Towards Labeling Moving Points

Approximation Algorithms for Free-Label Maximization

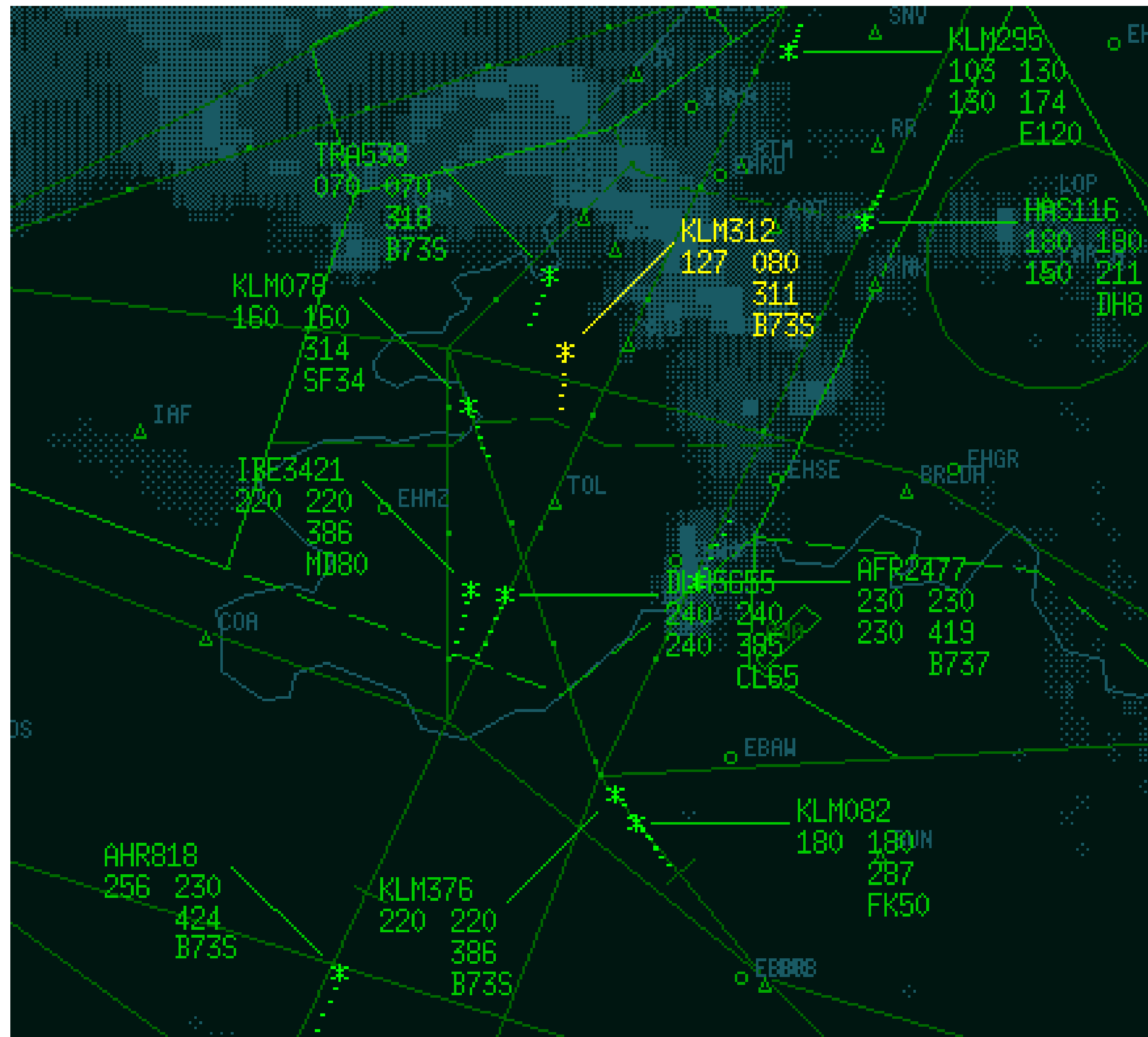
Mark de Berg
(mdberg@win.tue.nl)

Dirk H.P. Gerrits
(dirk@dirkgerrits.com)

