Optimal Any-angle Pathfinding

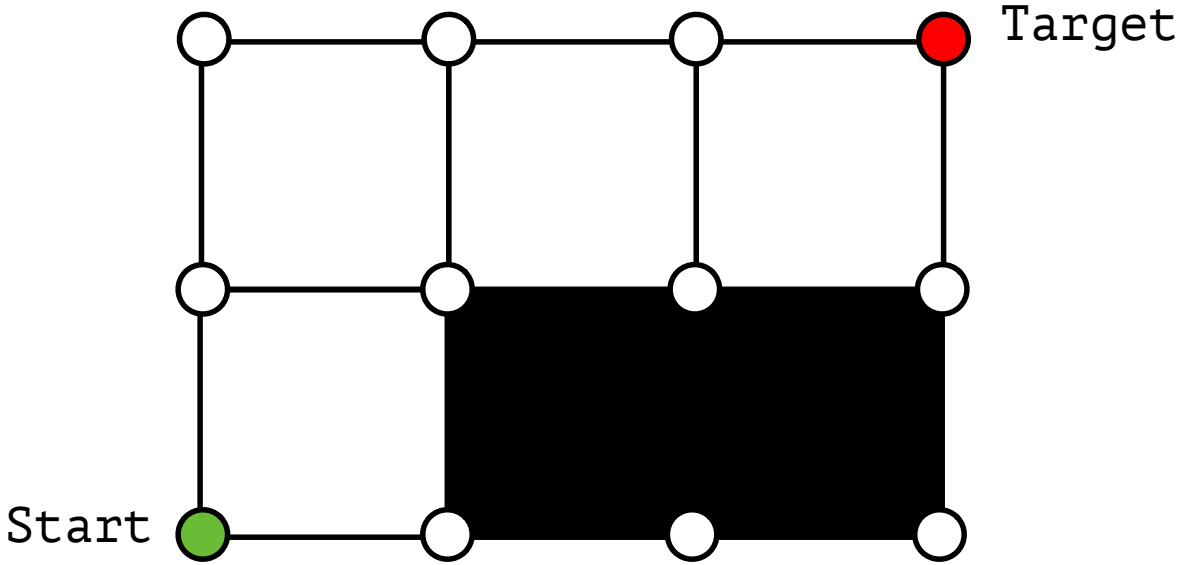
Daniel D. Harabor
Monash University
http://harabor.net/daniel

Joint work with:
Vural Aksakalli, Michael Cui, Alban Grastien and Dindar Öz

AAMAS Tutorial
2019-05-14

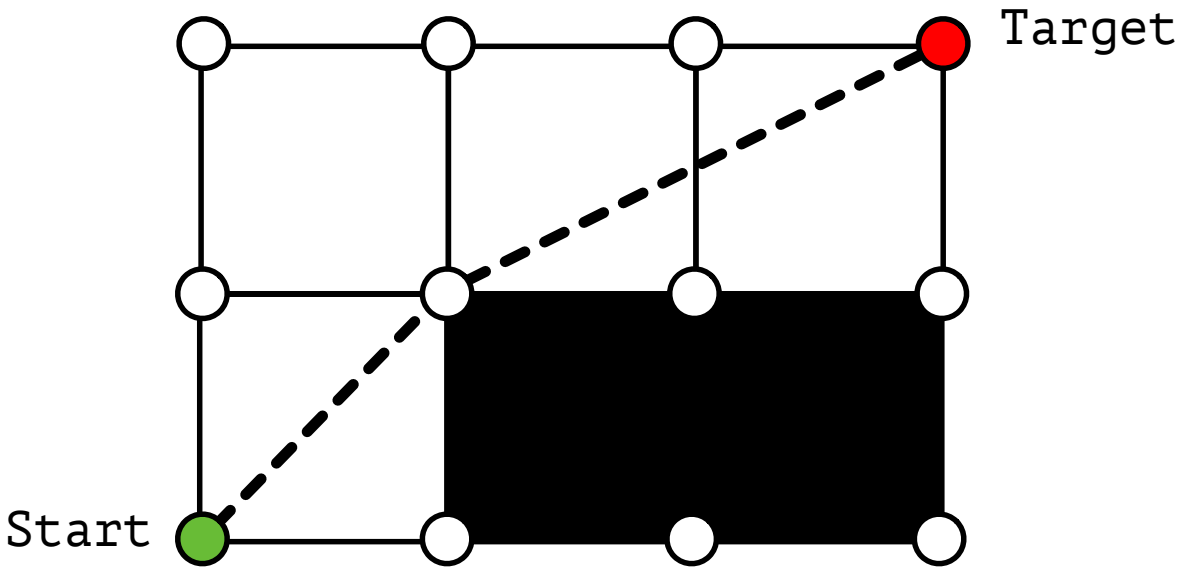
Suboptimality and Theta*

Theta* [Nash *et al.*, 2007] proceeds from one grid vertex to the next. This strategy is suboptimal since it expands nodes out of order.



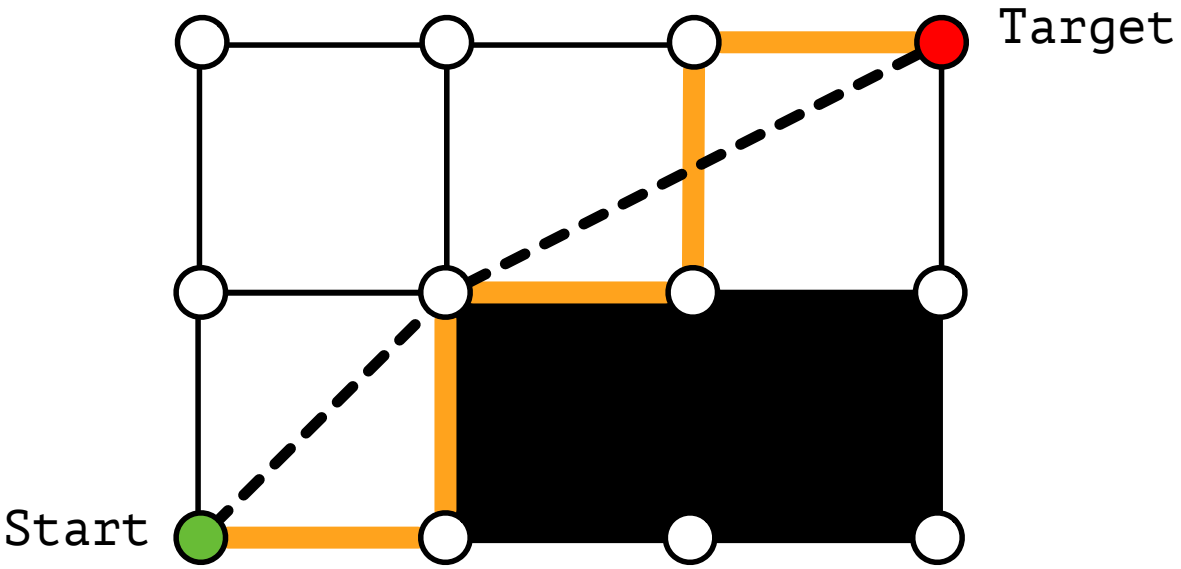
Suboptimality and Theta*

Theta* [Nash *et al.*, 2007] proceeds from one grid vertex to the next. This strategy is suboptimal since it expands nodes out of order.



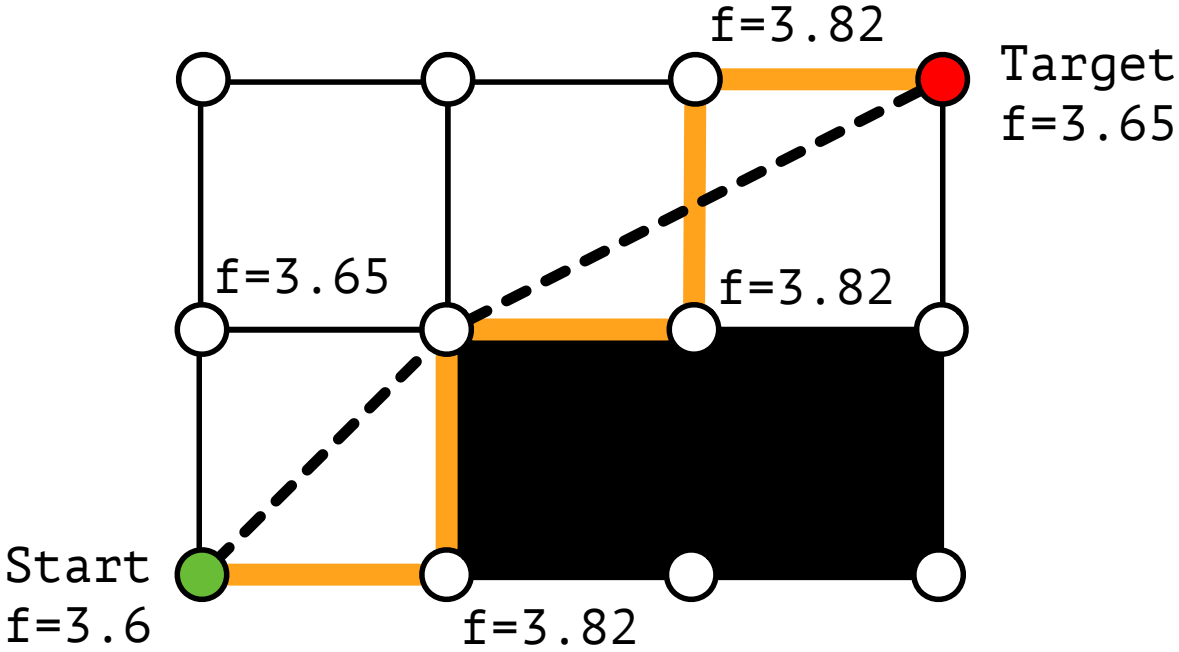
Suboptimality and Theta*

Theta* [Nash *et al.*, 2007] proceeds from one grid vertex to the next. This strategy is suboptimal since it expands nodes out of order.



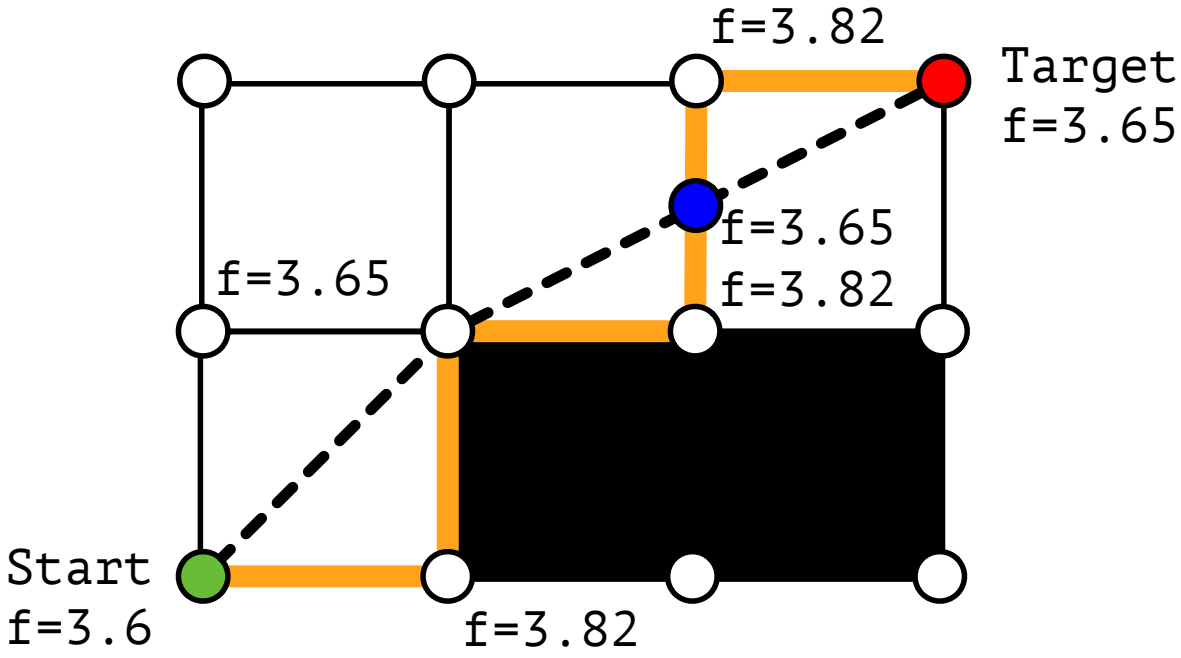
Suboptimality and Theta*

Theta* [Nash *et al.*, 2007] proceeds from one grid vertex to the next. This strategy is suboptimal since it expands nodes out of order.



Suboptimality and Theta*

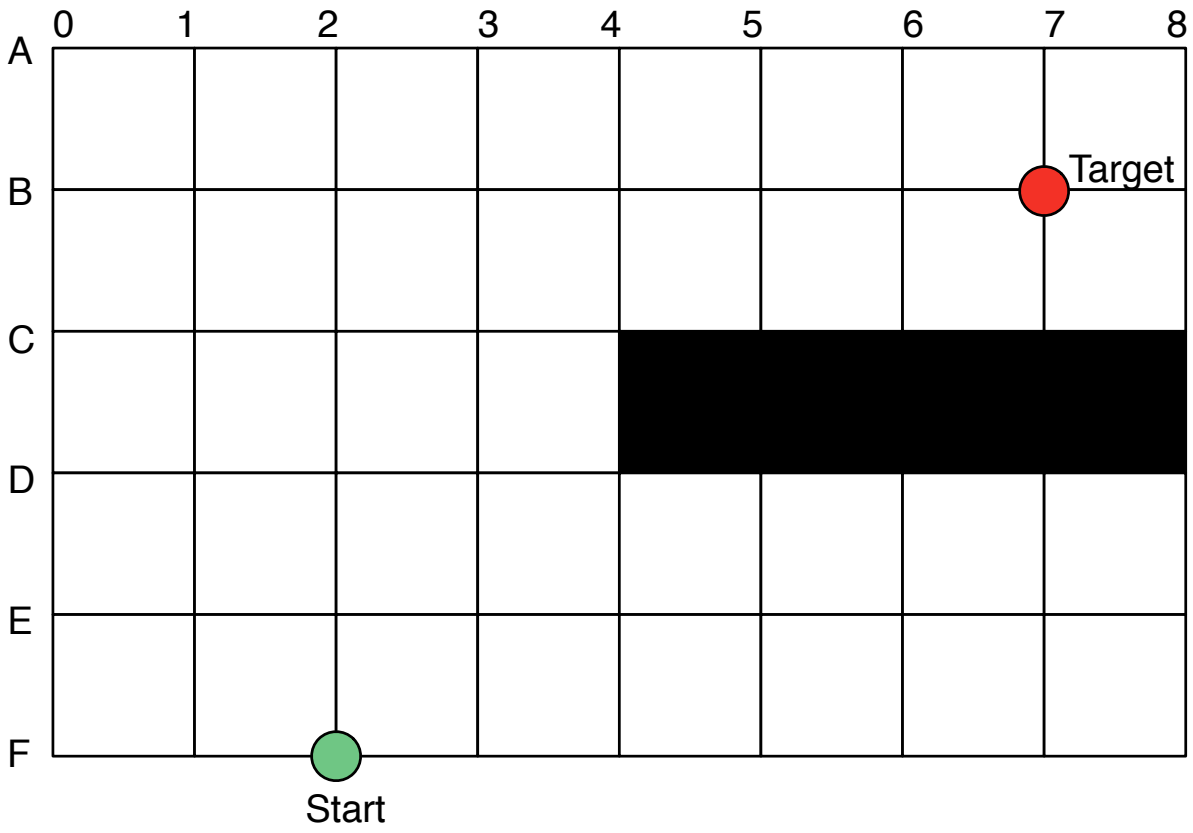
Theta* [Nash *et al.*, 2007] proceeds from one grid vertex to the next. This strategy is suboptimal since it expands nodes out of order.



Anya: An optimal any-angle pathfinding technique

Intuition

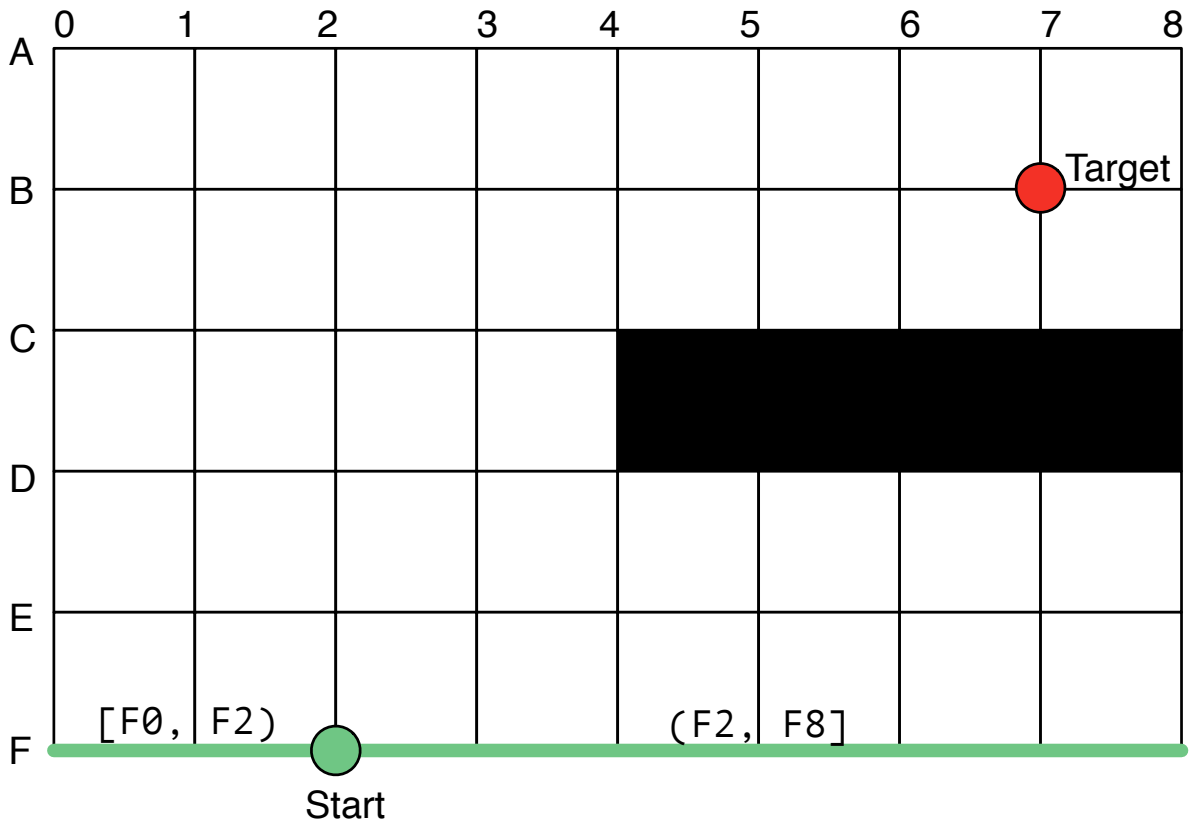
Expand *sets* of nodes together at one time [Harabor *et al.*, 2016]. A set is a contiguous intervals of points from along a single row.



Anya: An optimal any-angle pathfinding technique

Intuition

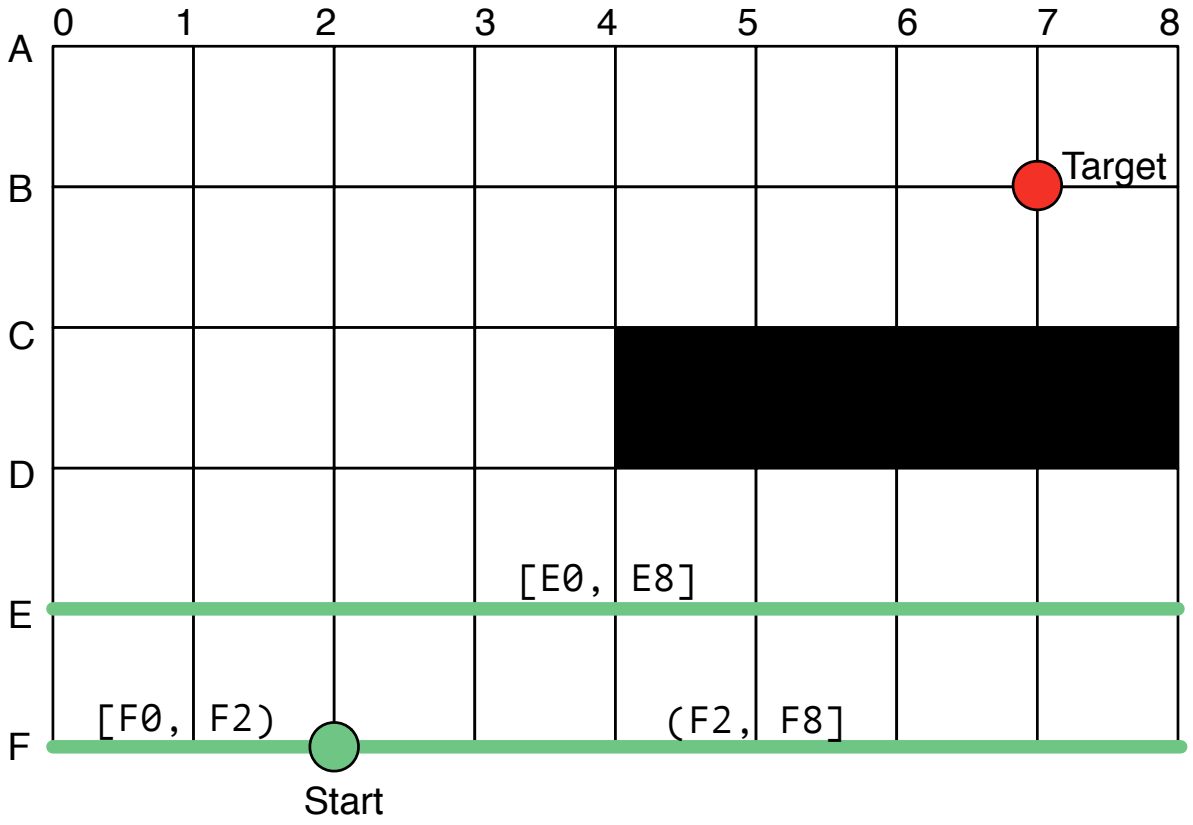
Expand *sets* of nodes together at one time [Harabor *et al.*, 2016]. A set is a contiguous intervals of points from along a single row.



Anya: An optimal any-angle pathfinding technique

Intuition

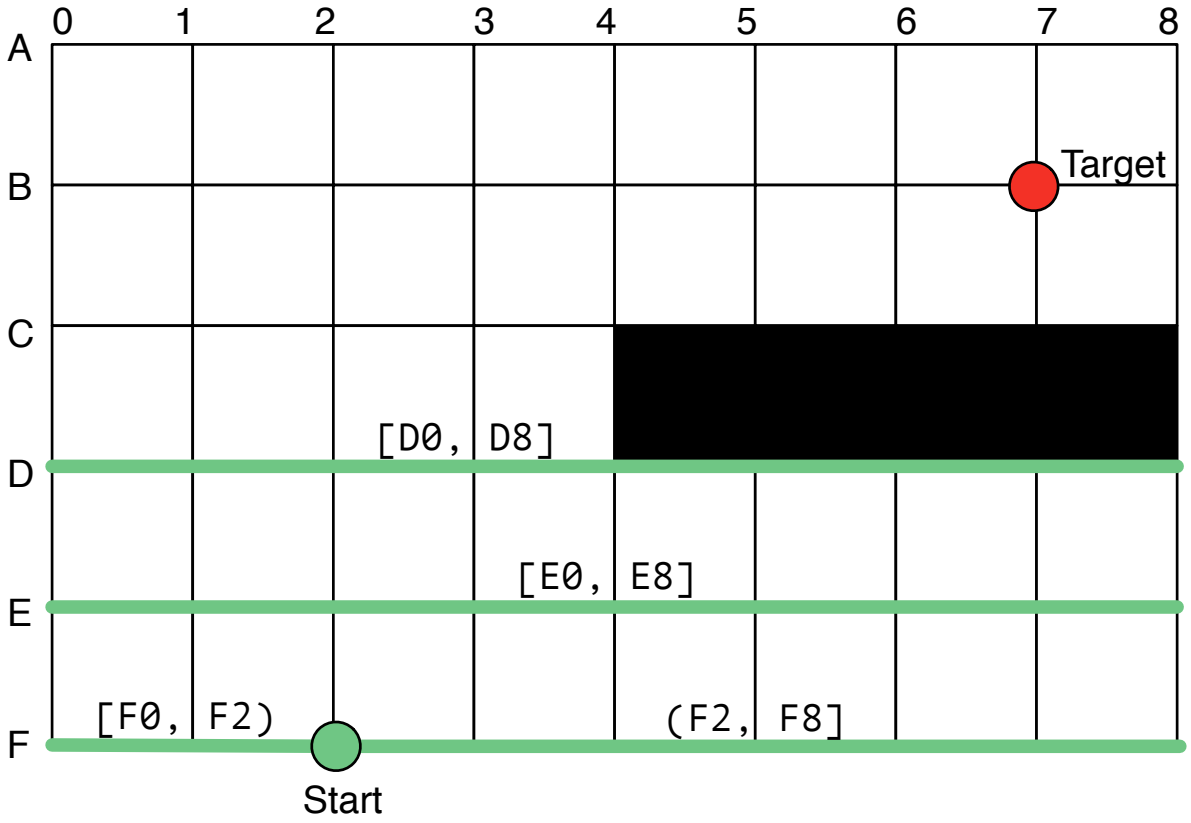
Expand *sets* of nodes together at one time [Harabor *et al.*, 2016]. A set is a contiguous intervals of points from along a single row.



