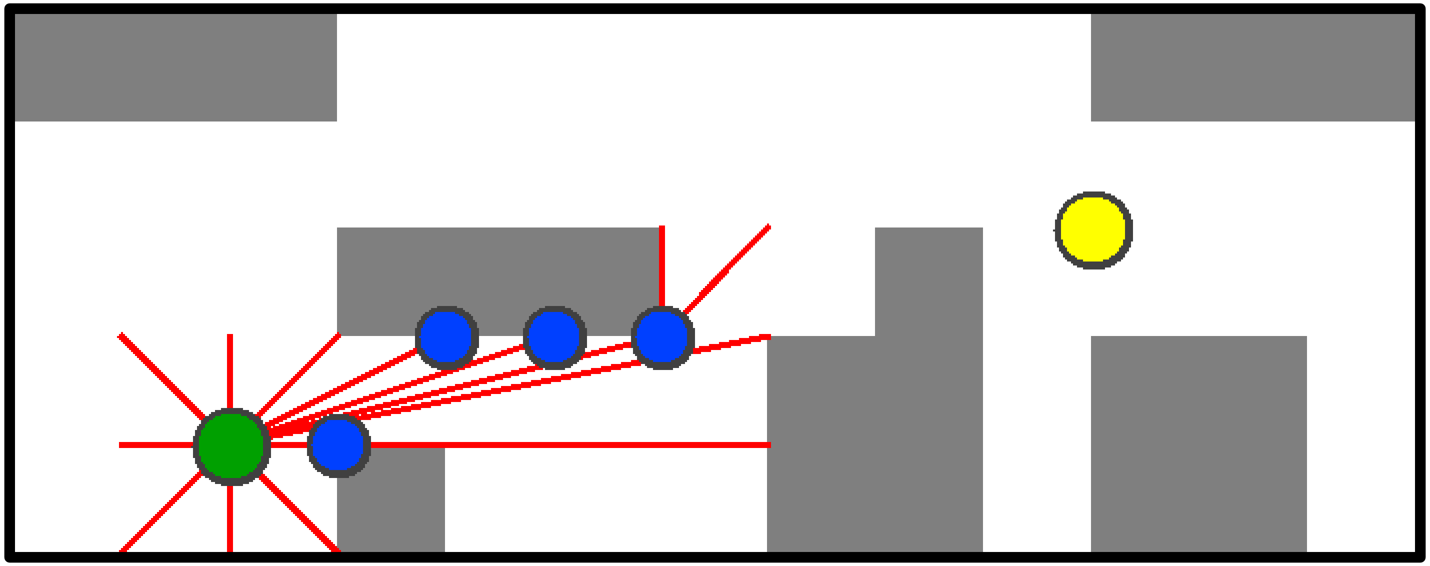
Basic Theta*



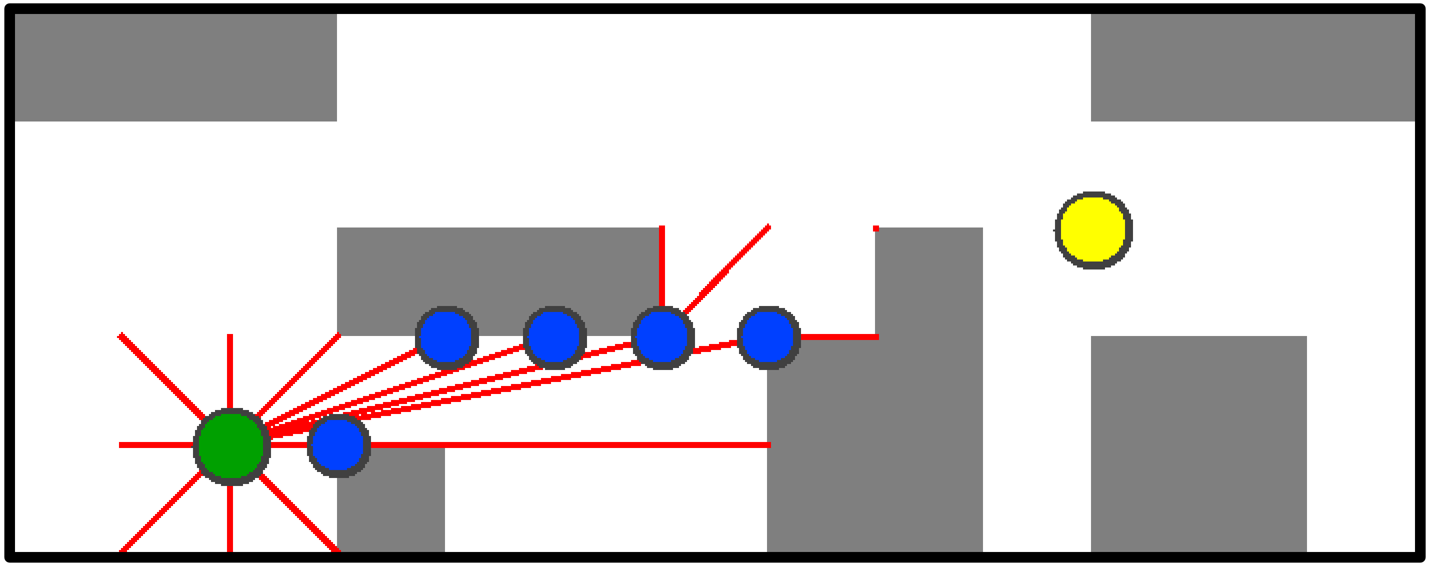
Basic Theta*



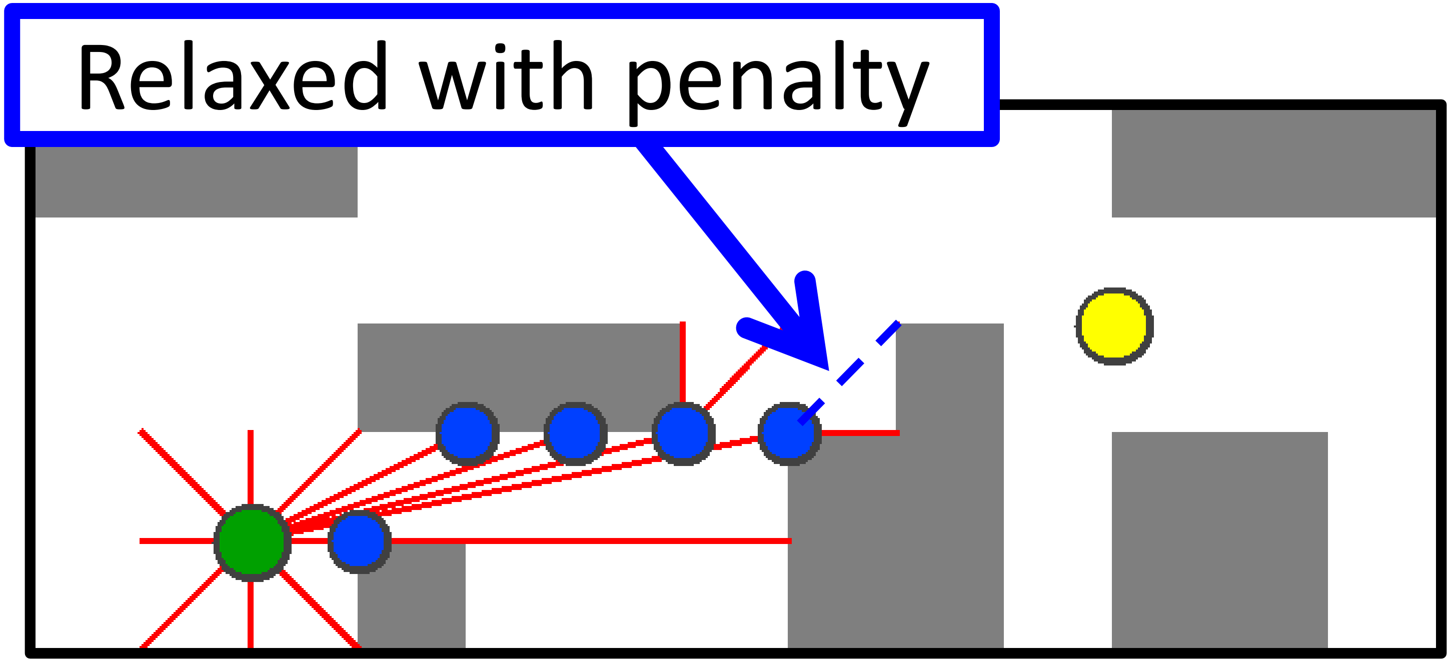
Strict Theta*



Strict Theta*

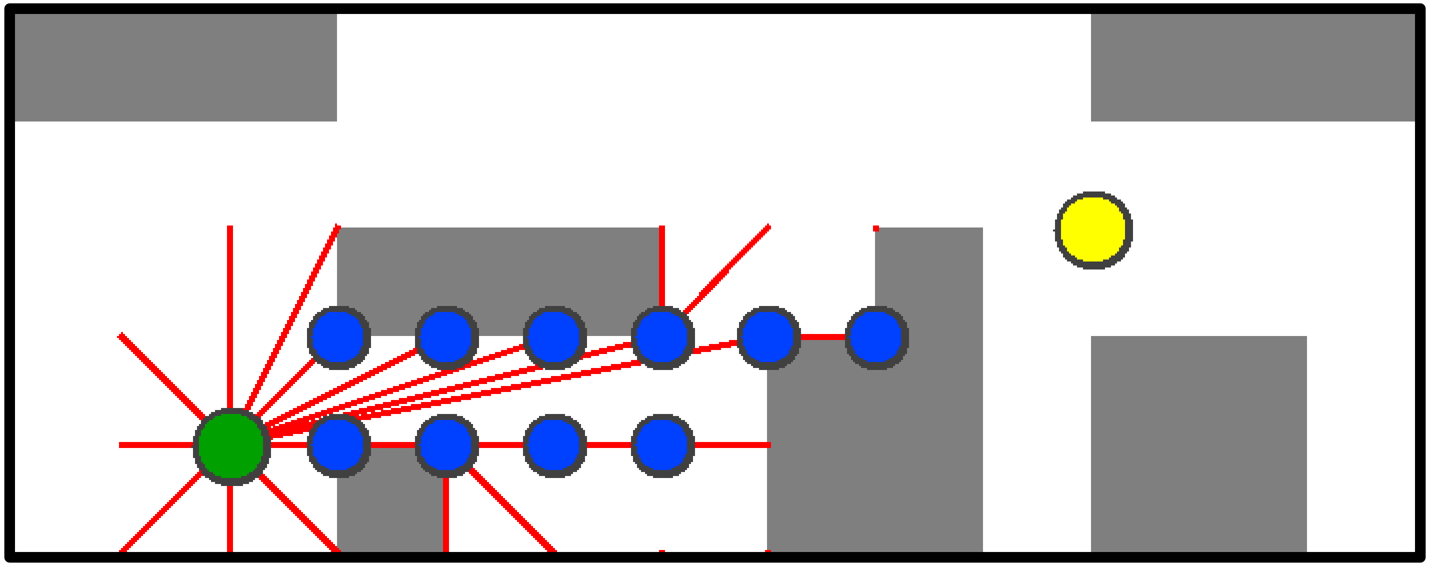


Strict Theta*

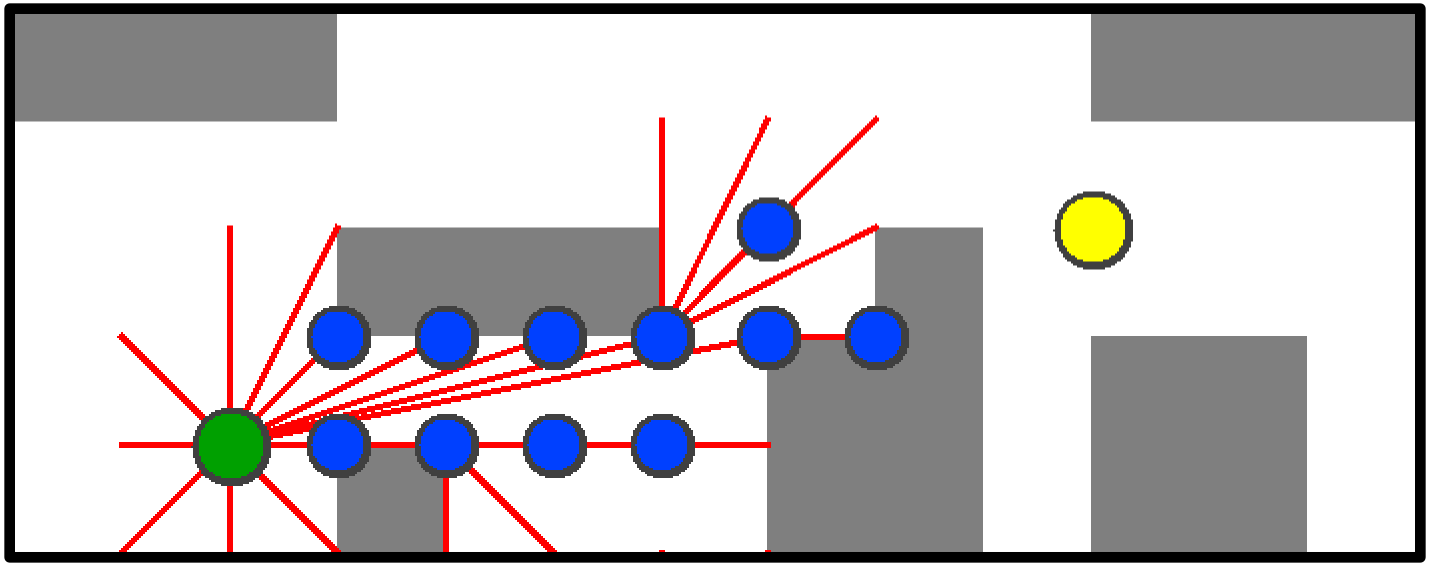


Relaxed with penalty

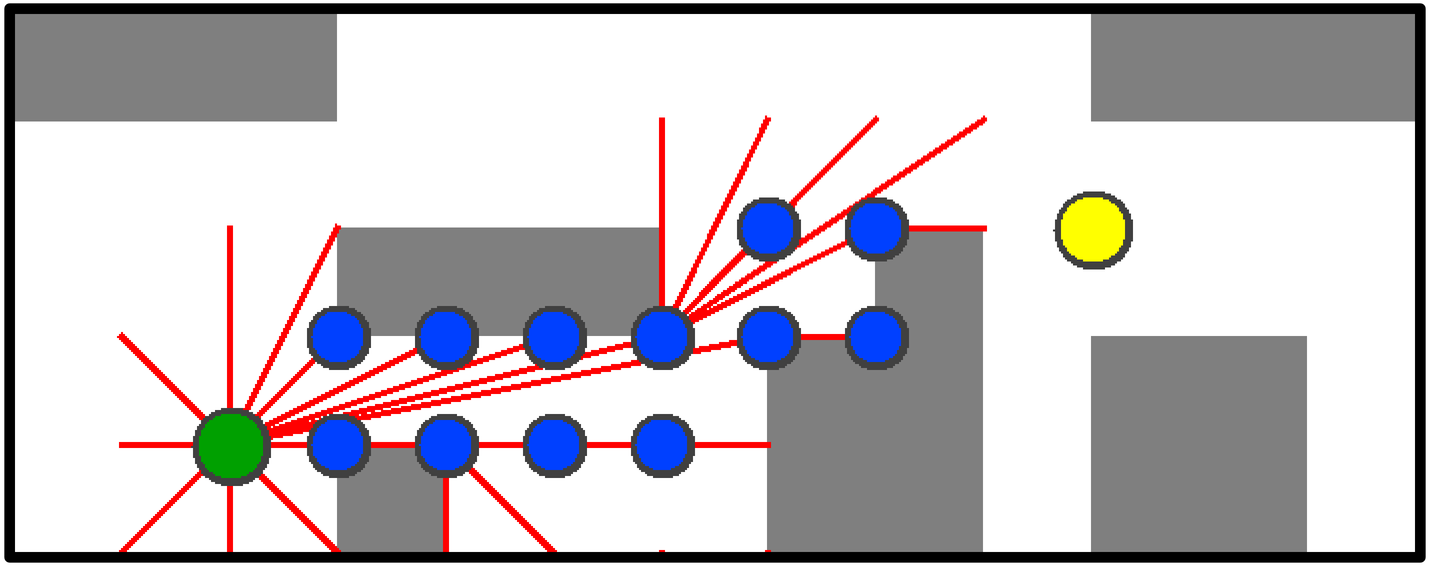
Strict Theta*



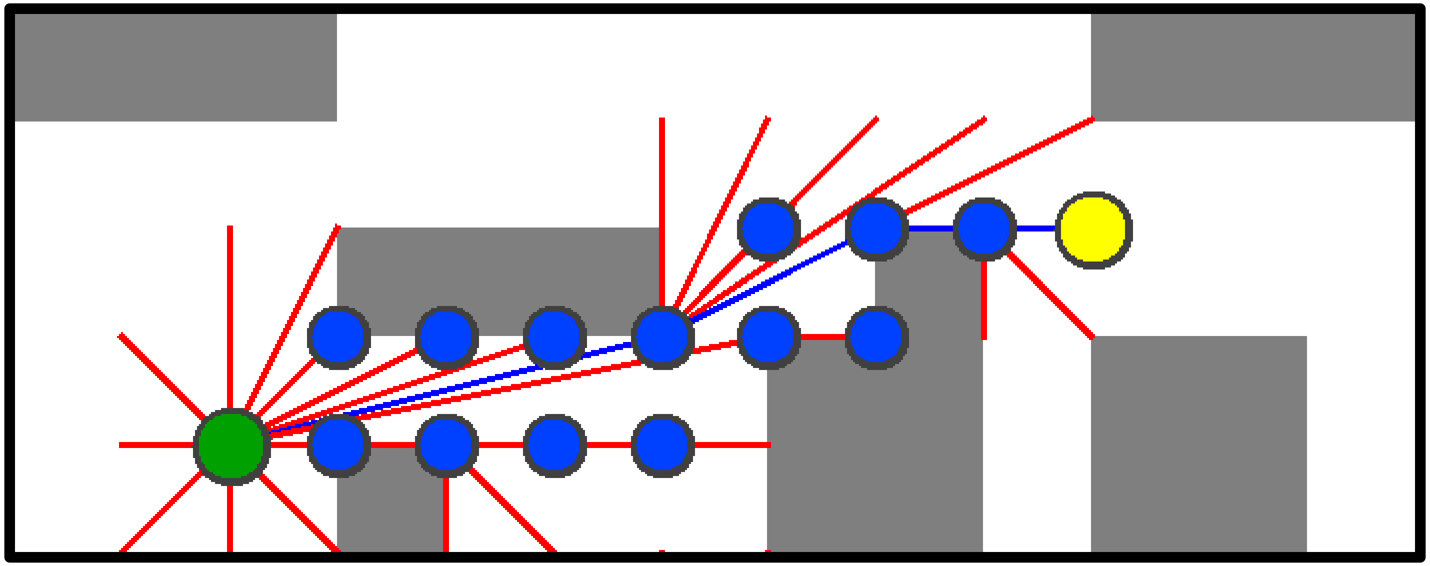
Strict Theta*



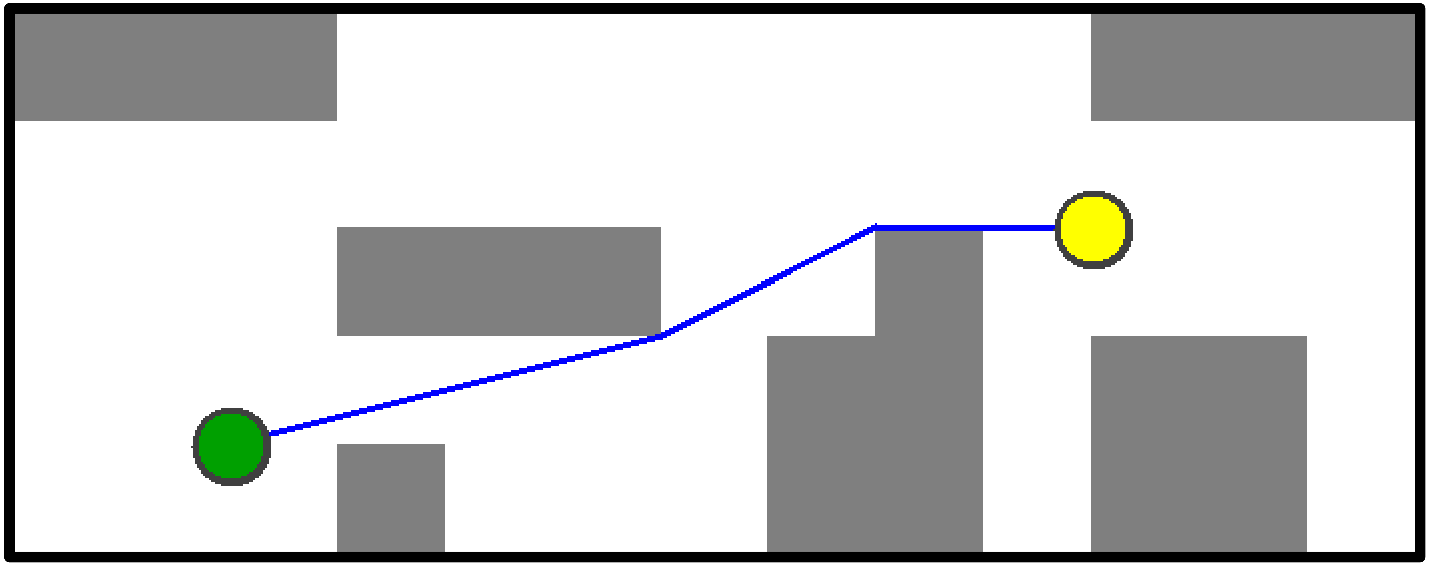
Strict Theta*



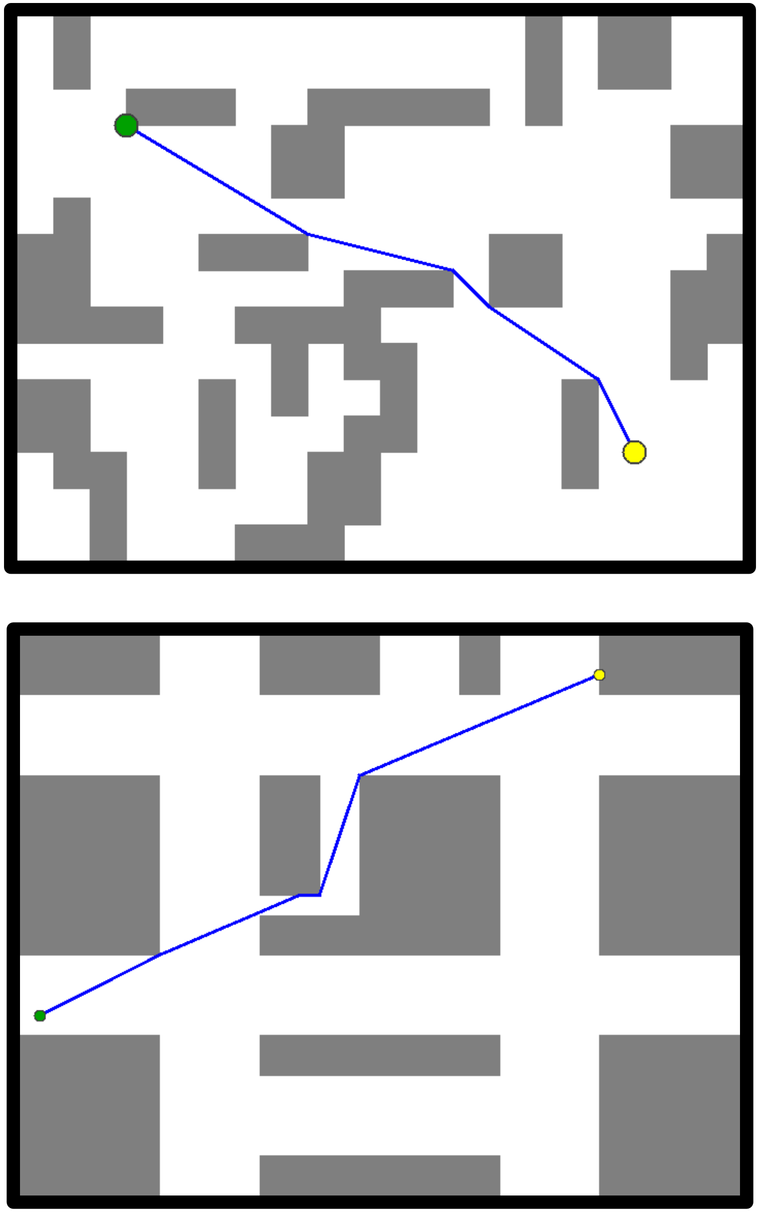
Strict Theta*



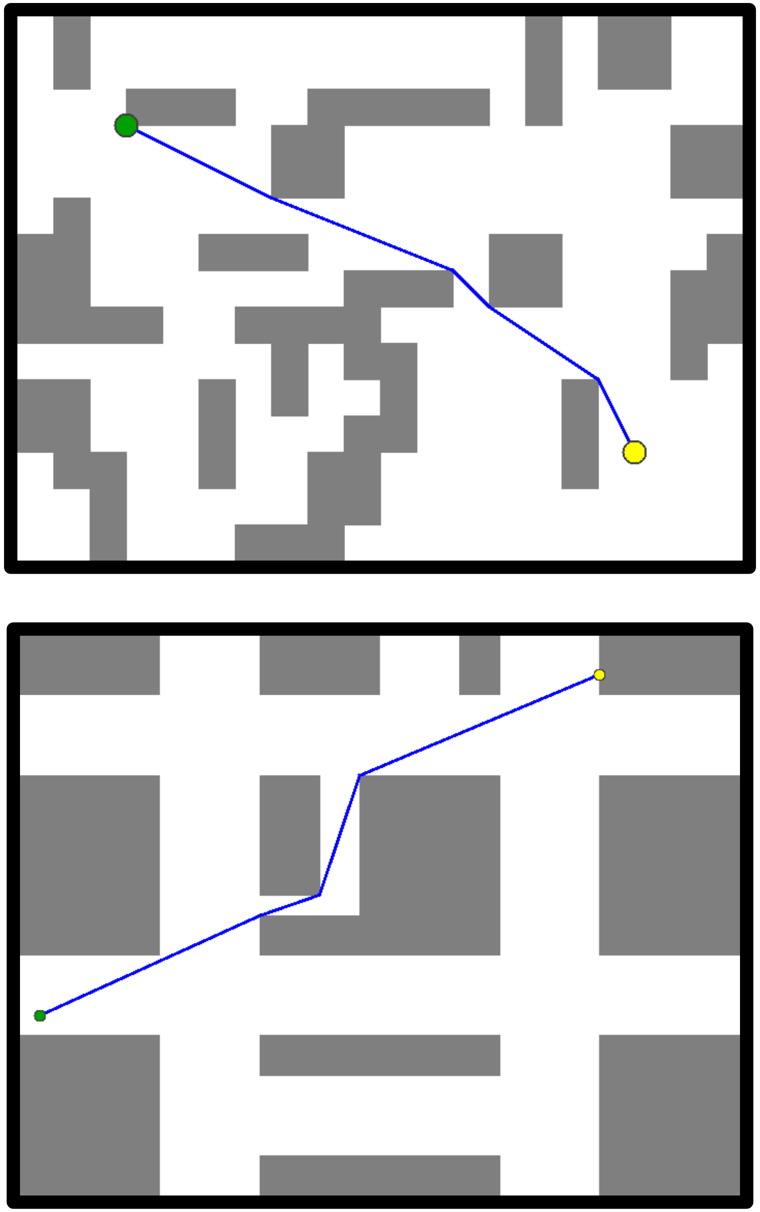
Strict Theta*

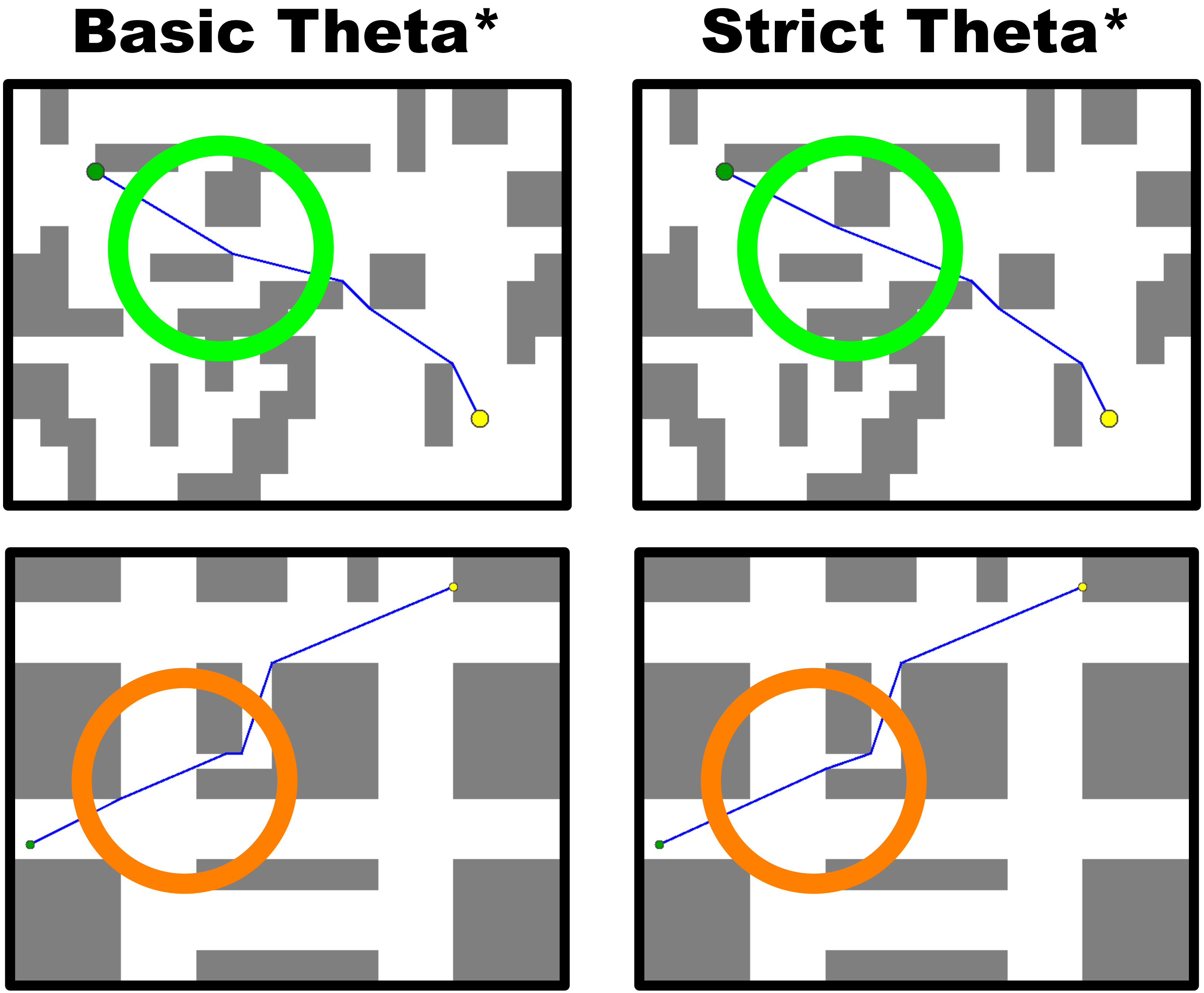


Basic Theta*



Strict Theta*