22 March 2010

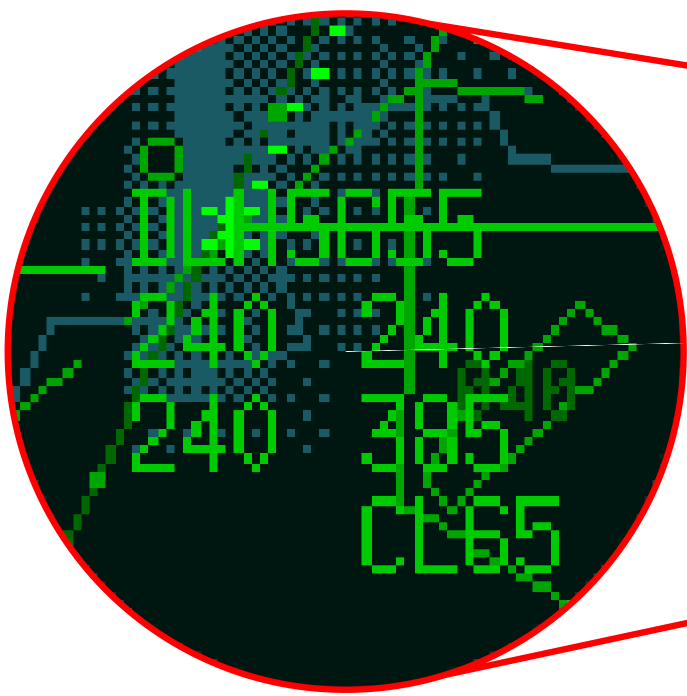
Motivation: air traffic control

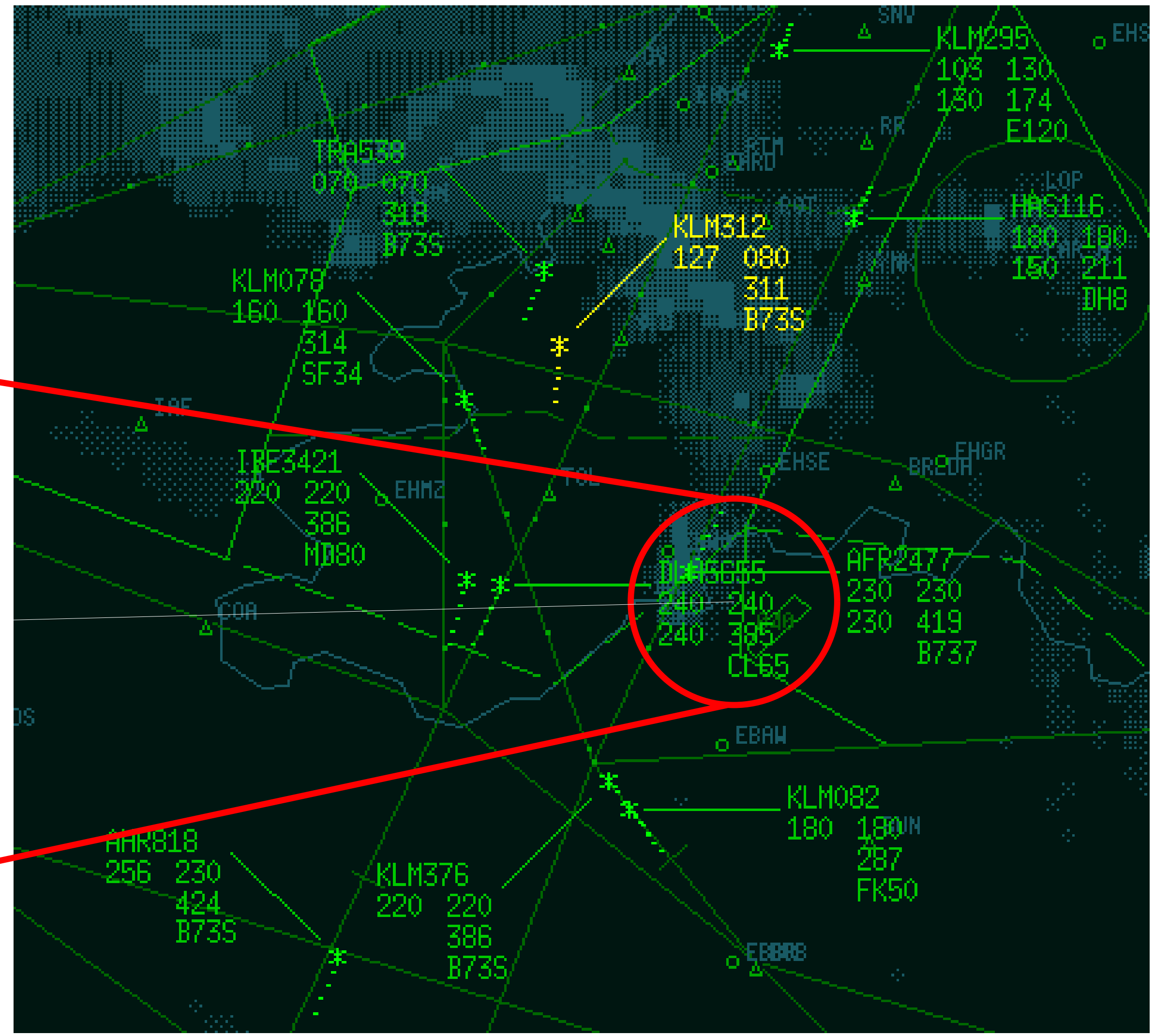
- Air traffic controllers monitor air traffic as moving dots with textual labels, and warn pilots of potential collisions.



Motivation: air traffic control

- Air traffic controllers monitor air traffic as moving dots with textual labels, and warn pilots of potential collisions.
- But they spend a lot of their time manually resolving label overlaps!