Anya: An optimal any-angle pathfinding technique

Intuition

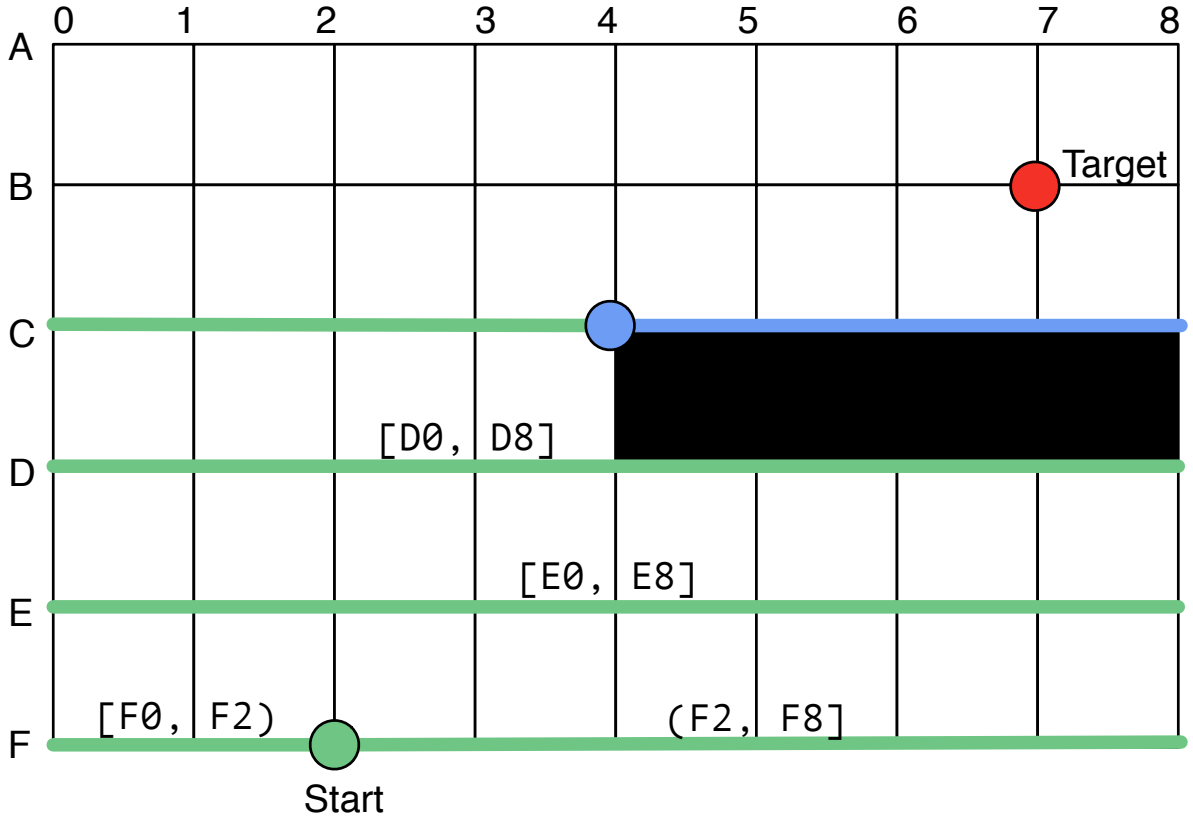
Expand *sets* of nodes together at one time [Harabor *et al.*, 2016]. A set is a contiguous intervals of points from along a single row.



Anya: An optimal any-angle pathfinding technique

Intuition

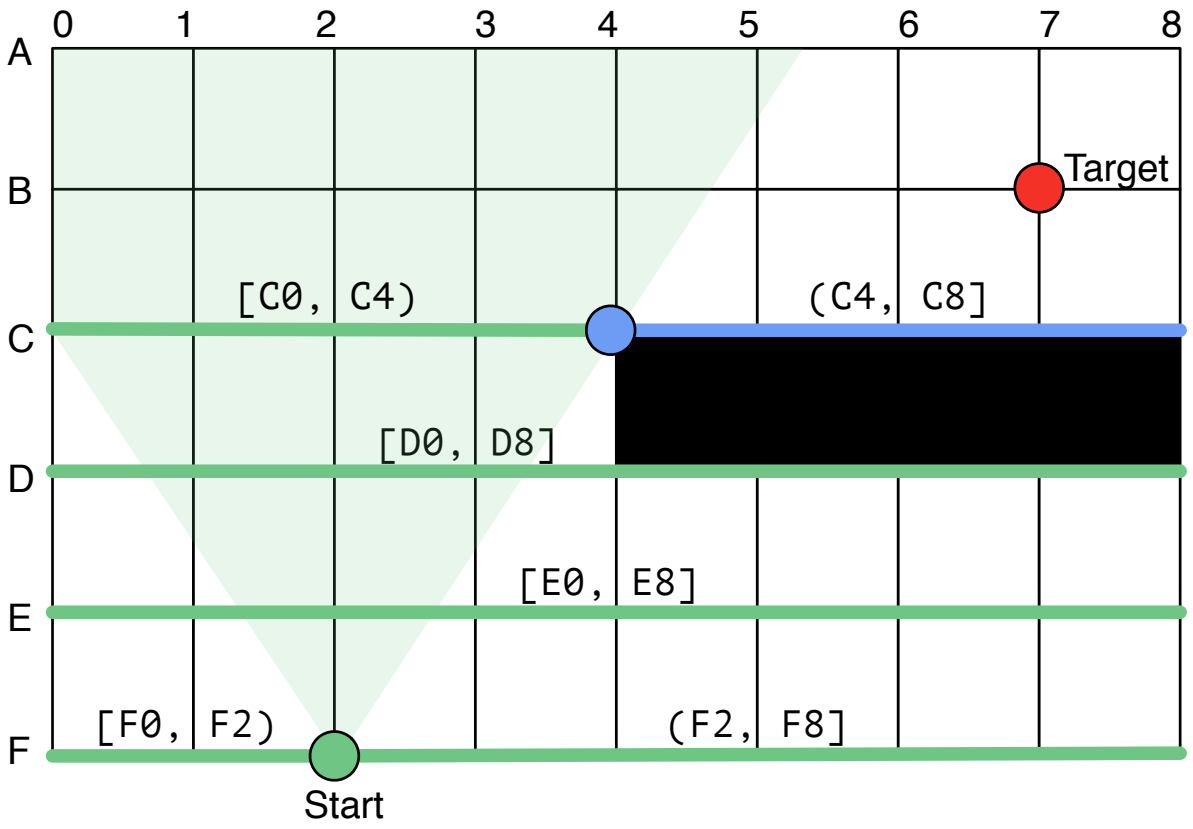
Expand *sets* of nodes together at one time [Harabor *et al.*, 2016]. A set is a contiguous intervals of points from along a single row.



Anya: An optimal any-angle pathfinding technique

Intuition

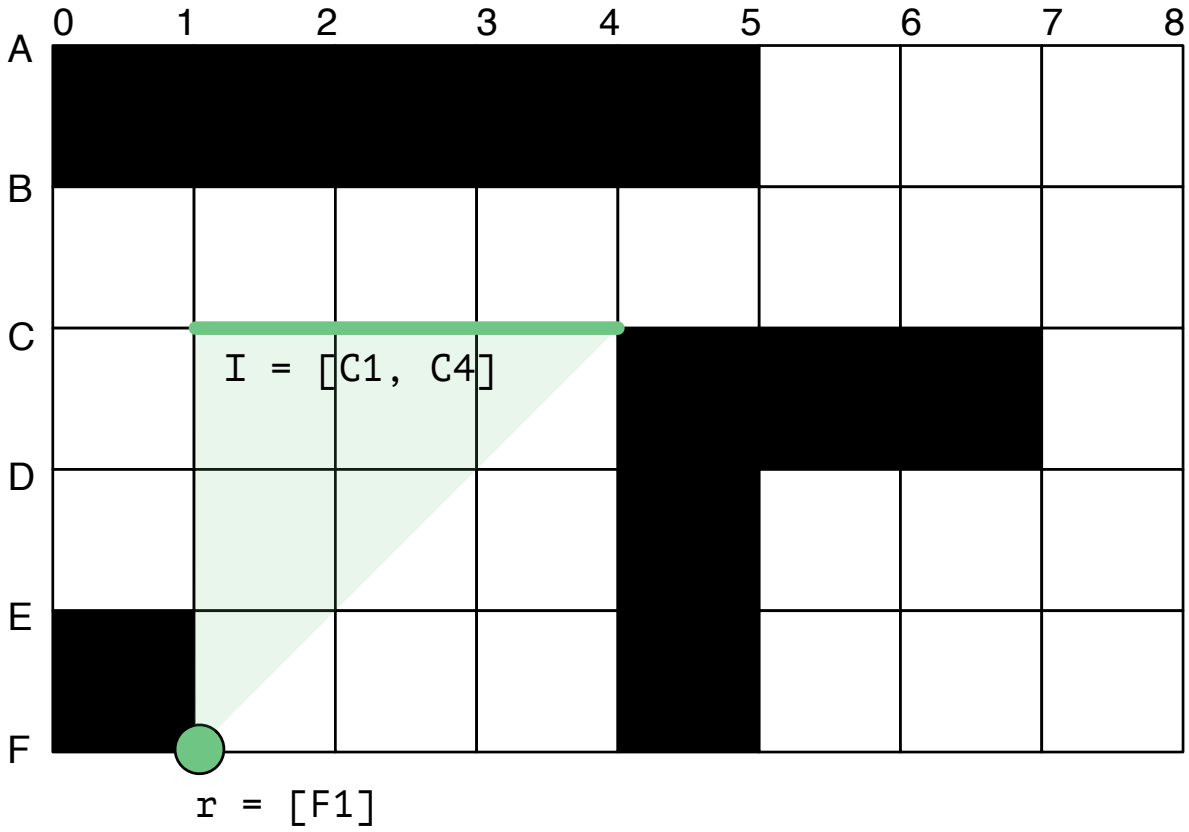
Expand *sets* of nodes together at one time [Harabor *et al.*, 2016]. A set is a contiguous intervals of points from along a single row.



Definition #1: Search Nodes

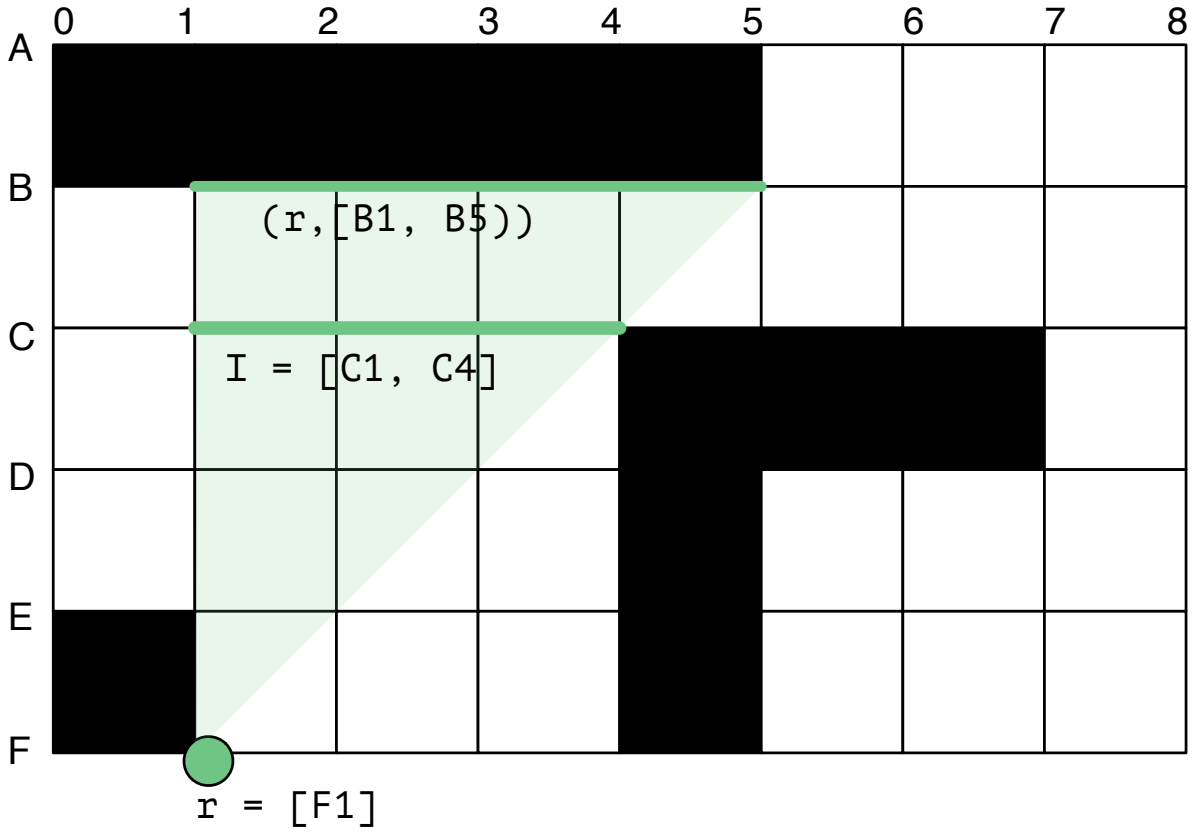
Every node is a tuple (I, r) where:

- r is a *root*; the most recent turning point.
- I is an interval of contiguous points, all visible from r.
- The *start node* has a point interval and a root "off the grid"



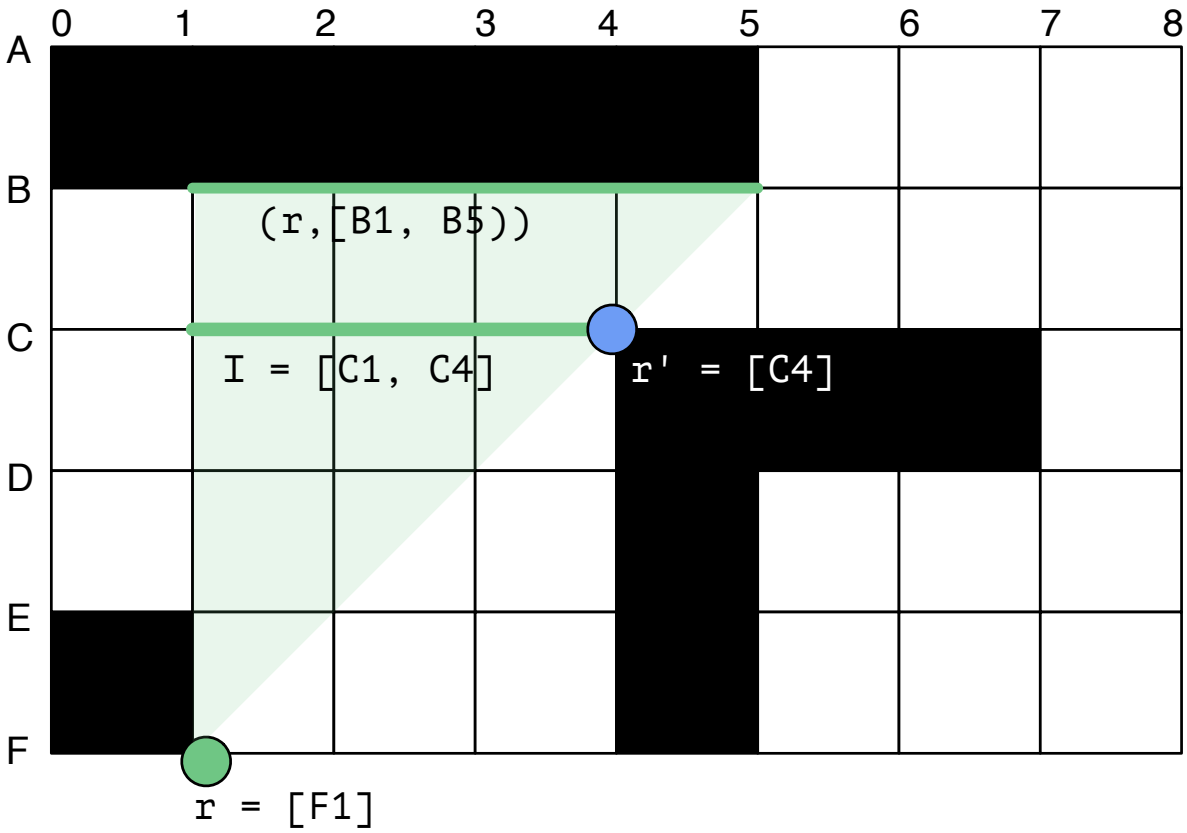
Definition #2: Successors

- Successors of node (I, r) are found by traveling from r and through I along a locally taut path.
- Two kinds of successors: *observable* and *non-observable*



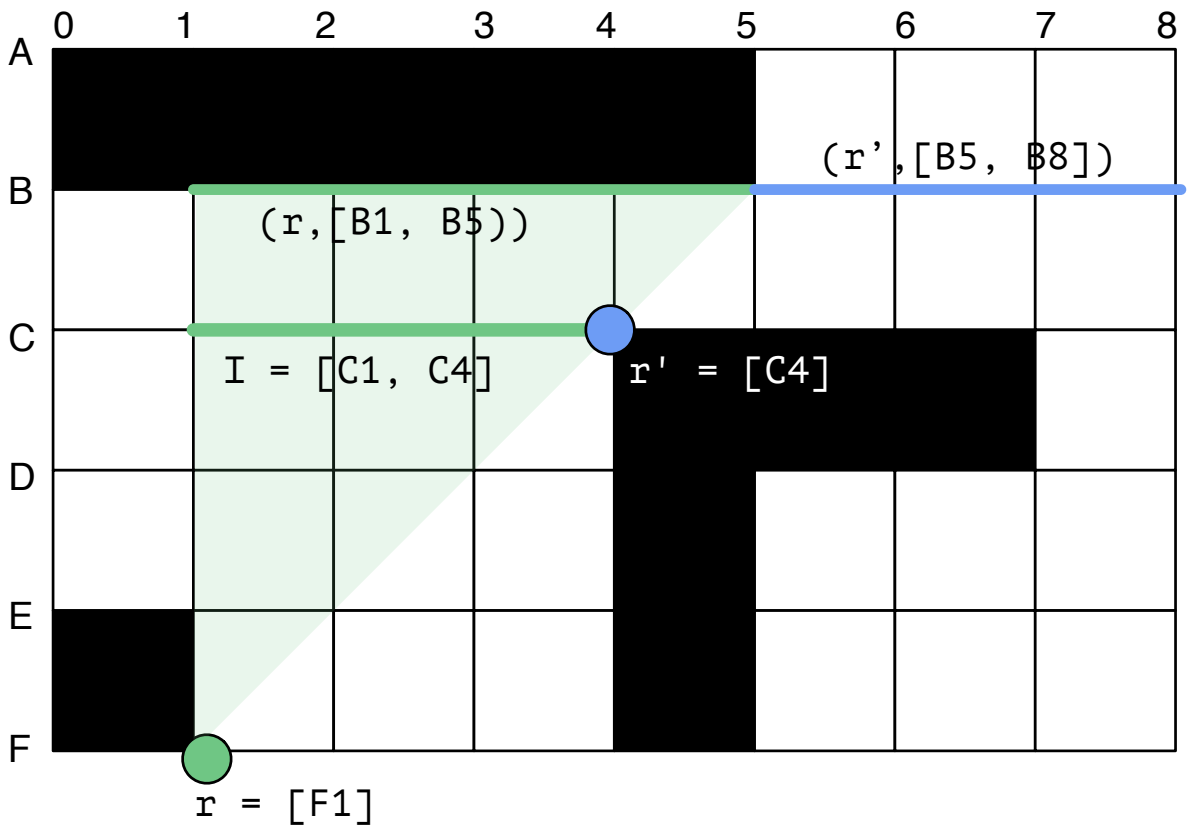
Definition #2: Successors

- Successors of node (I, r) are found by traveling from r and through I along a locally taut path.
- Two kinds of successors: *observable* and *non-observable*



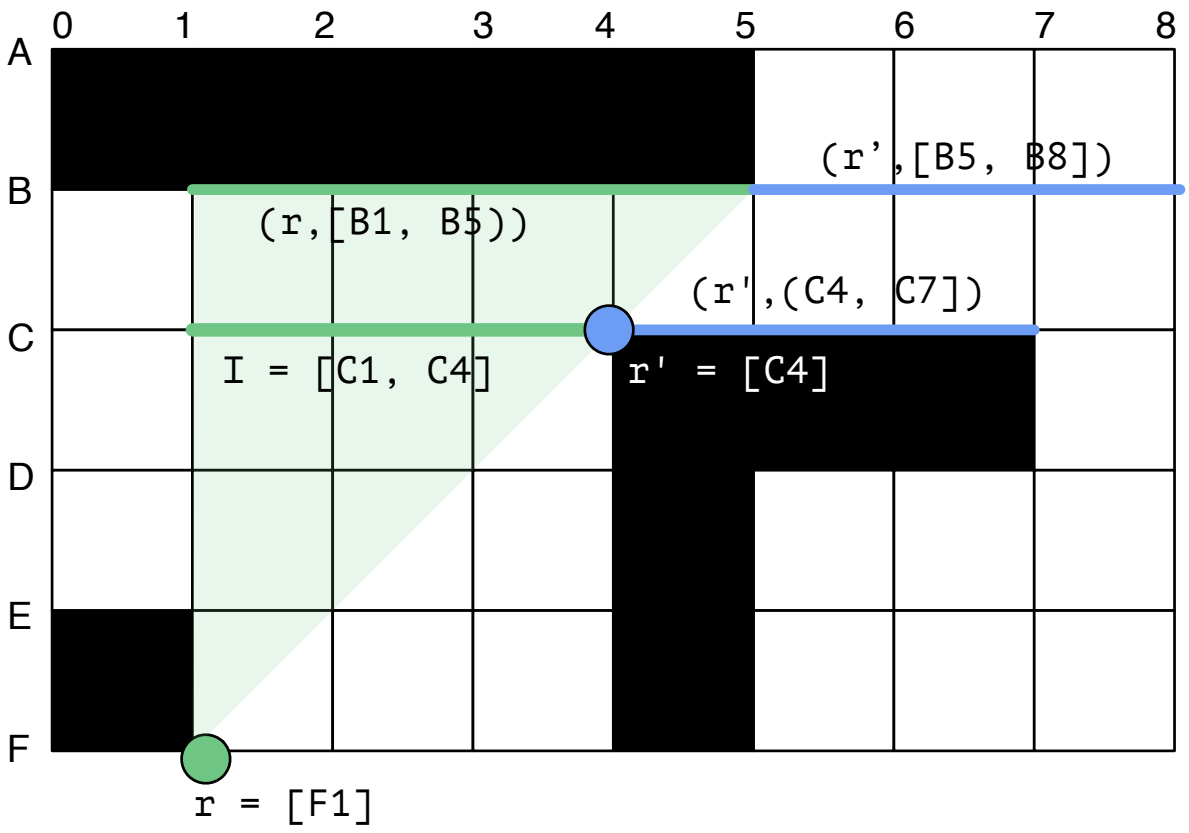
Definition #2: Successors

- Successors of node (I, r) are found by traveling from r and through I along a locally taut path.
- Two kinds of successors: *observable* and *non-observable*



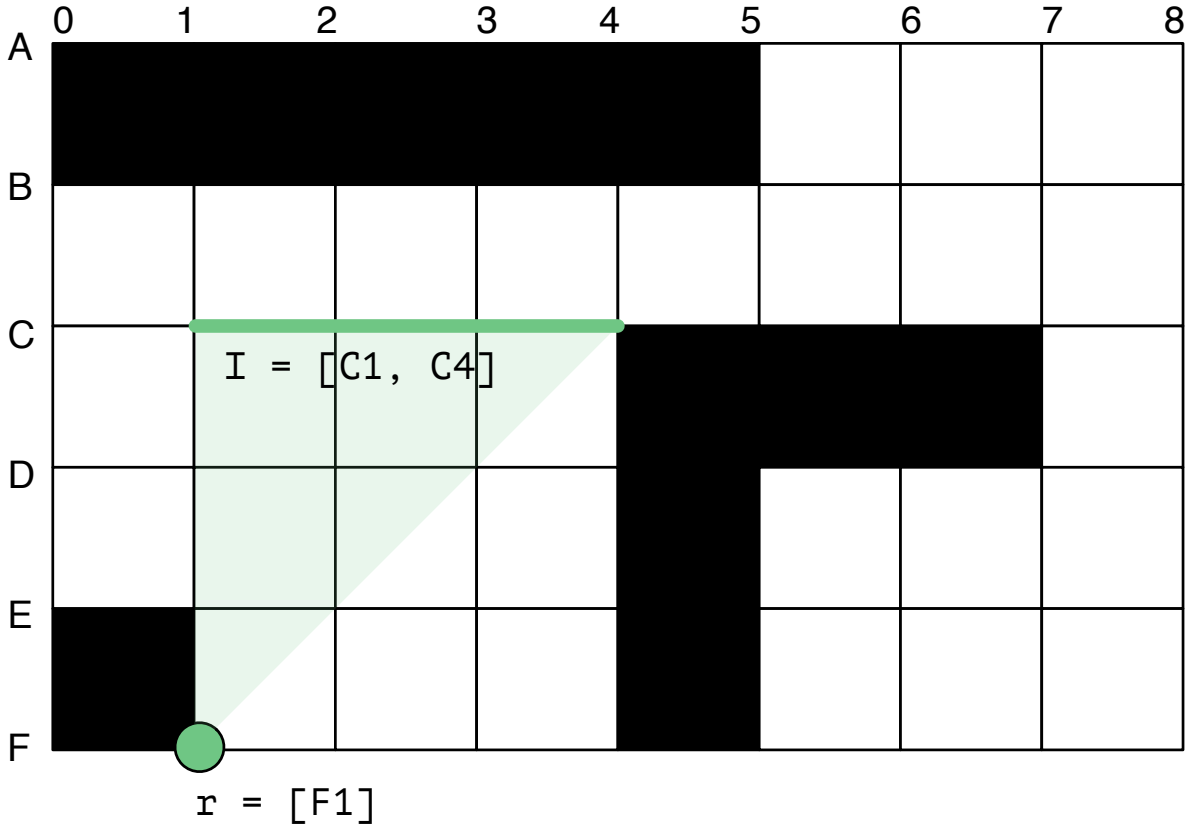
Definition #2: Successors

- Successors of node (I, r) are found by traveling from r and through I along a locally taut path.
- Two kinds of successors: *observable* and *non-observable*