Basic Theta*

Strict Theta*

Basic Theta*

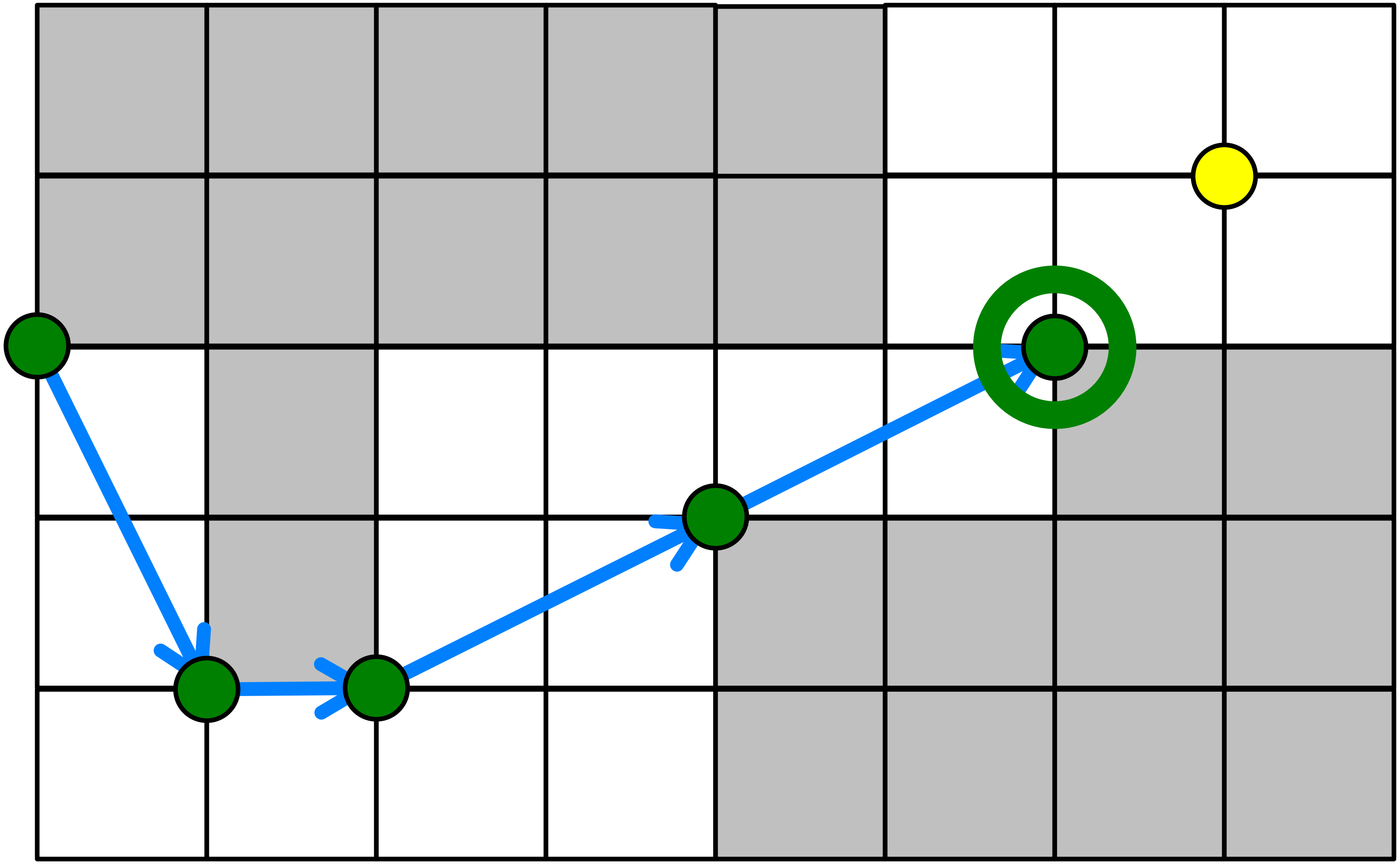
Strict Theta*

Basic Theta*

Strict Theta*

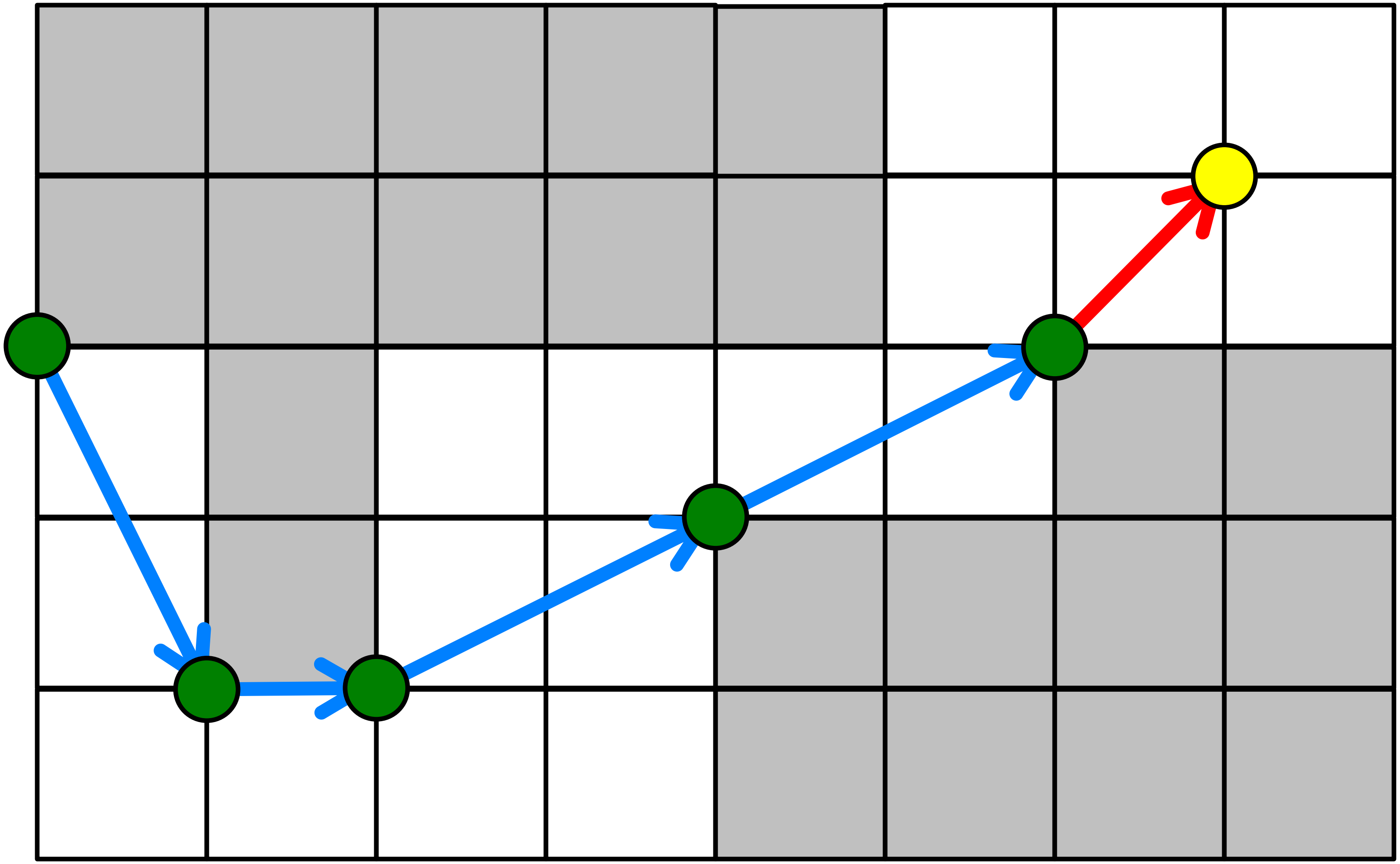
Recursive Strict Theta*

Recursive Strict Theta*

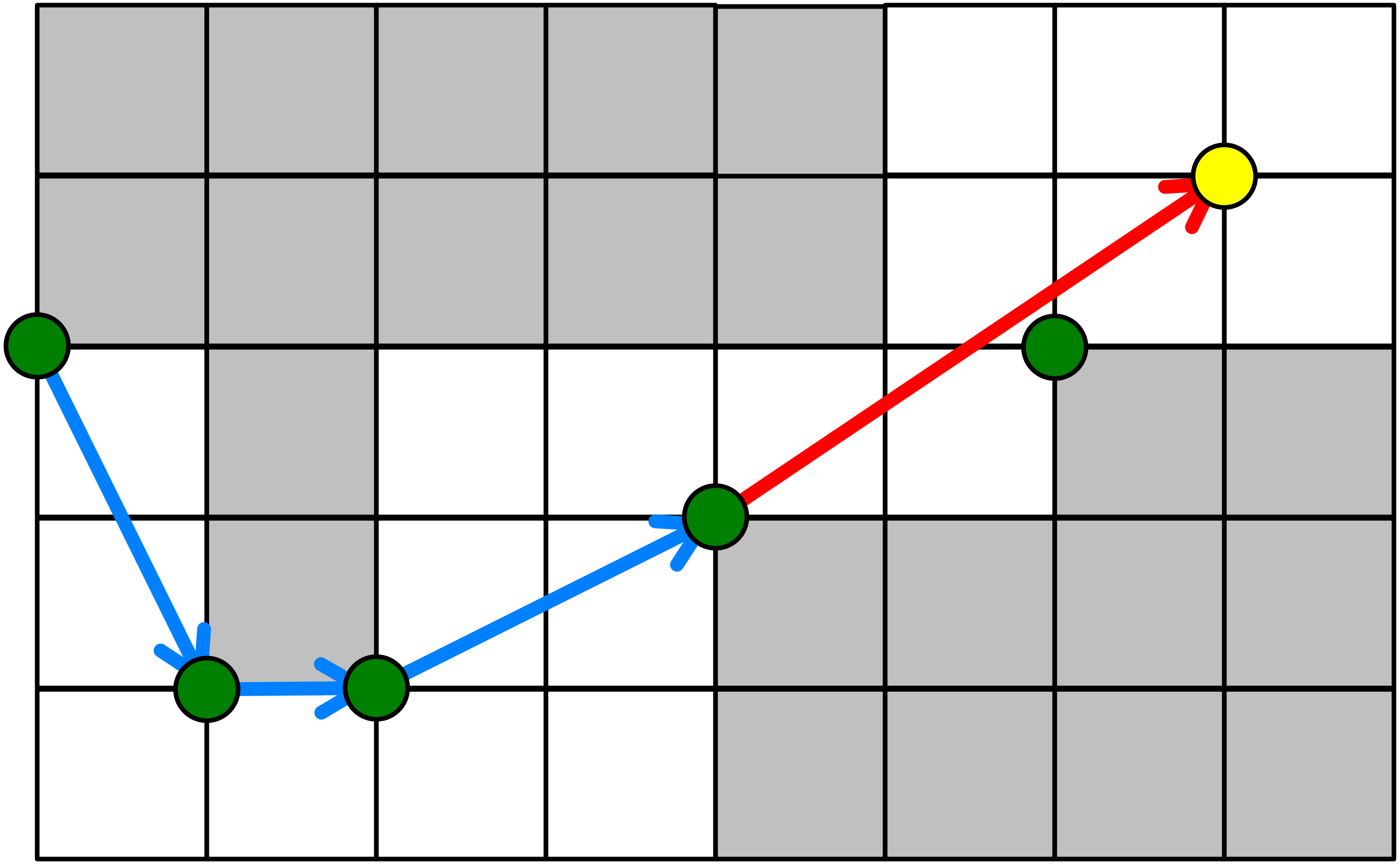


Recursive Strict Theta*

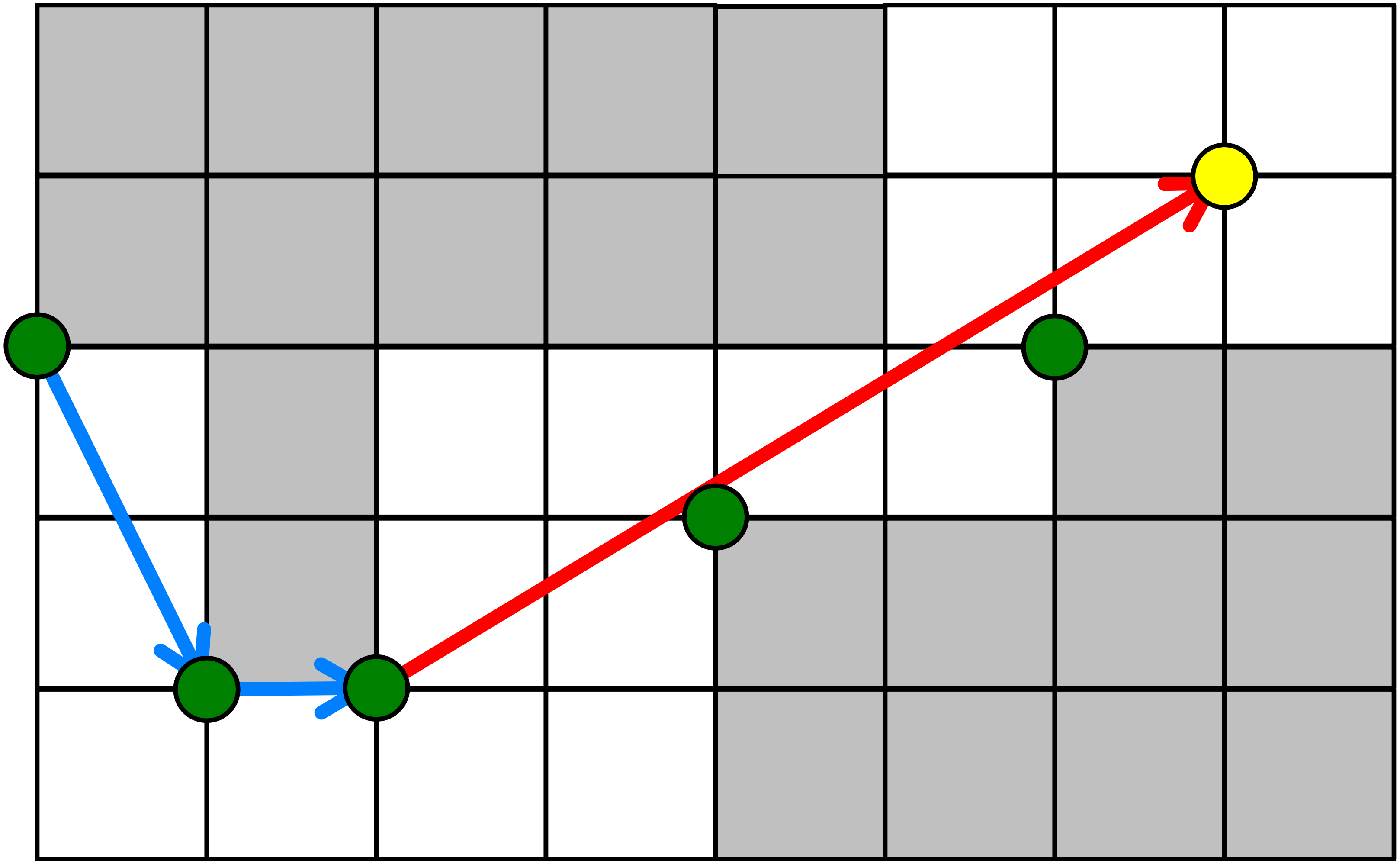
Recursive Strict Theta*



Recursive Strict Theta*

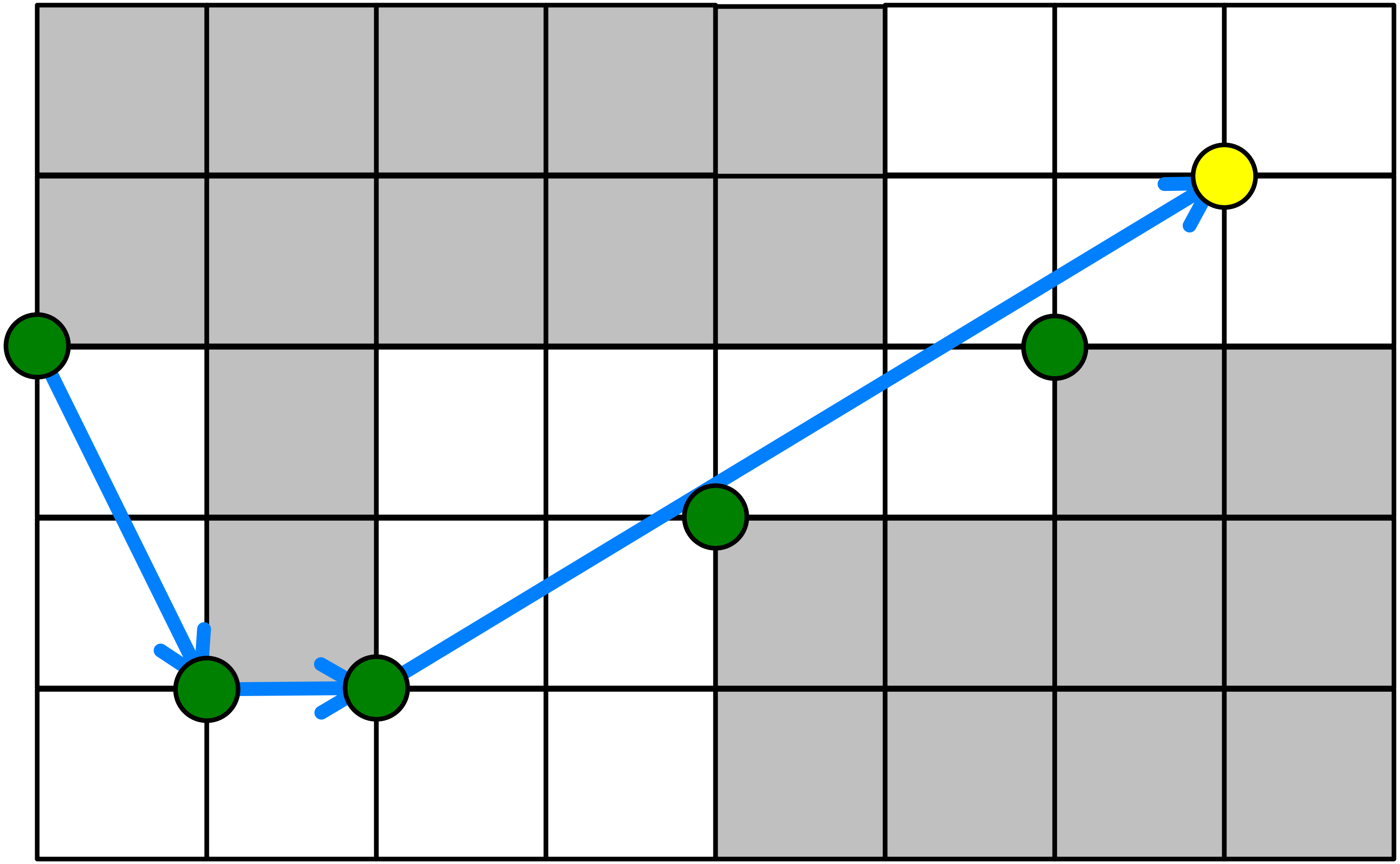


Recursive Strict Theta*



Recursive Strict Theta*

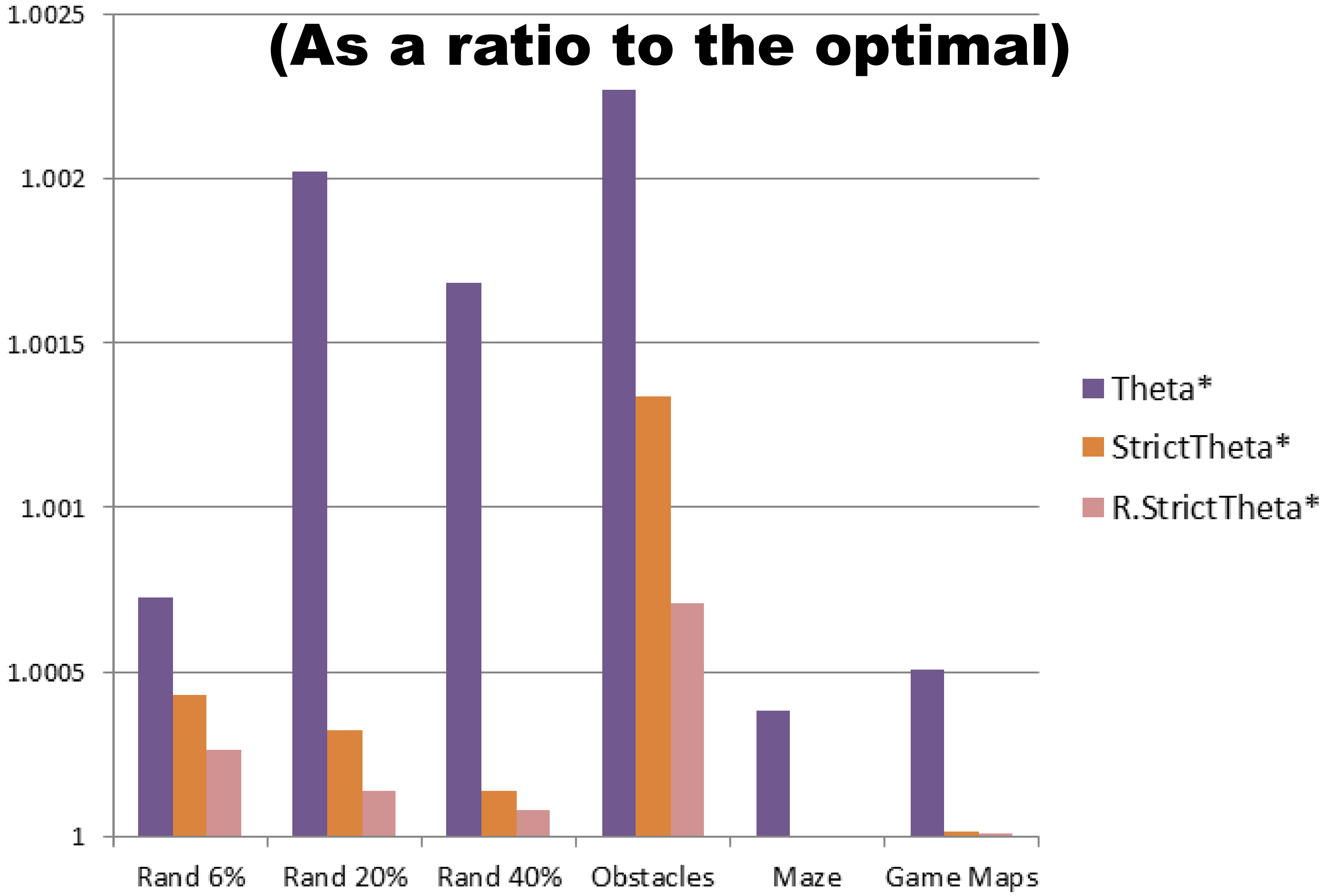
Recursive Strict Theta*



Path Length
(As a ratio to the optimal)

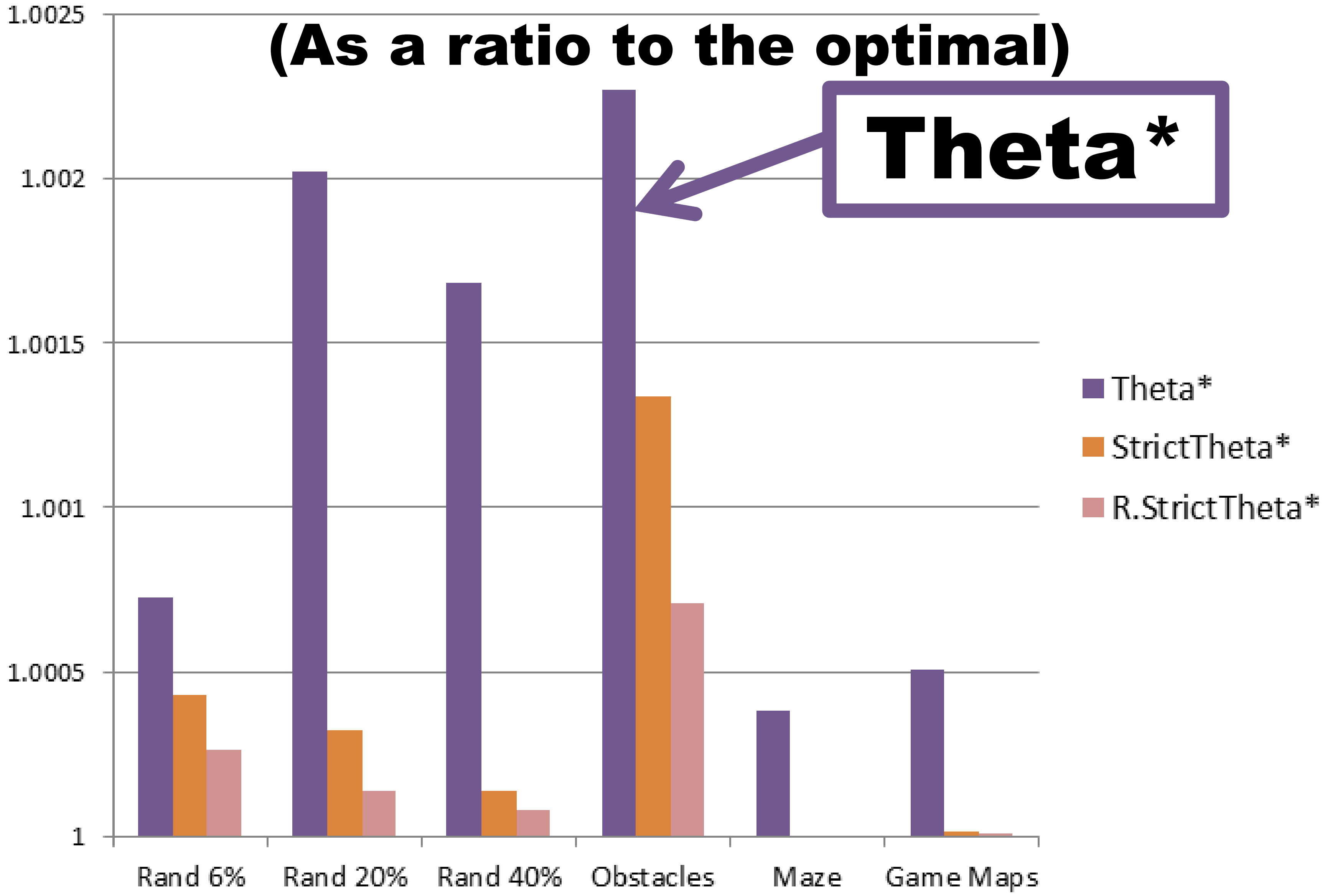
Path Length

(As a ratio to the optimal)