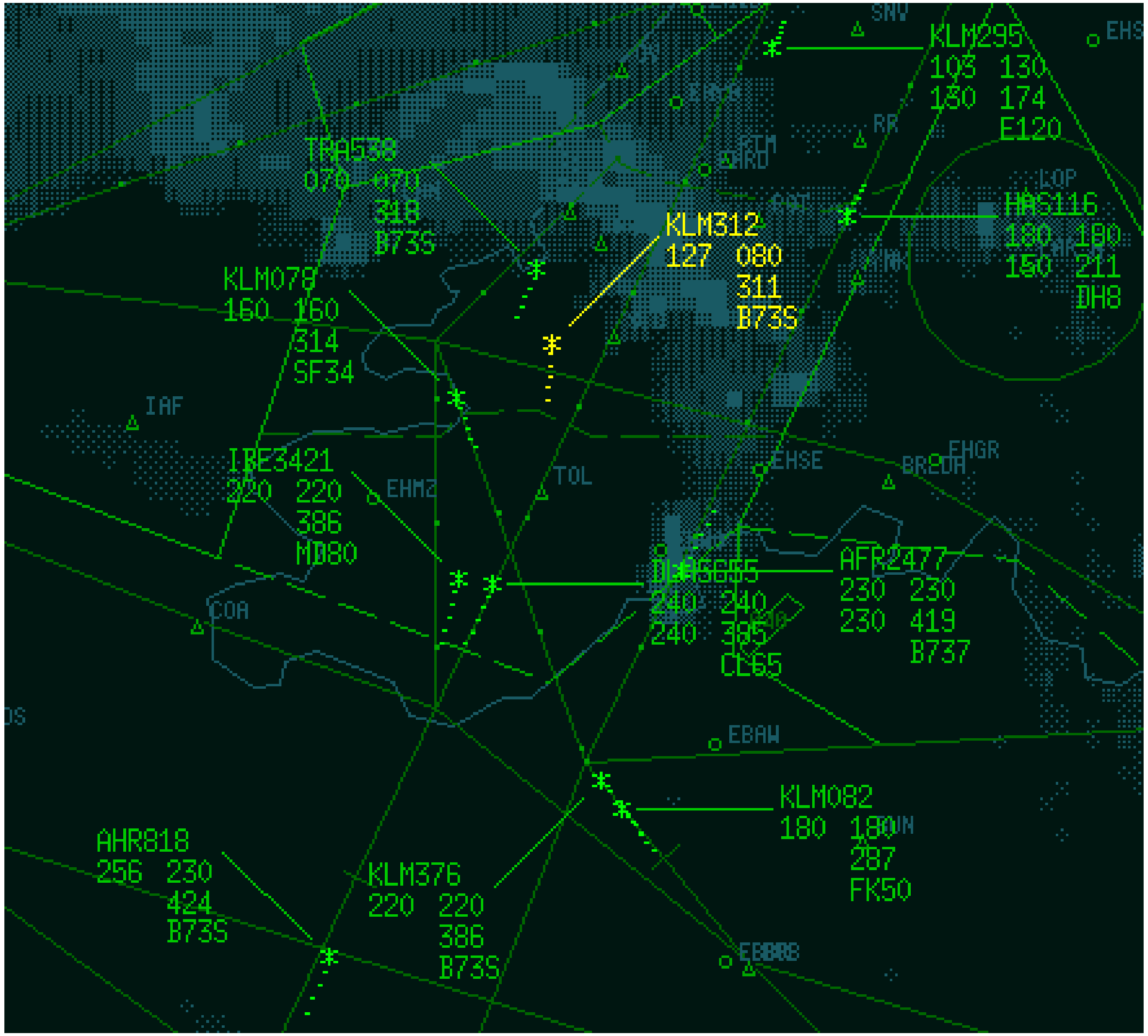


Motivation: air traffic control

- Air traffic controllers monitor air traffic as moving dots with textual labels, and warn pilots of potential collisions.

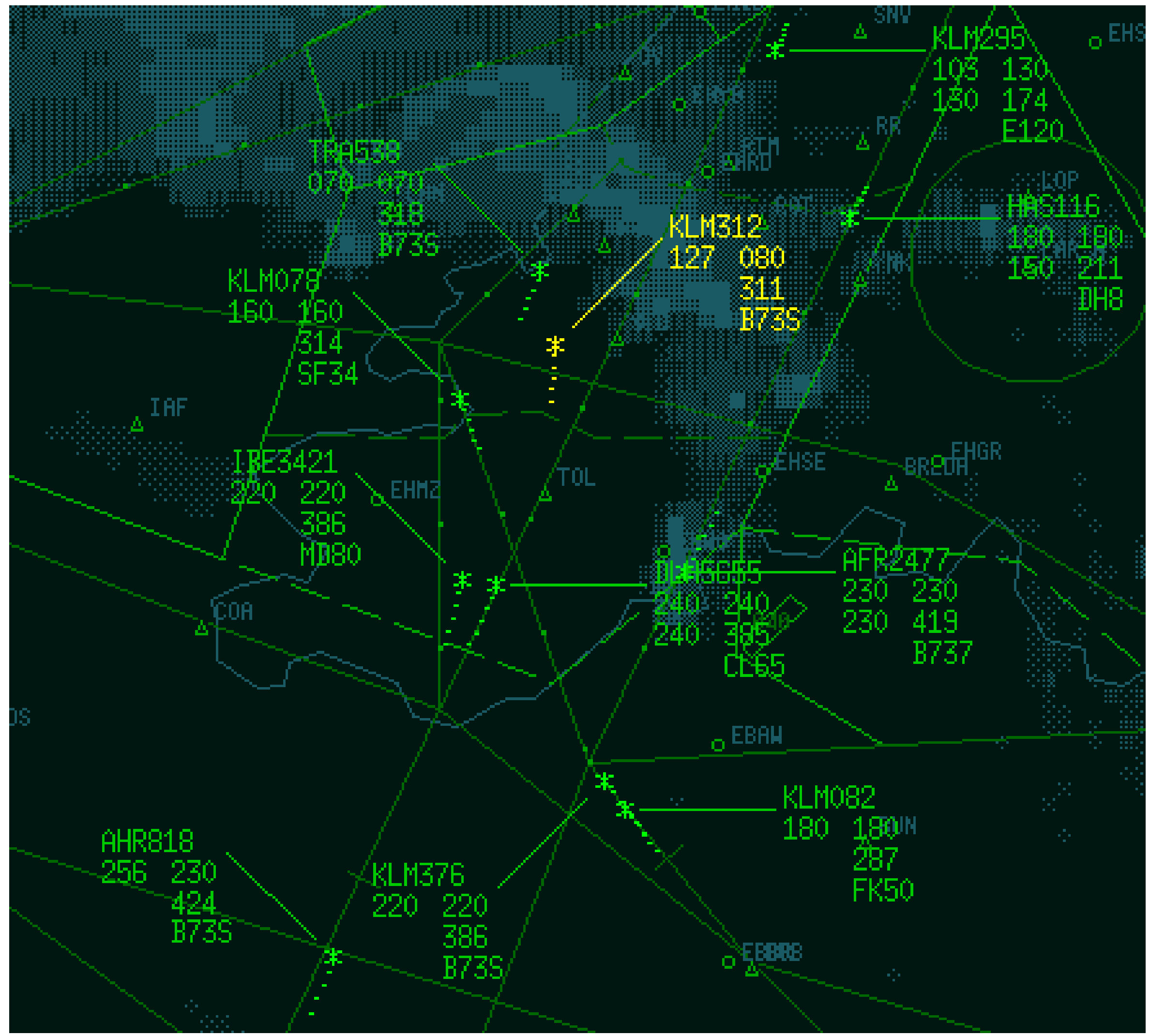
- But they spend a lot of their time manually resolving label overlaps!