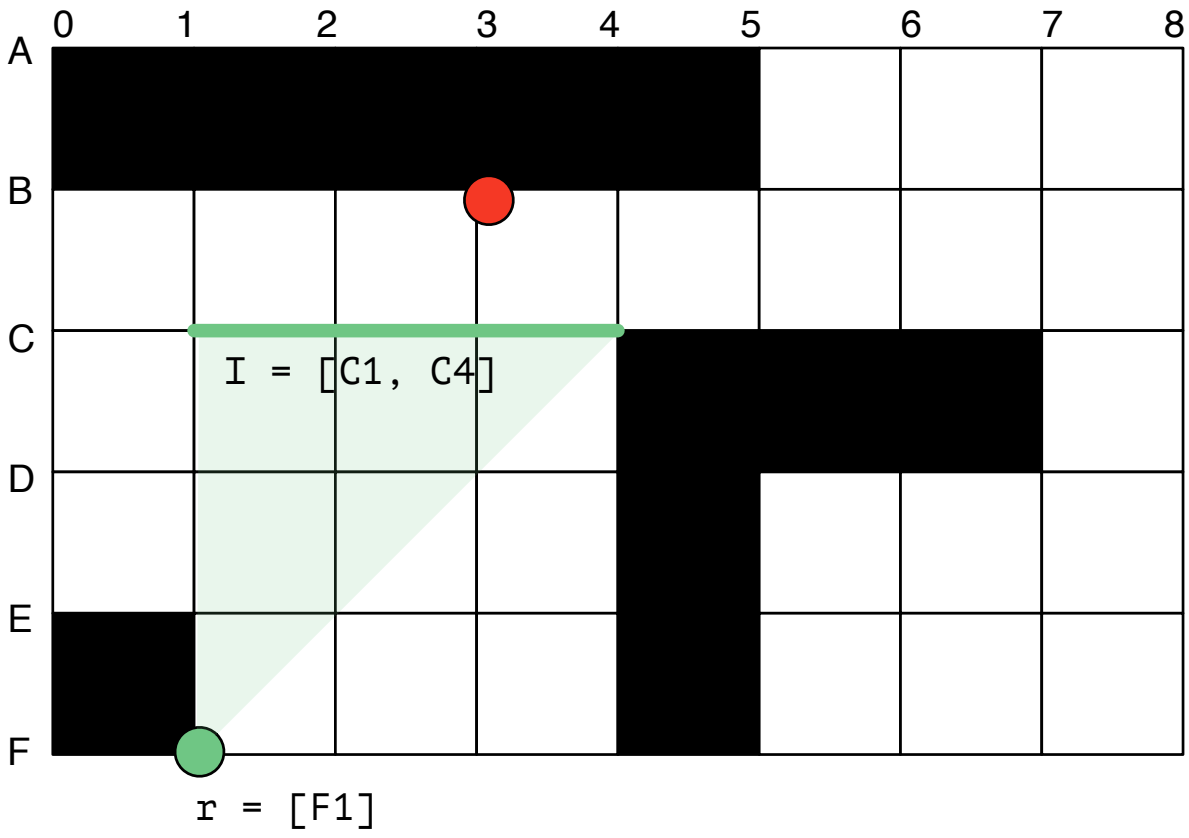
Evaluation Function

- From each interval I we choose a single point p which minimises the cost-to-go (i.e. the f-value of the node).



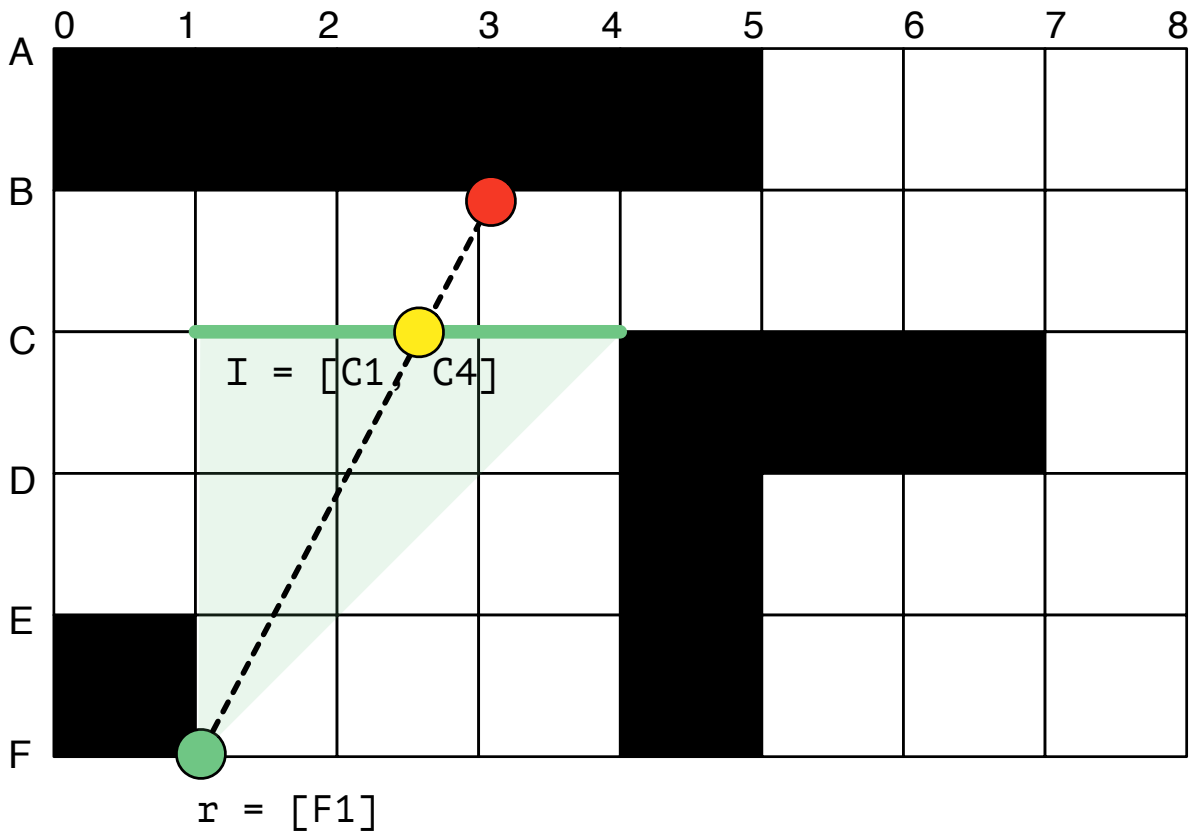
Evaluation Function

- From each interval I we choose a single point p which minimises the cost-to-go (i.e. the f-value of the node).



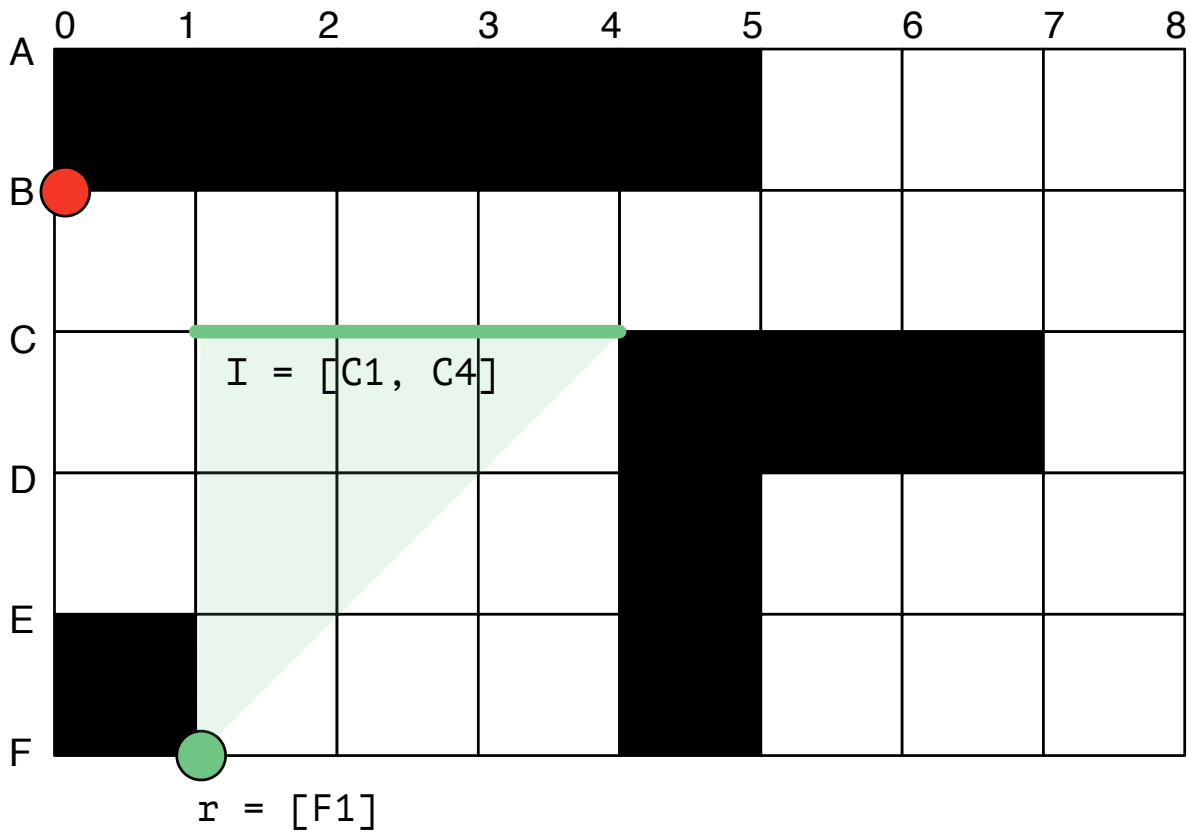
Evaluation Function

- From each interval I we choose a single point p which minimises the cost-to-go (i.e. the f-value of the node).



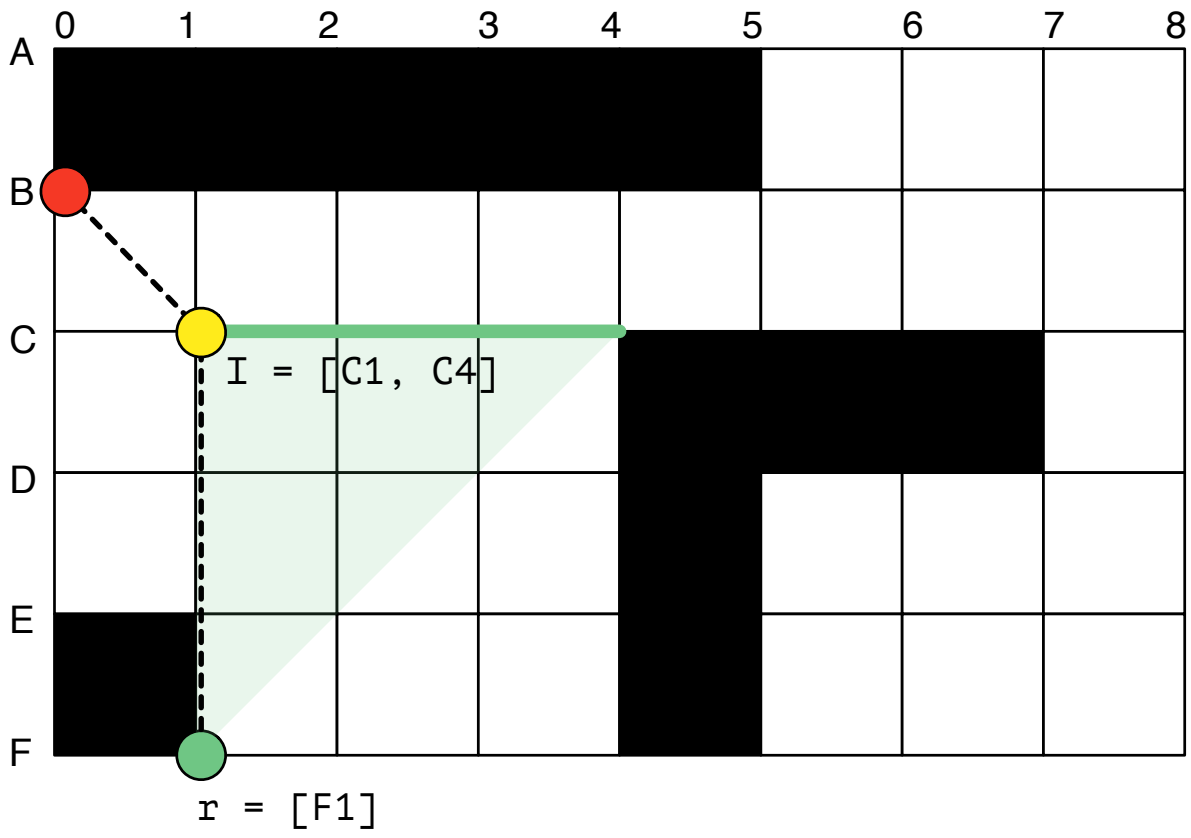
Evaluation Function

- From each interval I we choose a single point p which minimises the cost-to-go (i.e. the f-value of the node).



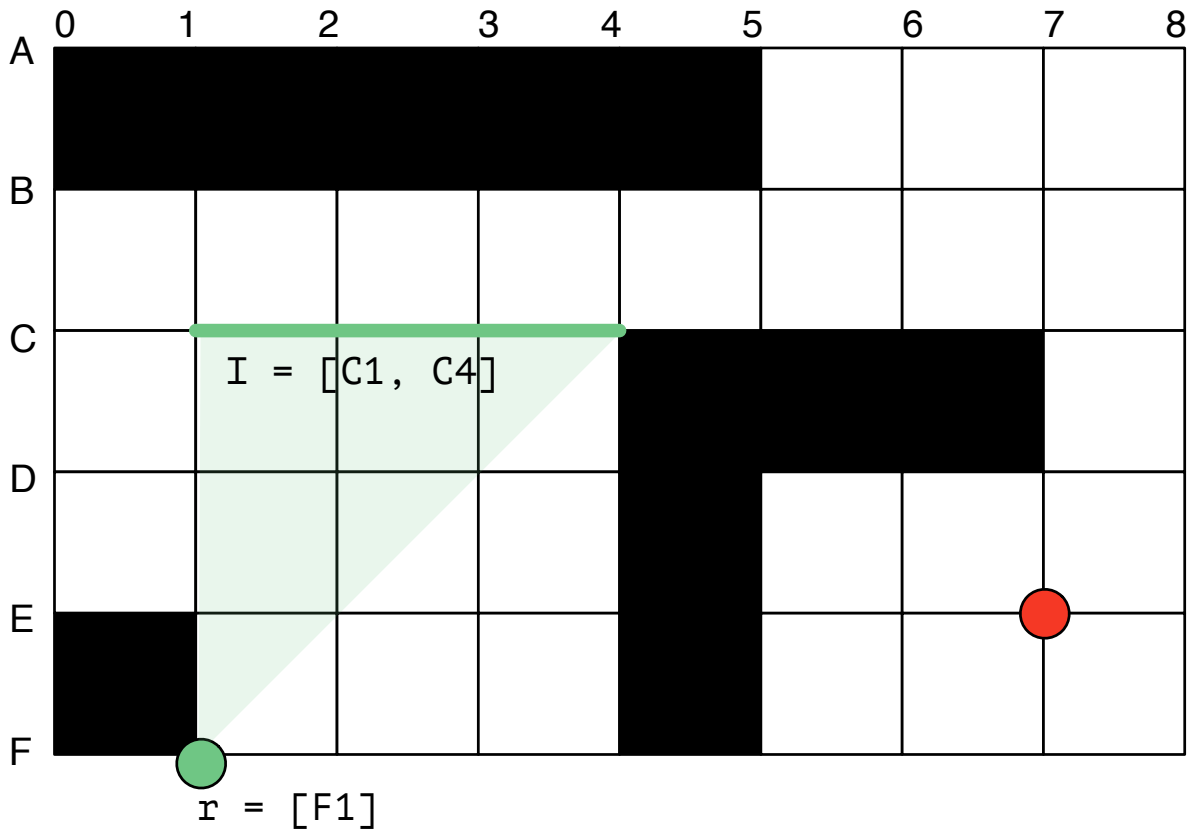
Evaluation Function

- From each interval I we choose a single point p which minimises the cost-to-go (i.e. the f-value of the node).



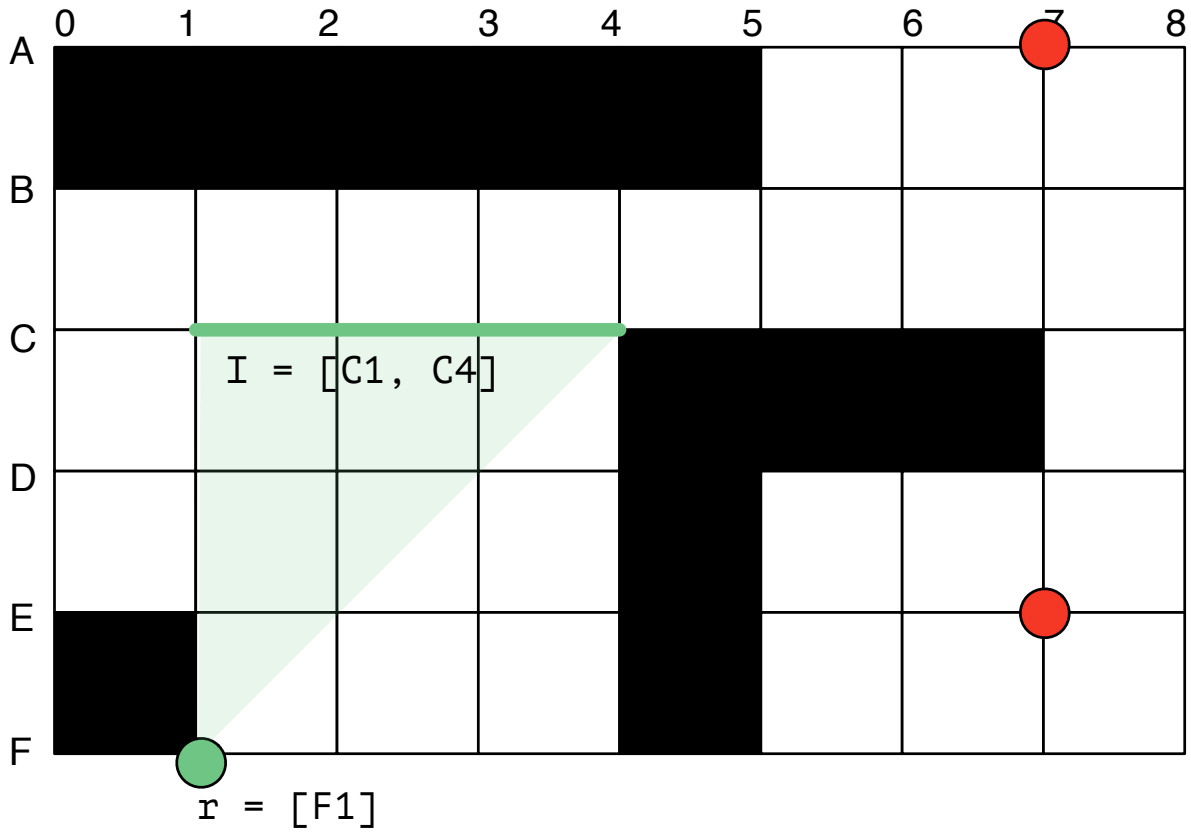
Evaluation Function

- From each interval I we choose a single point p which minimises the cost-to-go (i.e. the f-value of the node).



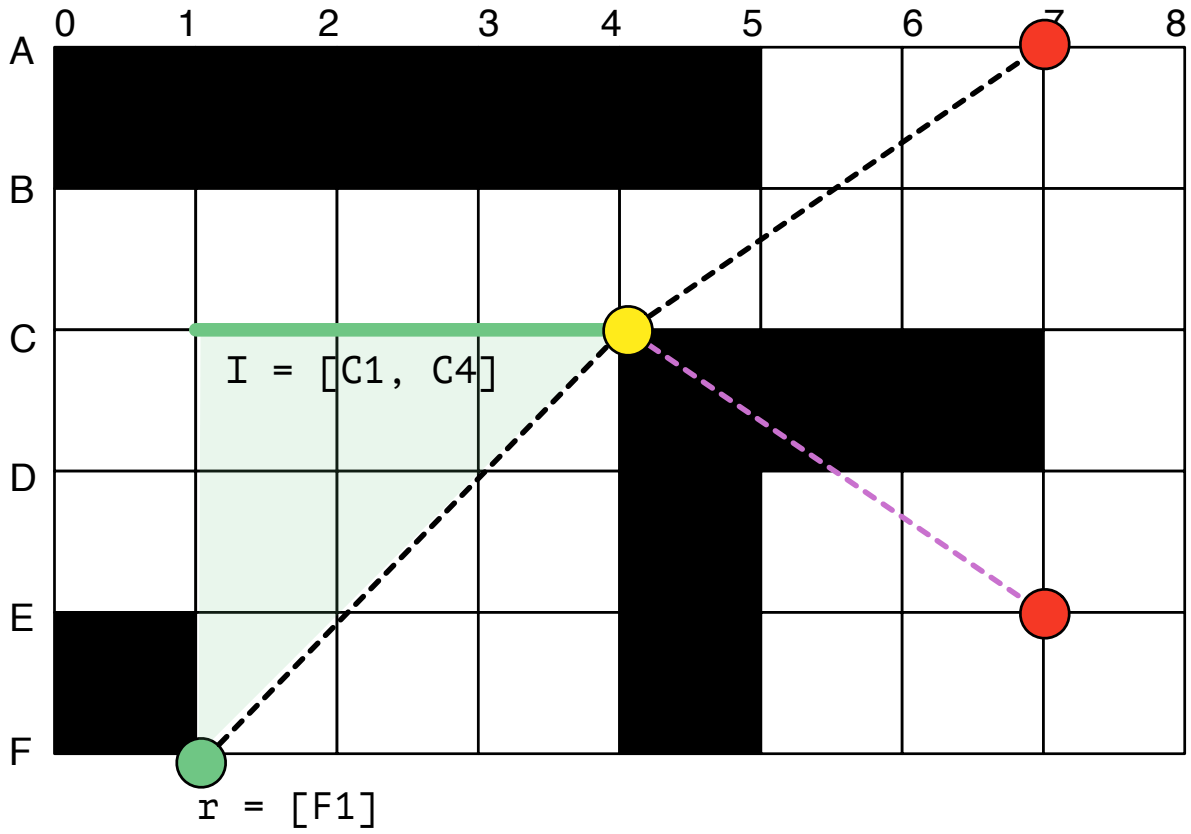
Evaluation Function

- From each interval I we choose a single point p which minimises the cost-to-go (i.e. the f-value of the node).



Evaluation Function

- From each interval I we choose a single point p which minimises the cost-to-go (i.e. the f-value of the node).



Theoretical properties

Completeness (Sketch)

- Every point is a corner or belongs to an interval.
- Every interval is visible from some predecessor.

Optimality (Sketch)

- Each representative point has a minimum f-value.
- The f-value of each successor is monotonically increasing.
- A node whose interval contains the target is eventually expanded.