Path Length

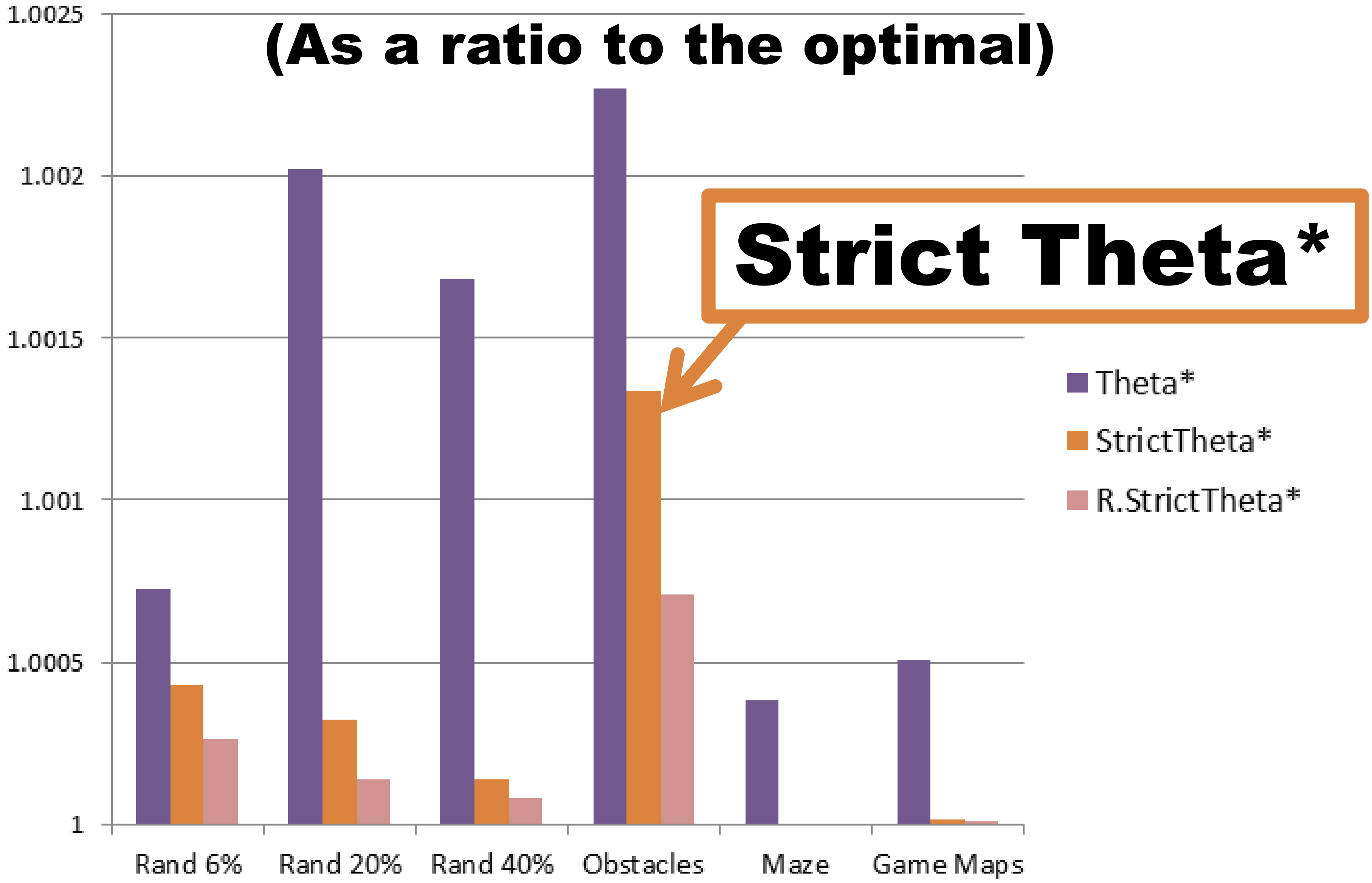
(As a ratio to the optimal)



Theta*

Path Length

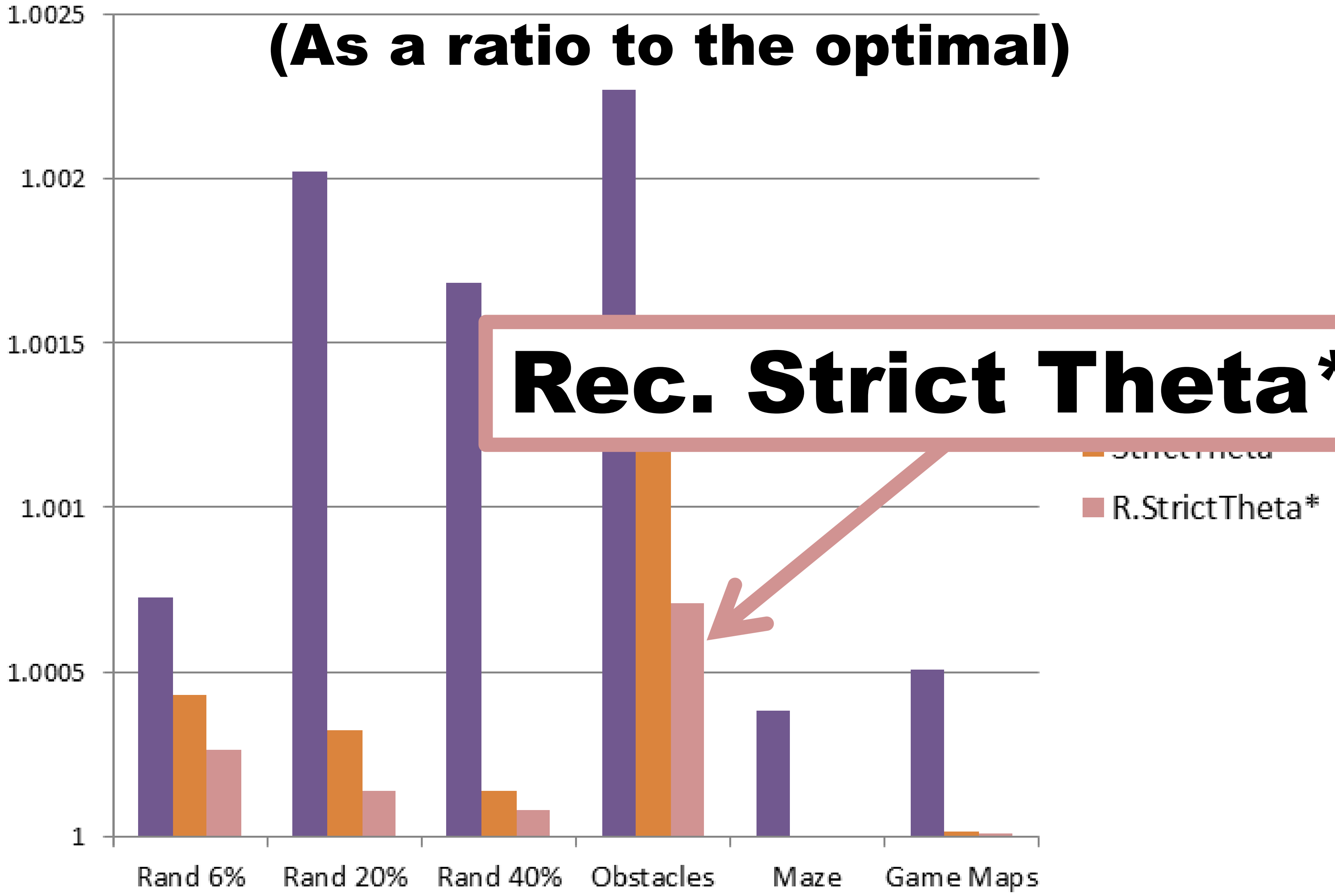
(As a ratio to the optimal)



Strict Theta*

Path Length

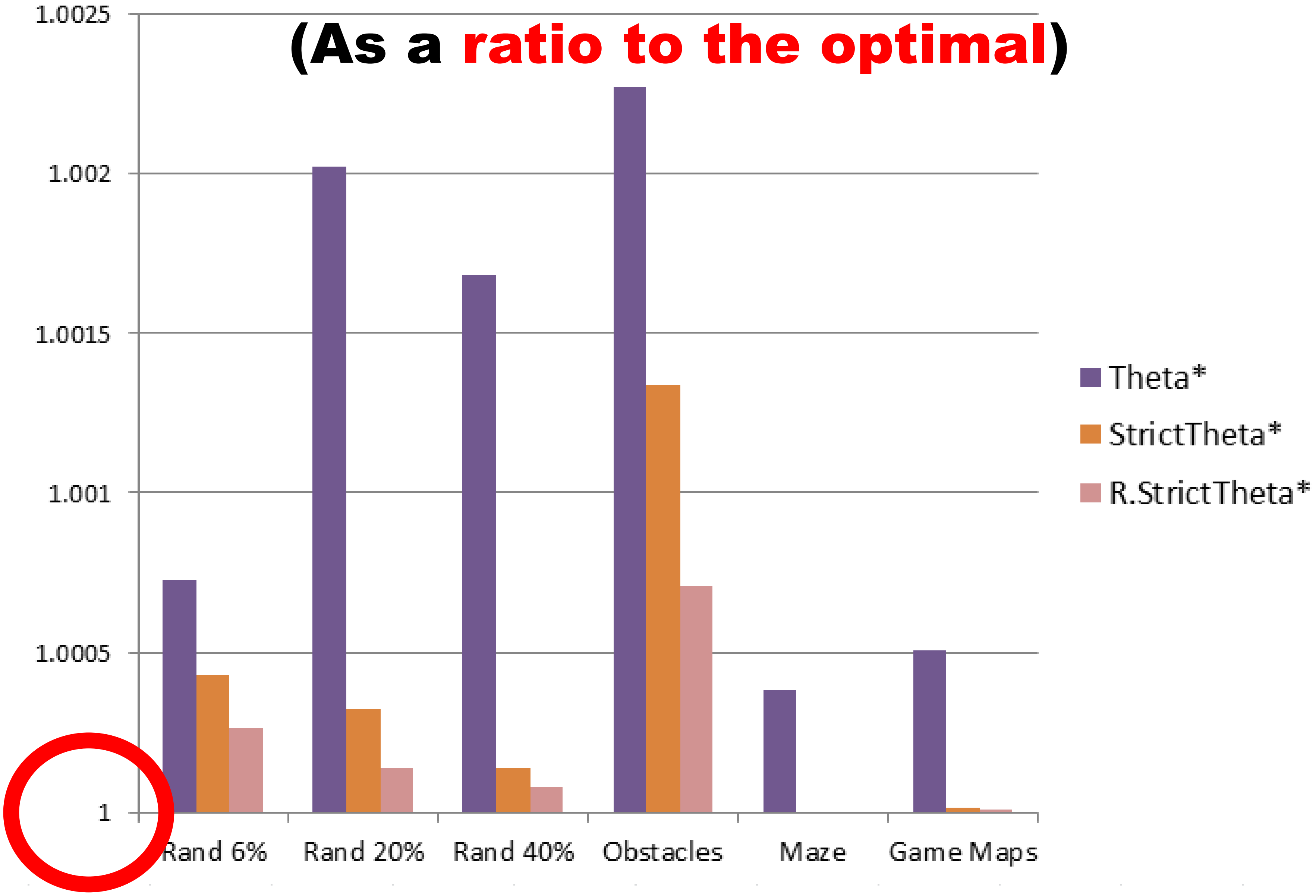
(As a ratio to the optimal)



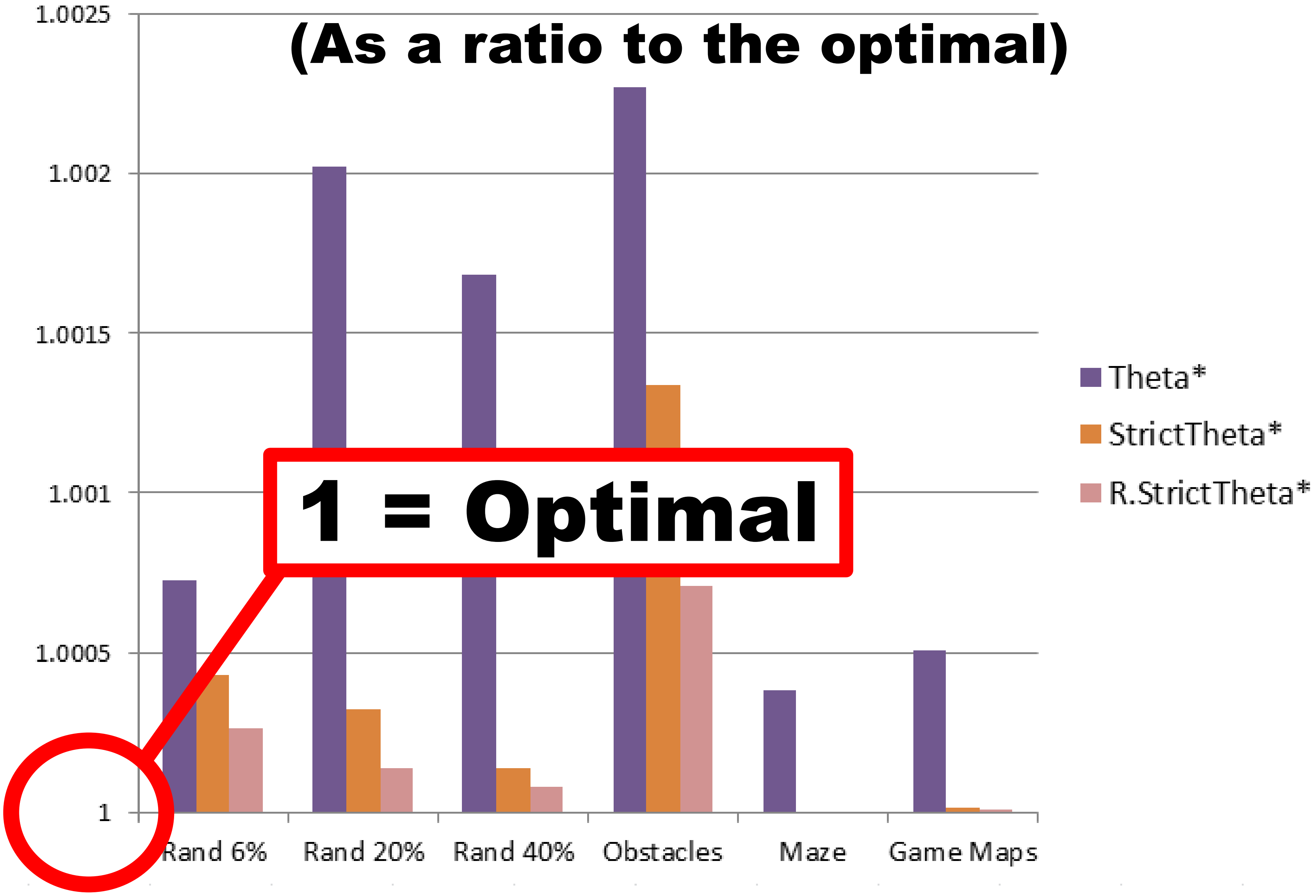
Rec. Strict Theta*

Path Length

(As a **ratio to the optimal**)



Path Length

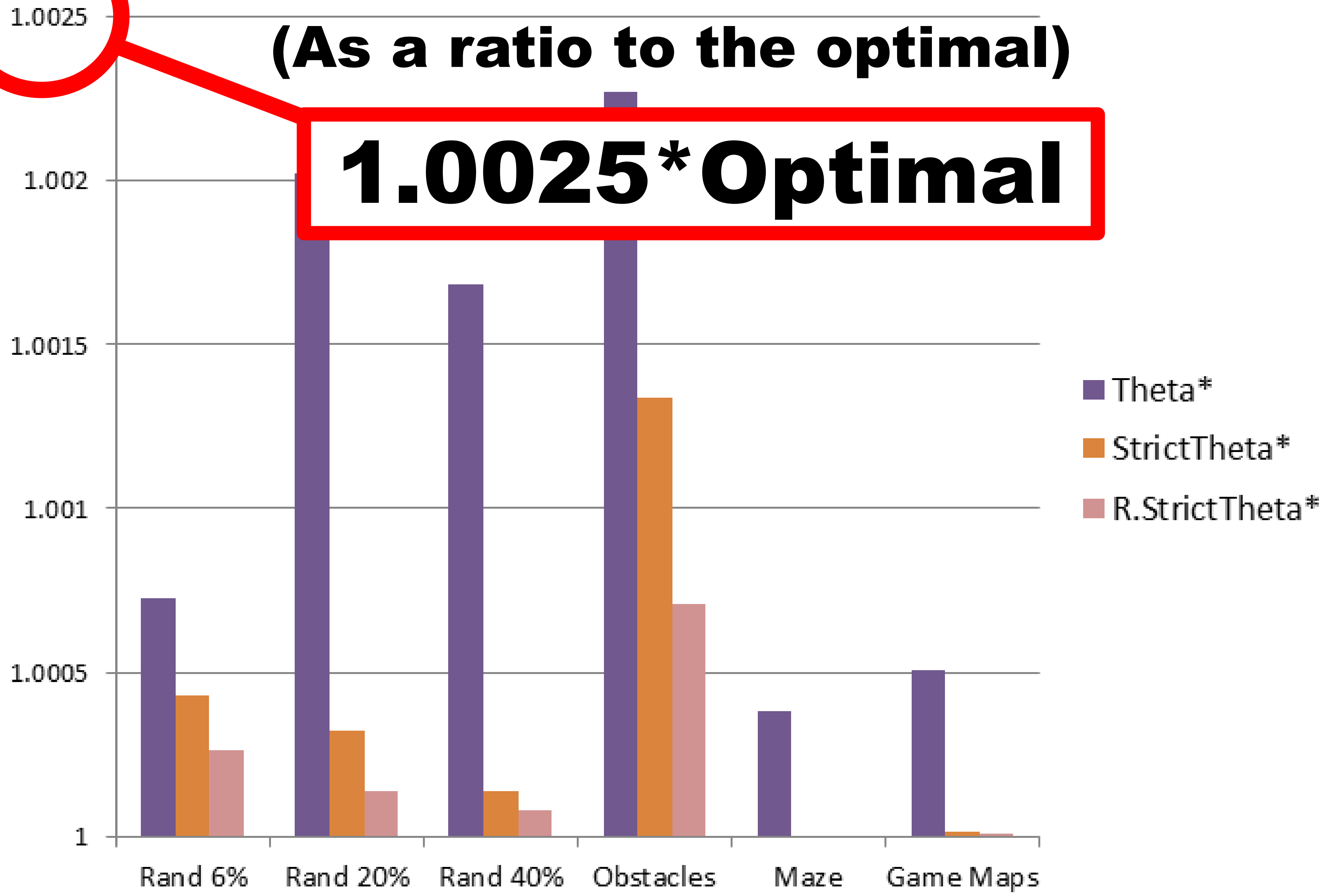


(As a ratio to the optimal)

1 = Optimal

Path Length

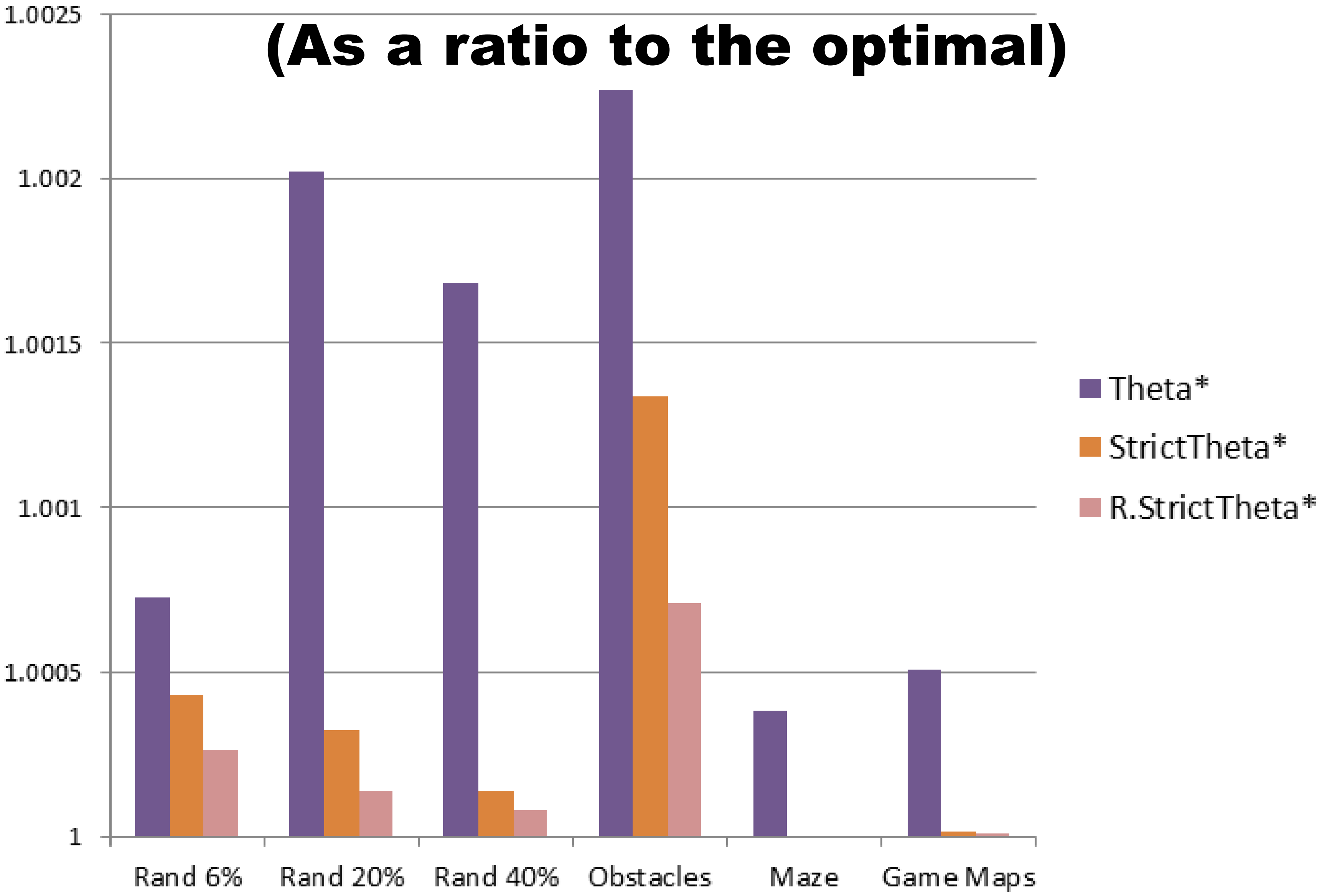
(As a ratio to the optimal)



1.0025*Optimal

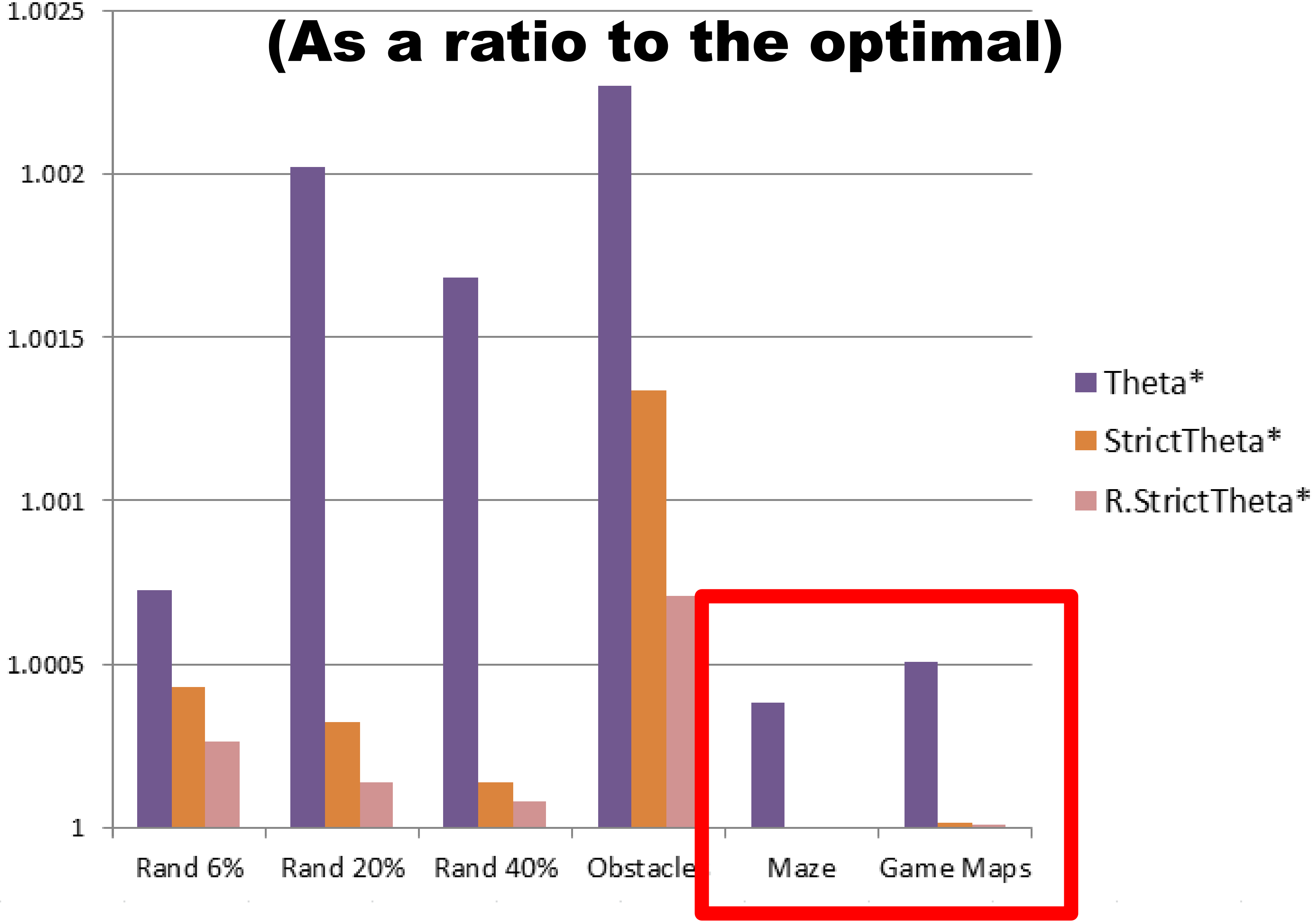
Path Length

(As a ratio to the optimal)