- With more air traffic each year, something needs to change.



Motivation: air traffic control

- Air traffic controllers monitor air traffic as moving dots with textual labels, and warn pilots of potential collisions.
- But they spend a lot of their time manually resolving label overlaps!
- With more air traffic each year, something needs to change.
- A lot is known on automated label placement, but only for static points.



Cartographic map labeling

...is the association of (textual) labels with features of a map.

Features can be:

- points (e.g. cities)
- polylines (e.g. rivers)
- polygons (e.g. provinces)

but we only consider points.

Cartographic map labeling

. . . is the association of (textual) labels with features of a map.

Features can be:

- points (e.g. cities)
- polylines (e.g. rivers)
- polygons (e.g. provinces)

but we only consider points.

Quality criteria:

- Clear correspondance between points and labels.
- Labels should be legible.

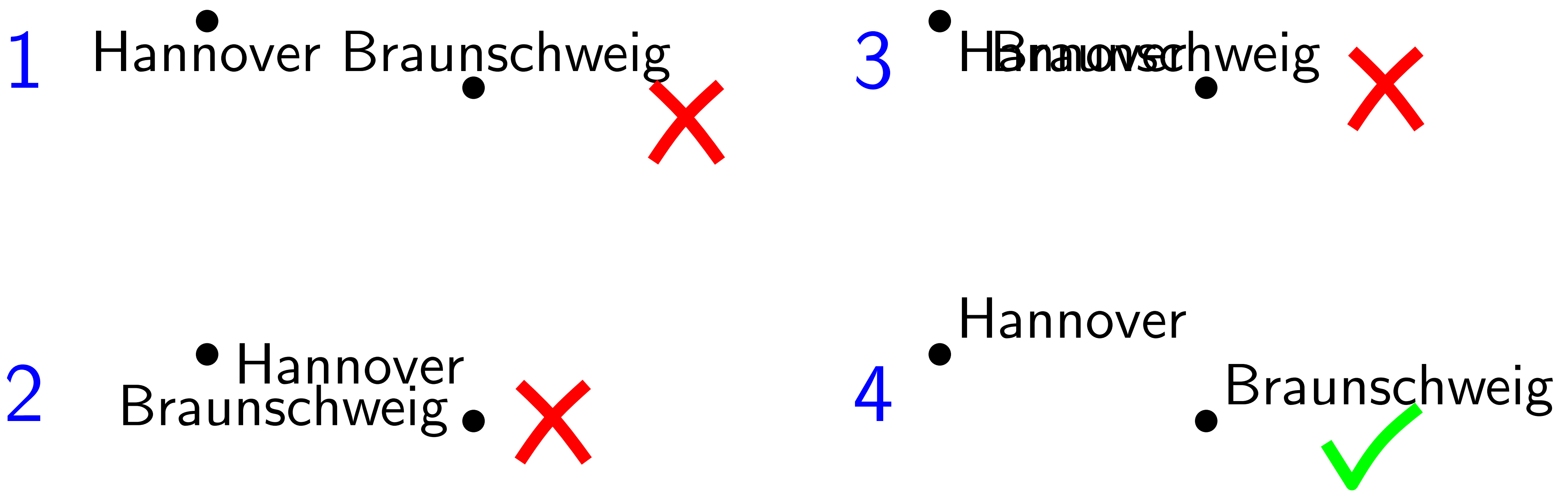
Formalizing the quality criteria

- Clear correspondance between points and labels.
- Labels should be legible.

Formalizing the quality criteria

- Clear correspondance between points and labels.
- Labels should be legible.

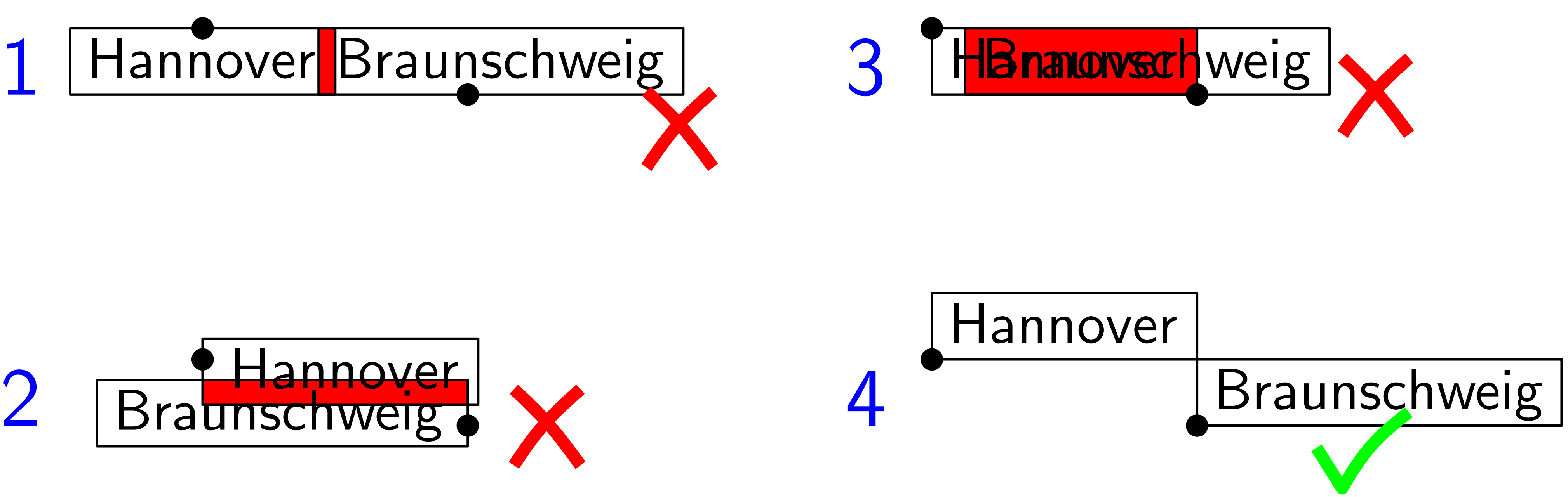
Keeping labels close to points is good, but not enough!