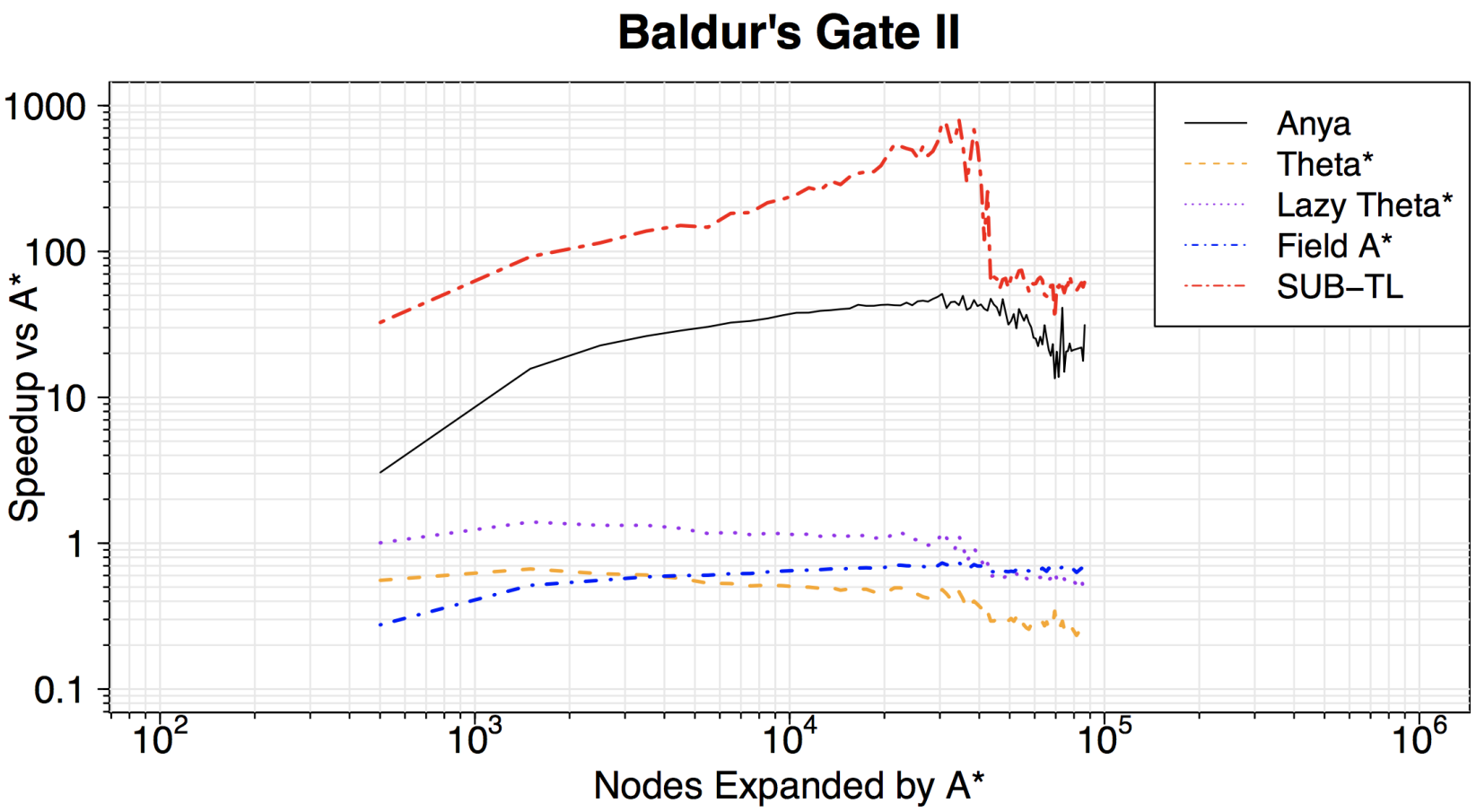
Online

Each search is performed entirely online and without reference to any pre-computed data structures or heuristics.

Full technical details in [Harabor *et al.*, 2016].

Results on Games Maps

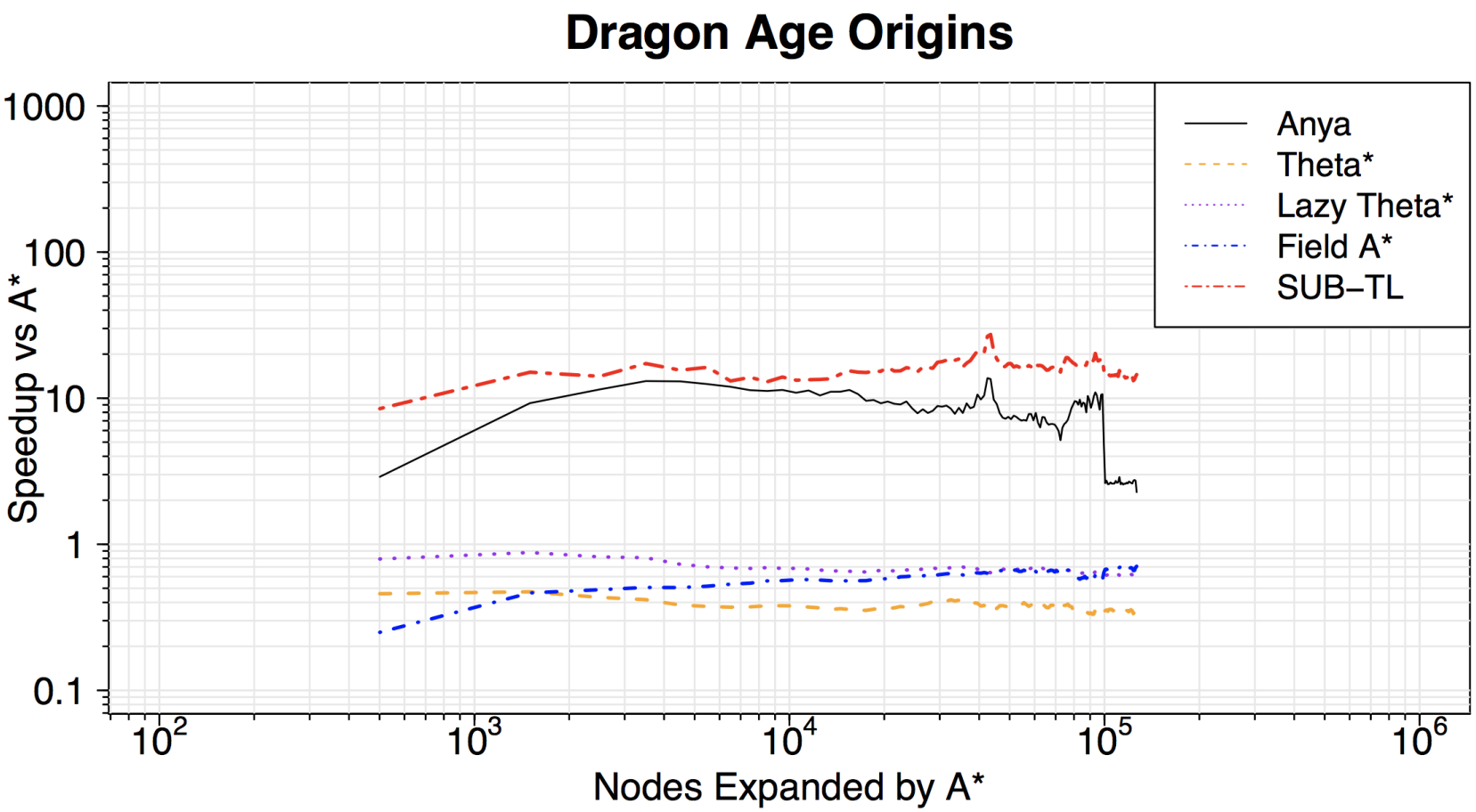
Speedup (time) vs grid A* on a range of game benchmarks appearing in Nathan Sturtevant's repository at `http:://movingai.com`.



75 maps, 93,160 problem instances.

Results on Games Maps

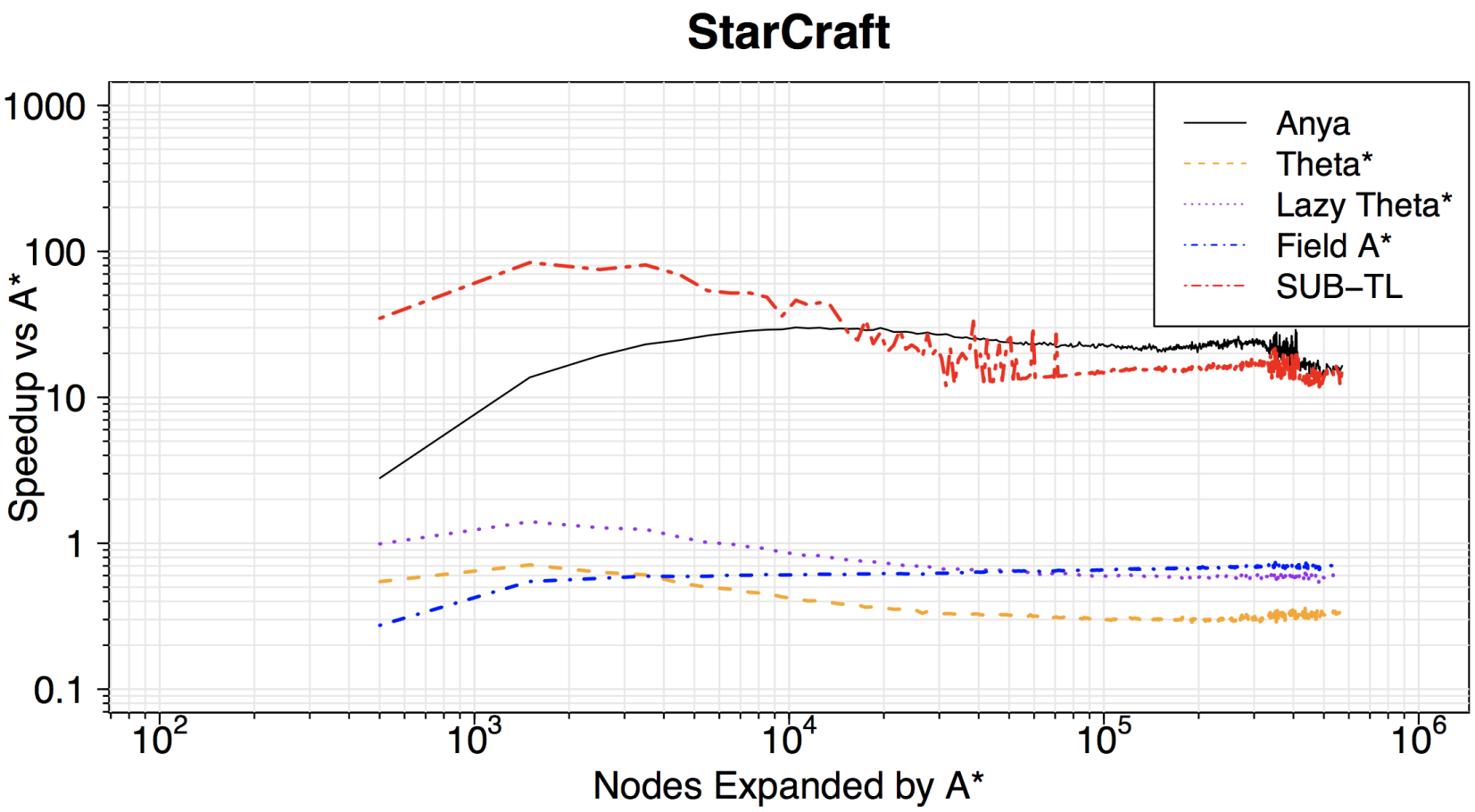
Speedup (time) vs grid A* on a range of game benchmarks appearing in Nathan Sturtevant's repository at `http://movingai.com`.



156 maps, 159,465 problem instances.

Results on Games Maps

Speedup (time) vs grid A* on a range of game benchmarks appearing in Nathan Sturtevant's repository at `http:://movingai.com`.



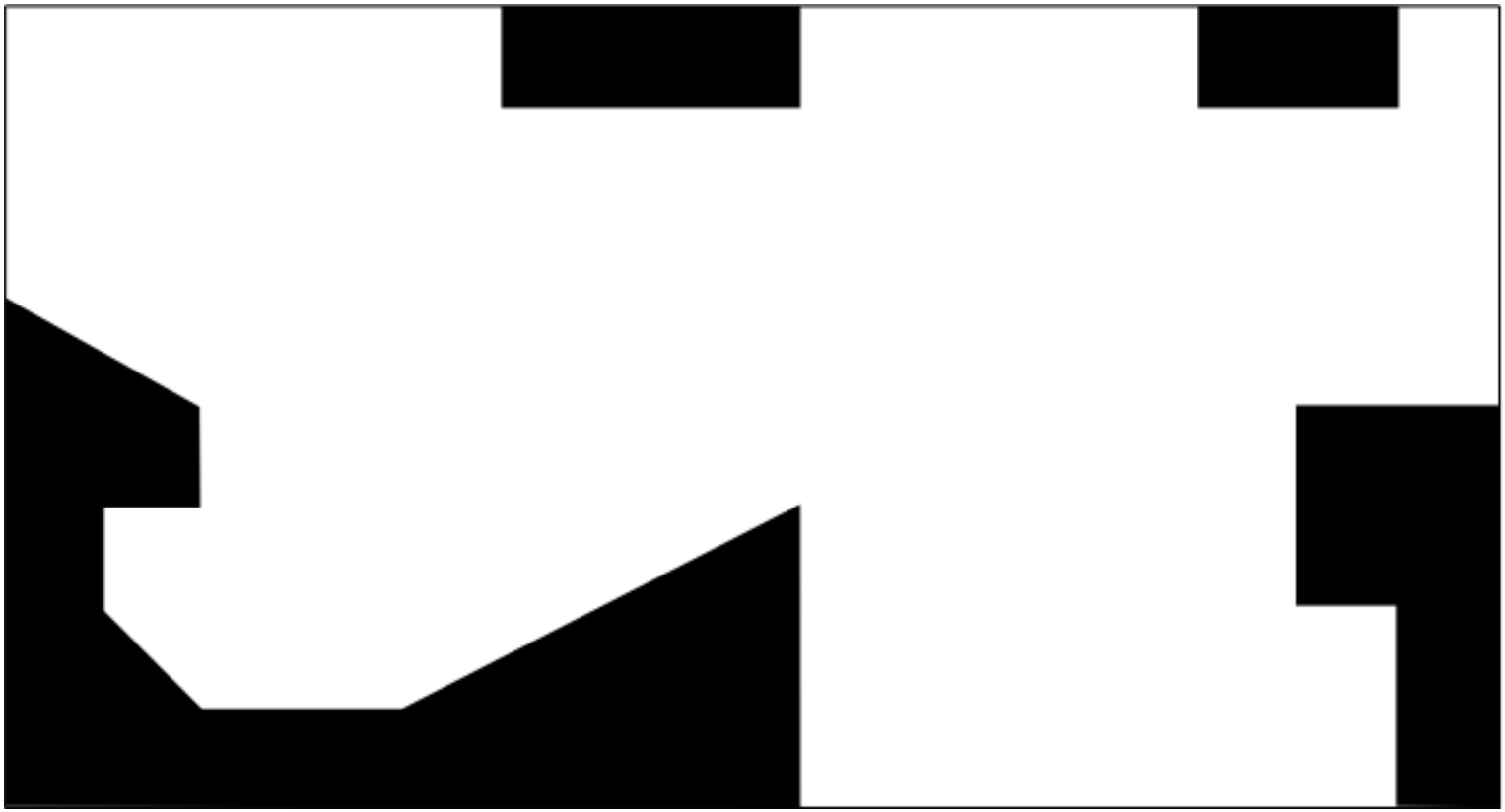
75 maps, 198,230 problem instances.

Euclidean Shortest Paths in 2D

Euclidean Shortest Path Problems in 2D

2D ESPP

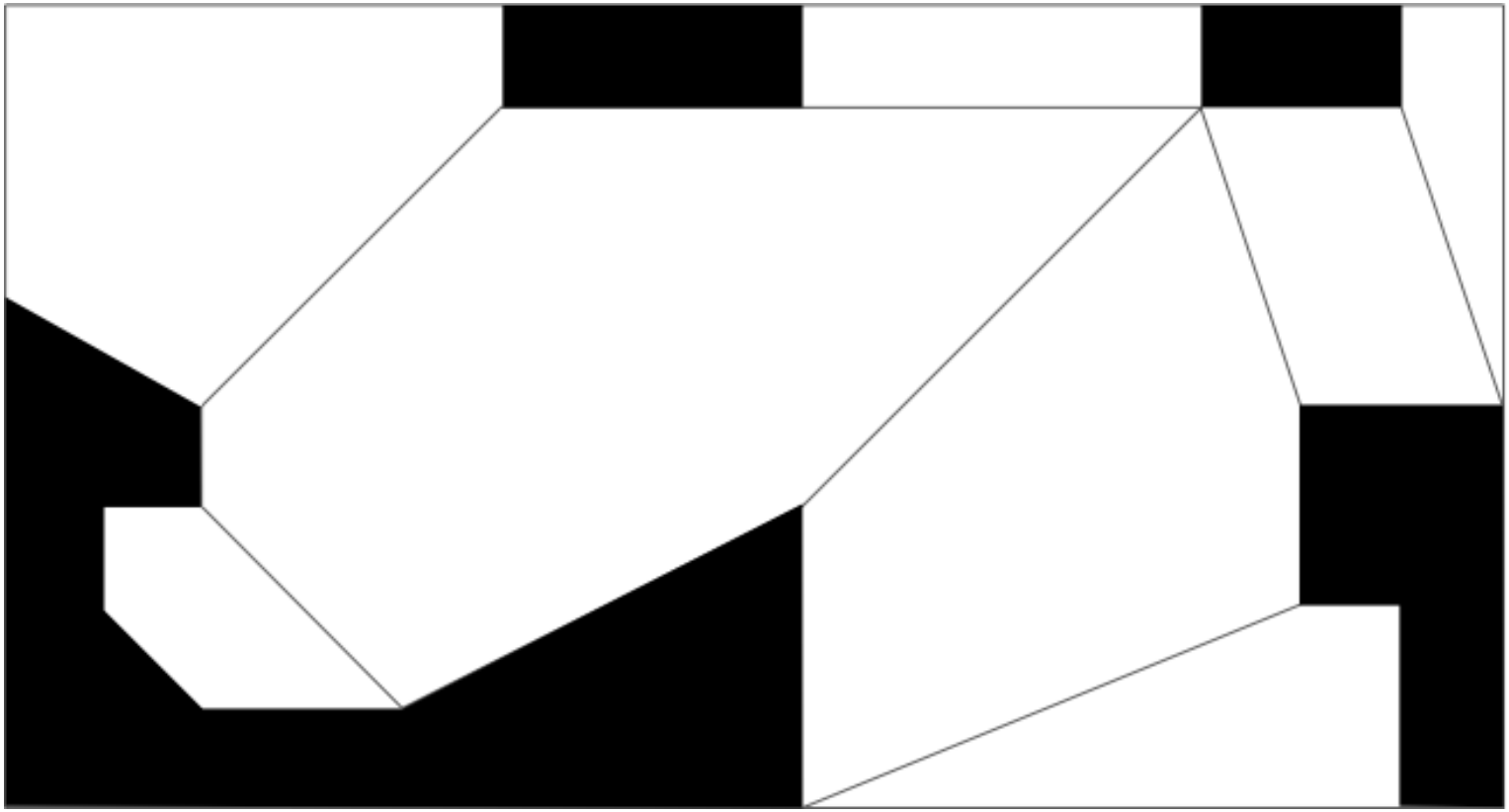
Find a shortest path among a set of polygonal obstacles.



2D ESPP on a Navigation Mesh

Navigation Mesh

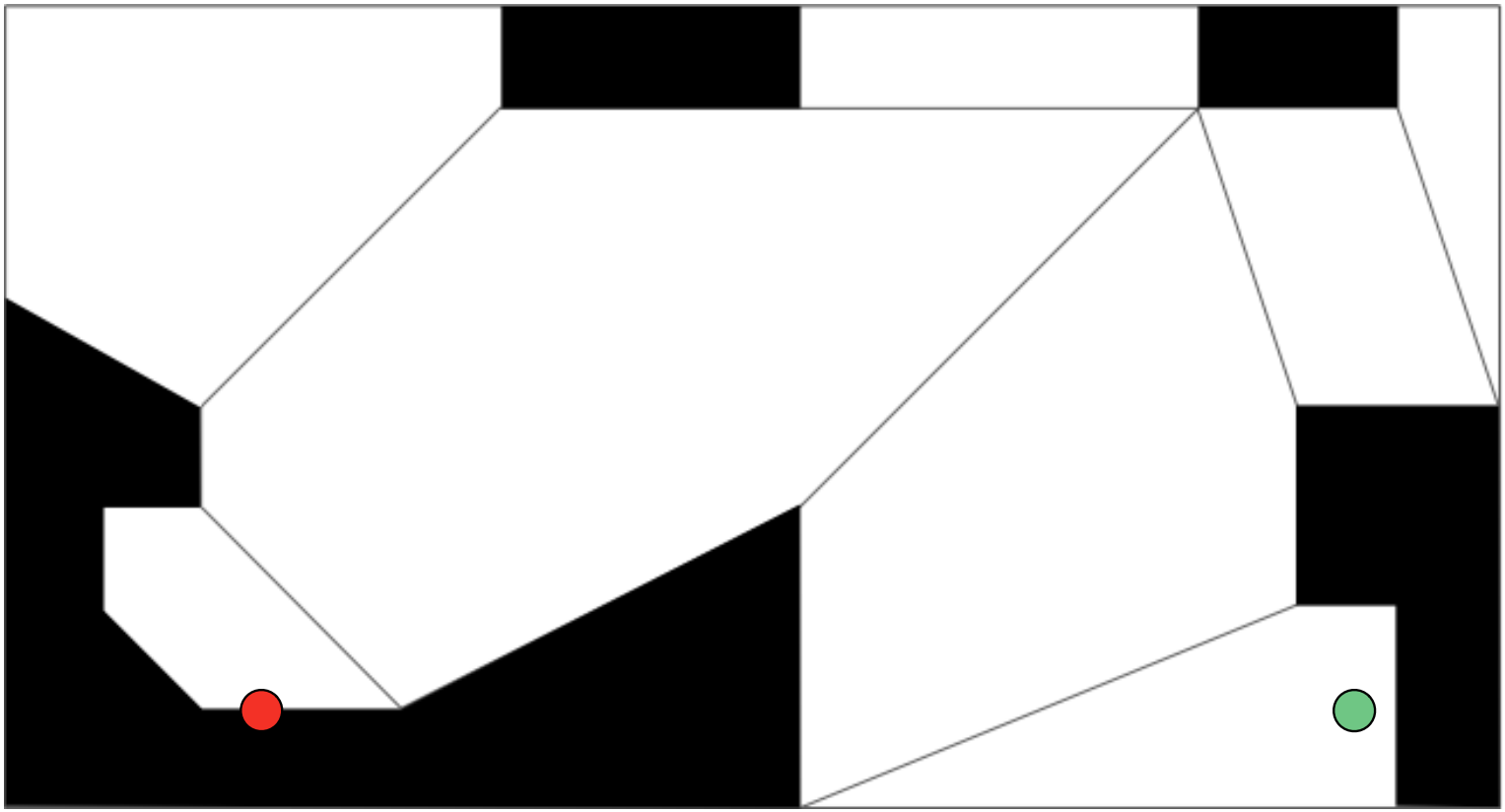
A partitioning of the traversable space into a collection of convex polygons. Cheap to build. Common in many application areas.