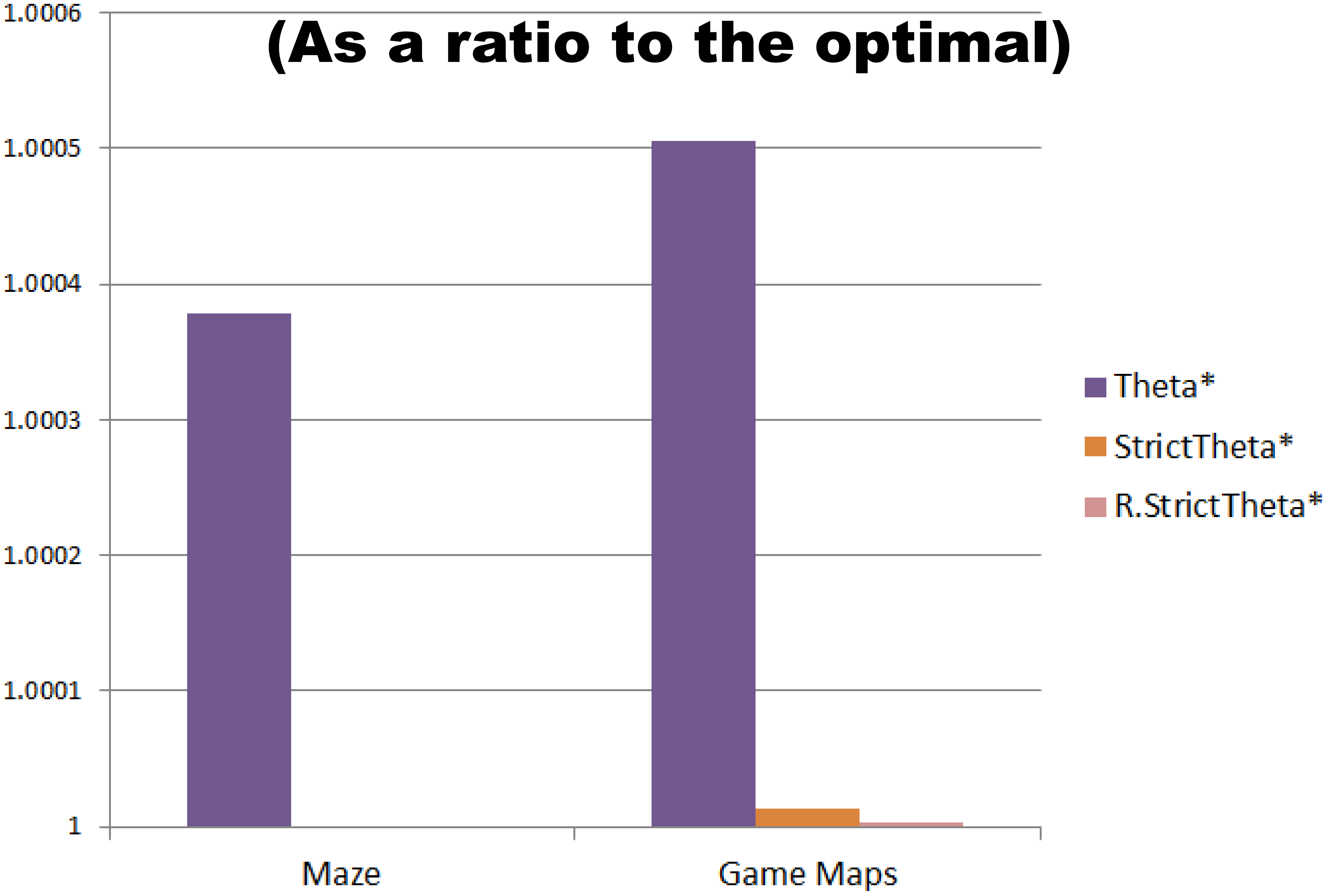
Path Length

(As a ratio to the optimal)



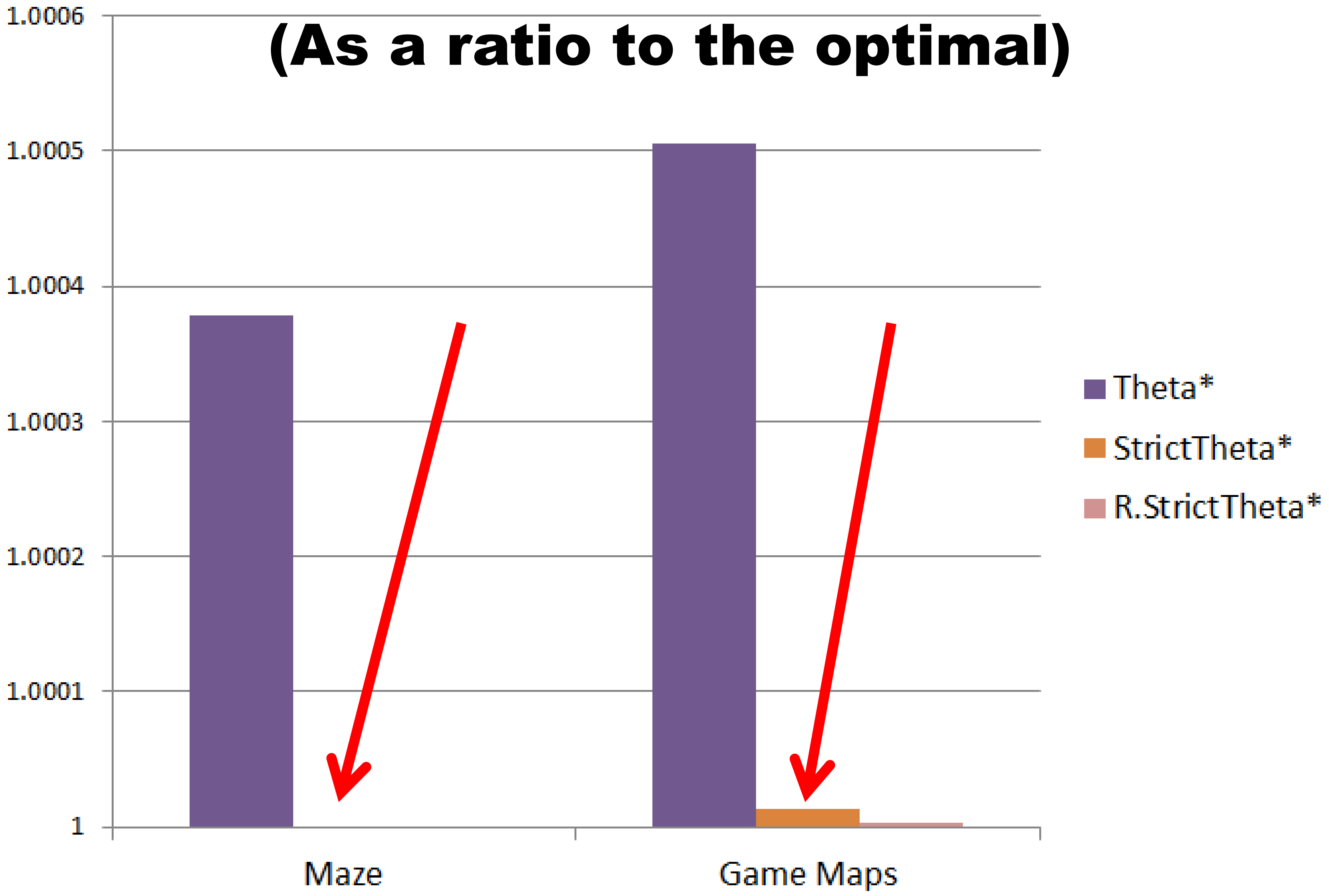
Path Length

(As a ratio to the optimal)



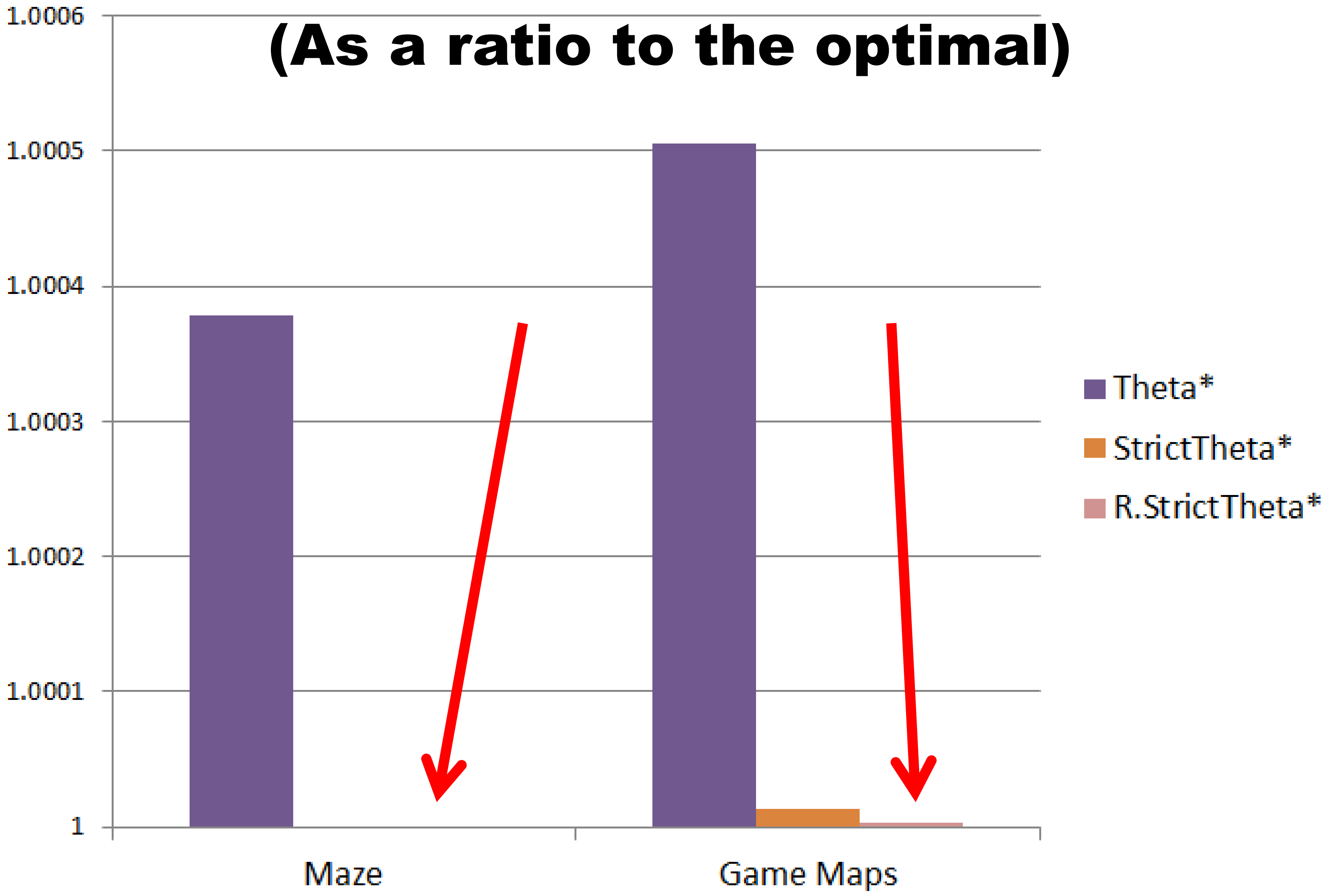
Path Length

(As a ratio to the optimal)



Path Length

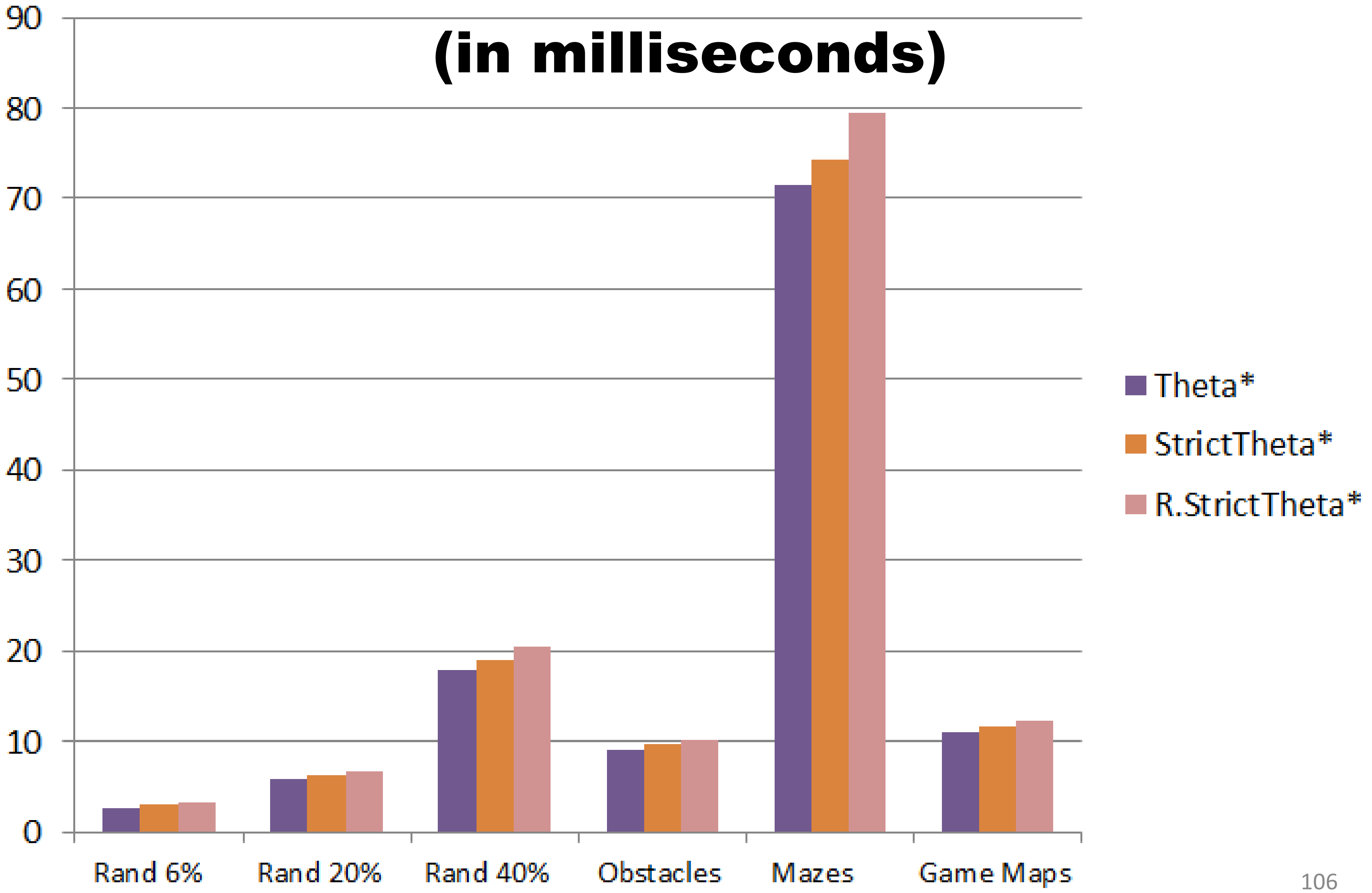
(As a ratio to the optimal)



Running Time

(Averaged, in milliseconds)

Running Time



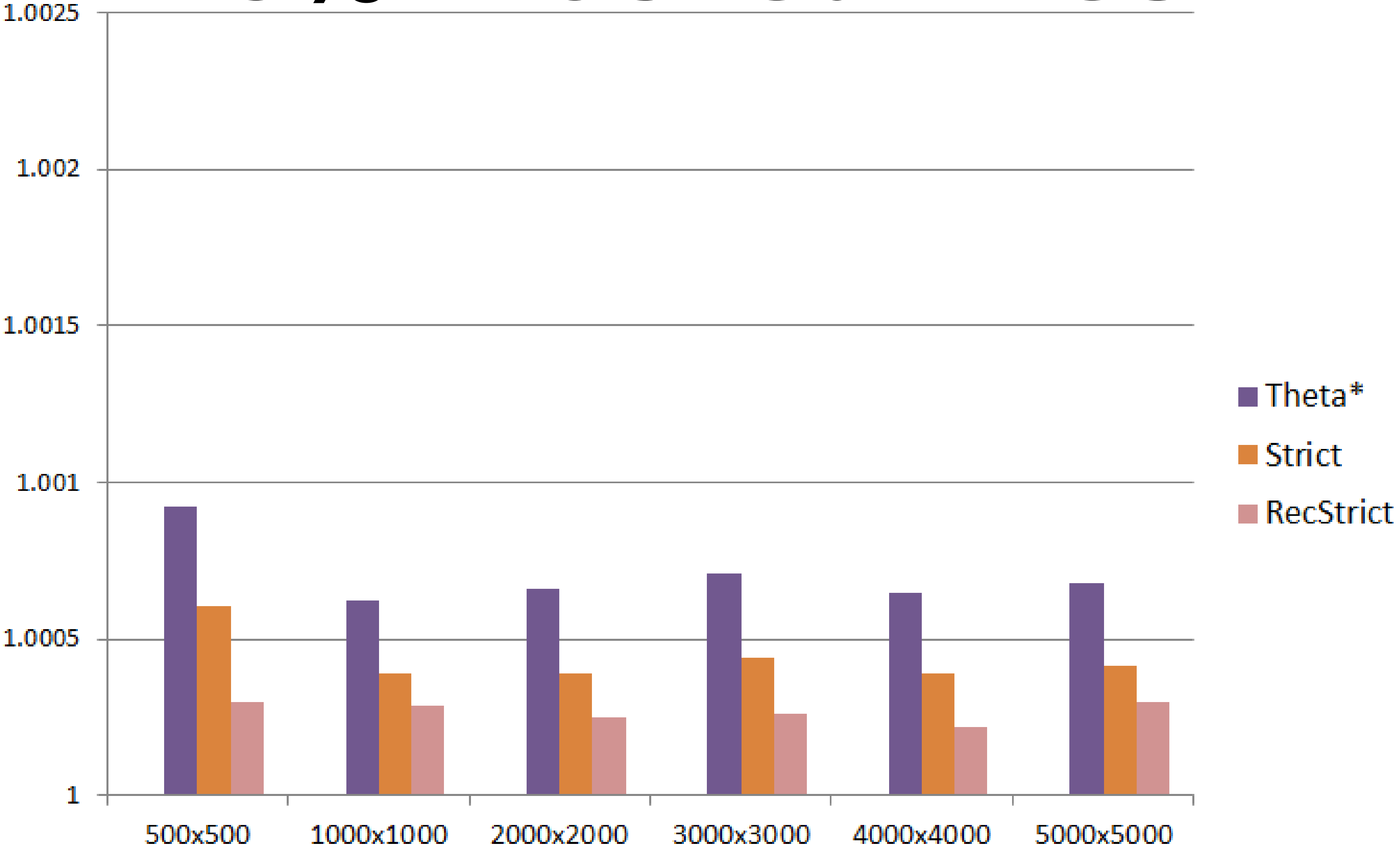
Results

Using large randomly generated maps of sizes:

500x500
1000x1000
2000x2000
3000x3000
4000x4000
5000x5000

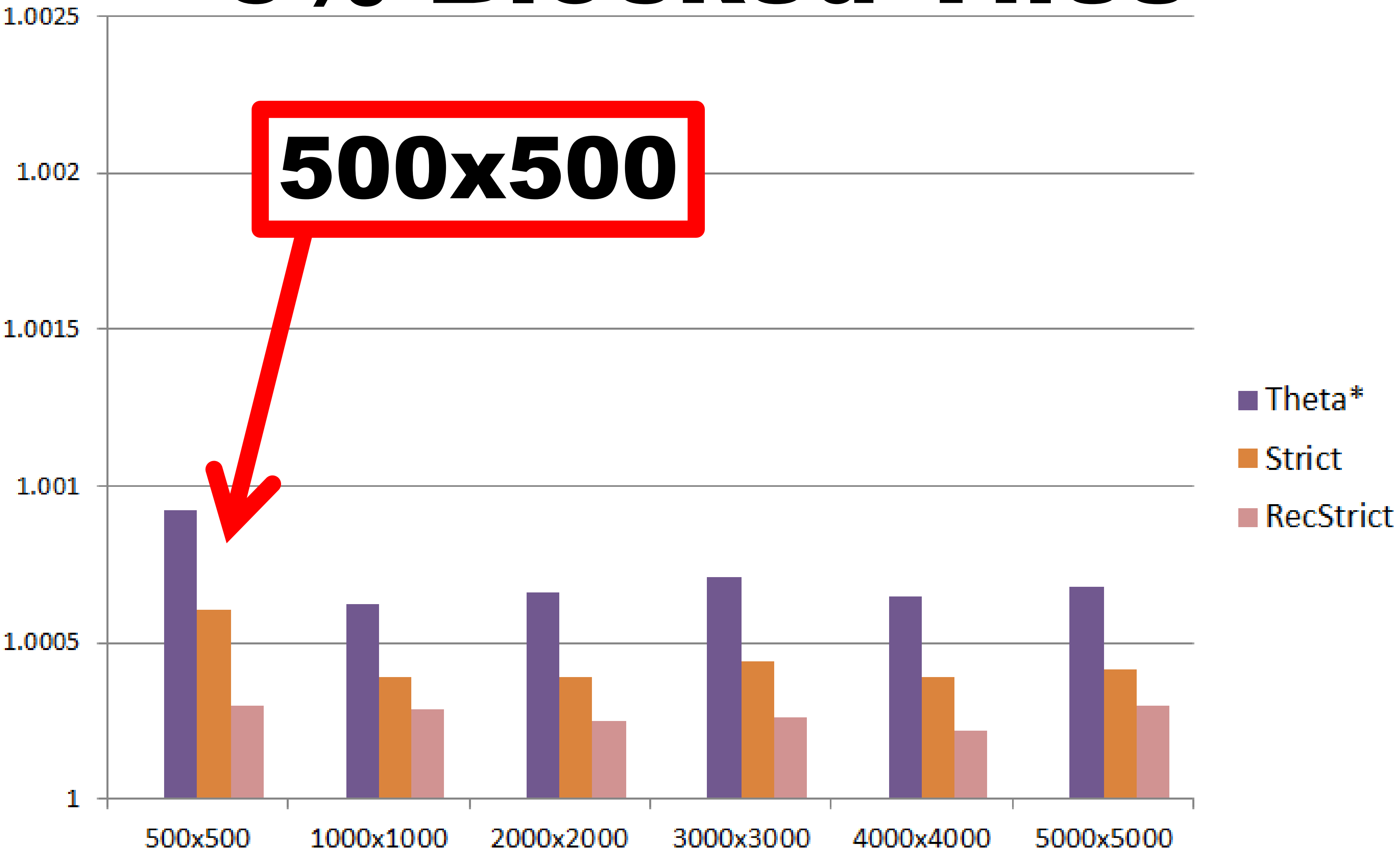
Path Length

6% Blocked Tiles



Path Length

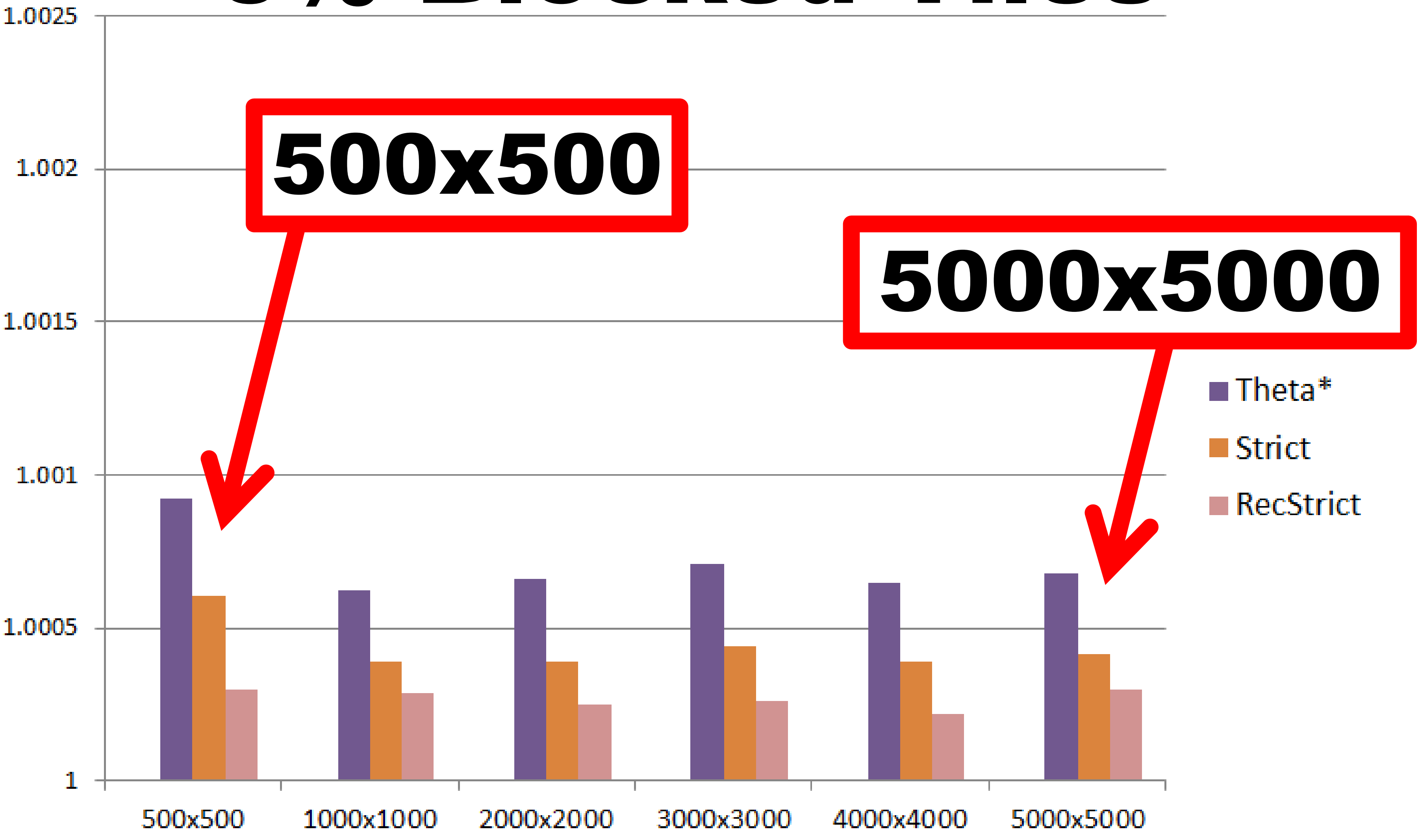
6% Blocked Tiles



500x500

Path Length

6% Blocked Tiles

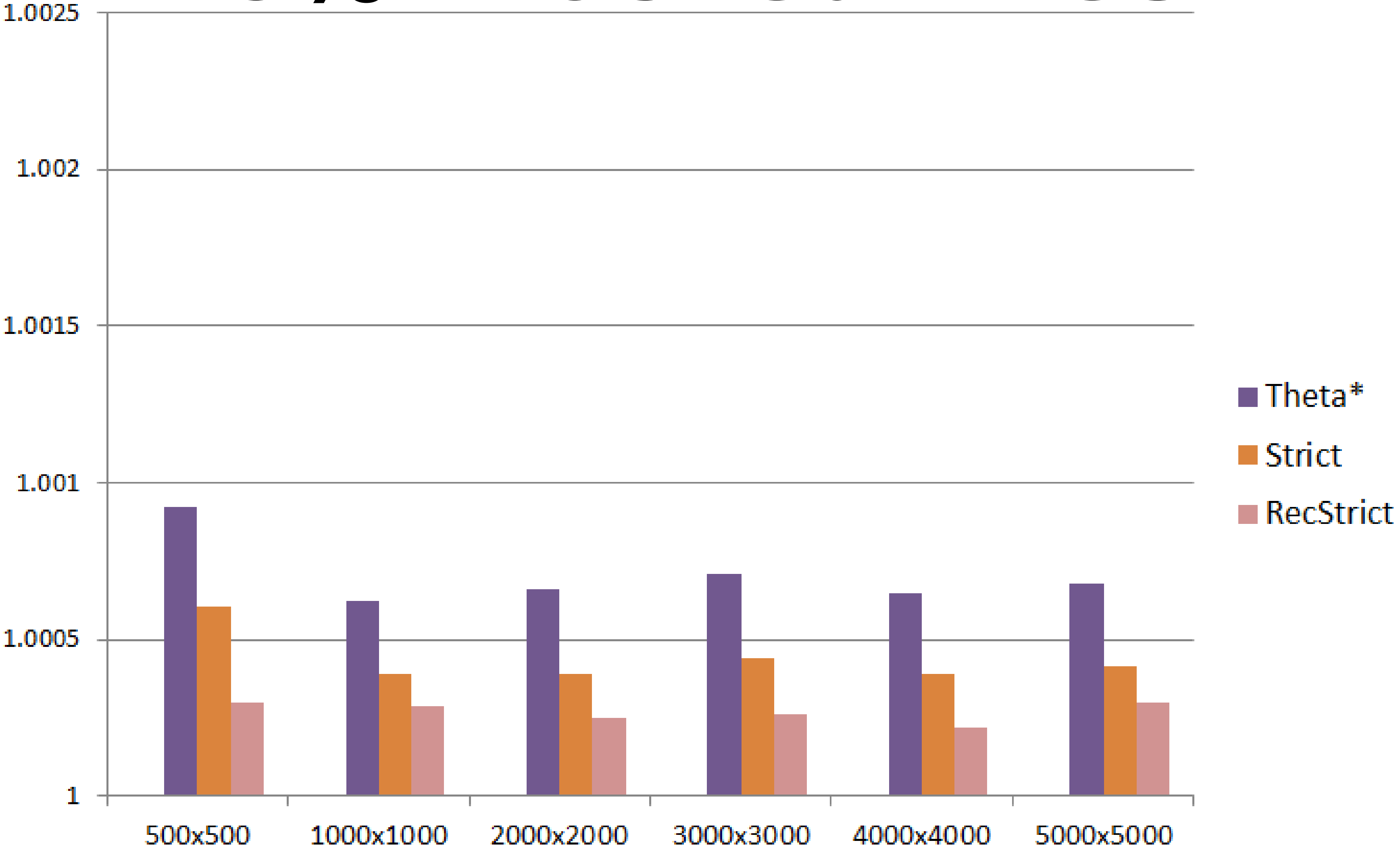


500x500

5000x5000

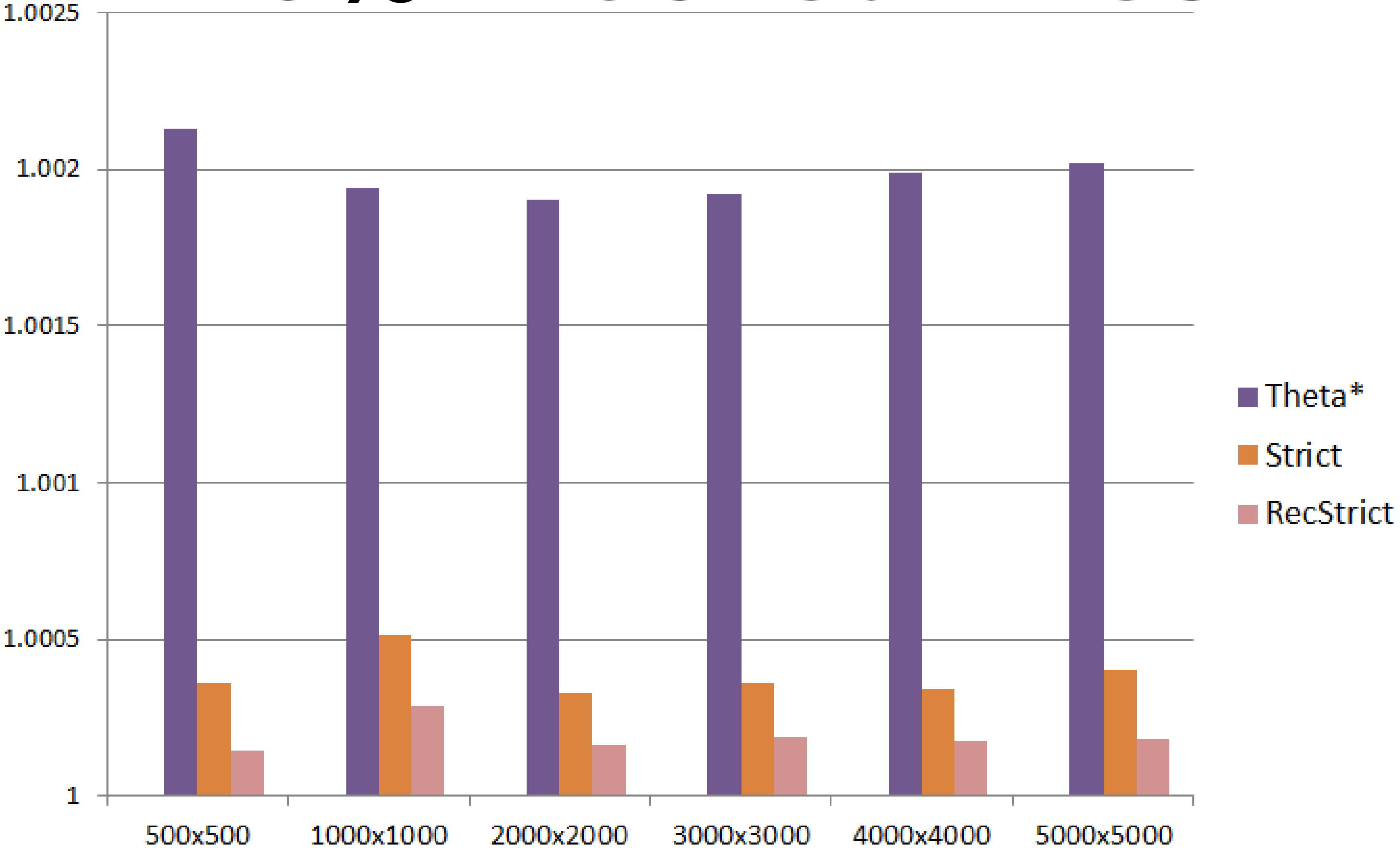
Path Length

6% Blocked Tiles



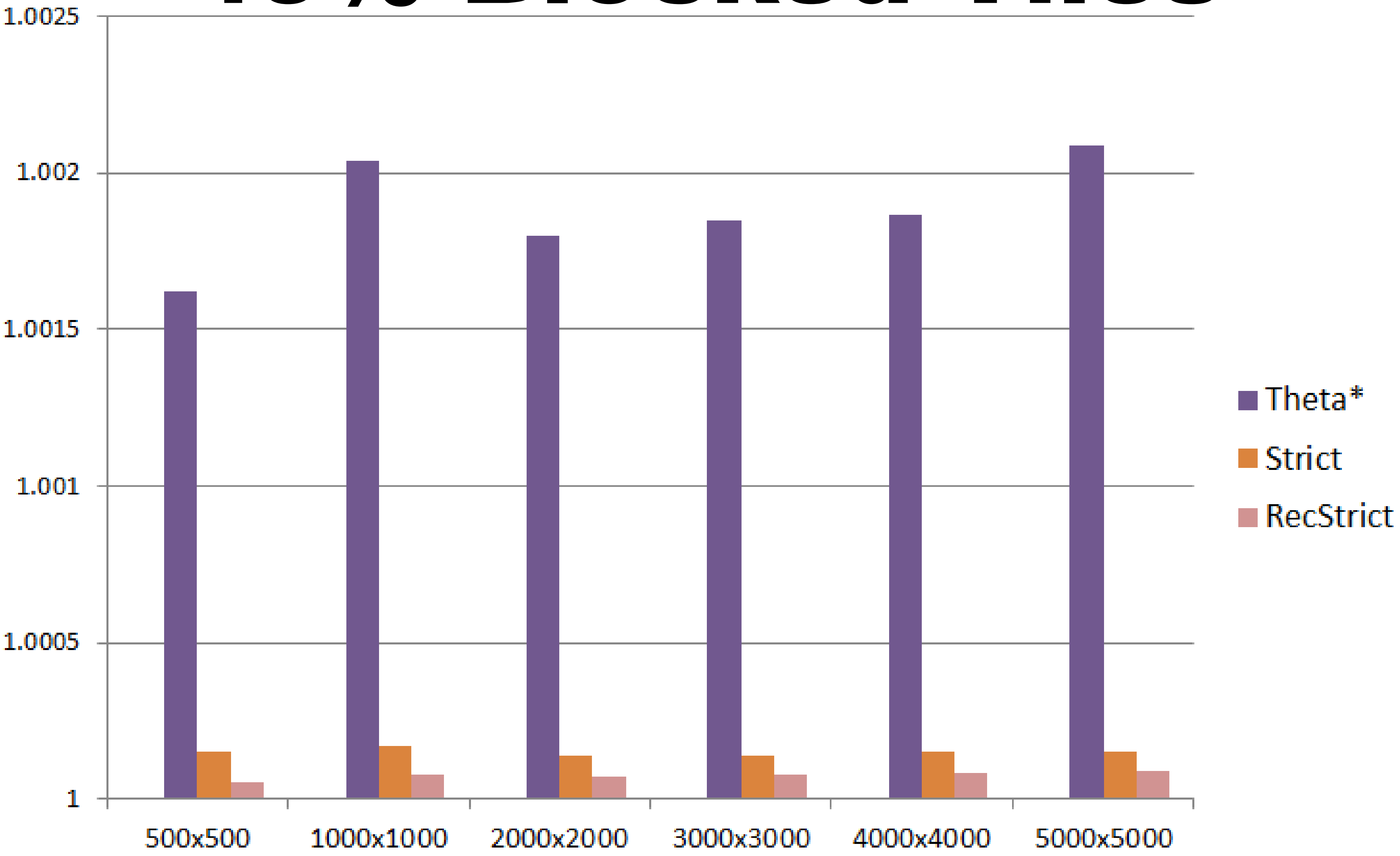
Path Length

20% Blocked Tiles



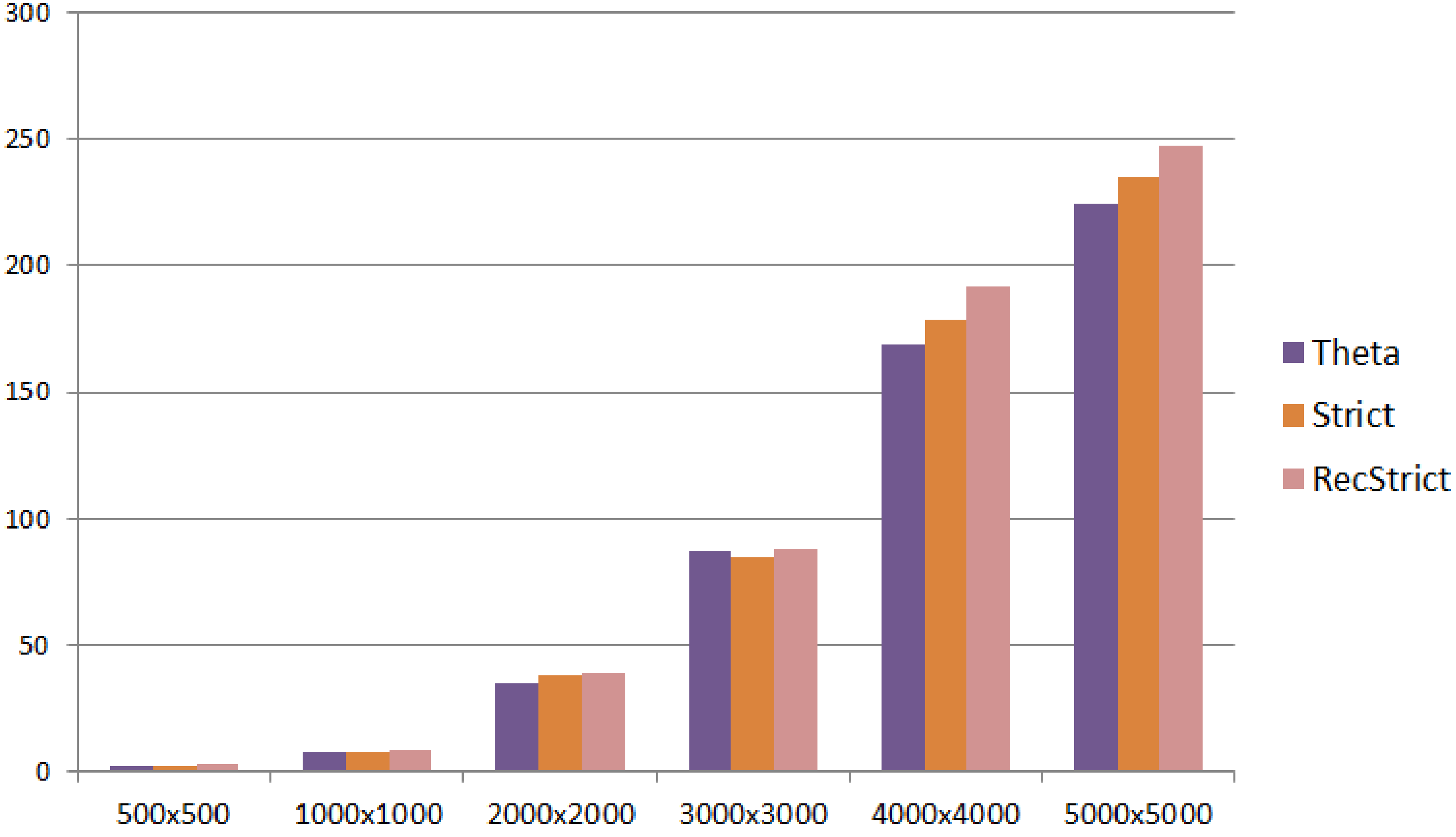
Path Length

40% Blocked Tiles

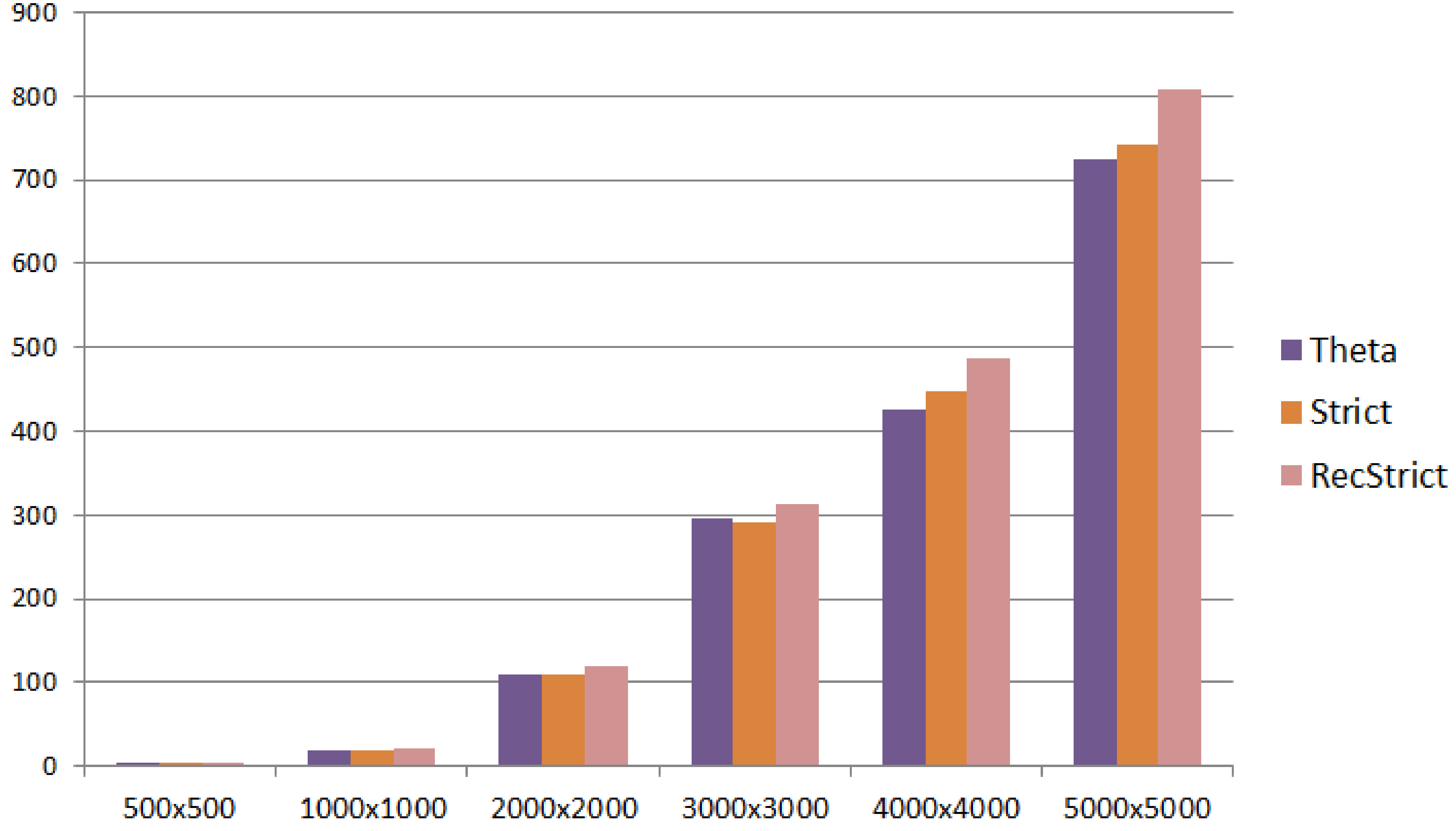


Running Time

6% Blocked Tiles

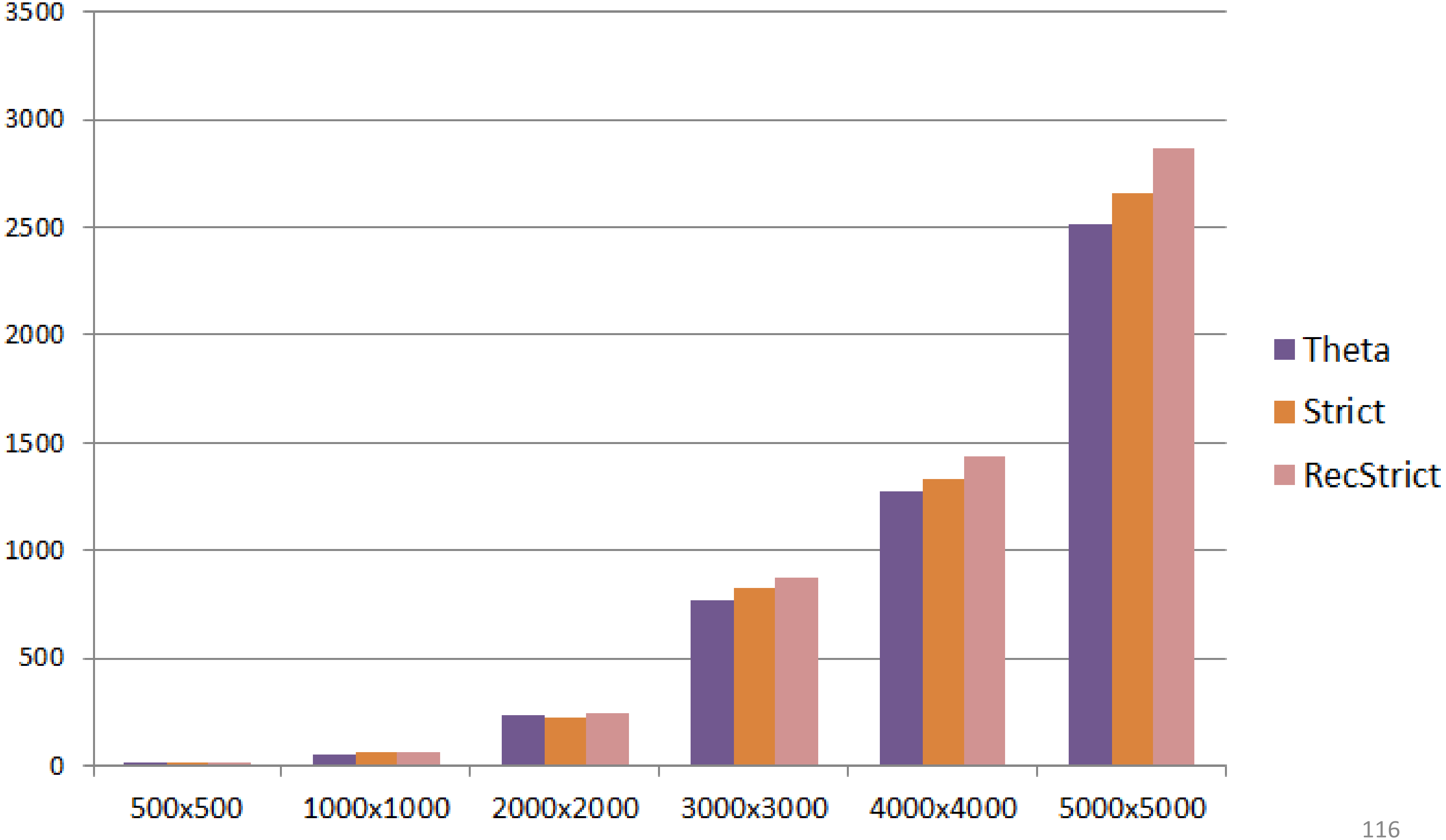


Running Time
20% Blocked Tiles



Running Time

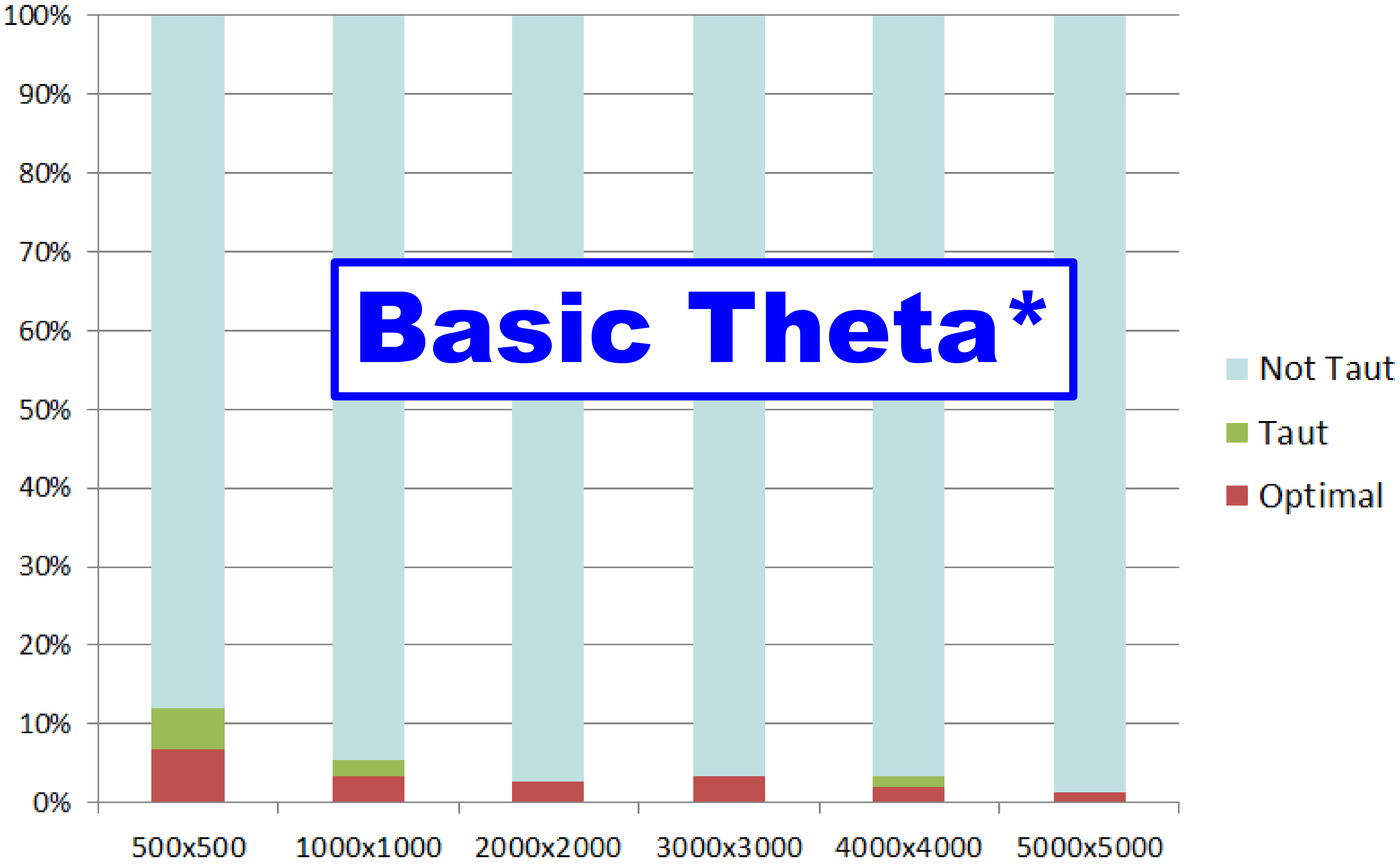
40% Blocked Tiles



Percentage Optimal / Taut

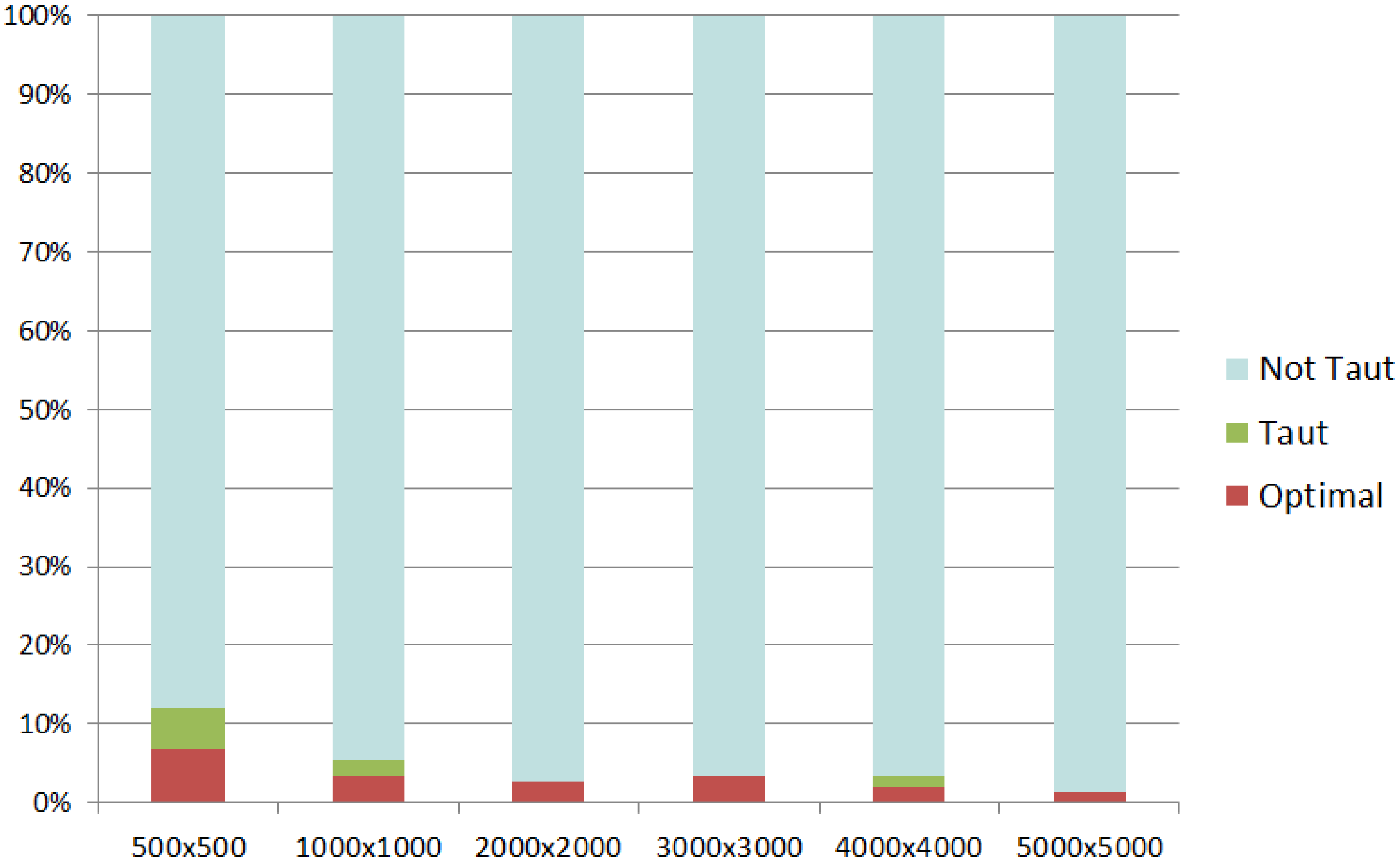
(Note: Optimal Paths are Always Taut)