Formalizing the quality criteria

- Clear correspondance between points and labels.
- Labels should be legible.

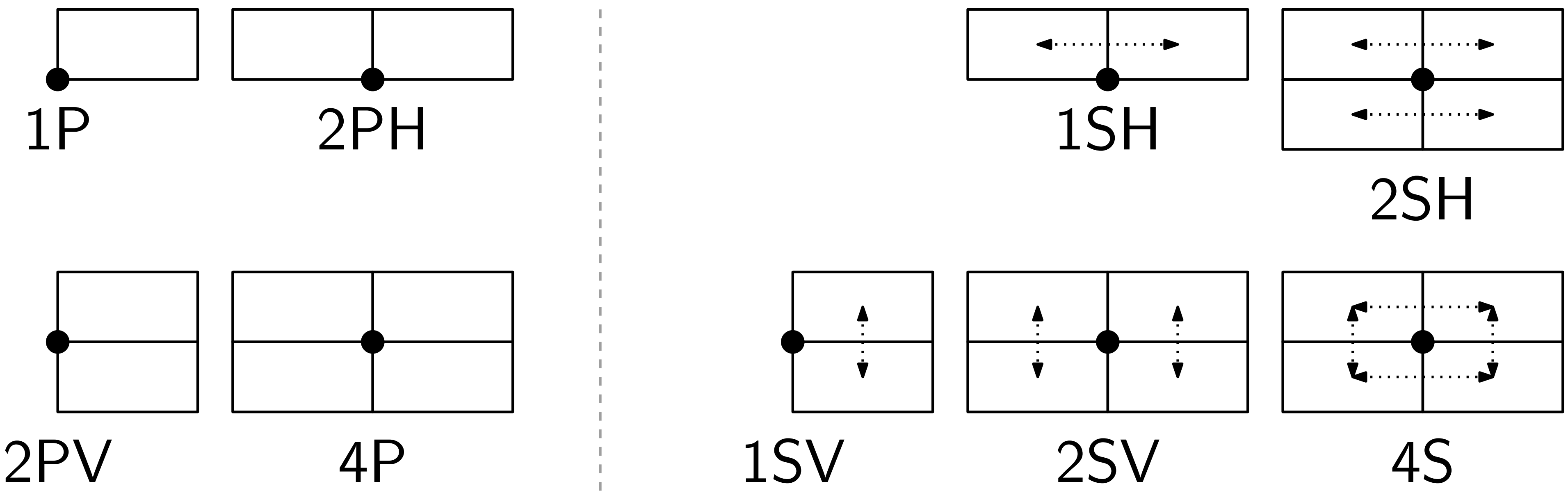
Keeping labels close to points is good, but not enough!



Replace each label by a slightly larger rectangle touching its point, and ensure these rectangles do not intersect each other.

Where to put the labels?

We assume each point has an associated set of rectangular label candidates. What this set looks like is the label model.

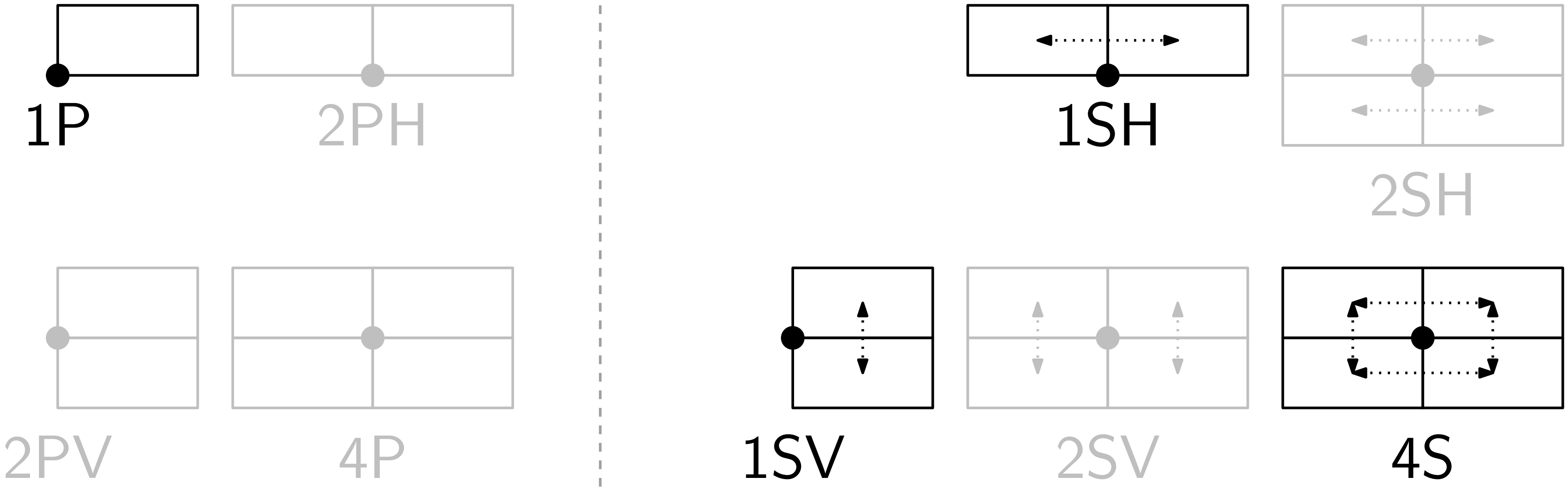


Fixed-position models
(finite set)

Slider models
(uncountably infinite set)

Where to put the labels?