From Any-angle Pathfinding to ESPP

Polyanya

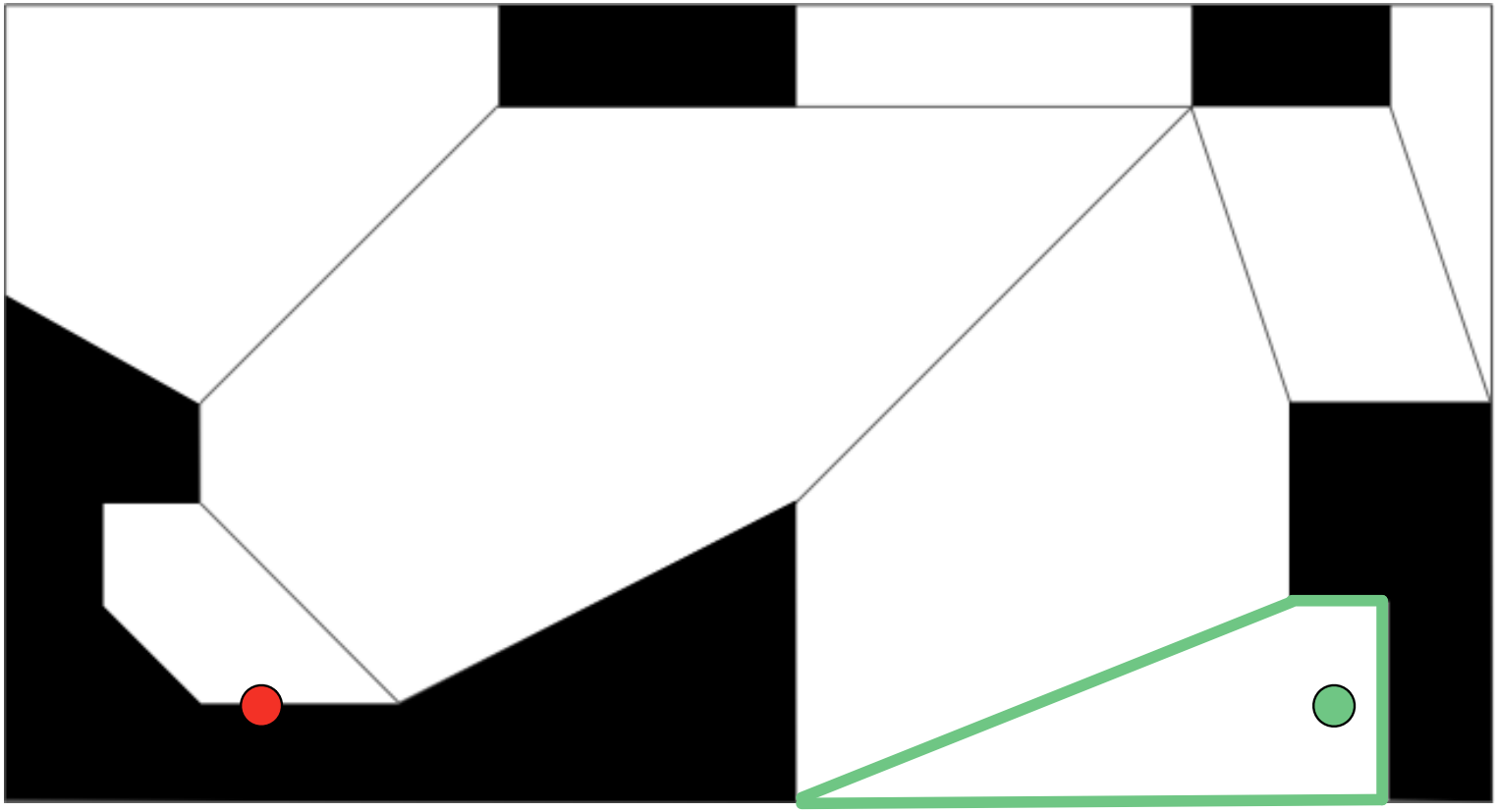
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

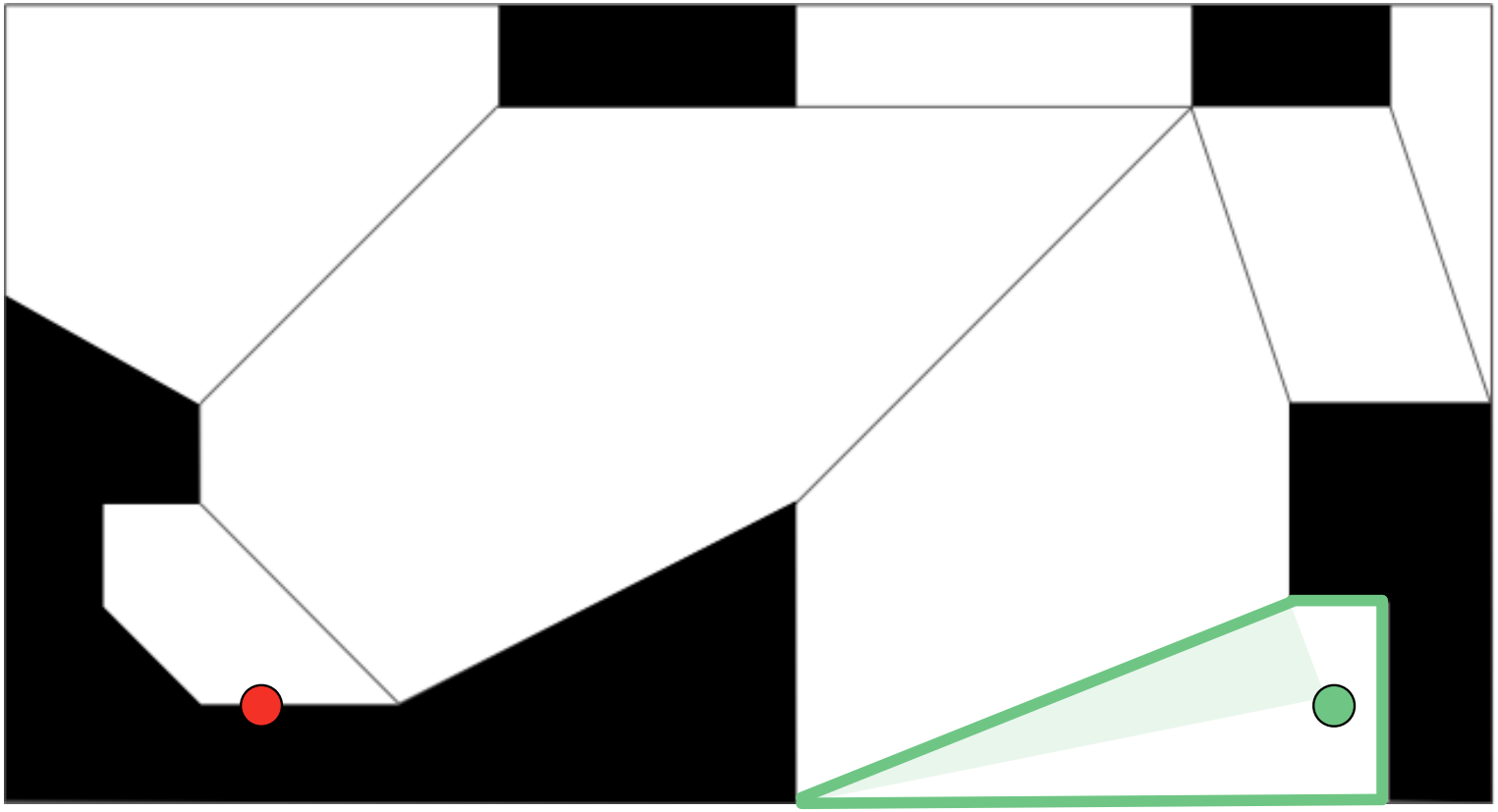
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

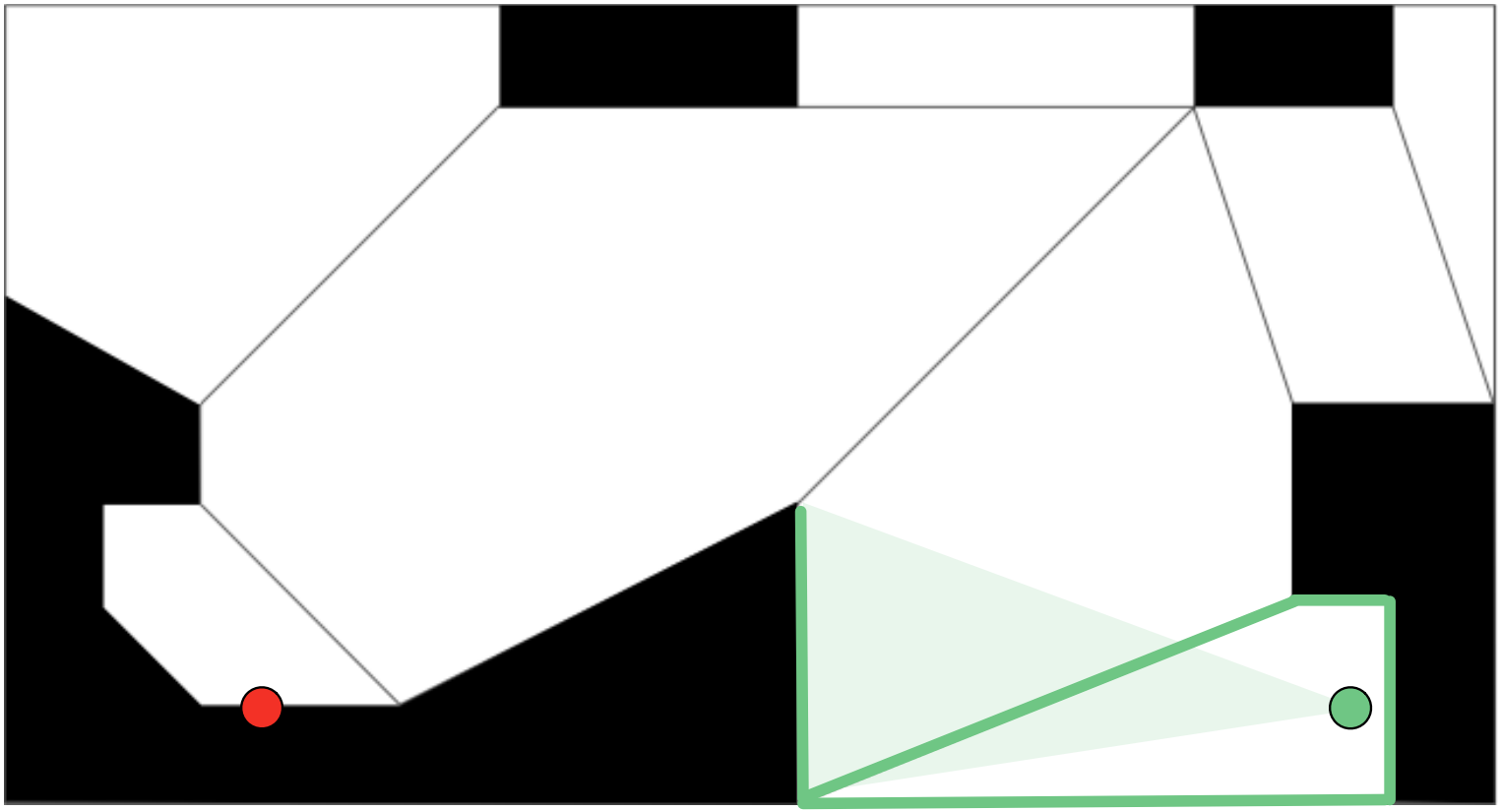
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

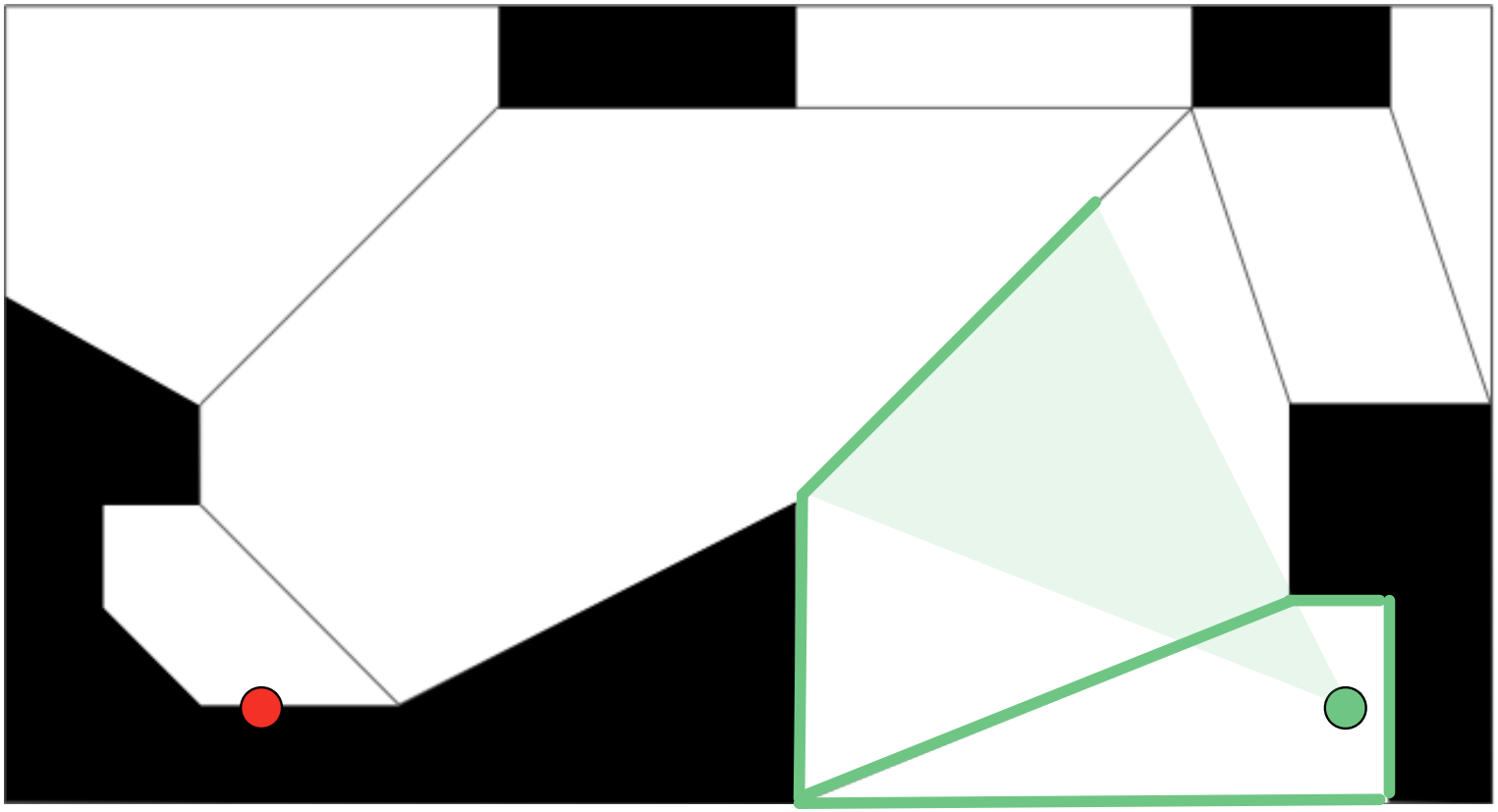
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

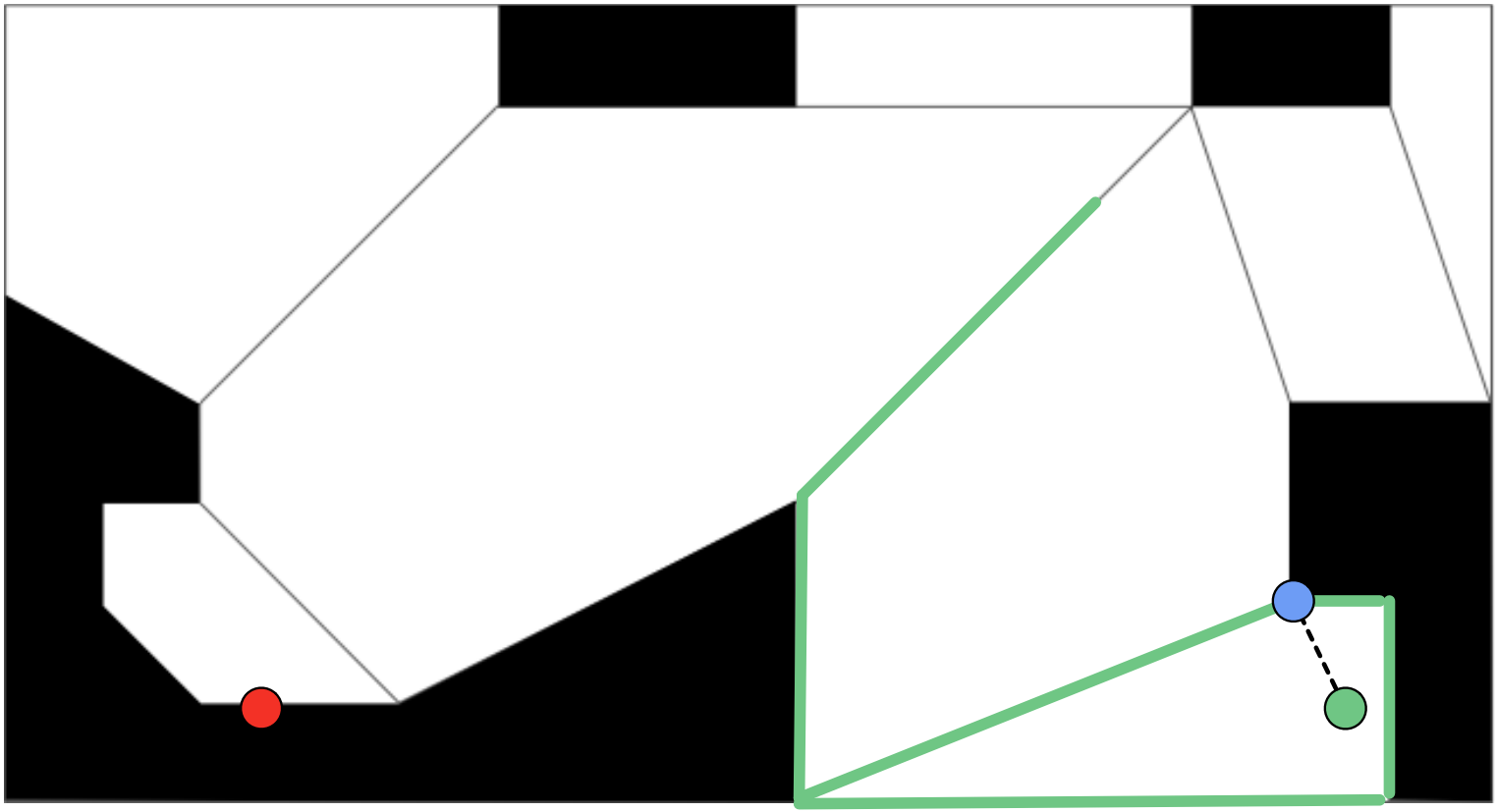
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

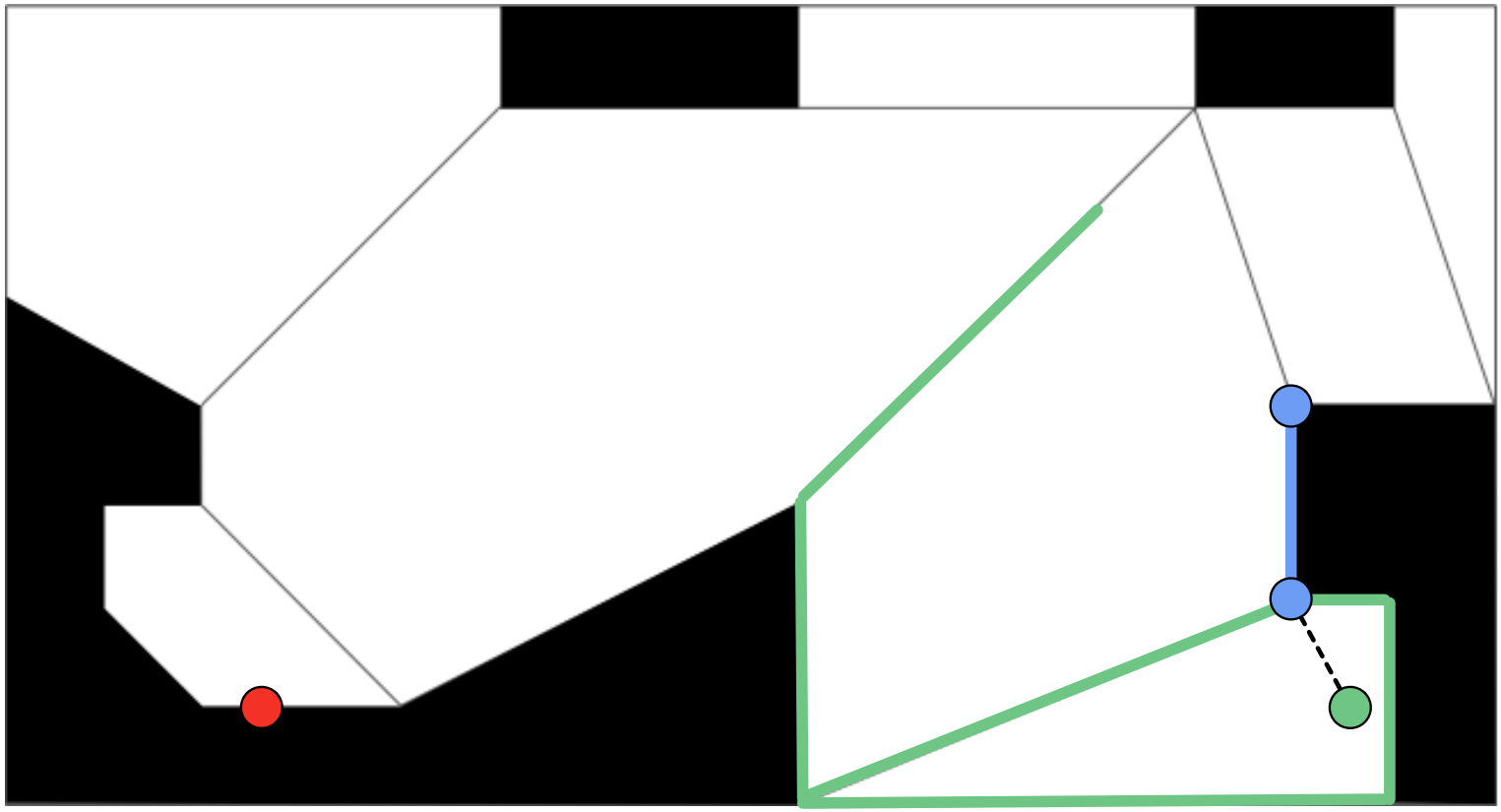
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

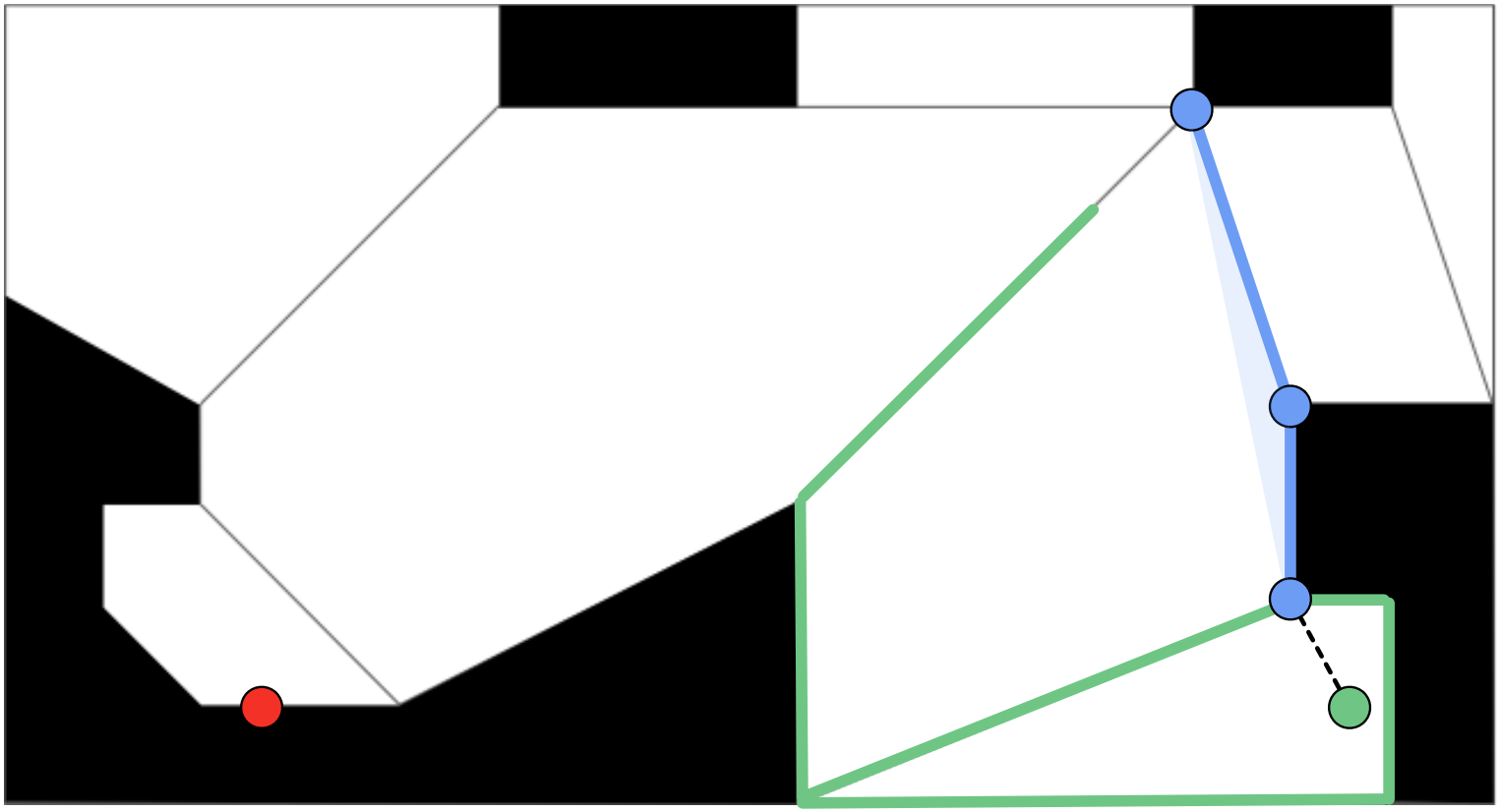
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

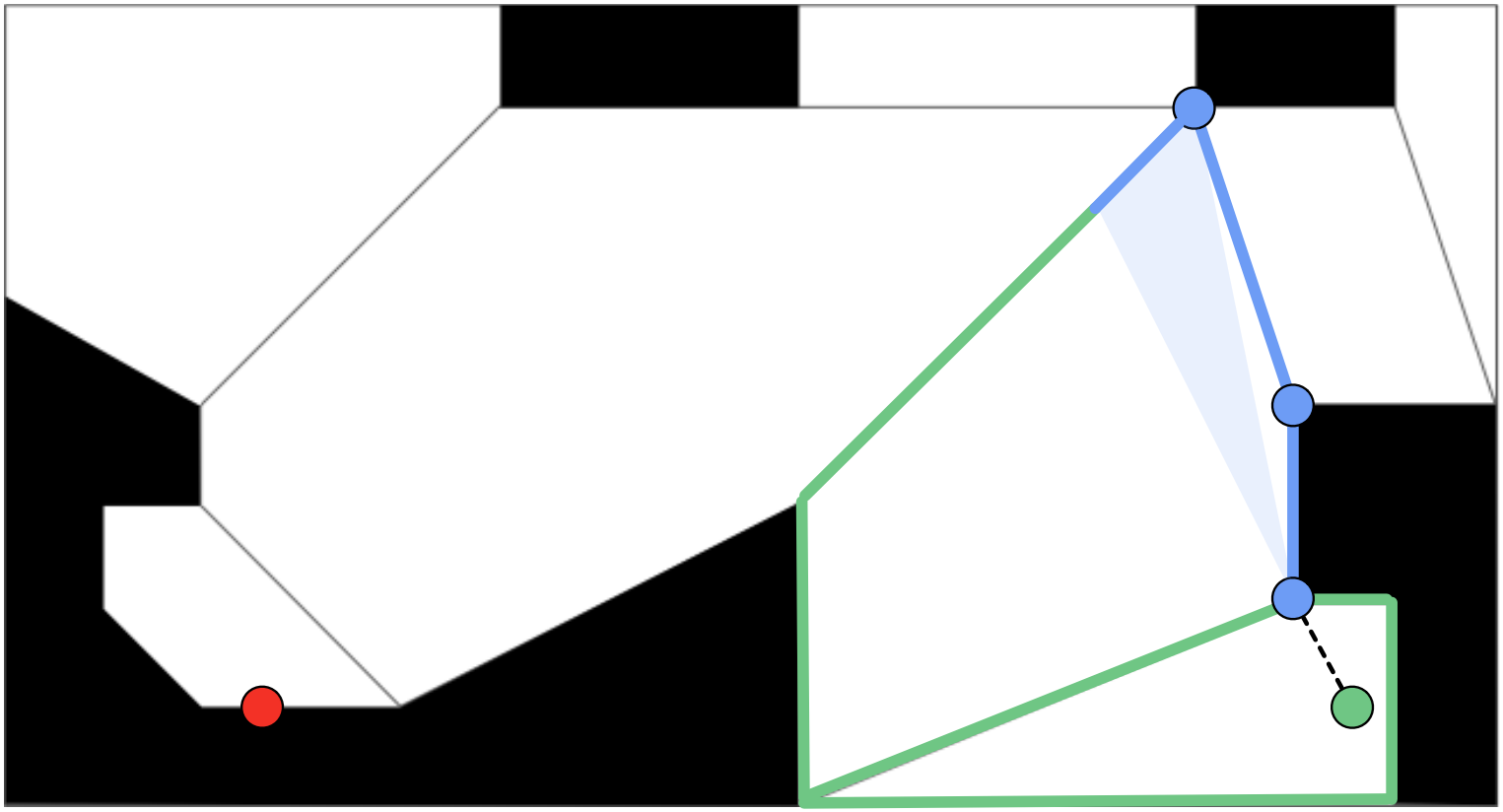
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