Percentage Optimal / Taut

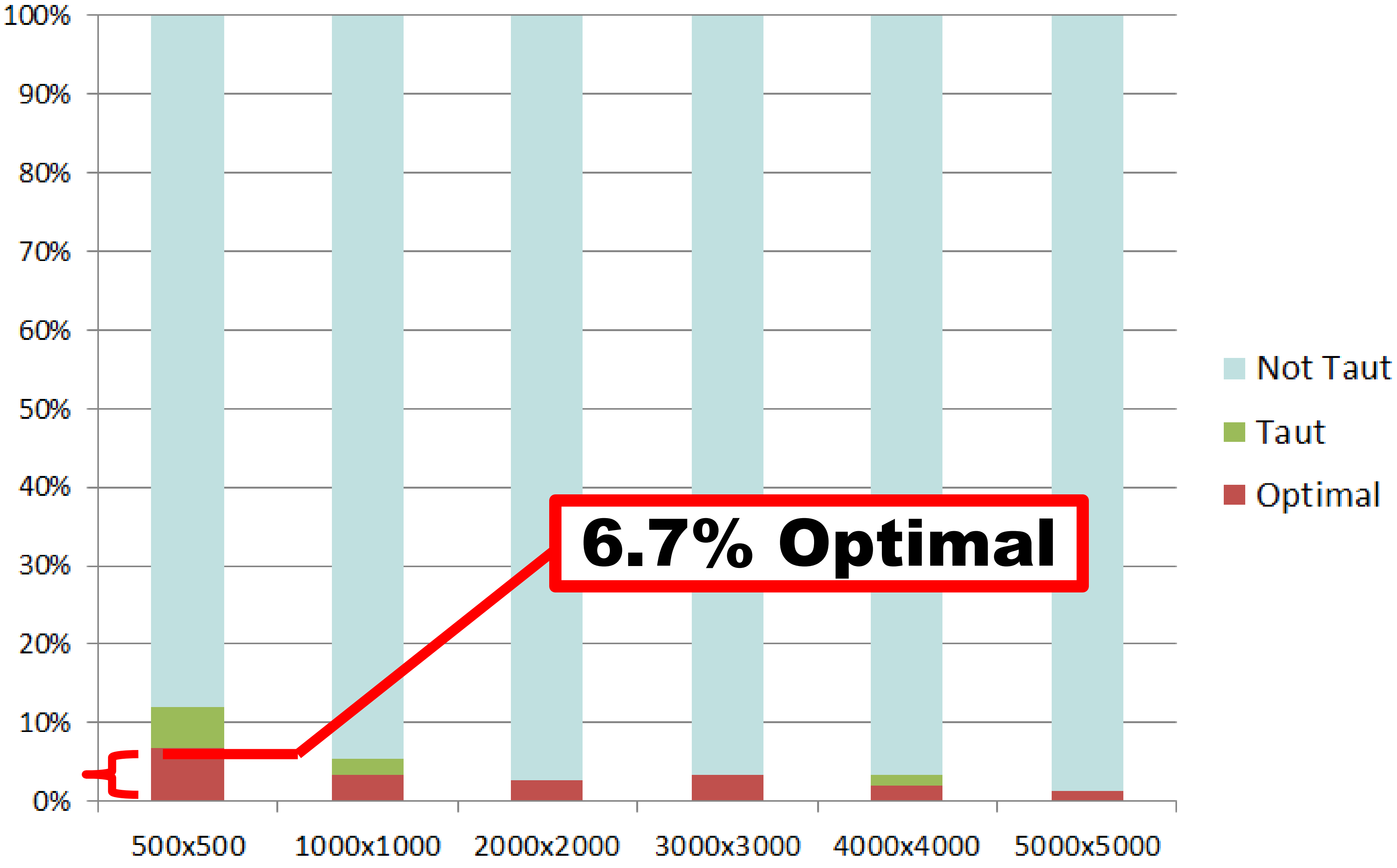


Basic Theta*

Percentage Optimal / Taut

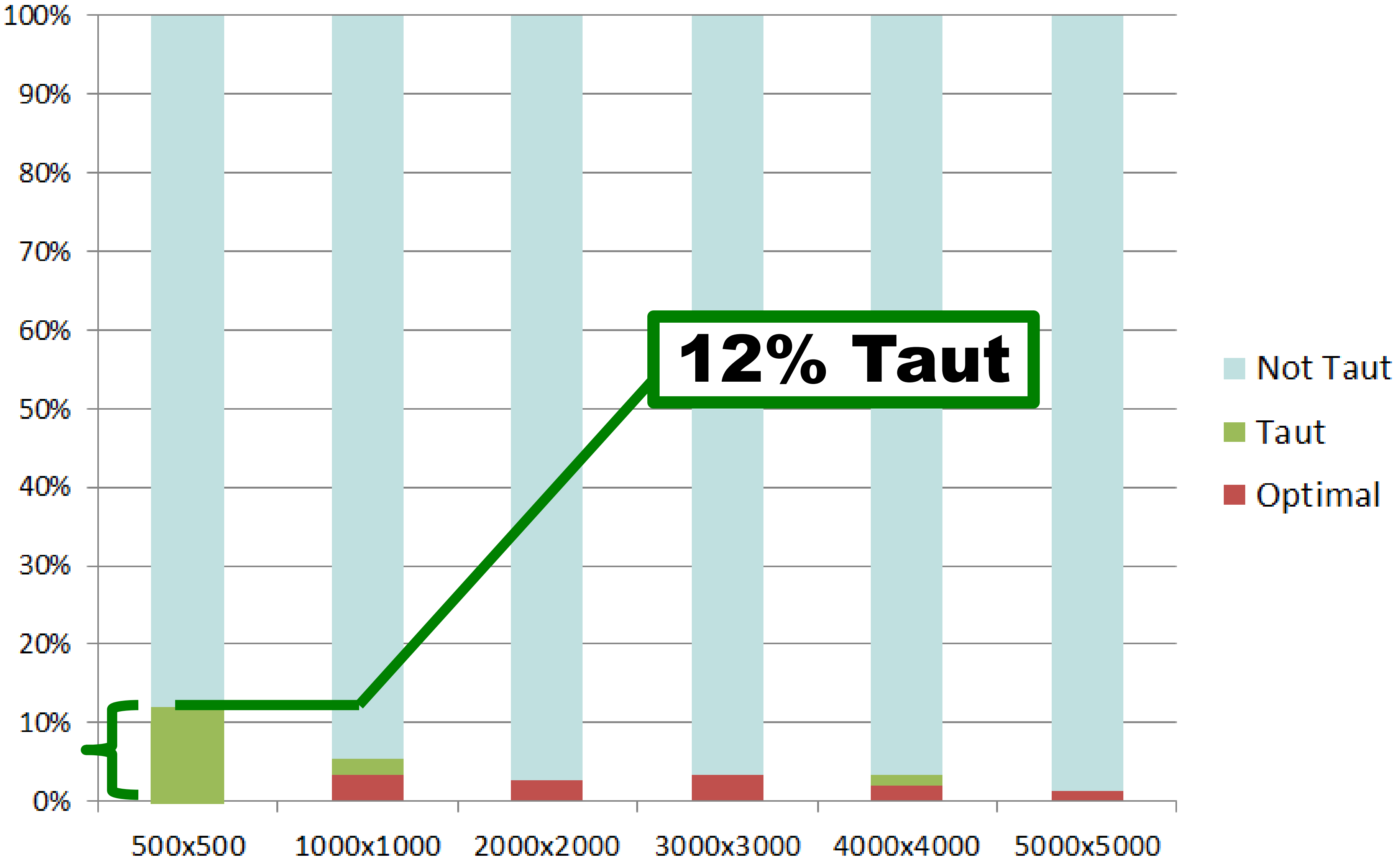


Percentage Optimal / Taut



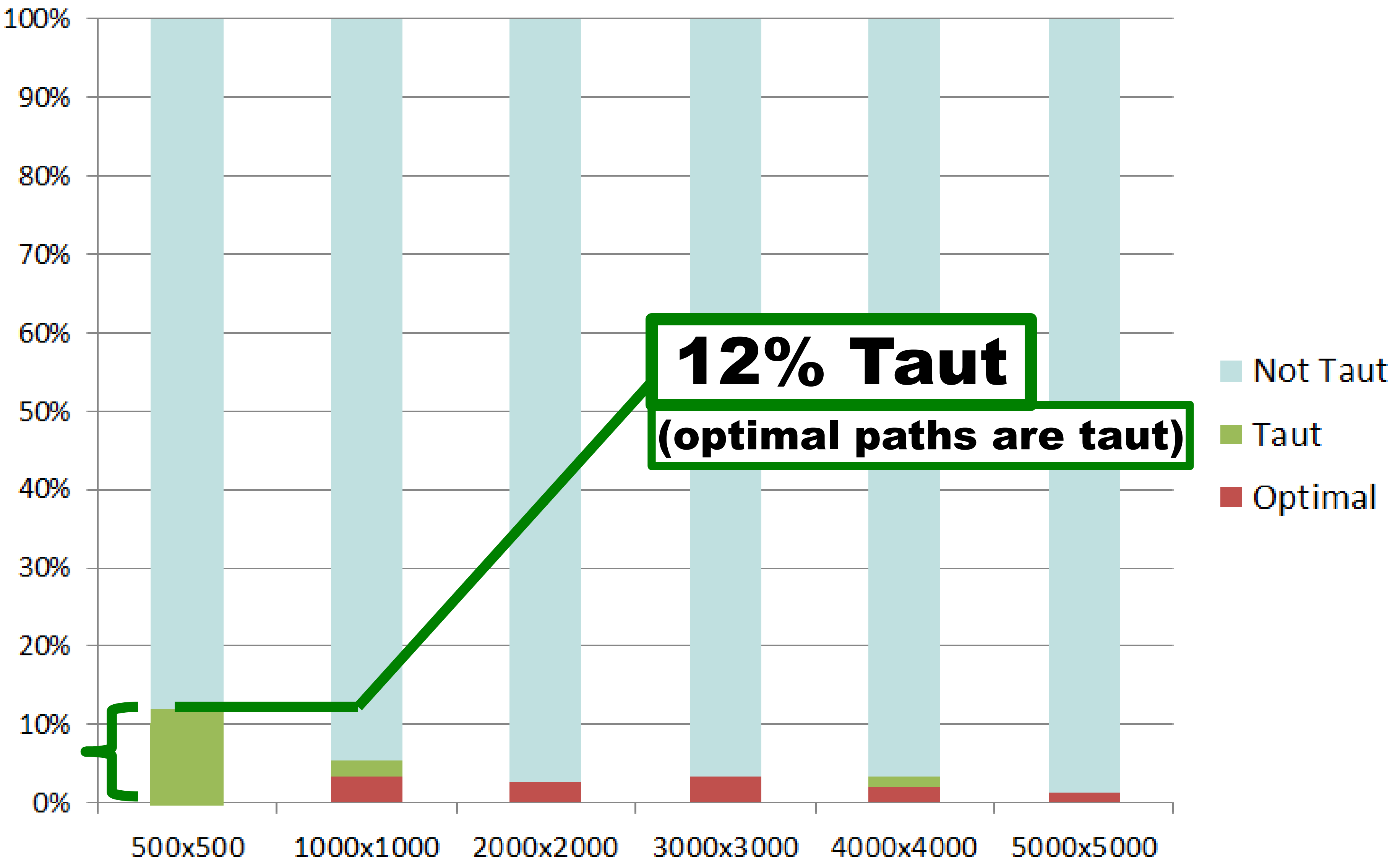
6.7% Optimal

Percentage Optimal / Taut



12% Taut

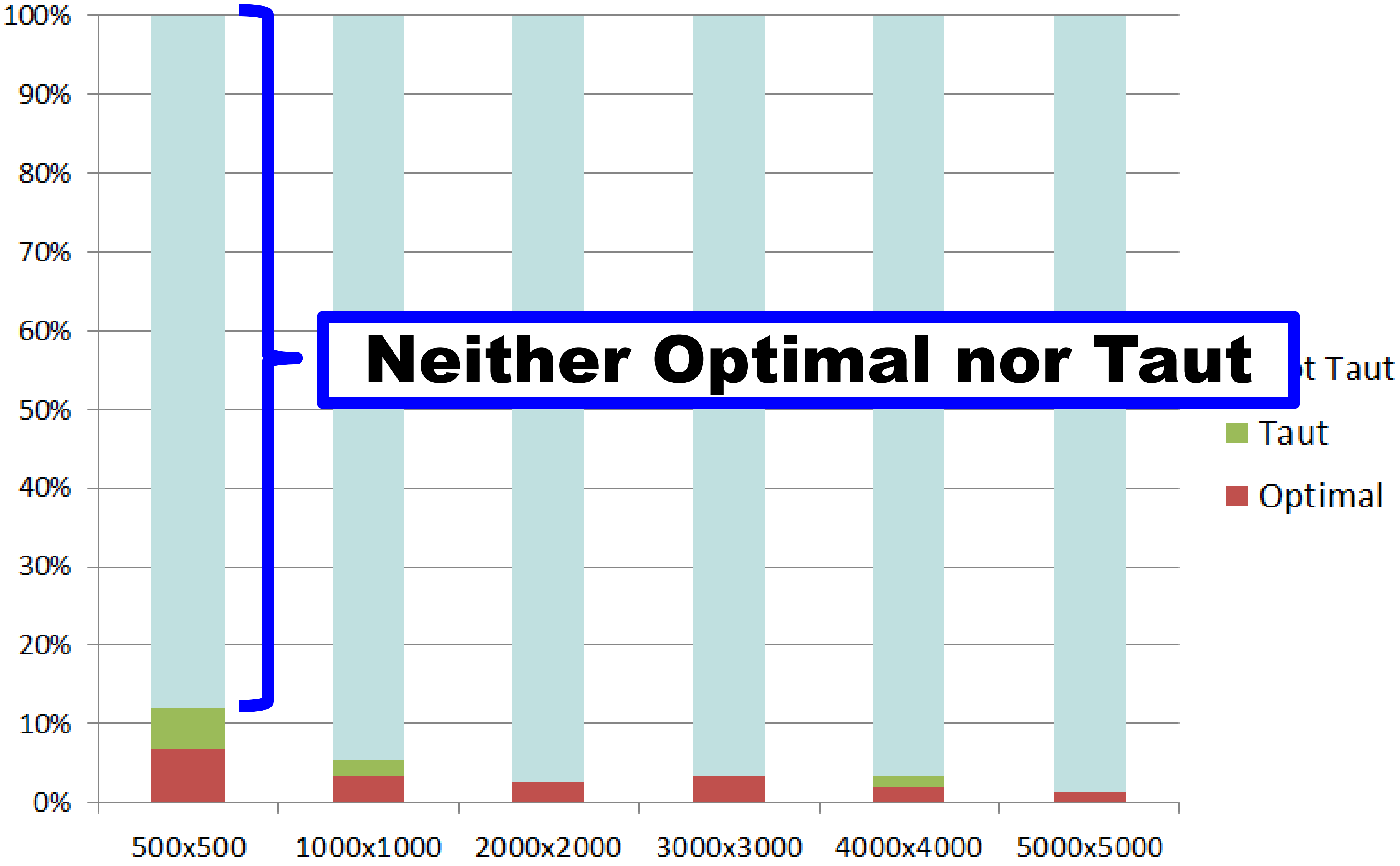
Percentage Optimal / Taut



12% Taut

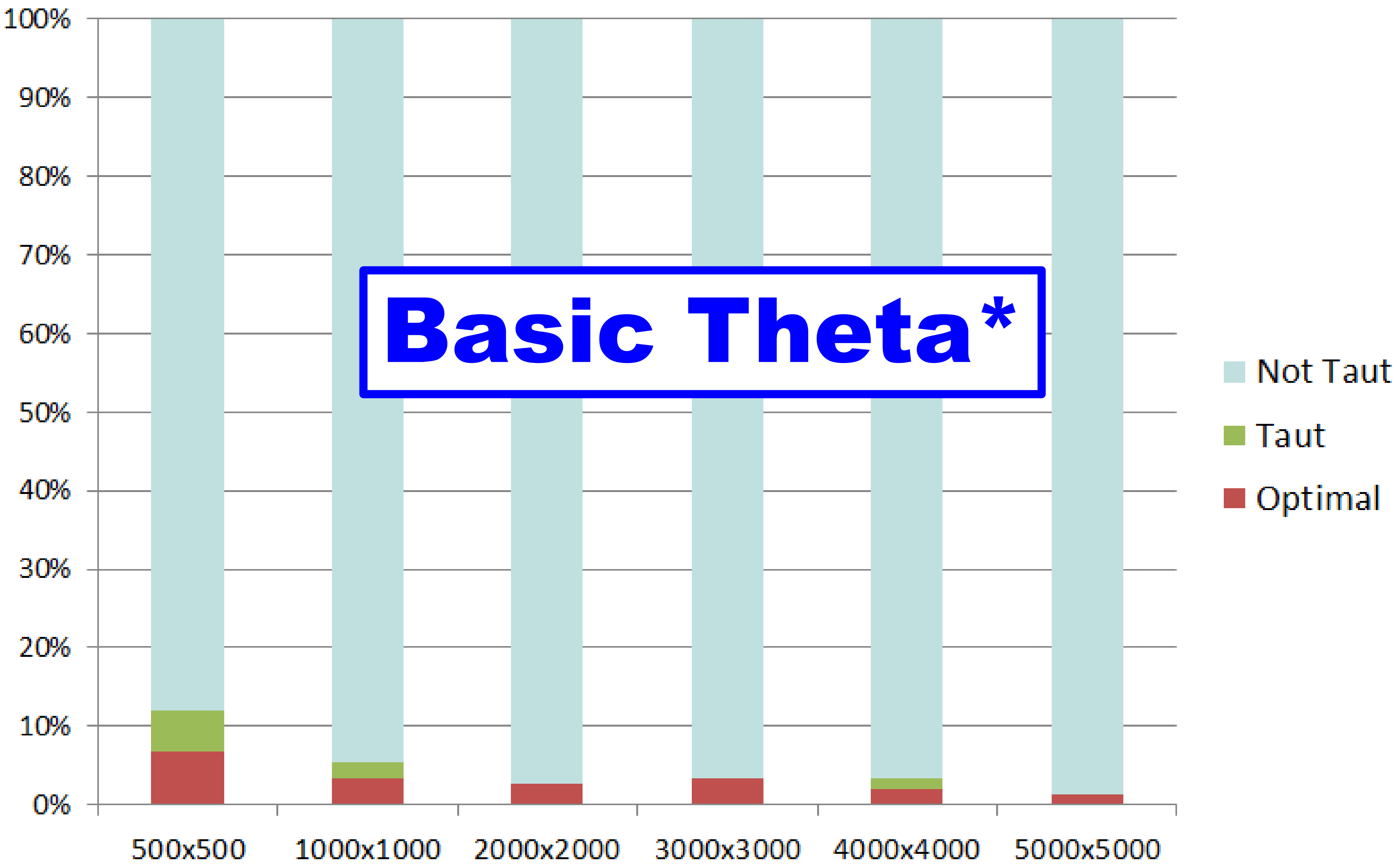
(optimal paths are taut)

Percentage Optimal / Taut



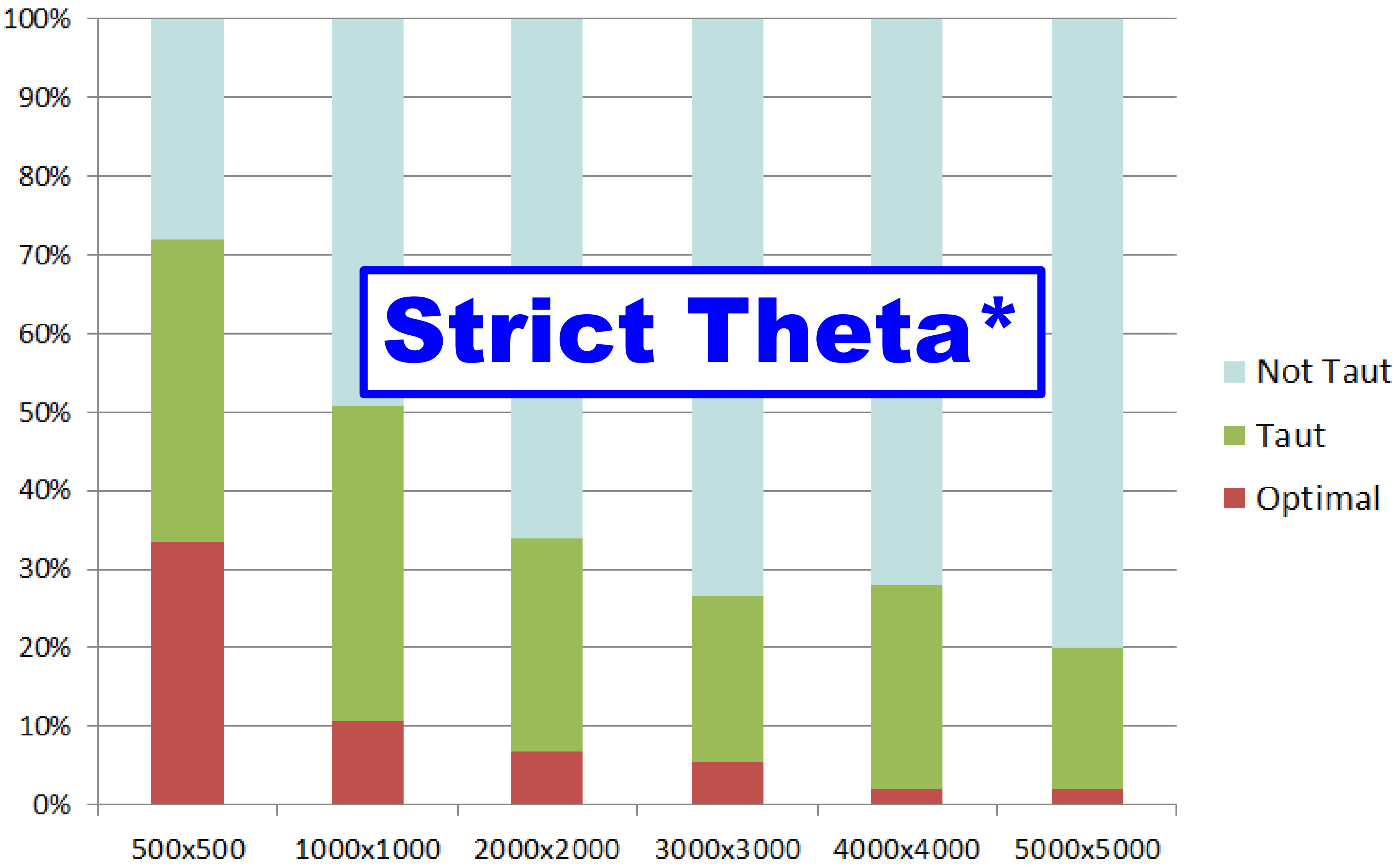
Neither Optimal nor Taut

Percentage Optimal / Taut



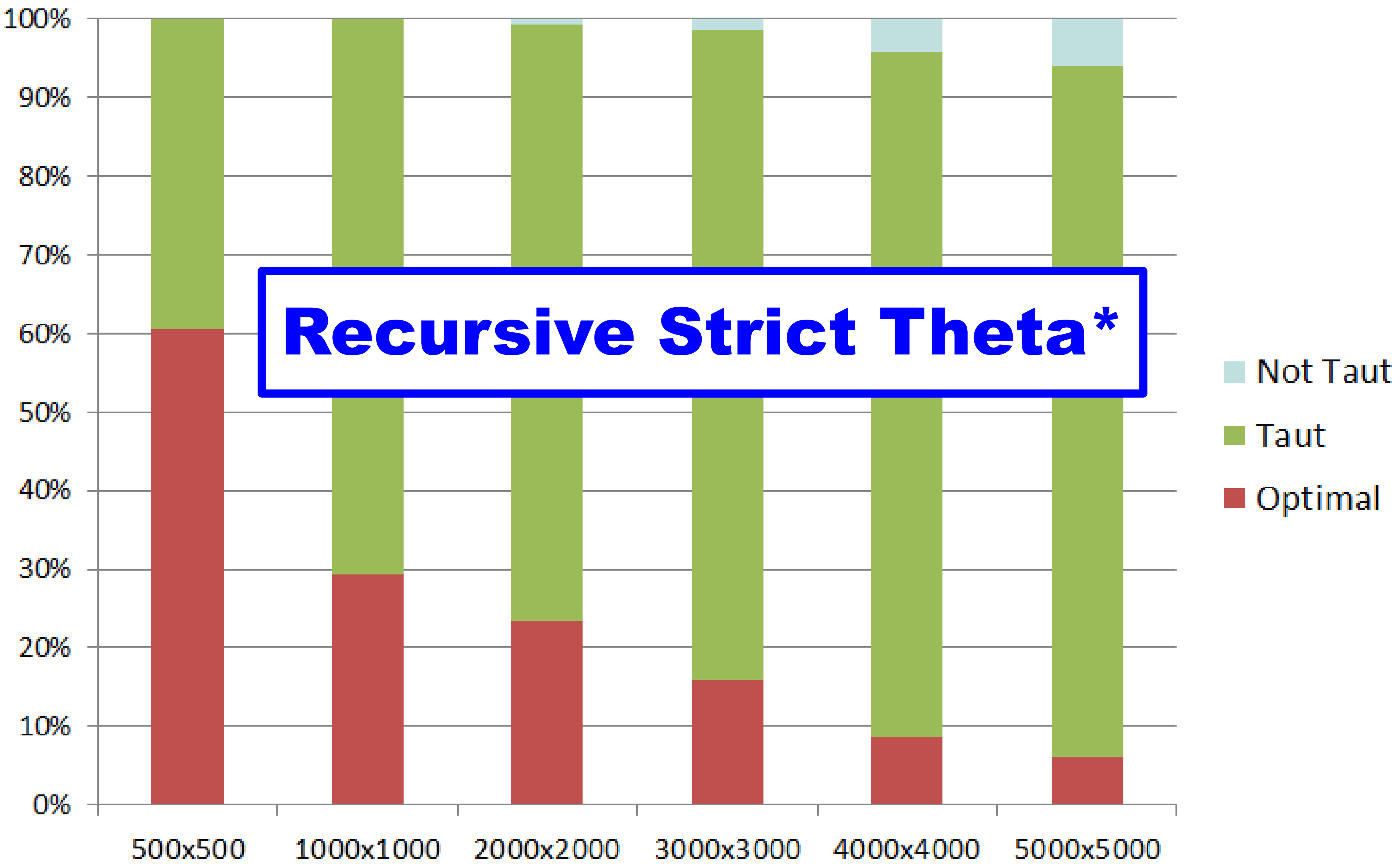
Basic Theta*

Percentage Optimal / Taut



Strict Theta*

Percentage Optimal / Taut



Recursive Strict Theta*

Strict Theta*

Easy to implement

Much shorter paths

Low runtime overhead

Implementation

github.com/Ohohcakester/Any-Angle-Pathfinding

Implementation

github.com/Ohohcakester/Any-Angle-Pathfinding

Google for "Any Angle Pathfinding"