We assume each point has an associated set of rectangular label candidates. What this set looks like is the label model.



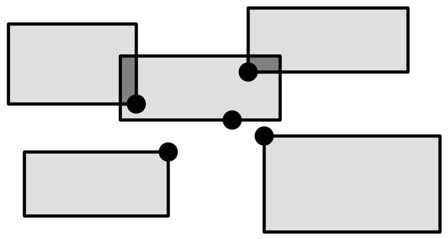
Fixed-position models
(finite set)

Slider models
(uncountably infinite set)

To label moving points continuously, only some models work.

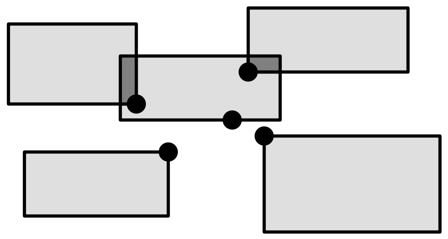
What if we cannot place all labels?

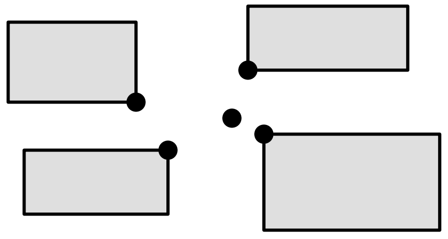
It is not always possible to label all points without intersections



What if we cannot place all labels?

It is not always possible to label all points without intersections





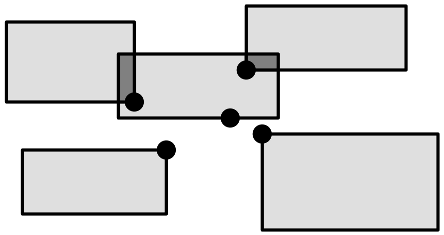
Solution 1: Remove some labels.

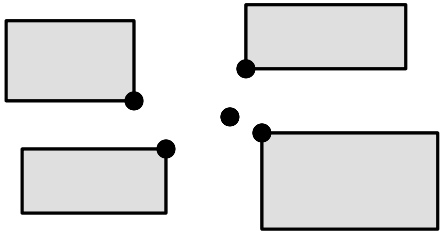
Remove fewest:

Keep most:

What if we cannot place all labels?

It is not always possible to label all points without intersections; NP-hard to decide ()



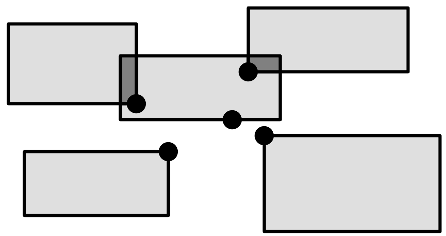


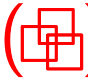
Solution 1: Remove some labels.

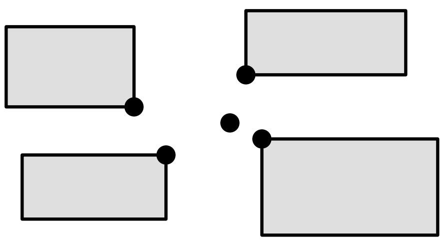
Remove fewest:

Keep most:

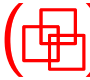
What if we cannot place all labels?

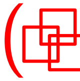
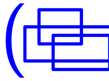
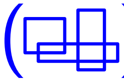


It is not always possible to label all points without intersections; NP-hard to decide ()



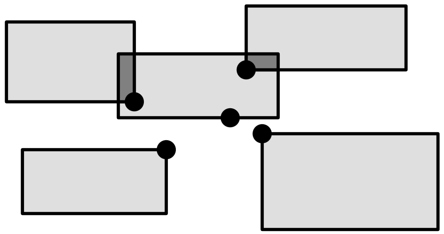
Solution 1: Remove some labels.

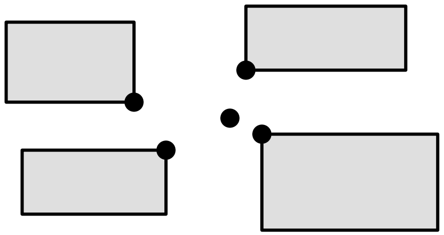
Remove fewest: inapproximable ()

Keep most: NP-hard (), 1/2-apx. & PTAS (), $O(1/\log\log n)$-apx. ()

What if we cannot place all labels?

It is not always possible to label all points without intersections; NP-hard to decide ( )

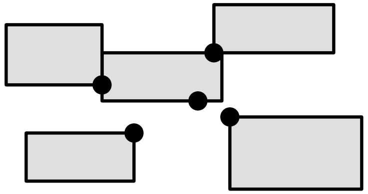




Solution 1: Remove some labels.

Remove fewest: inapproximable ( )

Keep most: NP-hard ( ), 1/2-apx. & PTAS ( ), $O(1/\log\log n)$-apx. ( )

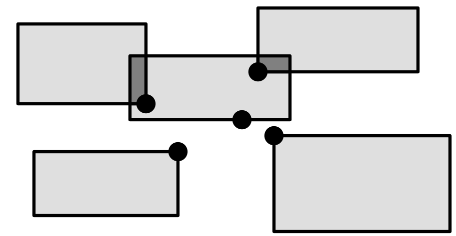


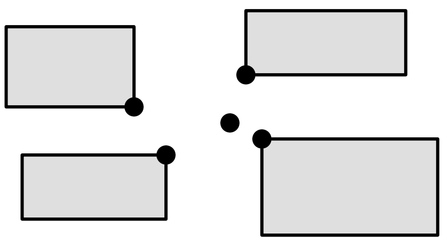
Solution 2: Shrink all labels uniformly.