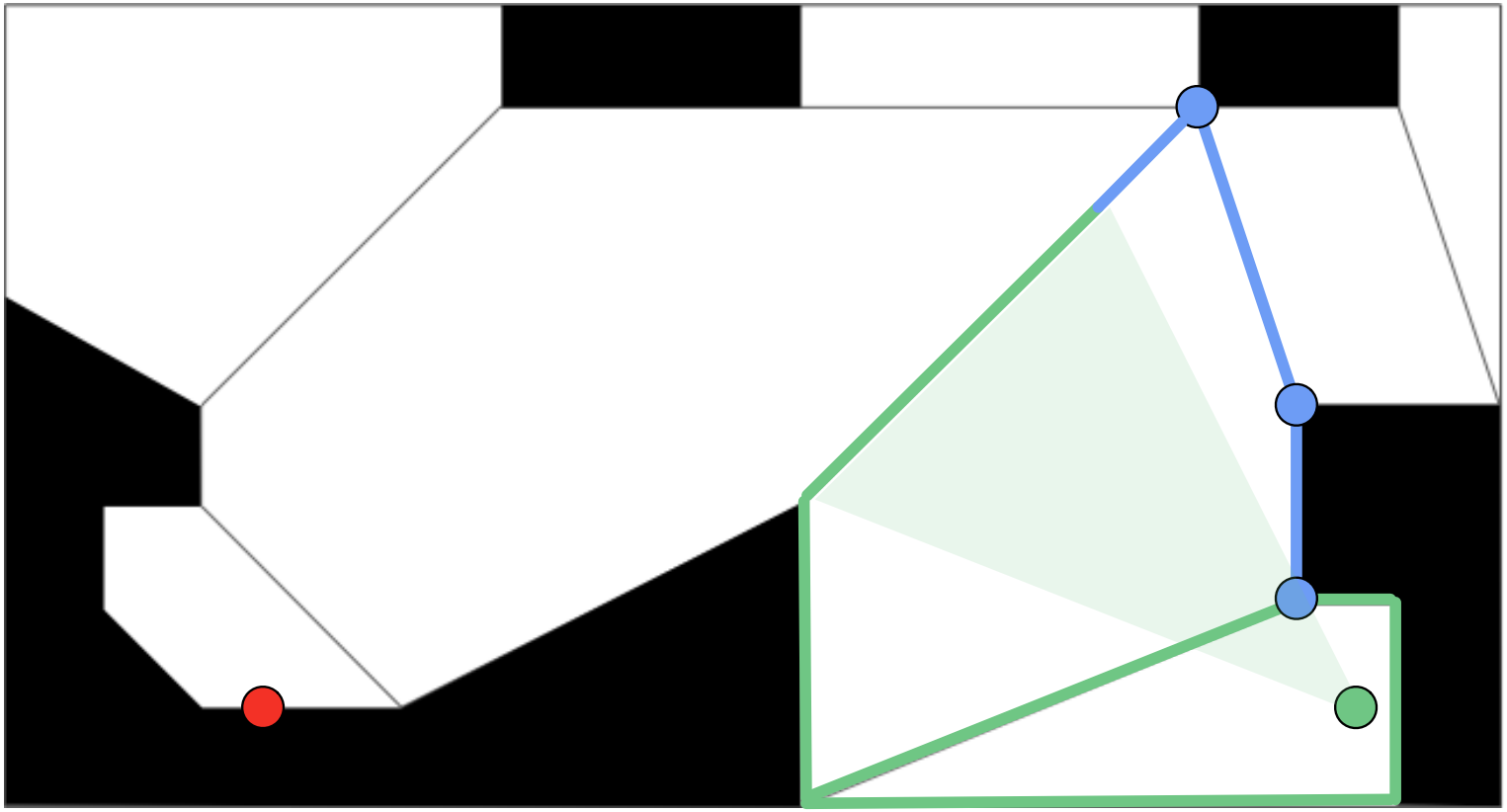
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

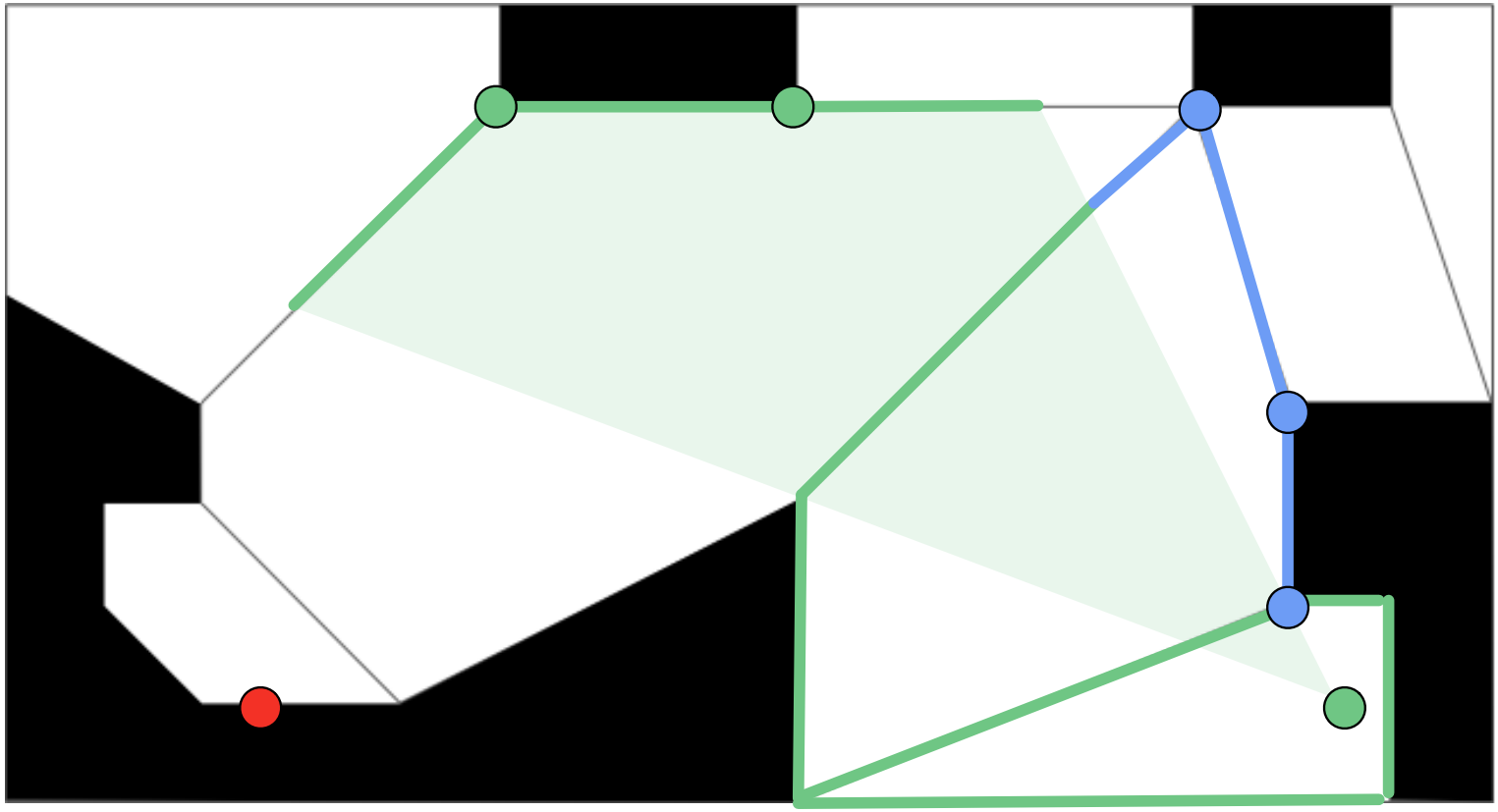
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

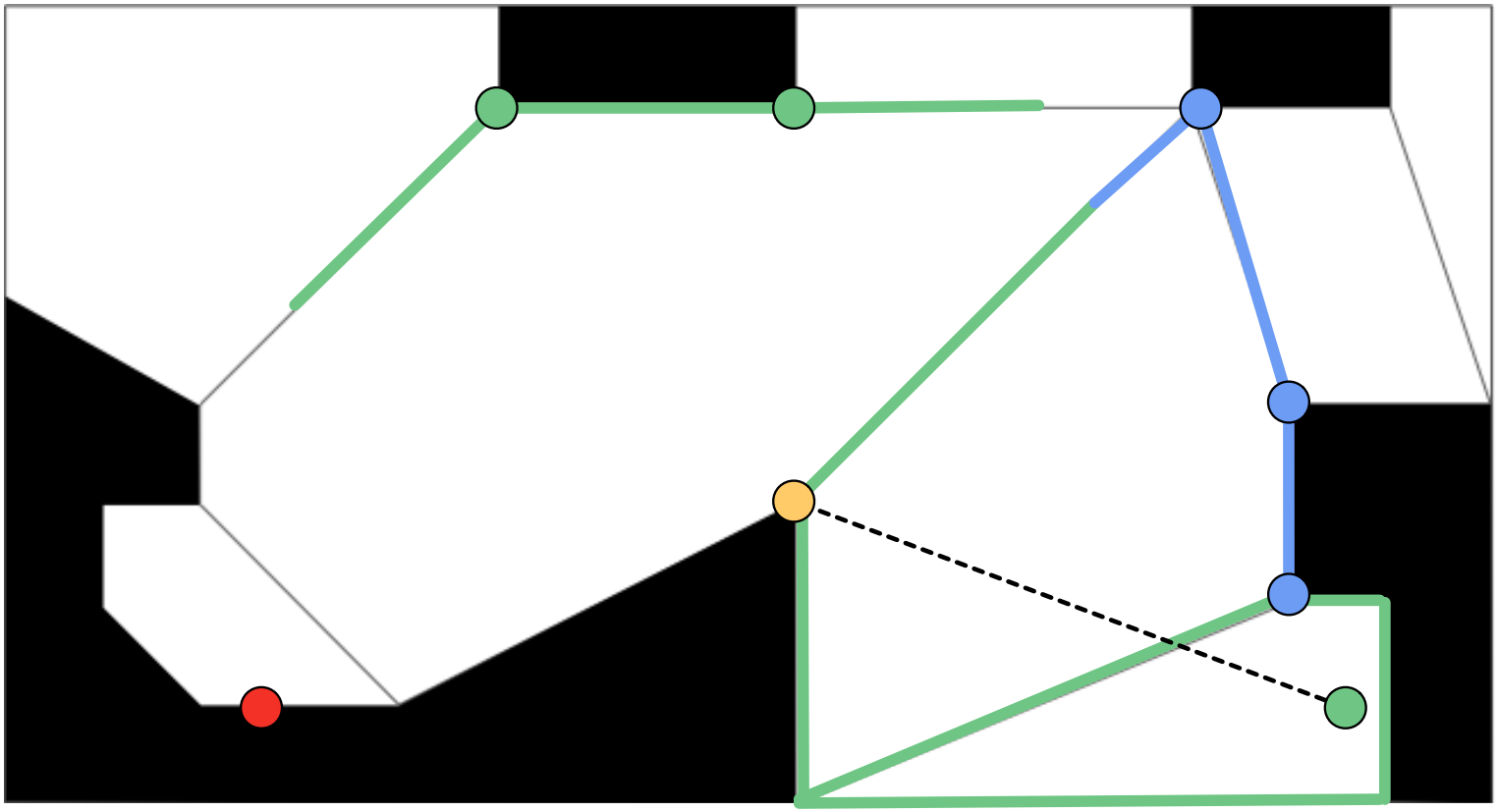
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

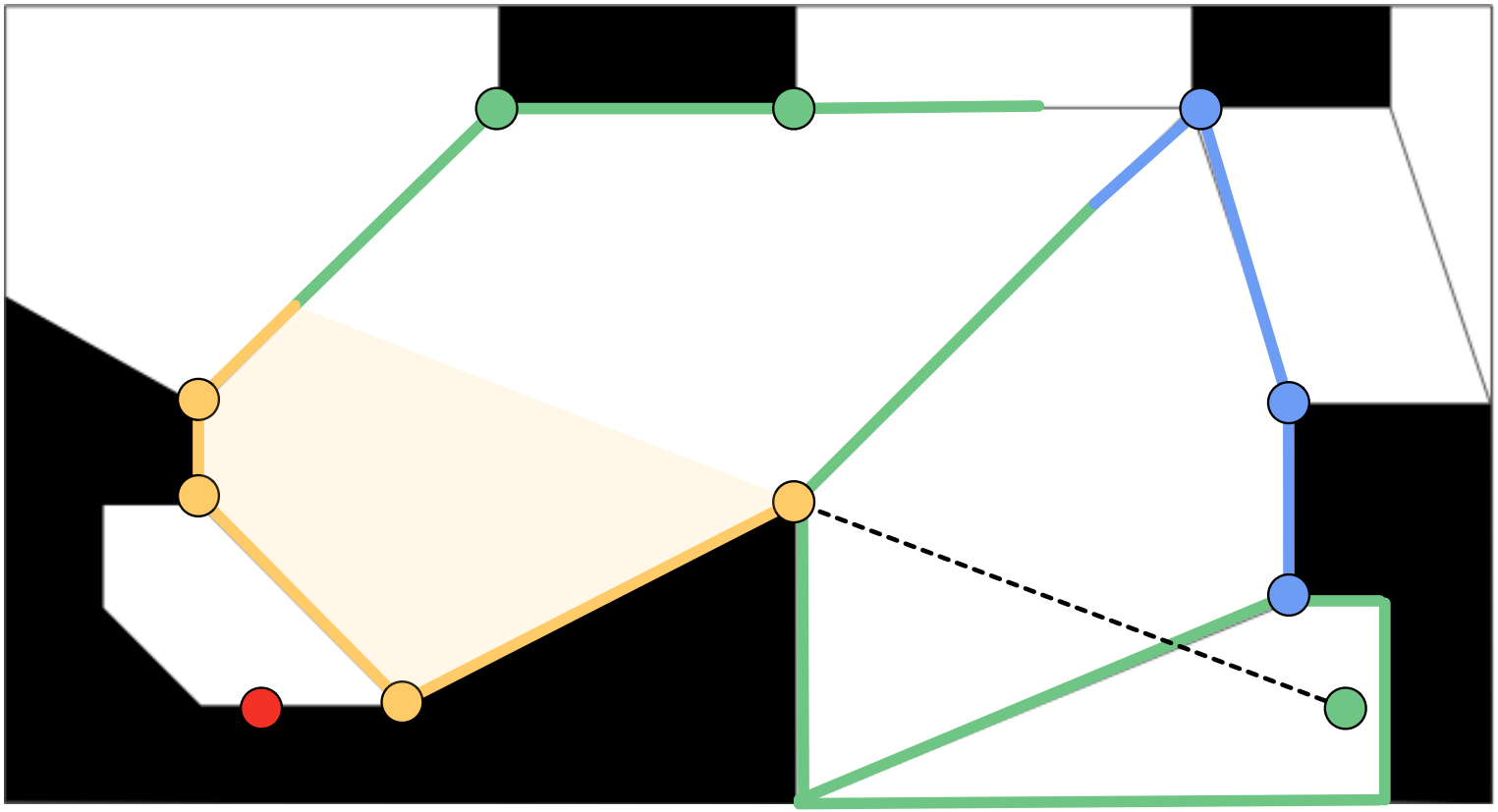
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

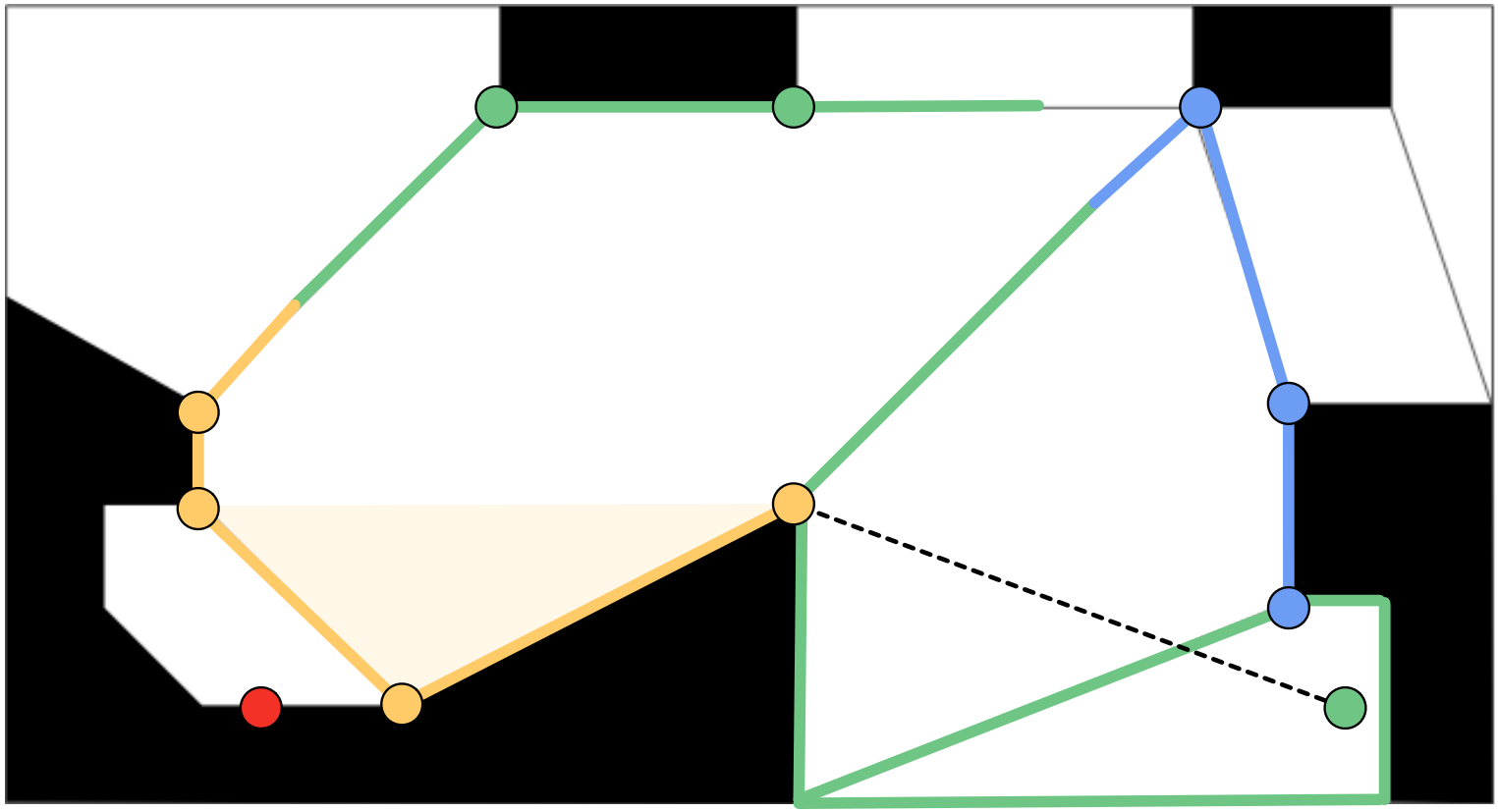
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

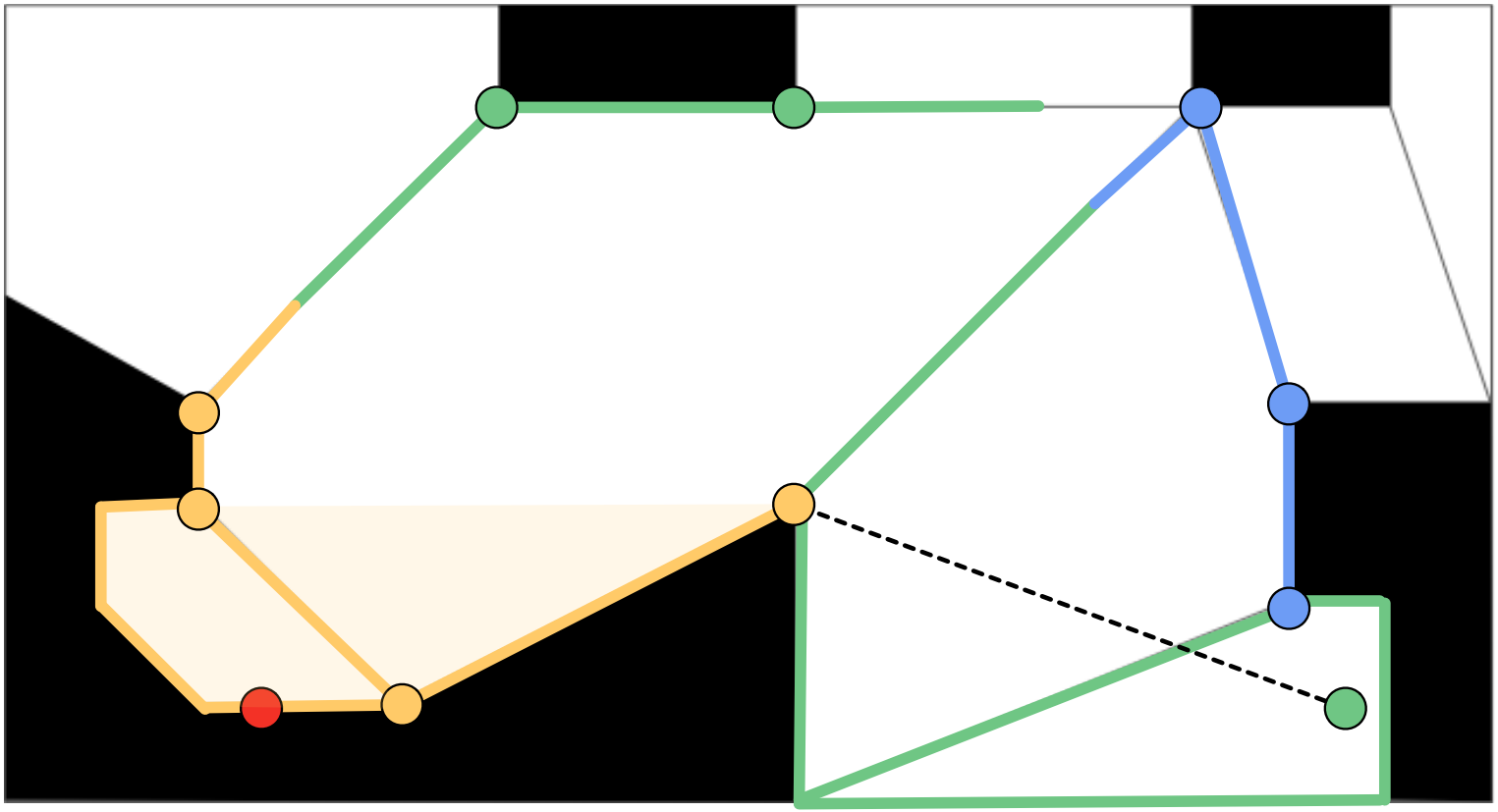
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