Strict Theta*

Shorter Motion Planning Using Taut Paths

Shunhao Oh
ohoh@u.nus.edu

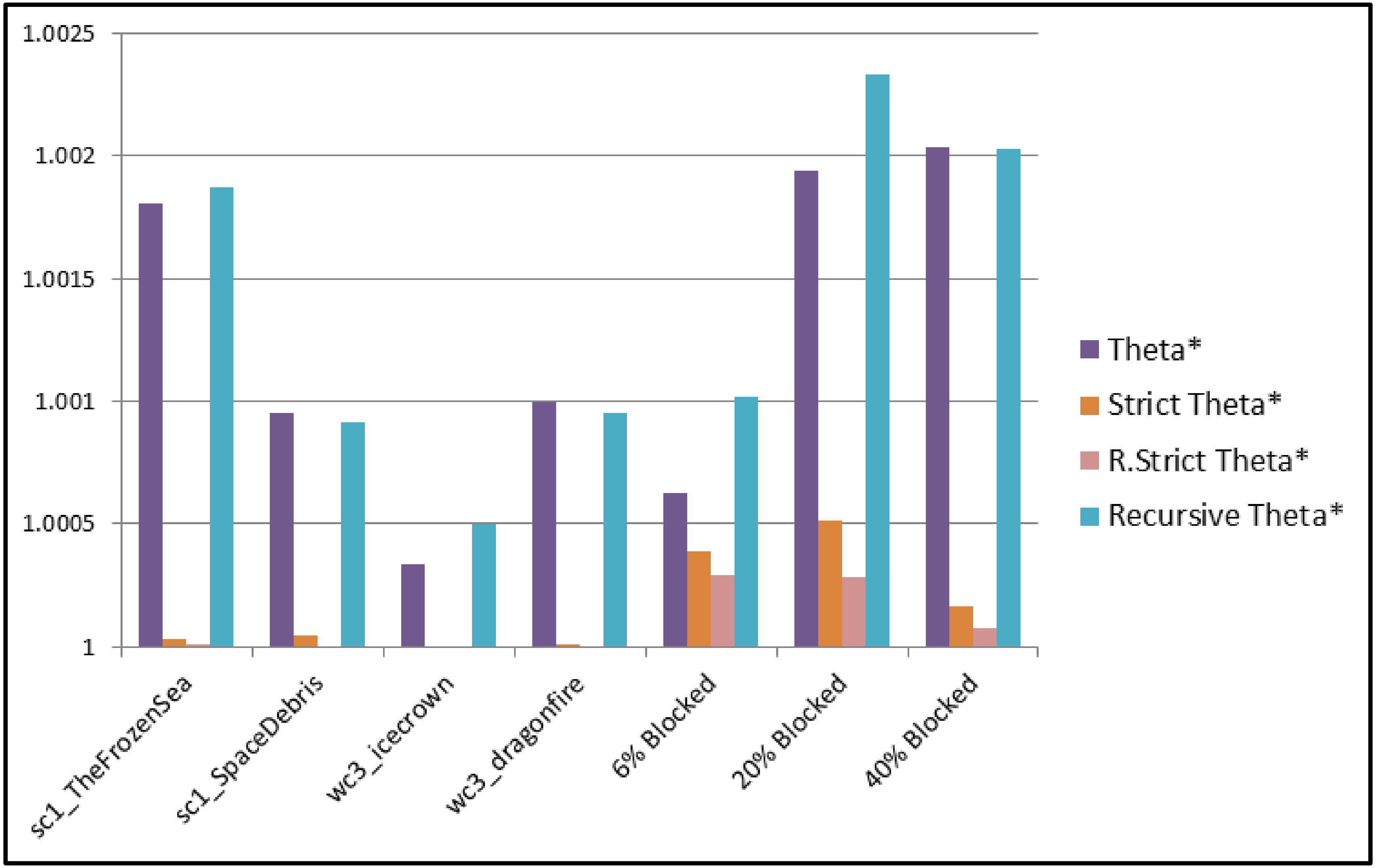
Hon Wai Leong
leonghw@comp.nus.edu.sg

Department of Computer Science
National University of Singapore

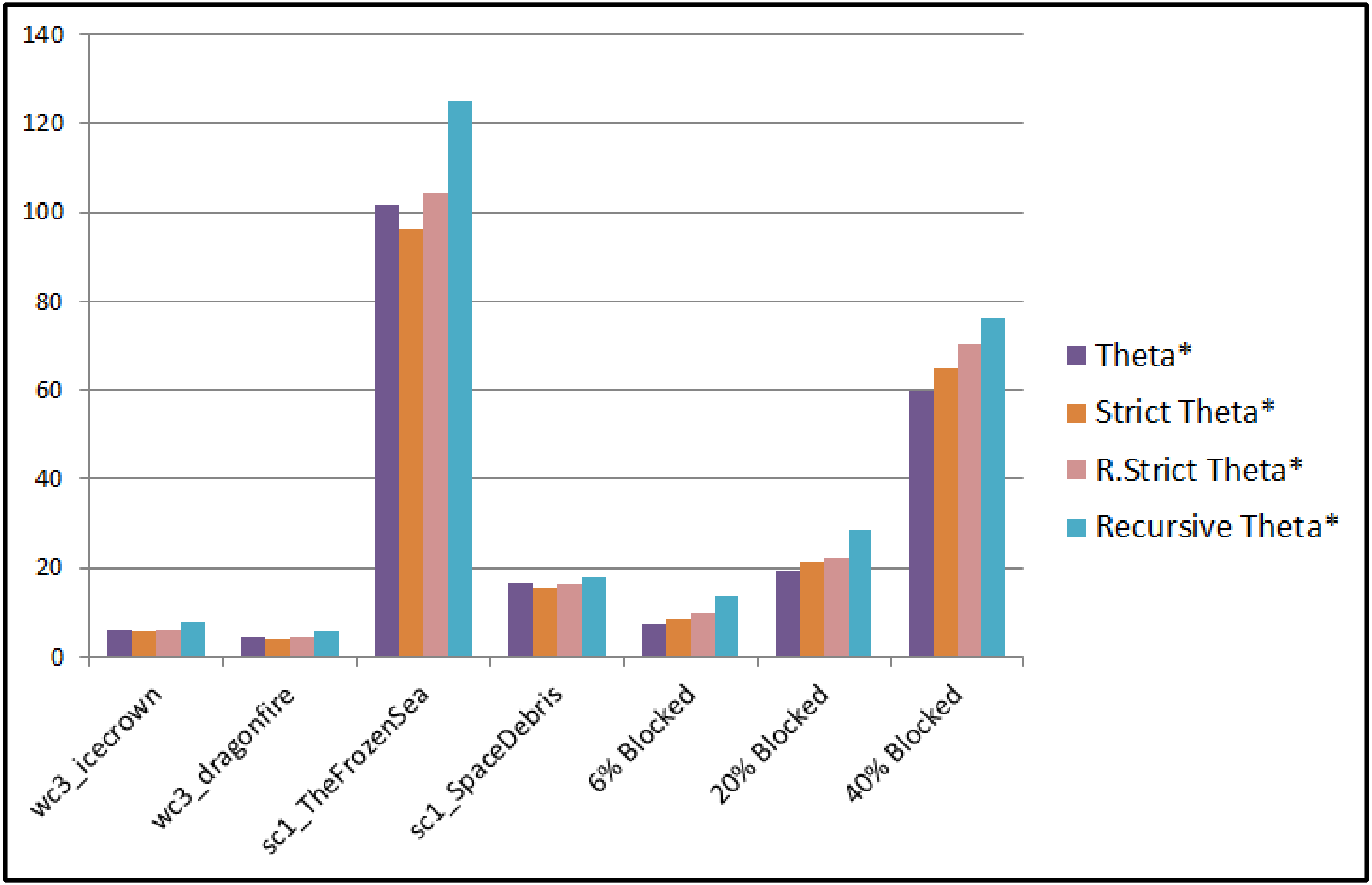
Implementation:

github.com/Ohohcakester/Any-Angle-Pathfinding

Path Length

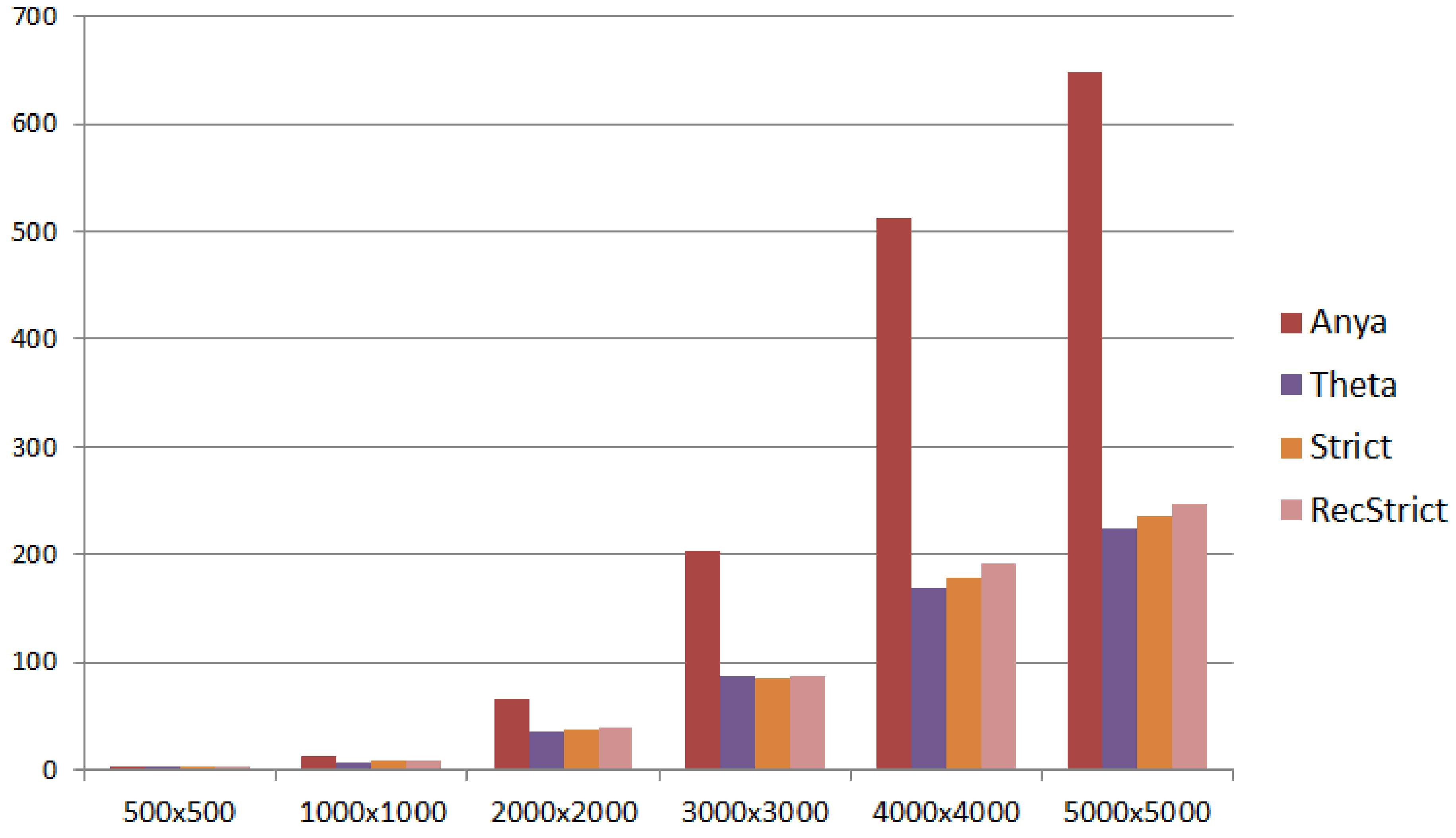

1.0025
1.002
1.0015
1.001
1.0005
1
sc1_TheFrozenSea
sc1_SpaceDebris
wc3_icecrown
wc3_dragonfire
6% Blocked
20% Blocked
40% Blocked
Theta*
Strict Theta*
R.Strict Theta*
Recursive Theta*