Largest size: APX-hard (,), 1/2-apx. ( ), $\sim$1/3-apx. ( )

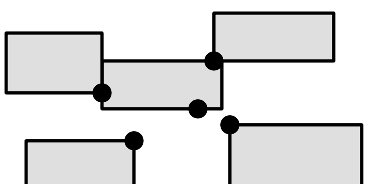
What if we cannot place all labels?

It is not always possible to label all points without intersections; NP-hard to decide ()



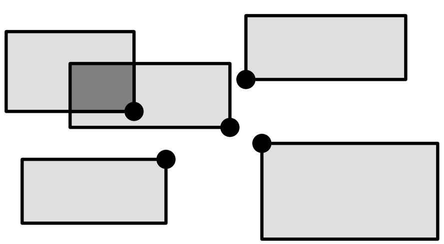


Solution 1: Remove some labels.
- Remove fewest: inapproximable ()
- Keep most: NP-hard (), 1/2-apx. & PTAS (), $O(1/\log\log n)$-apx. ()

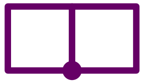
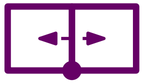


Solution 2: Shrink all labels uniformly.
- Largest size: APX-hard (,), 1/2-apx. (), $\sim$1/3-apx. ()

European aviation safety regulations disallow both of the above!

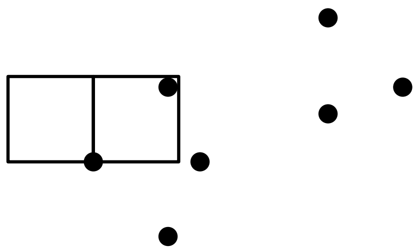


Solution 3: Allow label intersections.
- Fewest intersected: inapproximable ()
- Most free: NP-hard (), [1/24, 1/4]-apx. & PTAS () ← our results

$O(n \log n)$-time 1/4-approximation for , 

For each point p from left to right:

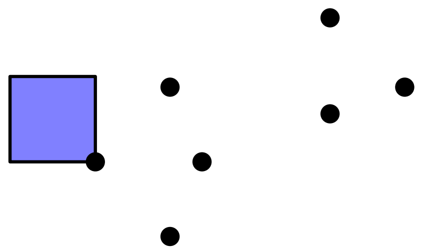
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.

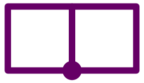
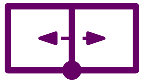


$O(n \log n)$-time 1/4-approximation for ,

For each point p from left to right:

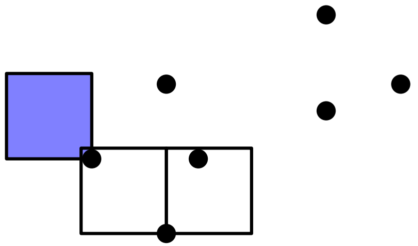
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.



$O(n \log n)$-time 1/4-approximation for , 

For each point p from left to right:

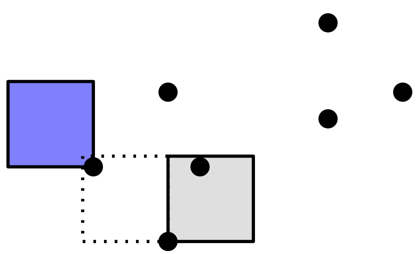
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.

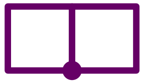
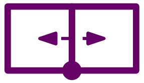


$O(n \log n)$-time 1/4-approximation for 

For each point p from left to right:

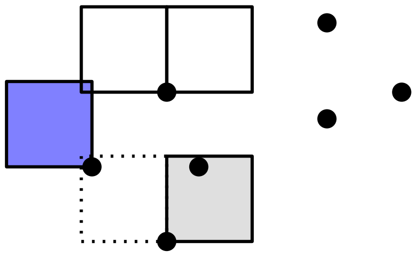
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.

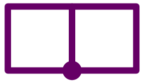
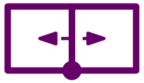


$O(n \log n)$-time 1/4-approximation for , 

For each point p from left to right:

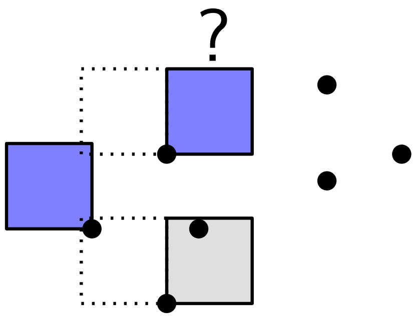
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.

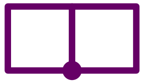
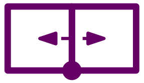


$O(n \log n)$-time 1/4-approximation for , 

For each point p from left to right:

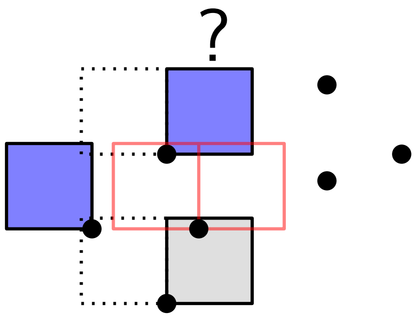
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.



$O(n \log n)$-time 1/4-approximation for , 

For each point p from left to right:

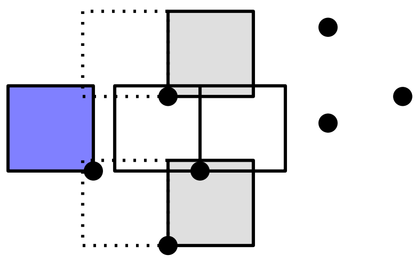
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.