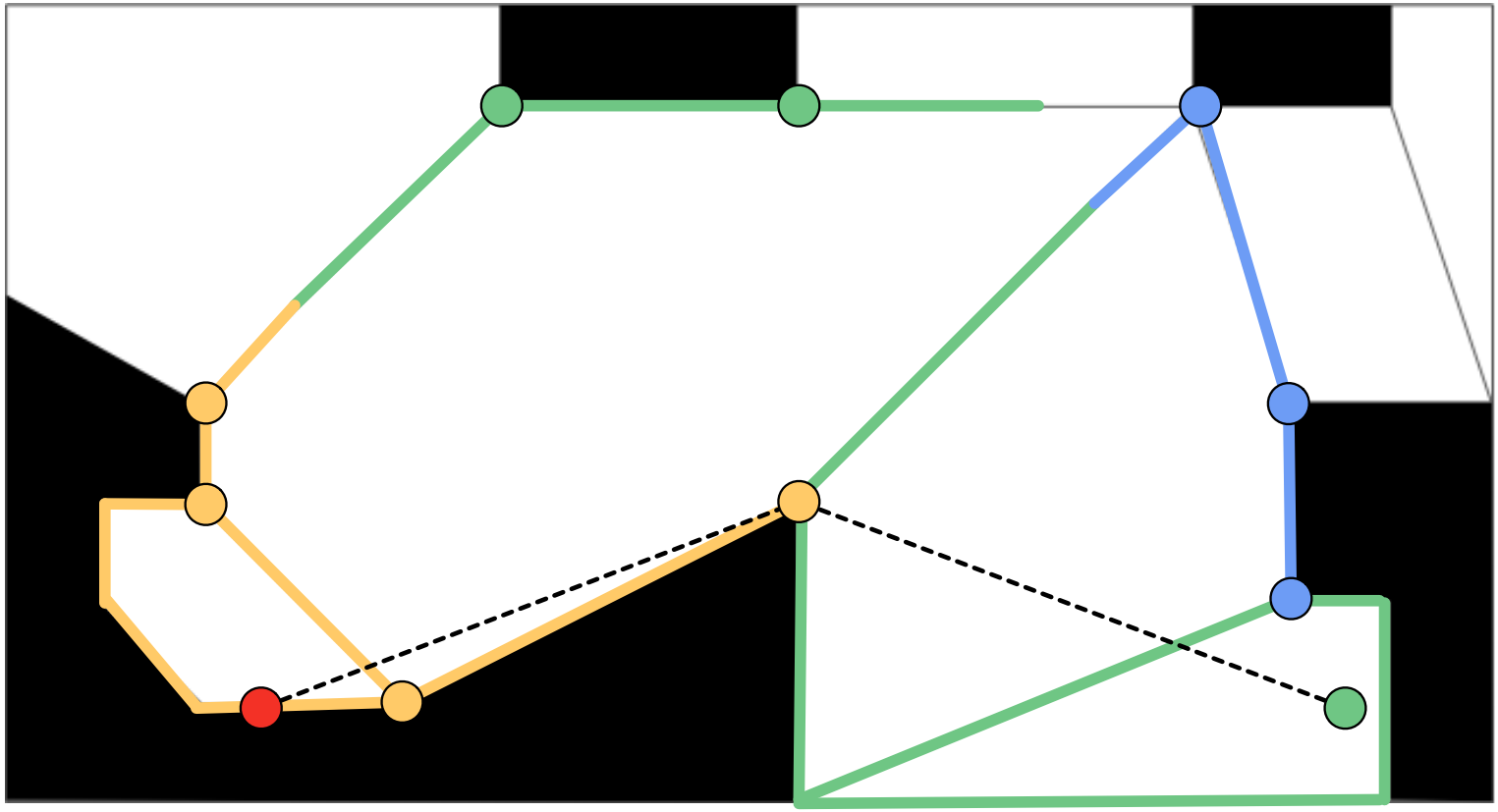
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



From Any-angle Pathfinding to ESPP

Polyanya

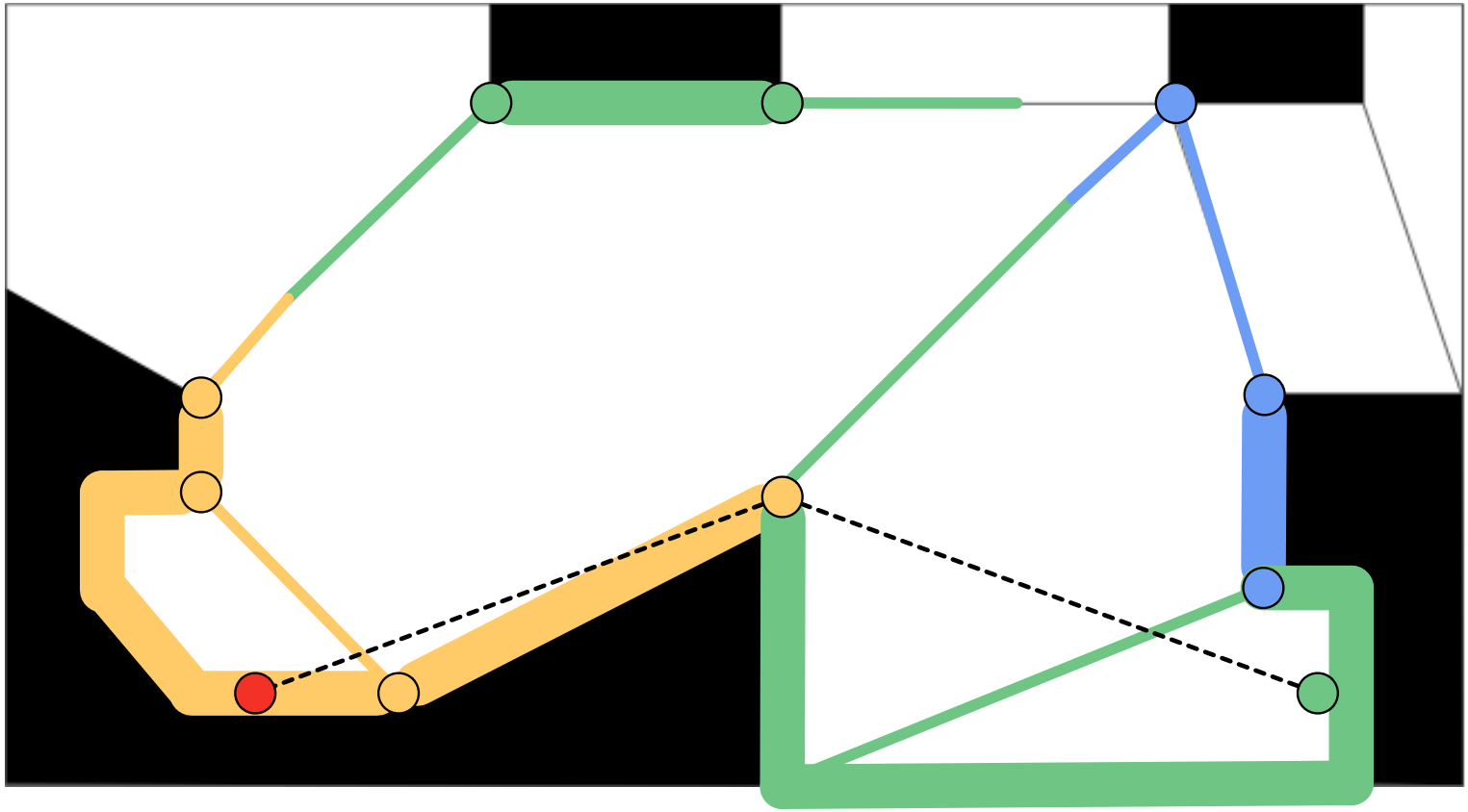
Polyanya [Cui *et al.*, 2017] extends and generalises Anya, from any-angle pathfinding on a grid to ESPP in the plane.



Further optimisations

Dead-end pruning

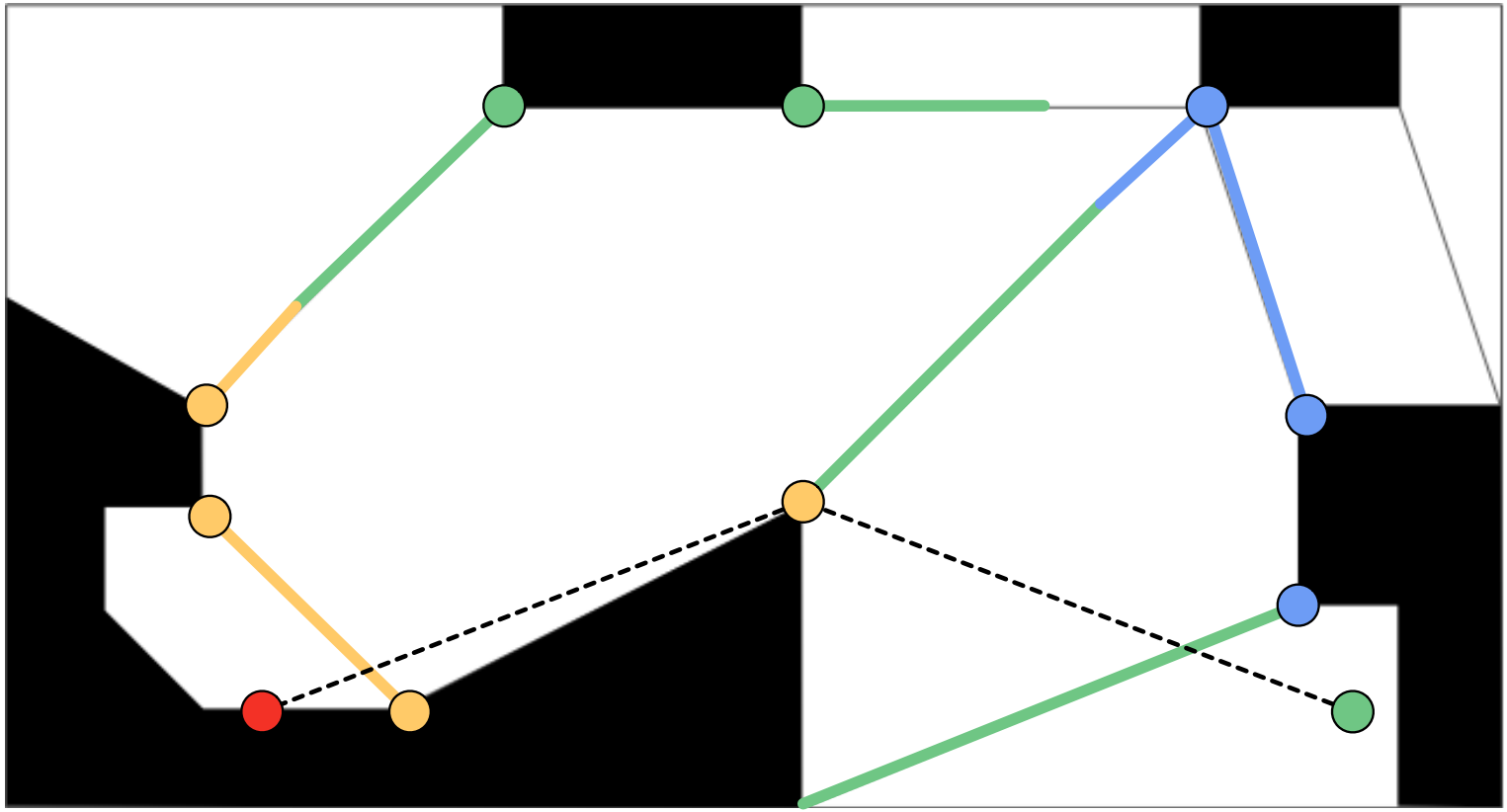
Prune all nodes that "push" into obstacles or into polygons that have only one entry edge.



Further optimisations

Dead-end pruning

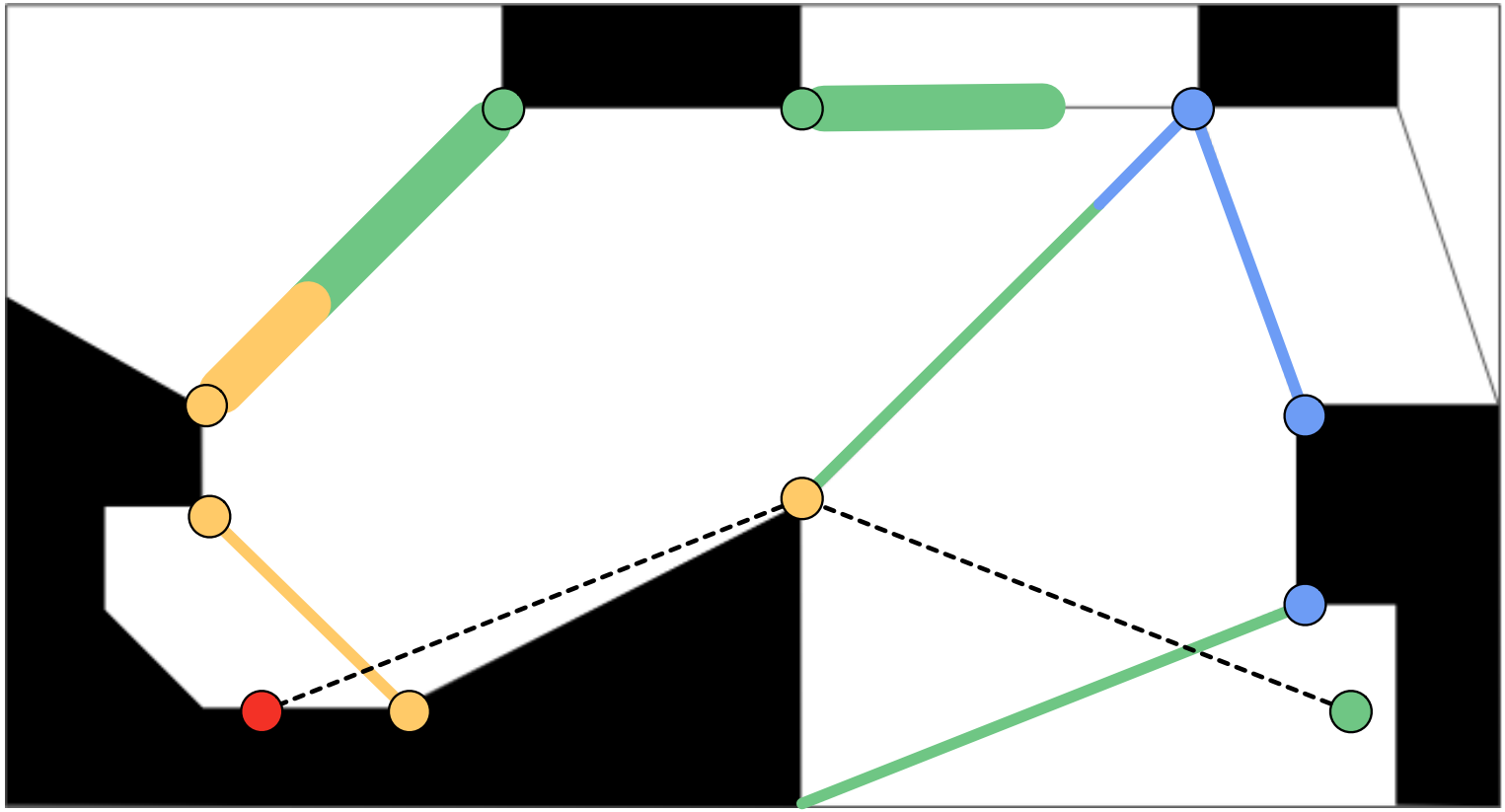
Prune all nodes that "push" into obstacles or into polygons that have only one entry edge.



Further optimisations

Dead-end pruning

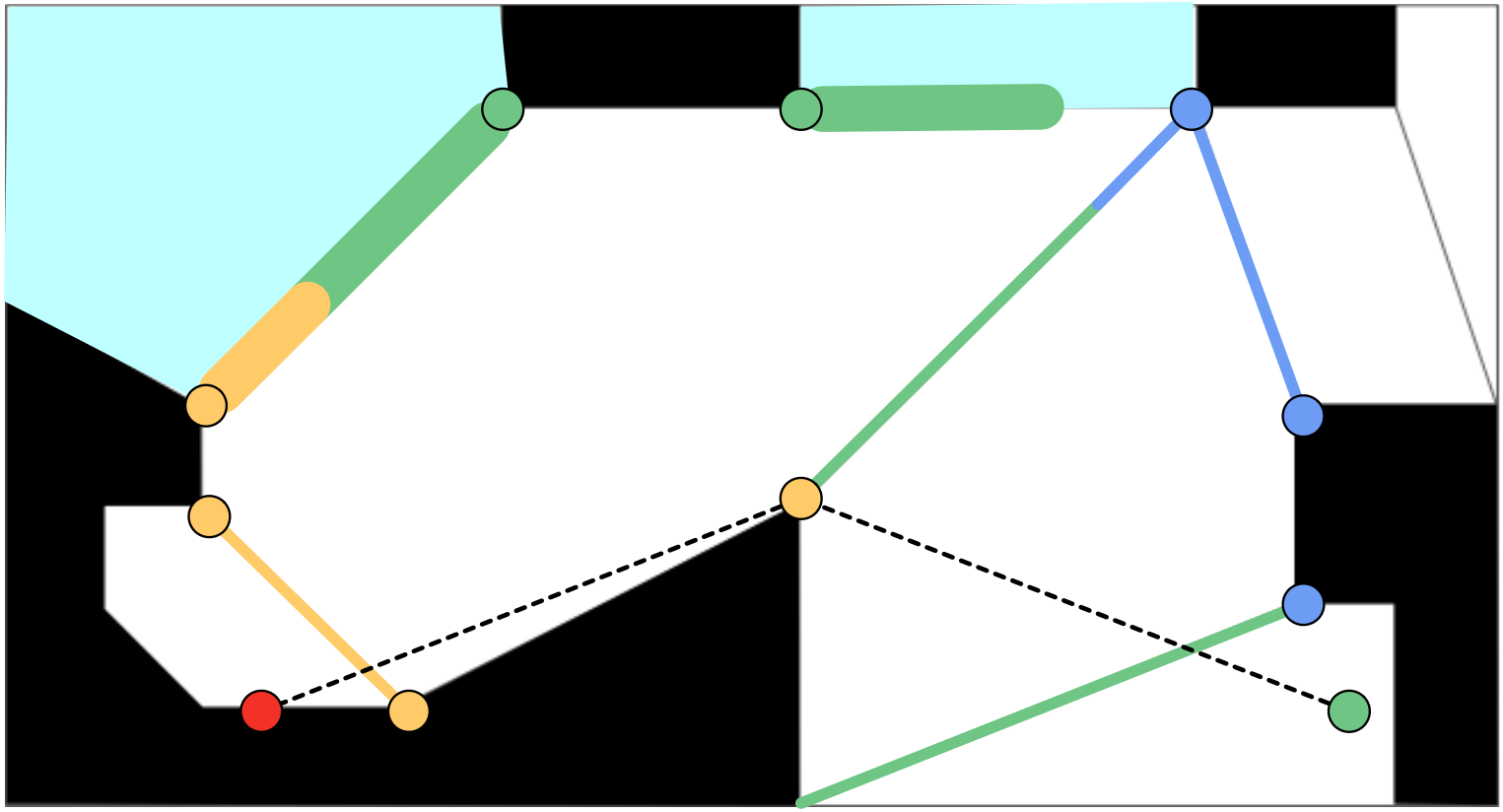
Prune all nodes that "push" into obstacles or into polygons that have only one entry edge.



Further optimisations

Dead-end pruning

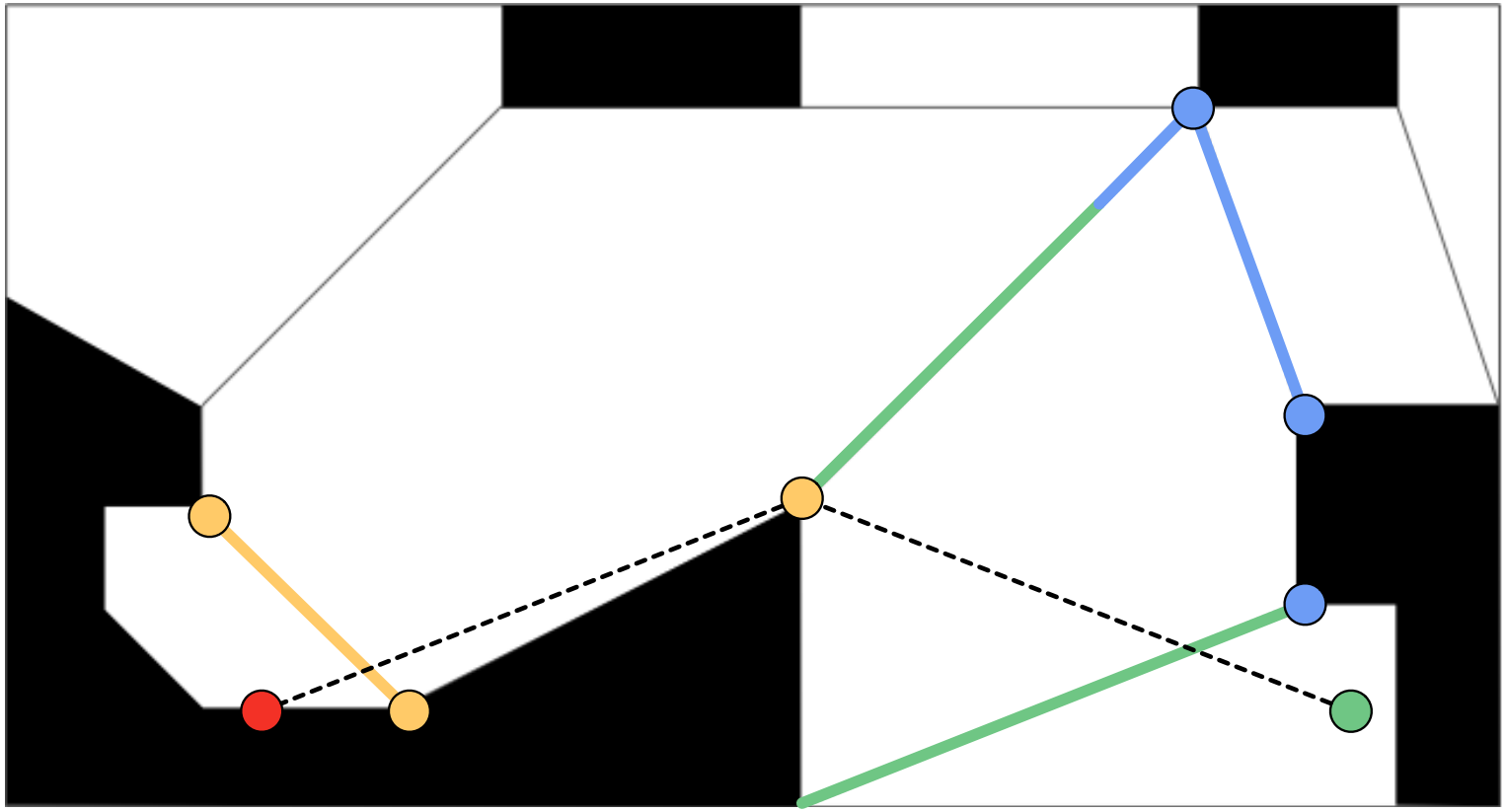
Prune all nodes that "push" into obstacles or into polygons that have only one entry edge.



Further optimisations

Dead-end pruning

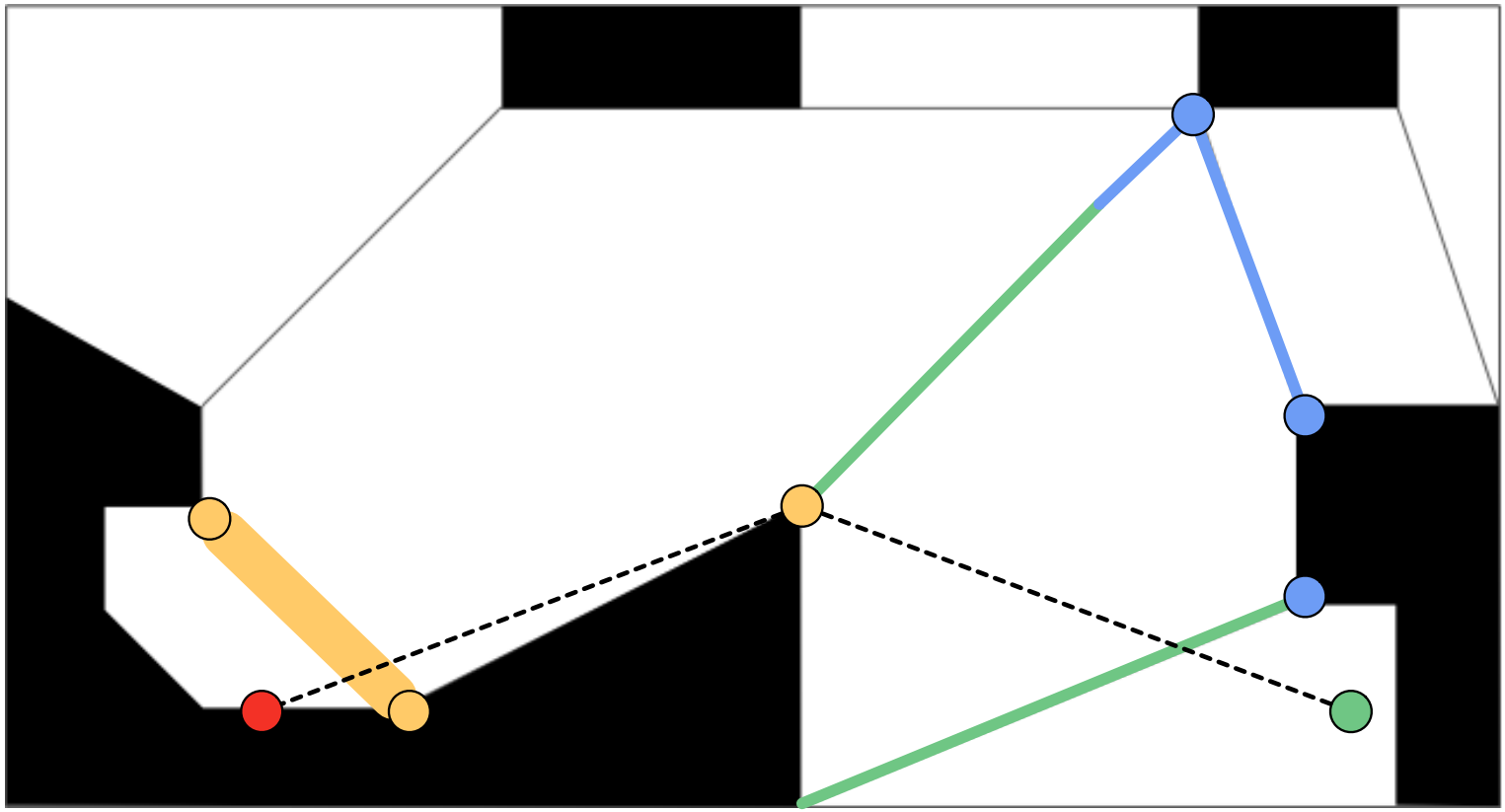
Prune all nodes that "push" into obstacles or into polygons that have only one entry edge.