Running Time



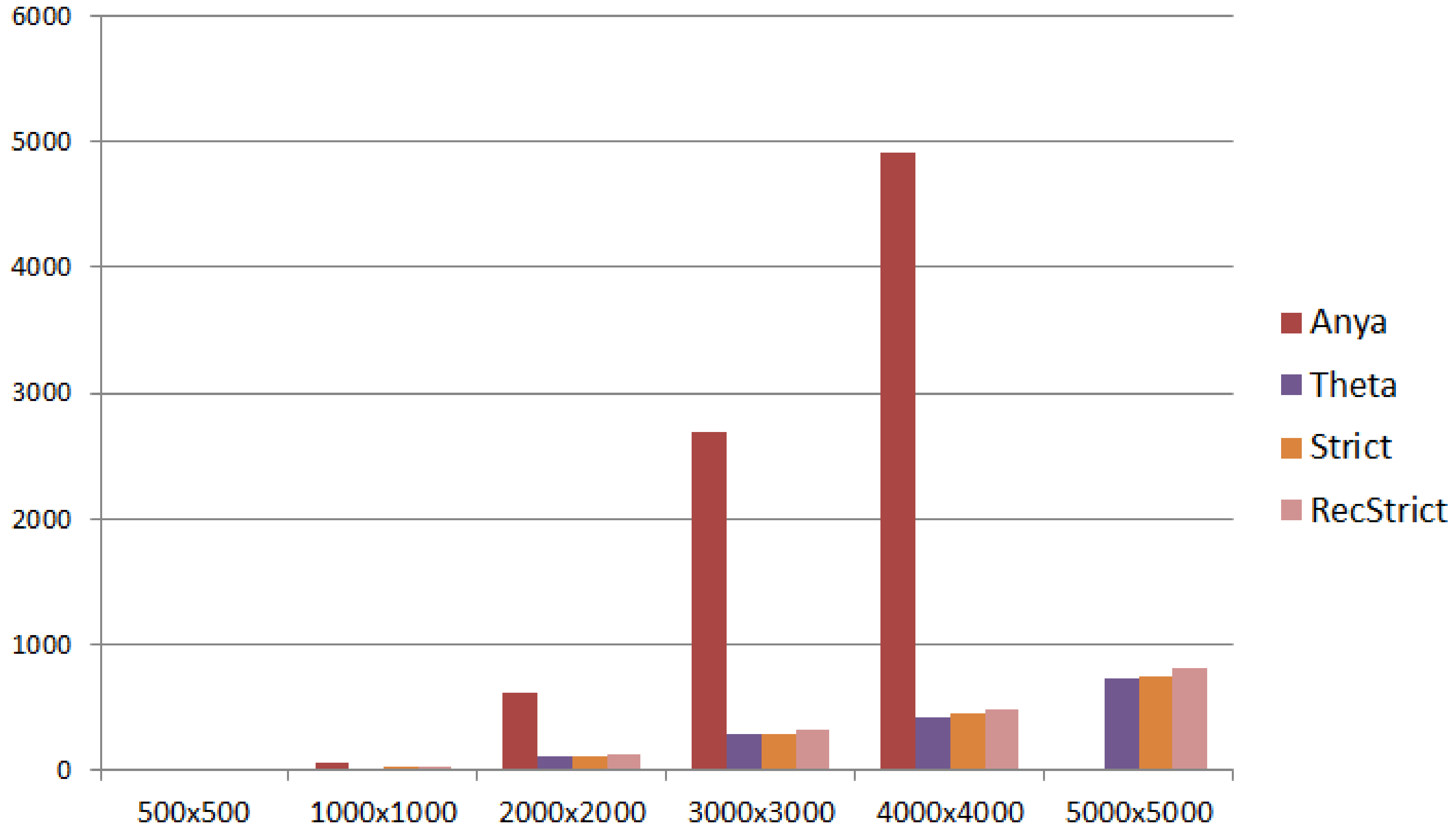
Running Time

6% Blocked Tiles



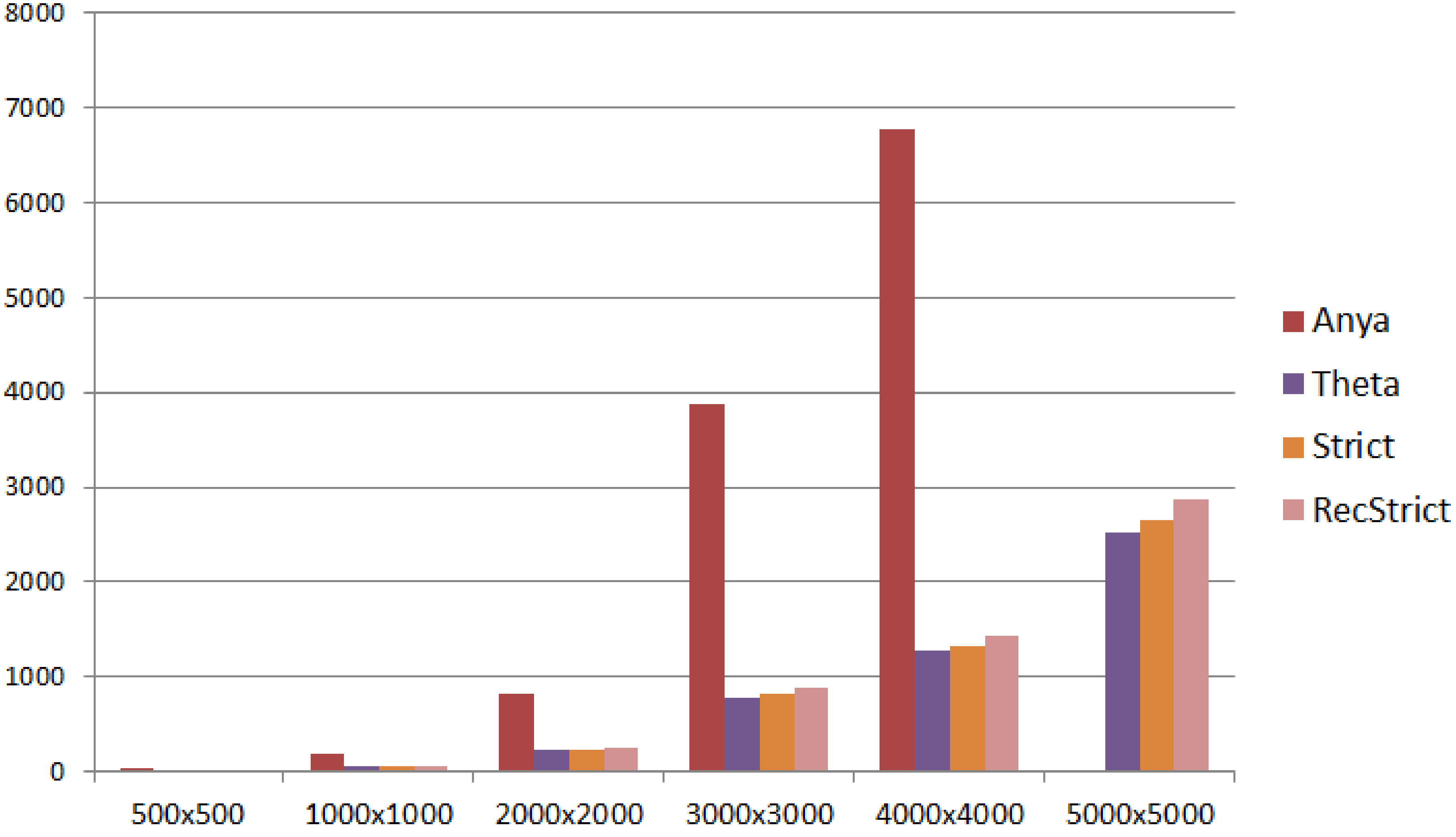
Running Time

20% Blocked Tiles

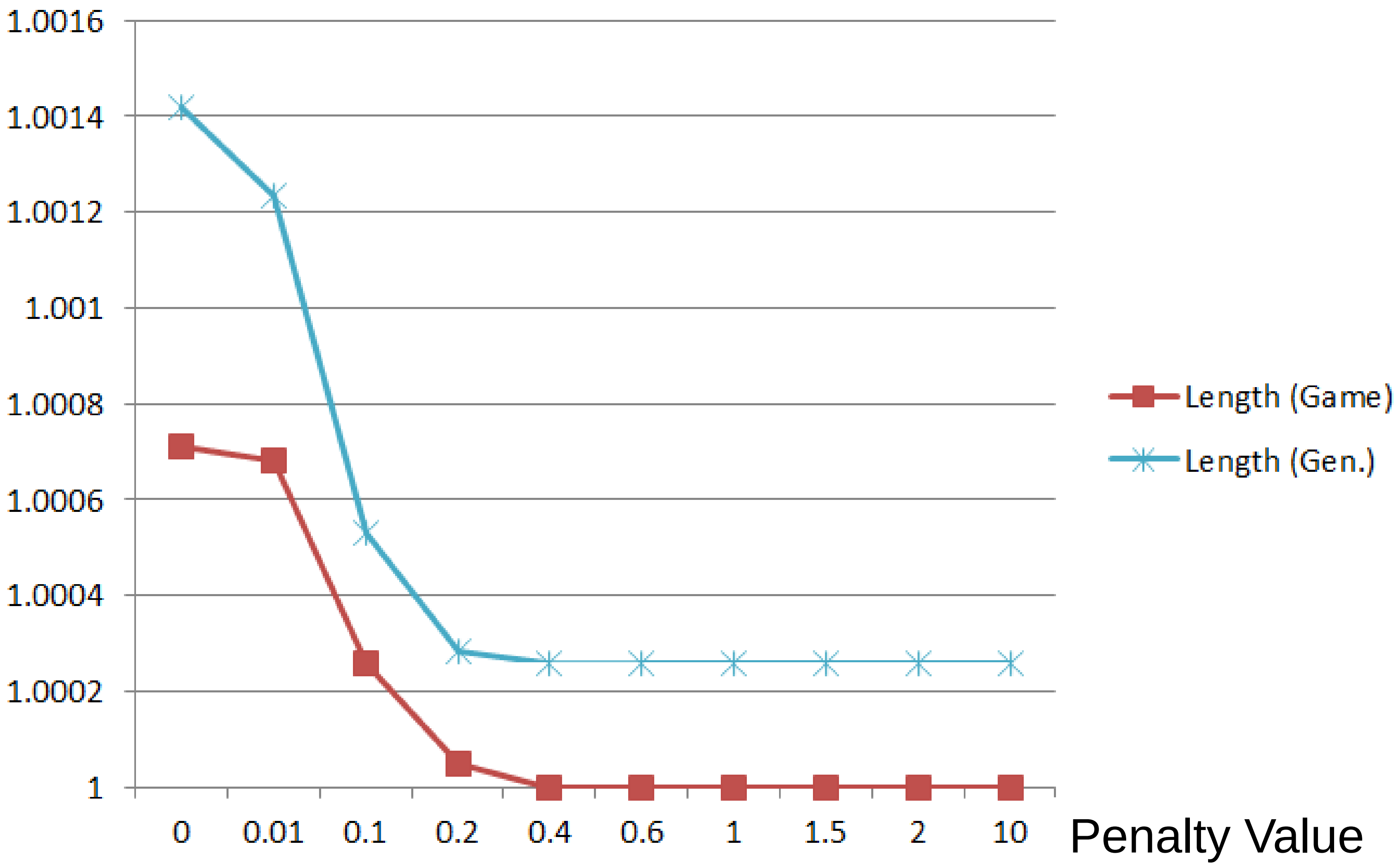


Running Time

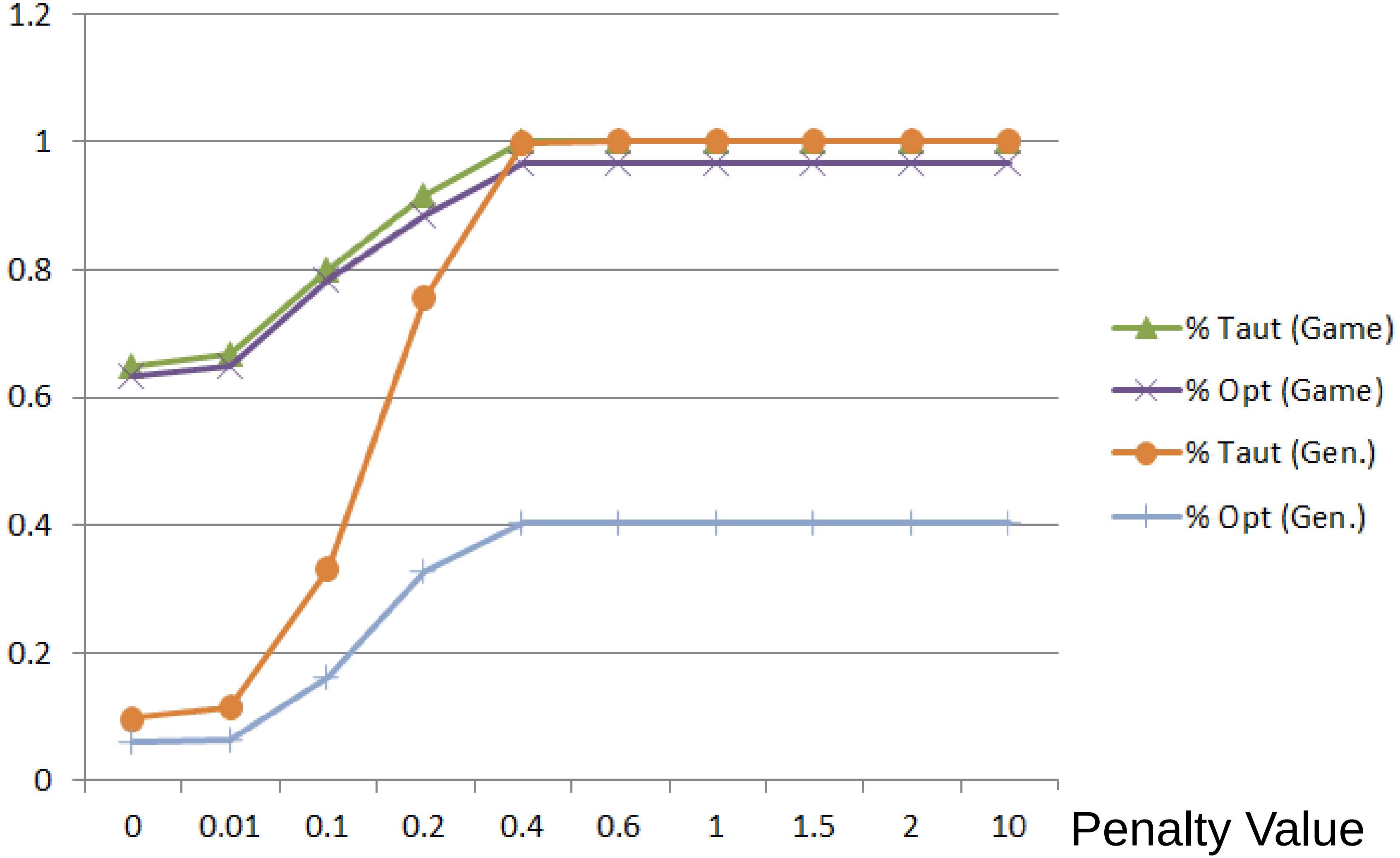
40% Blocked Tiles



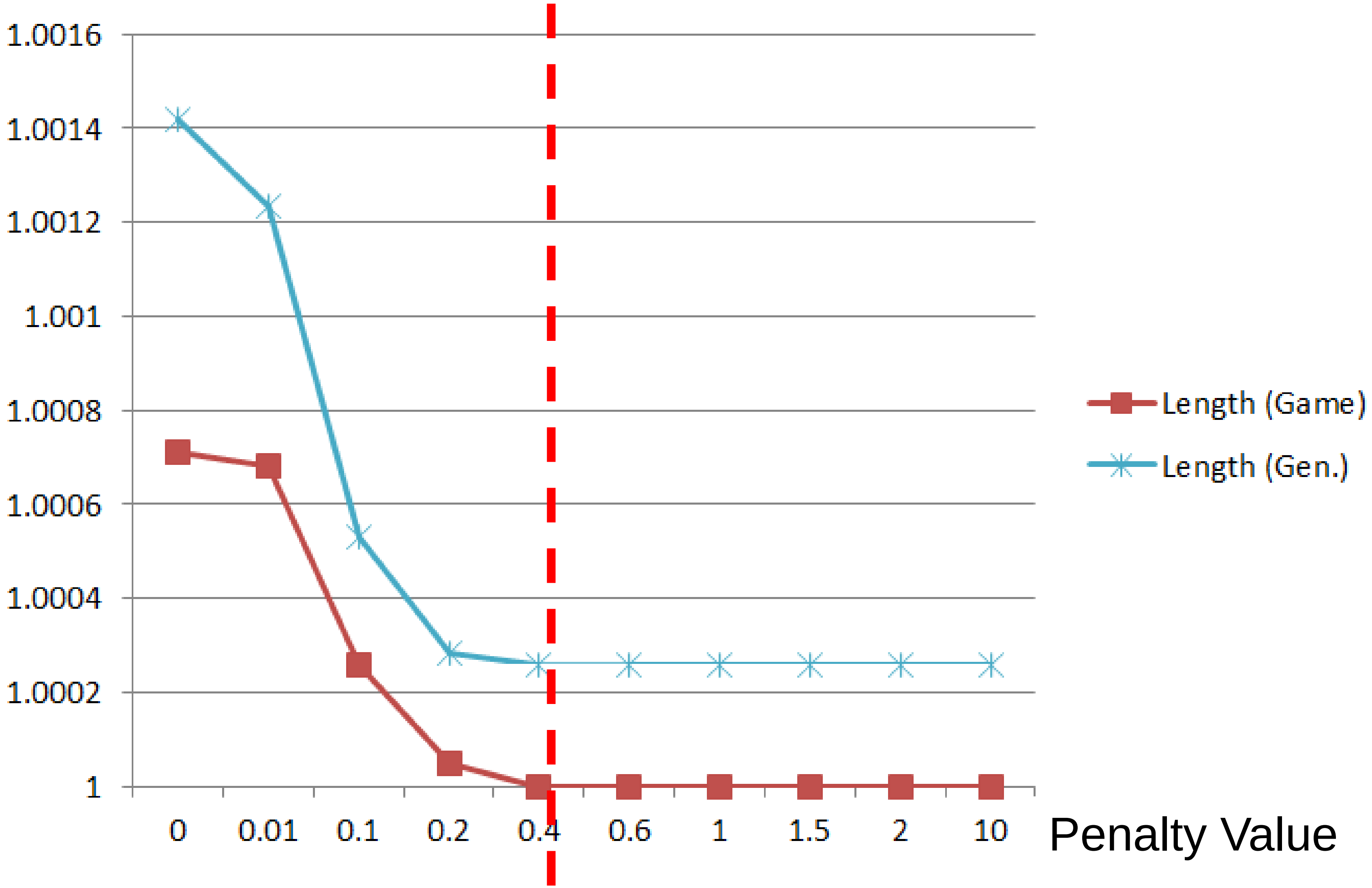
Path Length vs Penalty Value



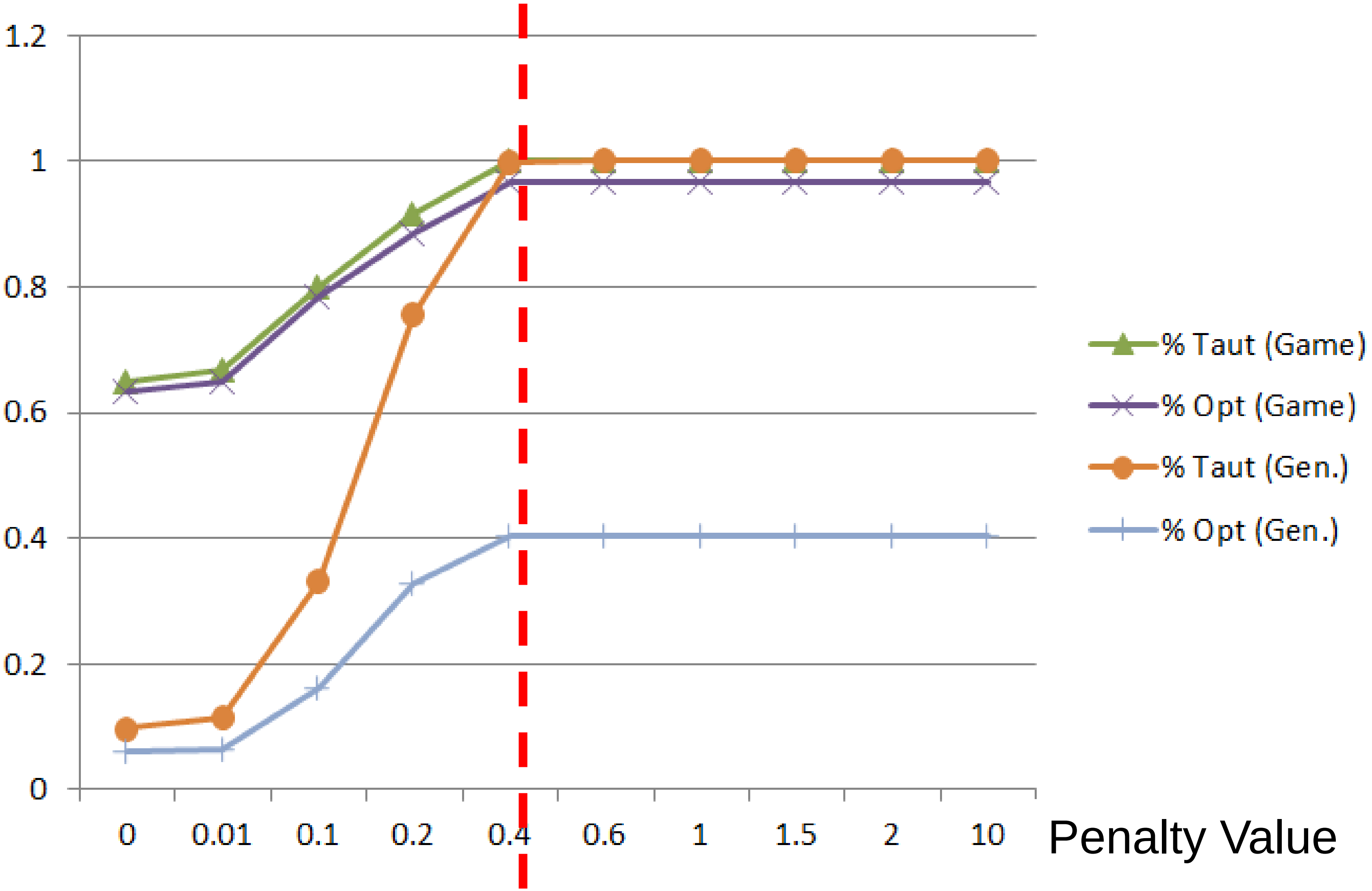
% Optimal/Taut vs Penalty Value

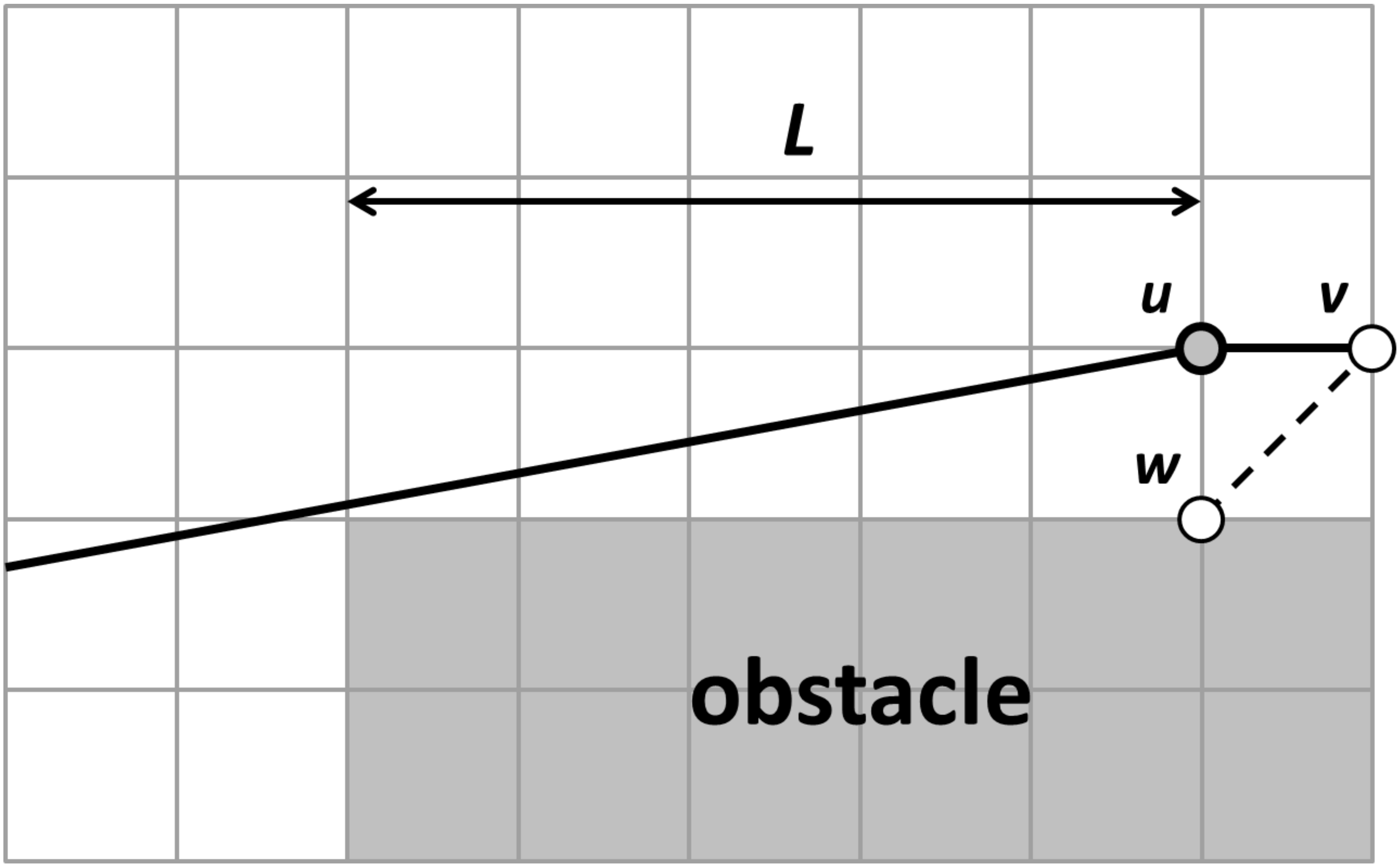


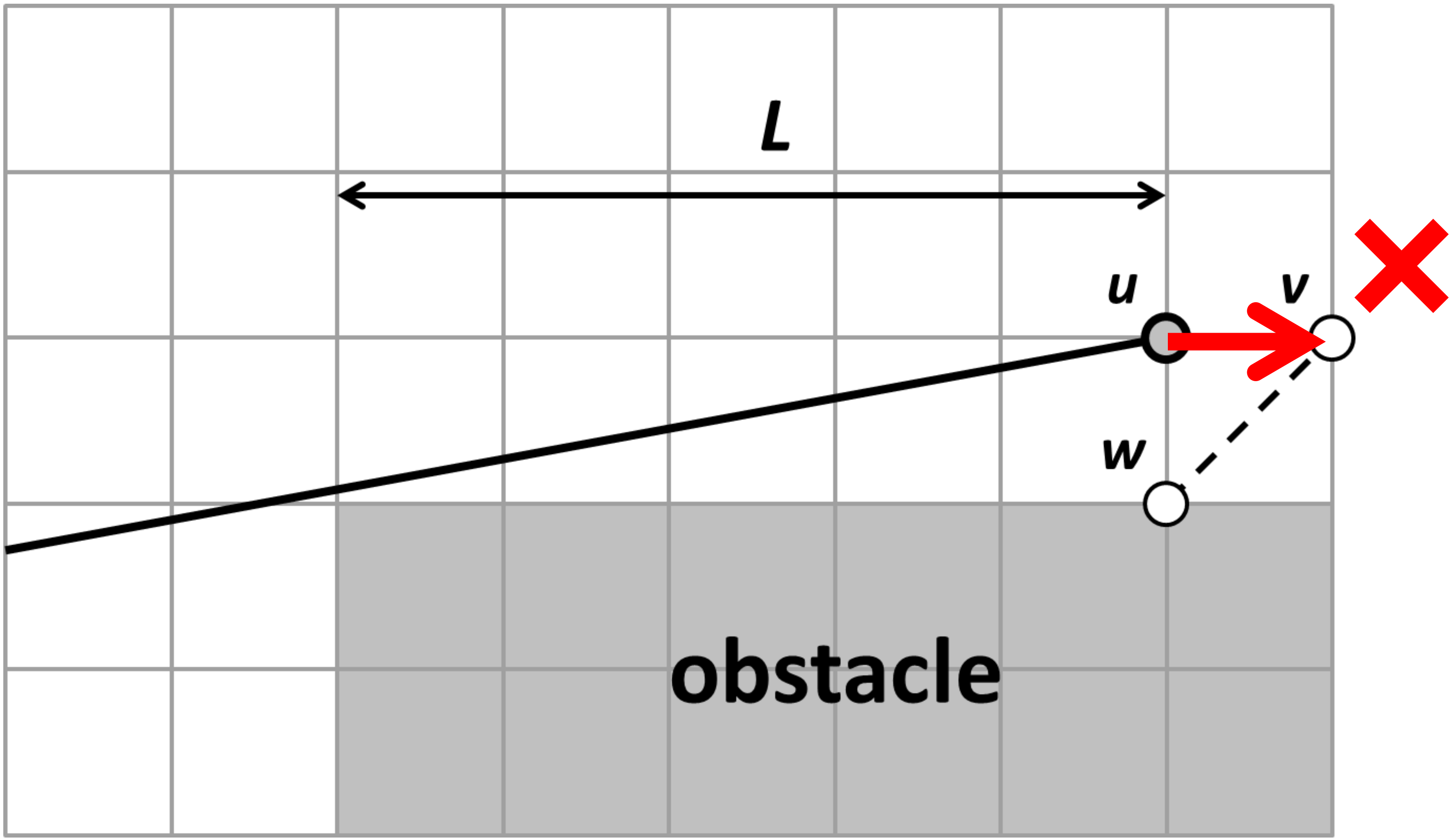
Path Length vs Penalty Value

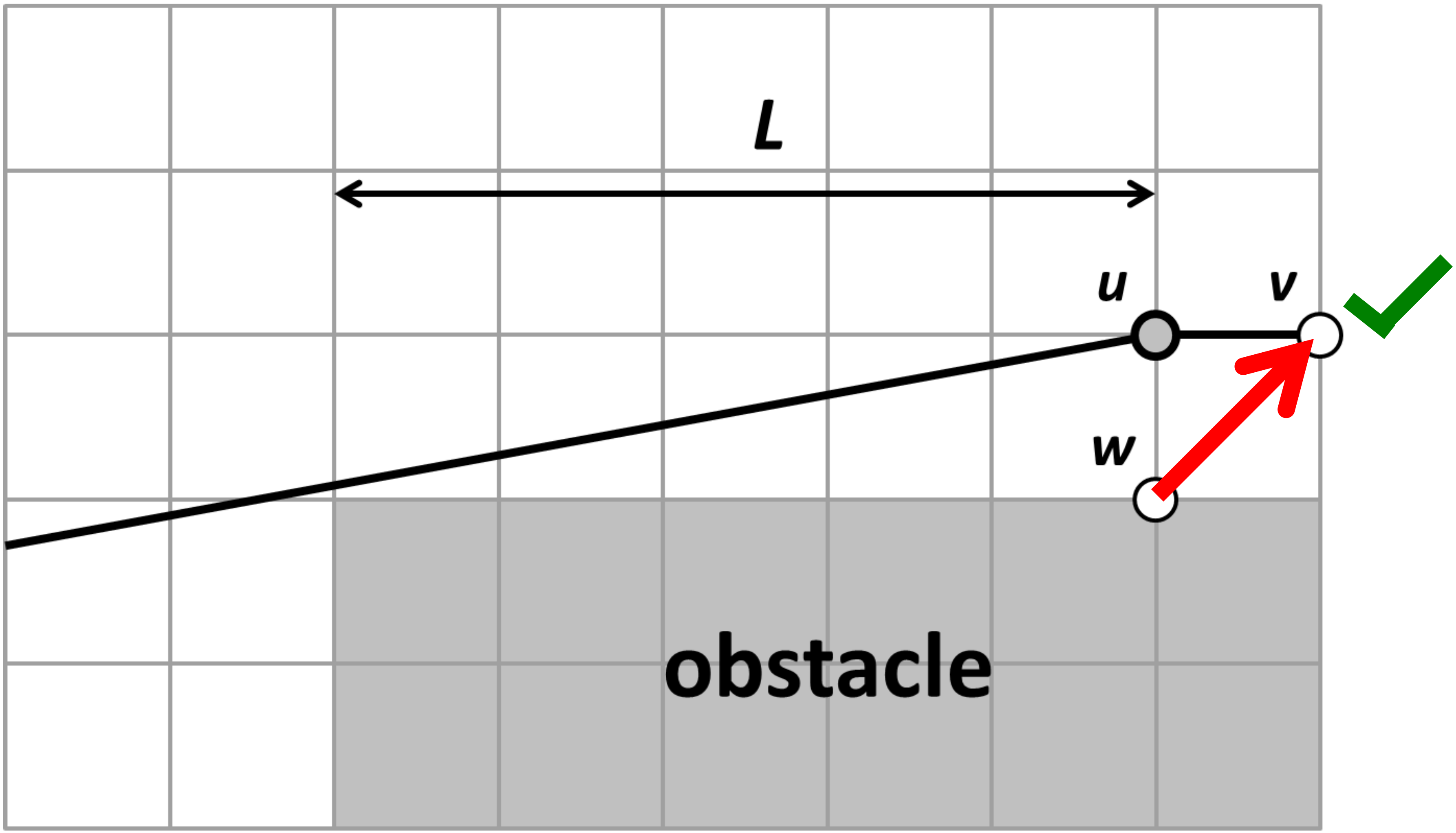


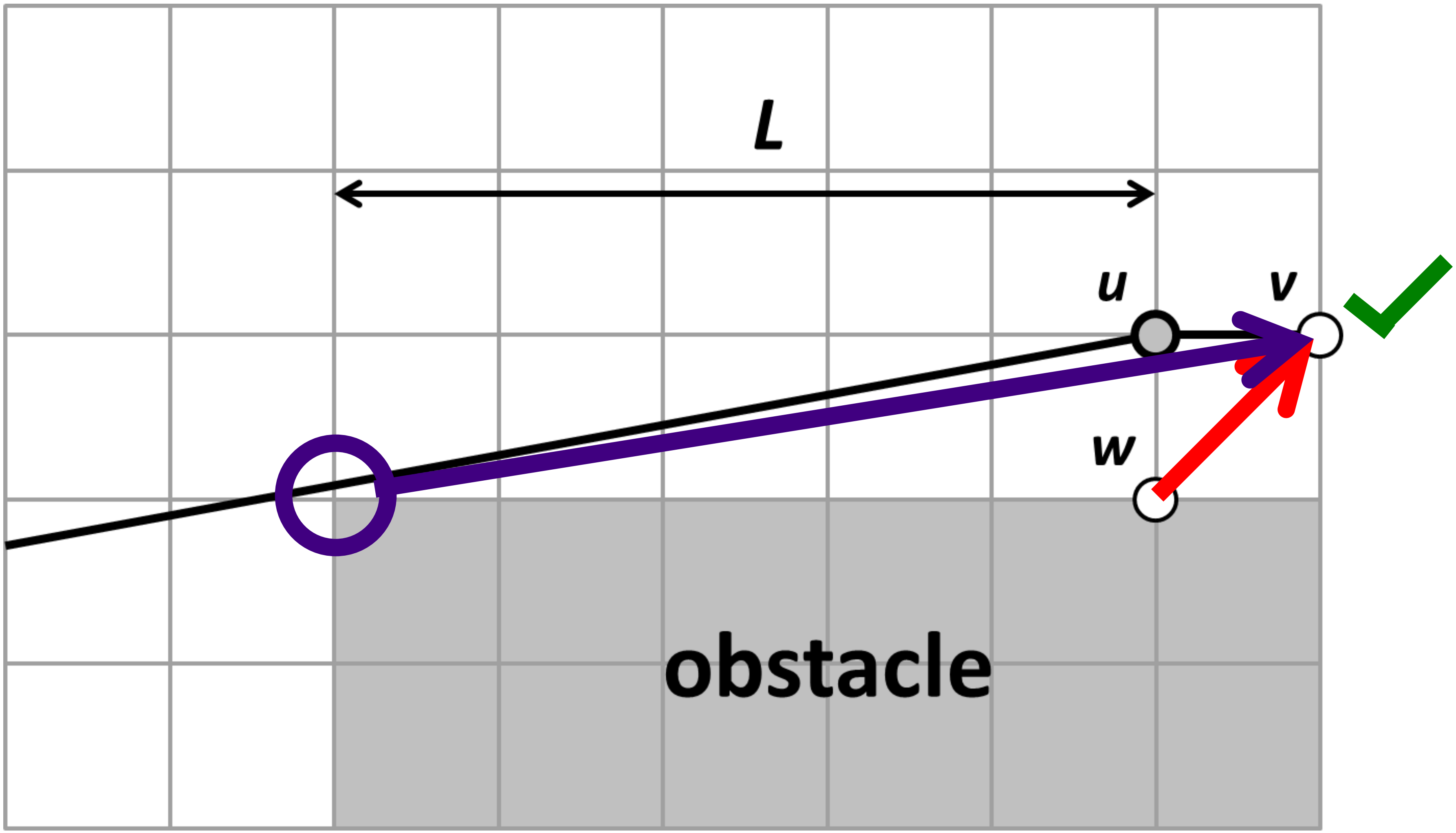
% Optimal/Taut vs Penalty Value

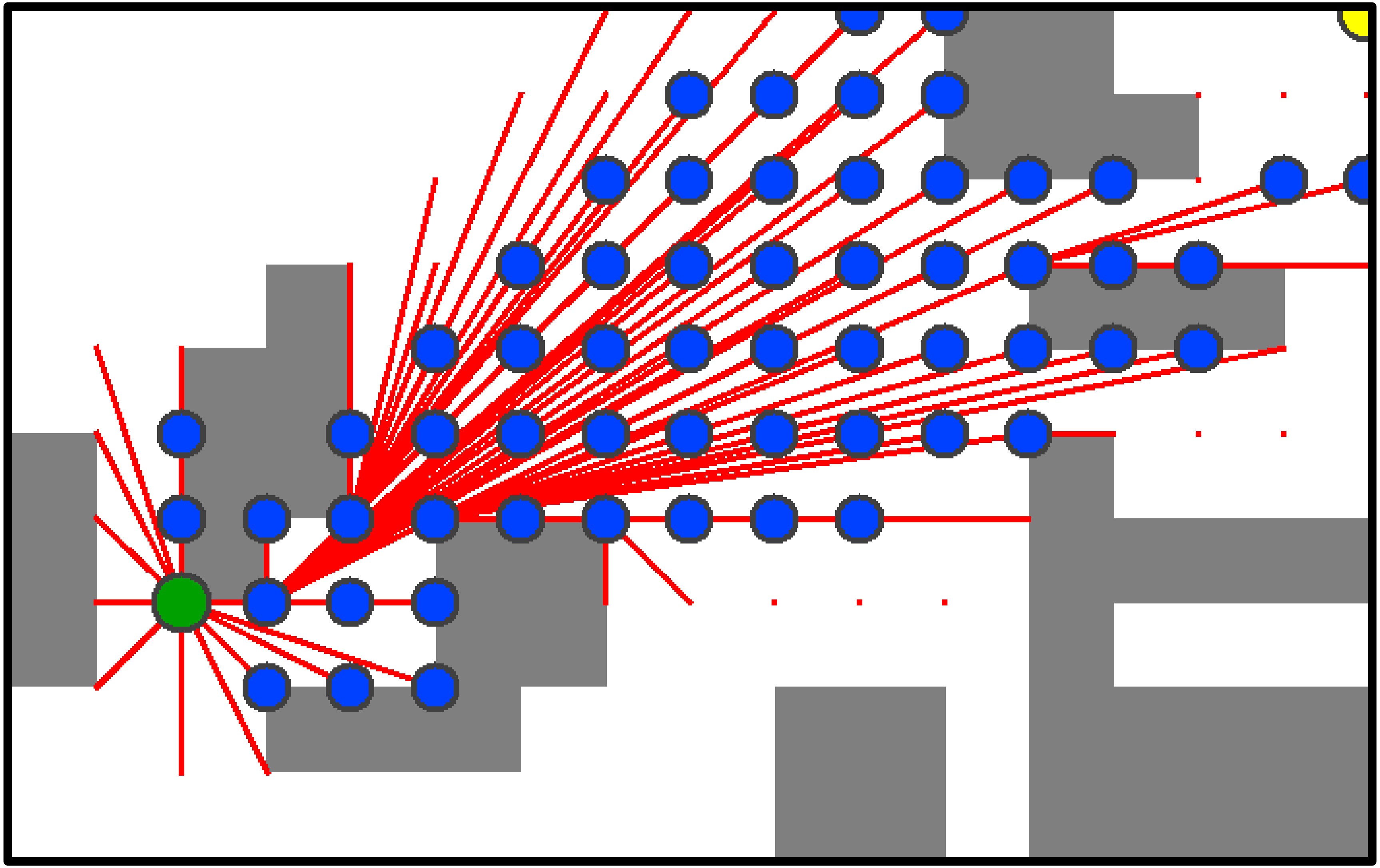



L
u
v
w
obstacle

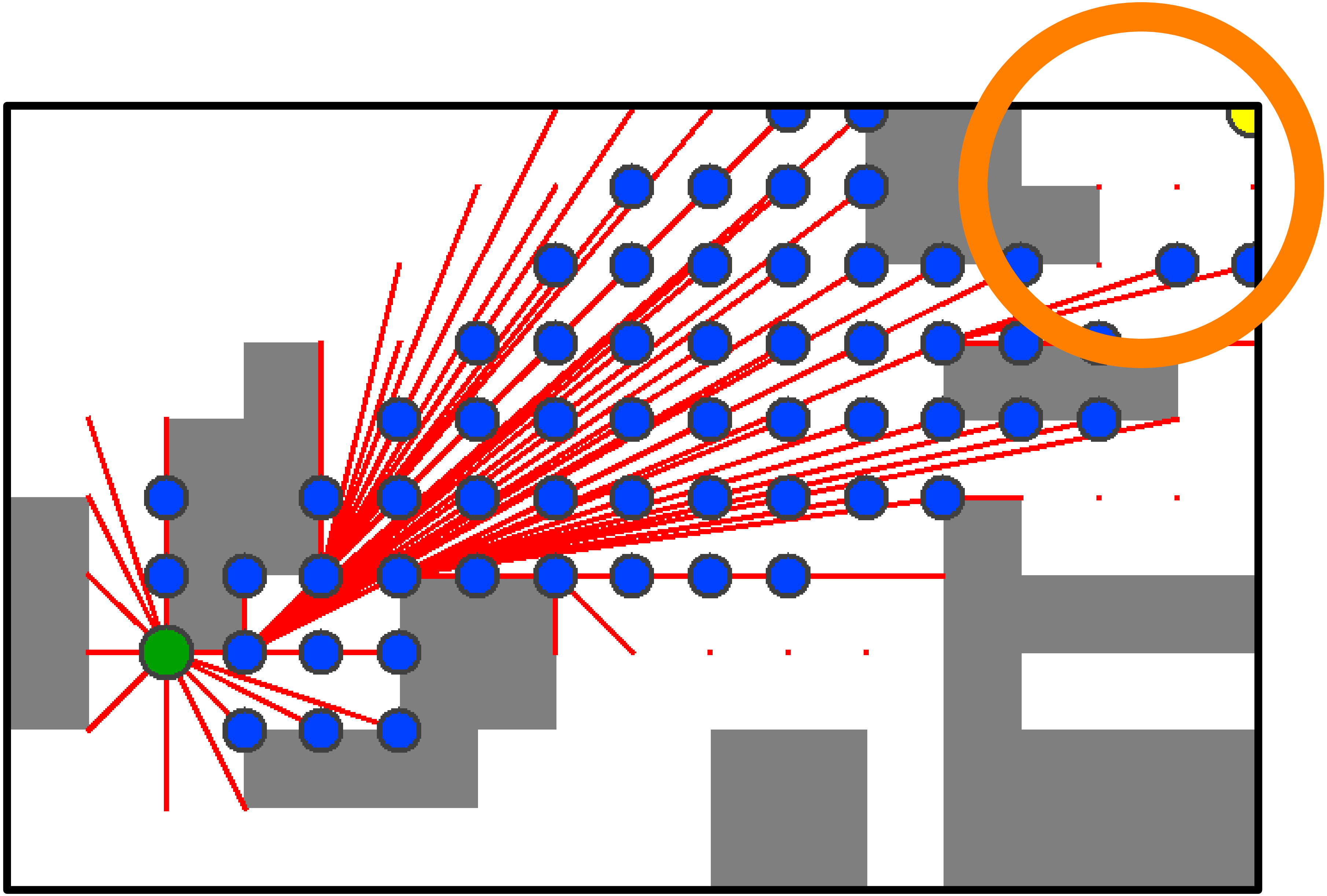

L
u
v
w
obstacle


L
u
v
w
obstacle

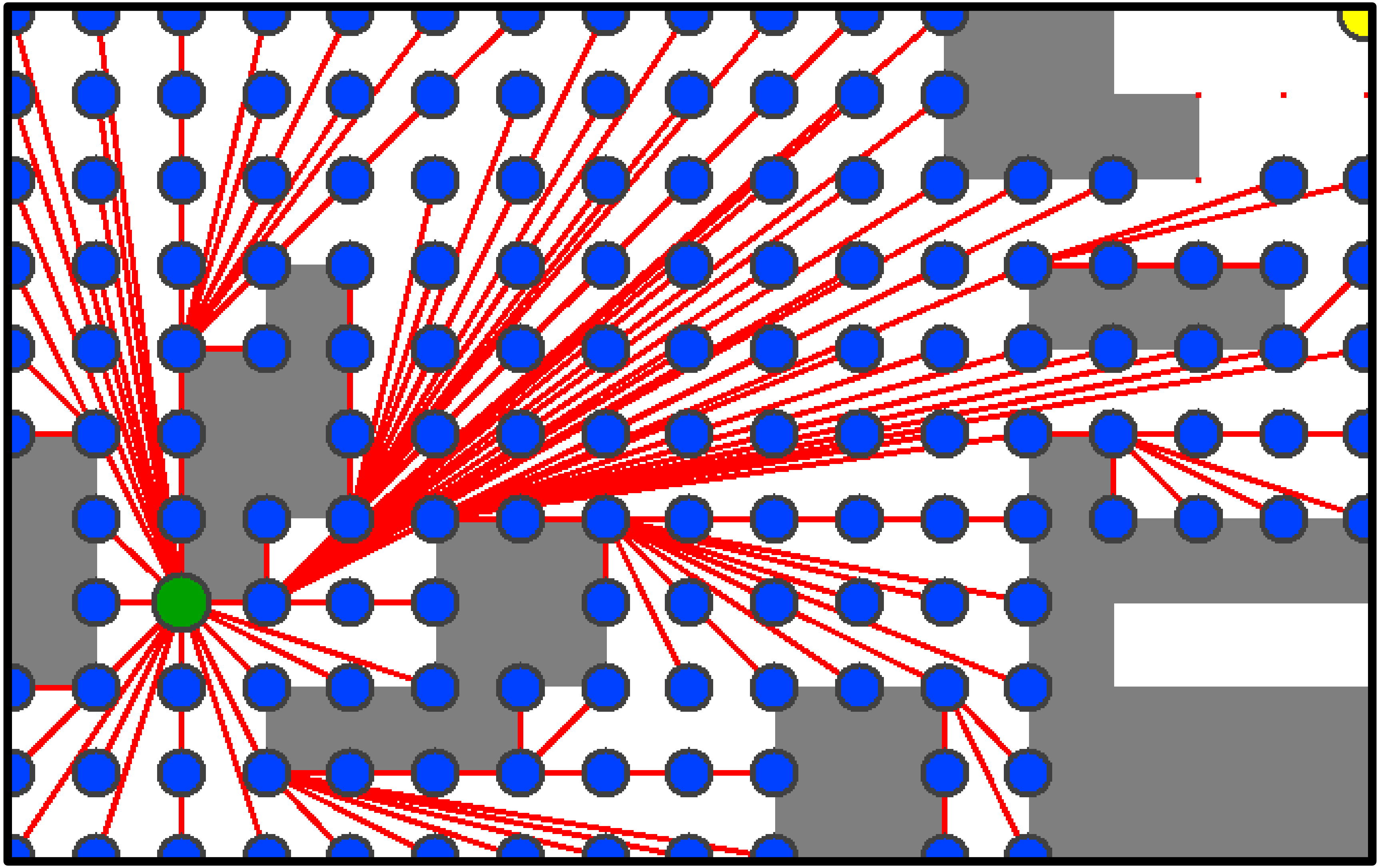

L
u
v
w
obstacle



Strict Theta*



Strict Theta*



Strict Theta*

ACKNOWLEDGEMENT

This presentation benefitted from

PowerPointLabs

a PowerPoint plugin for creating
better presentations with less effort.

PowerPointLabs
is available for free at
http://PowerPointLabs.info