Further optimisations

Intermediate Pruning

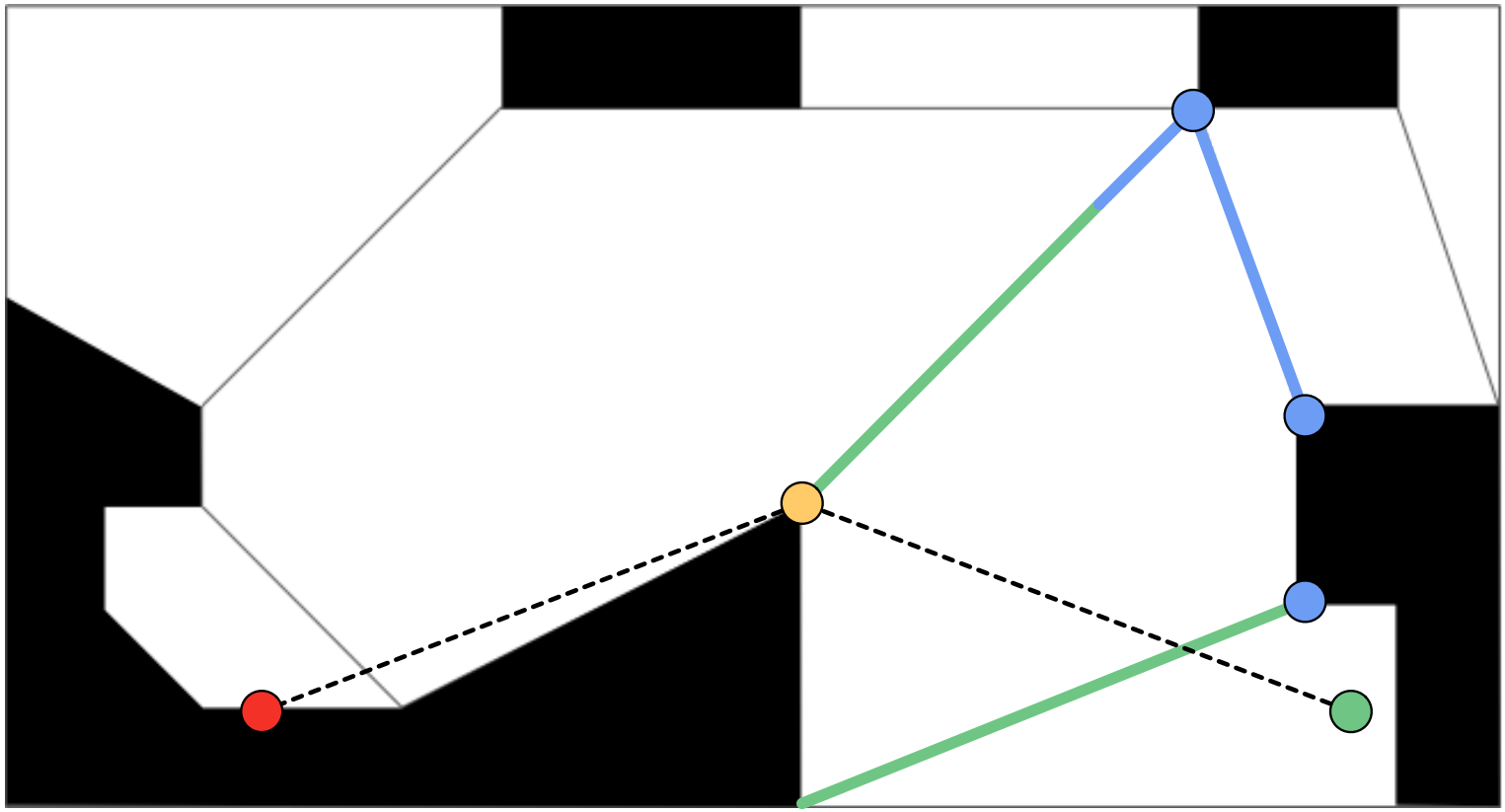
Immediately and recursively expand any node that has only a single successor.



Further optimisations

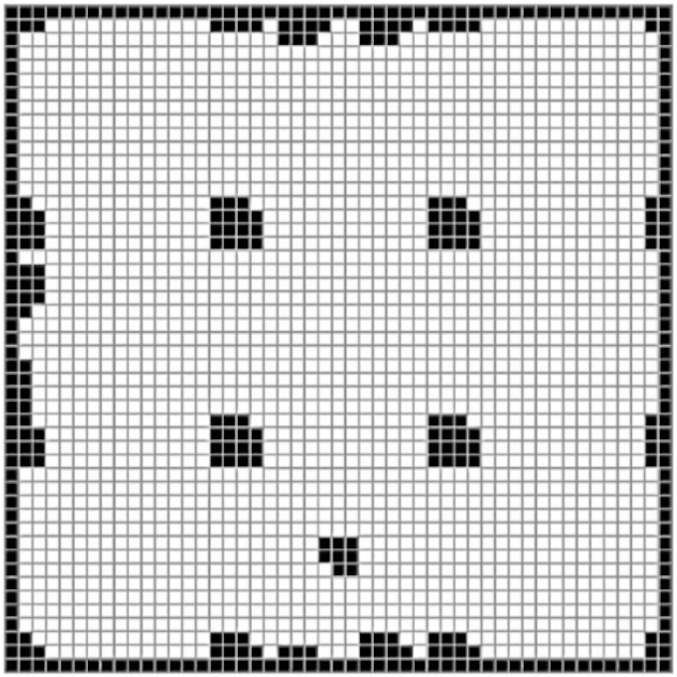
Intermediate Pruning

Immediately and recursively expand any node that has only a single successor.



Mesh selection

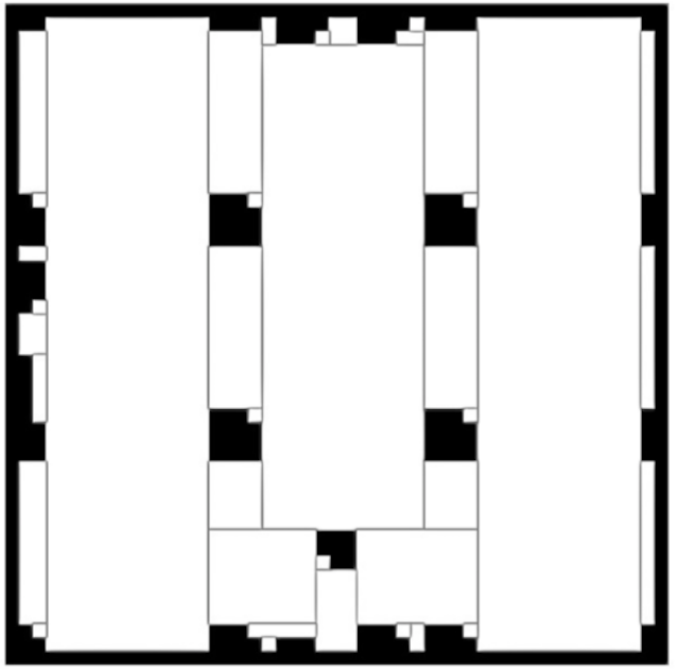
We generted a variety of meshes including: grids, rectangles, Constrained Delaunay Triangulations (CDT), and greedily merged CDTs.
Rule of thumb: Bigger polys means better performance.



This example grid (ARENA) appears in the game Dragon Age Origins.

Mesh selection

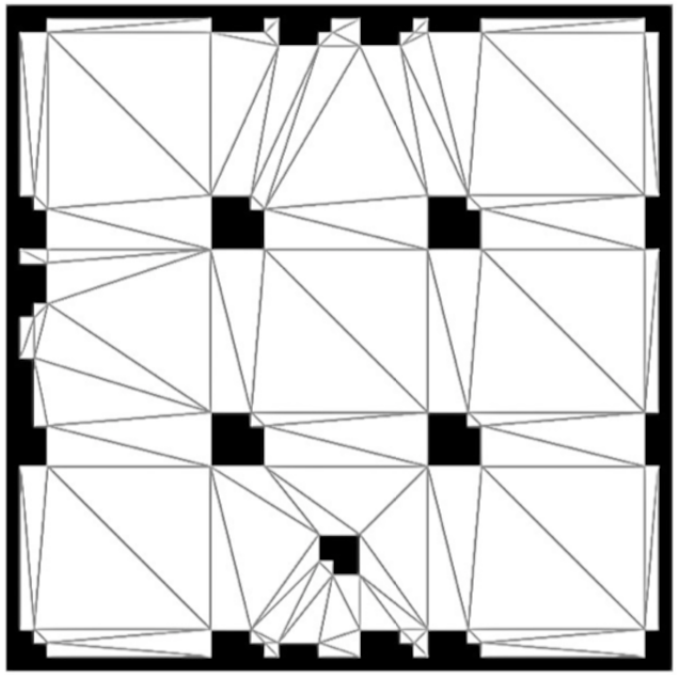
We generted a variety of meshes including: grids, rectangles, Constrained Delaunay Triangulations (CDT), and greedily merged CDTs.
Rule of thumb: Bigger polys means better performance.



Rectangle mesh, computed by greedily merging grid tiles.

Mesh selection

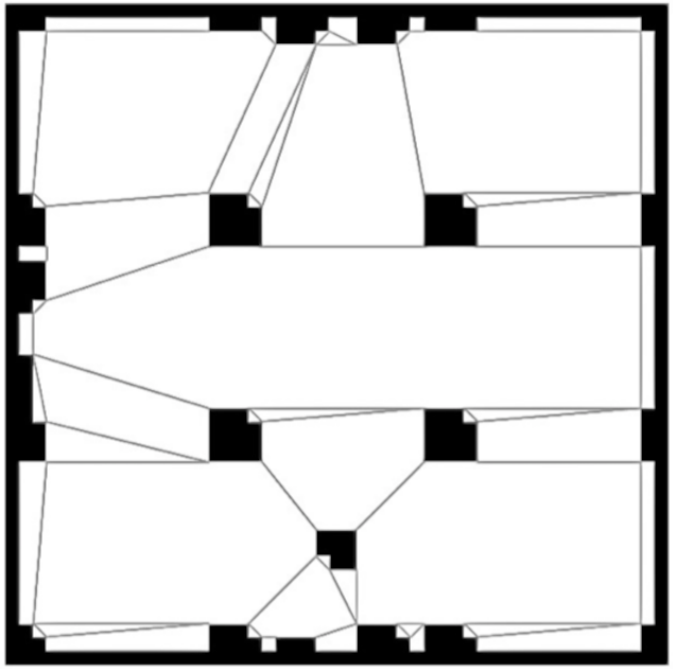
We generted a variety of meshes including: grids, rectangles, Constrained Delaunay Triangulations (CDT), and greedily merged CDTs.
Rule of thumb: Bigger polys means better performance.



Constrained Delaunay Triangulation, computed with Fade2D

Mesh selection

We generted a variety of meshes including: grids, rectangles, Constrained Delaunay Triangulations (CDT), and greedily merged CDTs.
Rule of thumb: Bigger polys means better performance.



Merged CDT, computed by greedily merging triangles.

Results vs. Anya

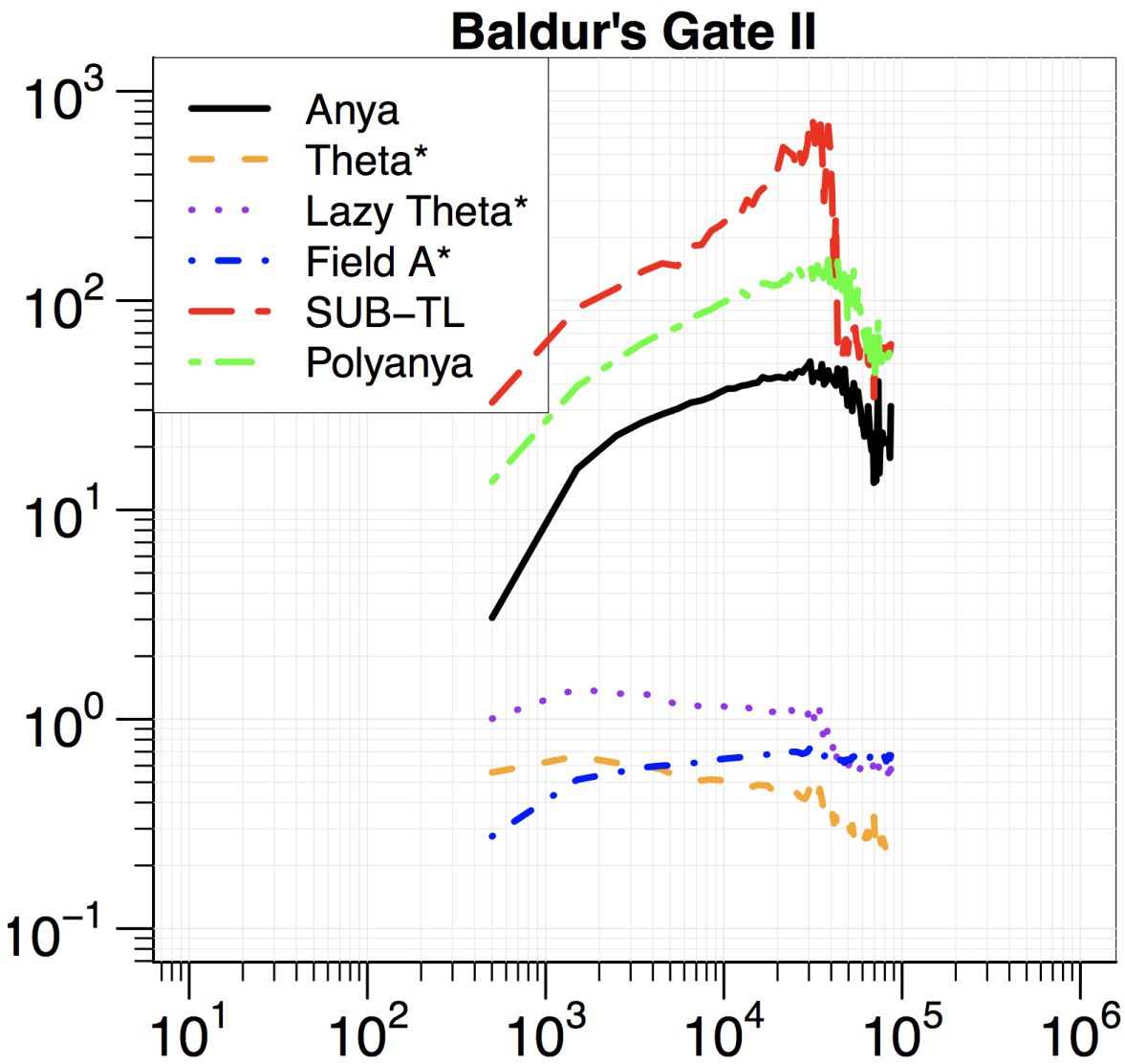
Experiment

Anya on grids vs Polyanya on different mesh types. Performance measured as **node expansions** and **search time speedup** (vs grid A*).

Benchmark	**Average Expansions**				**Average Speedup**			
	Anya	Polyanya			Anya	Polyanya		
		CDT	M-CDT	Rect		CDT	M-CDT	Rect
BG-II	79	67	**37**	131	19.9	40.2	**55.7**	23.7
DA2	228	261	**120**	388	2.5	10.3	**17.6**	7.0
DA:O	956	908	**499**	1725	8.2	8.3	**13.5**	5.3
Mazes	6633	7079	1708	**1614**	22.8	103	**158.2**	157.2
Random	17476	30615	**14243**	17934	0.9	0.6	**1.0**	0.7
Rooms	1431	1862	895	**844**	10	63.8	106.0	**107.8**
SC1	1379	1444	**755**	2537	21	18.1	**29.4**	9.4
WC3	89	109	**55**	132	3.2	30.3	**43.7**	21.0

Table: Highlighted are best results for each metric and benchmark pair.

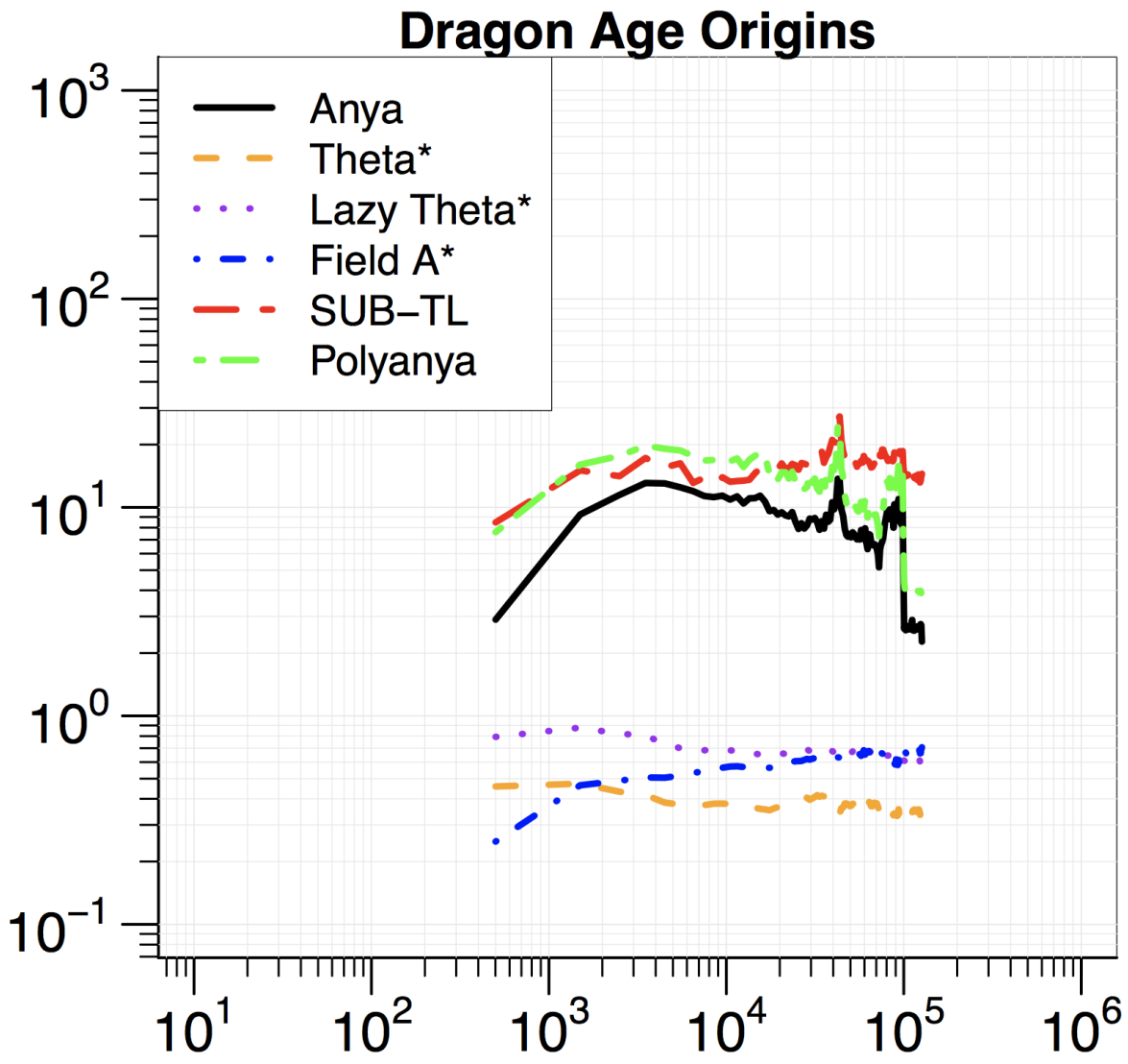
Results on Game Maps



y-axis:
speedup (time) vs grid A*.

x-axis:
problem instances, ordered by difficulty (measured as node expansions required by grid A*).

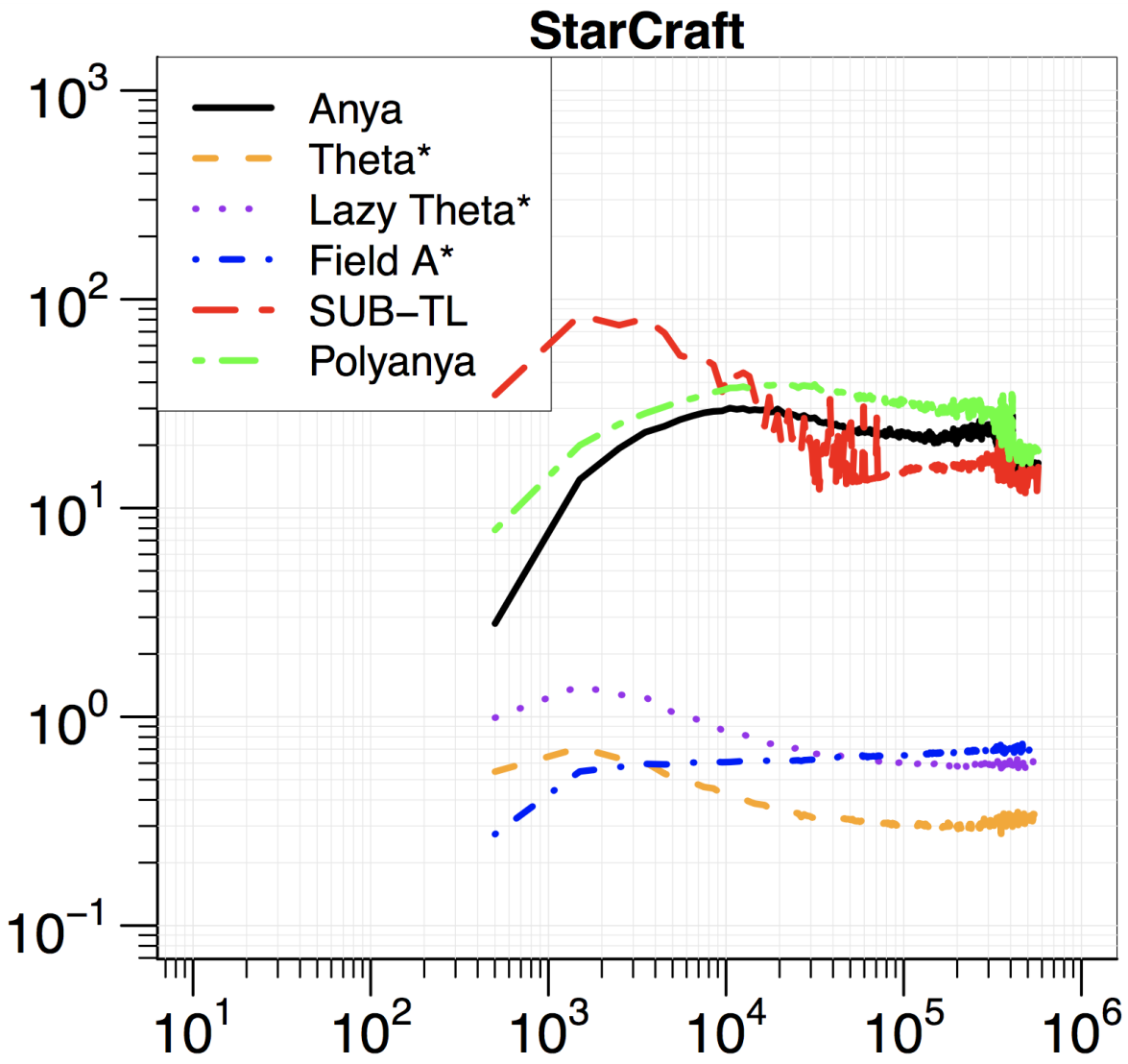
Results on Game Maps



y-axis:
speedup (time) vs grid A*.

x-axis:
problem instances, ordered by difficulty (measured as node expansions required by grid A*).

Results on Game Maps



y-axis:
speedup (time) vs grid A*.

x-axis:
problem instances, ordered by difficulty (measured as node expansions required by grid A*).

Perspectives and Future Work

Problems in 2D

Progress in this area

- We have a wide variety of benchmarks
- We understand the characteristics of 2D domains that make search difficult. For example:
  - Grids: Symmetric paths
  - Roads: Highway dimension
- We have sophisticated search algorithms, heuristics and preprocessing techniques
- We can solve 2D problems extremely well and often with very few compromises.

Challenges:

- Maps with dynamically changing costs (up and down)
- Euclidean pathfinding with weighted regions
- Maps with moving obstacles

Problems in 3D

Progress in this area

Some progress has been made:

- We have benchmarks, though only from games [Brewer and Sturtevant, 2018]
- Canonical orderings have been generalised to 3D
- Theta* is generalised to 3D

Challenges

- Orders more states (tens and sometimes hundreds of millions)
- Is 3D search challenging for the same reasons as 2D?
  - We know the continuous 3D problem is intractable in general [Canny and Reif, 1987]
  - But not much else; e.g.
  - Do there exist bounded suboptimal methods analogous to Anya and Polyanya?
- How effective are our existing methods in 3D?

References I

Daniel Brewer and Nathan R Sturtevant.
Benchmarks for pathfinding in 3d voxel space.
In *Eleventh Annual Symposium on Combinatorial Search*, 2018.

John Canny and John Reif.
New Lower Bound Techniques for Robot Motion Planning Problems.
In *Proceedings of the 28th Annual Symposium on Foundations of Computer Science*, SFCS '87, pages 49–60, Washington, DC, USA, 1987. IEEE Computer Society.

Michael L. Cui, Daniel Harabor, and Alban Grastien.
Compromise-free Pathfinding on a Navigation Mesh.
In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2017.

Daniel Harabor, Alban Grastien, Dindar Öz, and Vural Aksakalli.
Optimal Any-angle Pathfinding in Practice.
Journal of Artificial Intelligence Research, 56(1):89–118, May 2016.

References II

Alex Nash, Kenny Daniel, Sven Koenig, and Ariel Felner.
Theta*: Any-Angle Path Planning on Grids.
In *Proceedings of National Conference on Artificial Intelligence (AAAI)*, pages 1177–1183, 2007.