$O(n \log n)$-time 1/4-approximation for

For each point p from left to right:

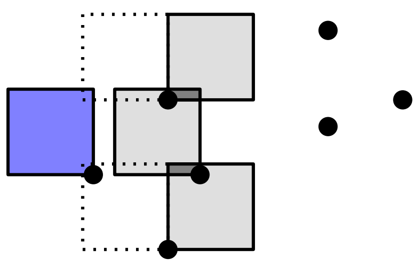
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.



$O(n \log n)$-time $1/4$-approximation for ,

For each point p from left to right:

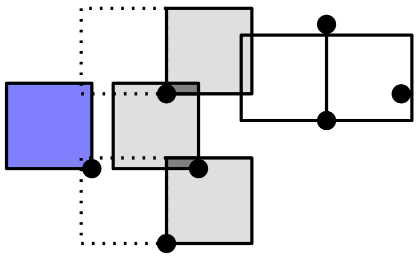
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.

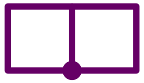
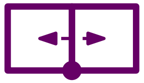


$O(n \log n)$-time 1/4-approximation for

For each point p from left to right:

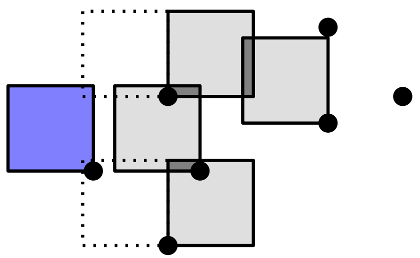
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.

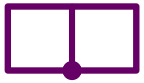
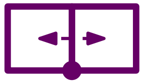


$O(n \log n)$-time 1/4-approximation for , 

For each point p from left to right:

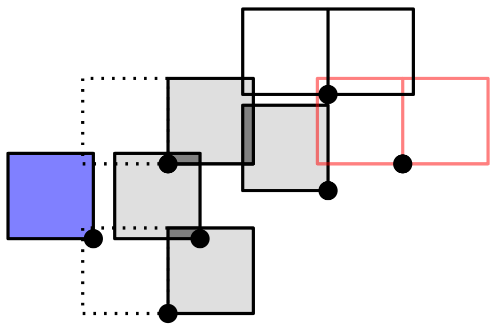
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.

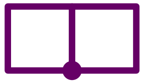
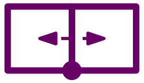


$O(n \log n)$-time $1/4$-approximation for , 

For each point p from left to right:

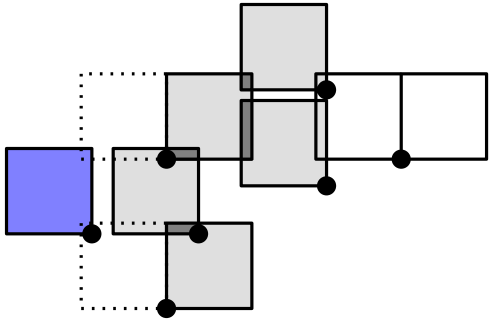
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.

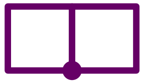
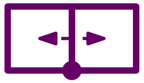


$O(n \log n)$-time 1/4-approximation for , 

For each point p from left to right:

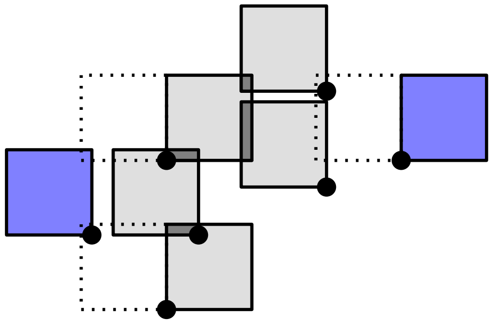
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.

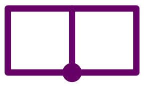
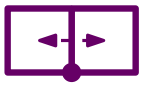


$O(n \log n)$-time $1/4$-approximation for , 

For each point p from left to right:

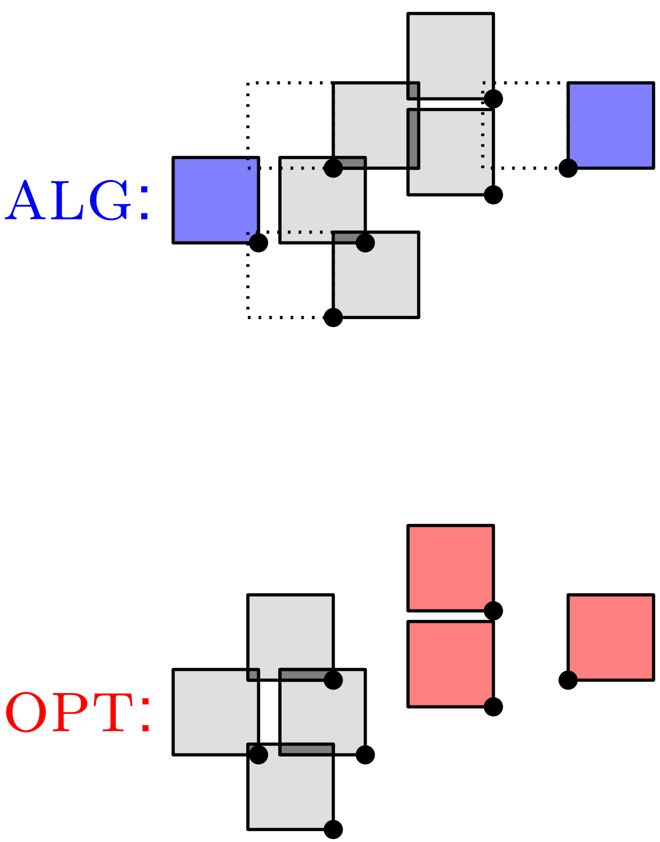
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.

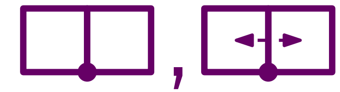


$O(n \log n)$-time 1/4-approximation for , 

For each point p from left to right:

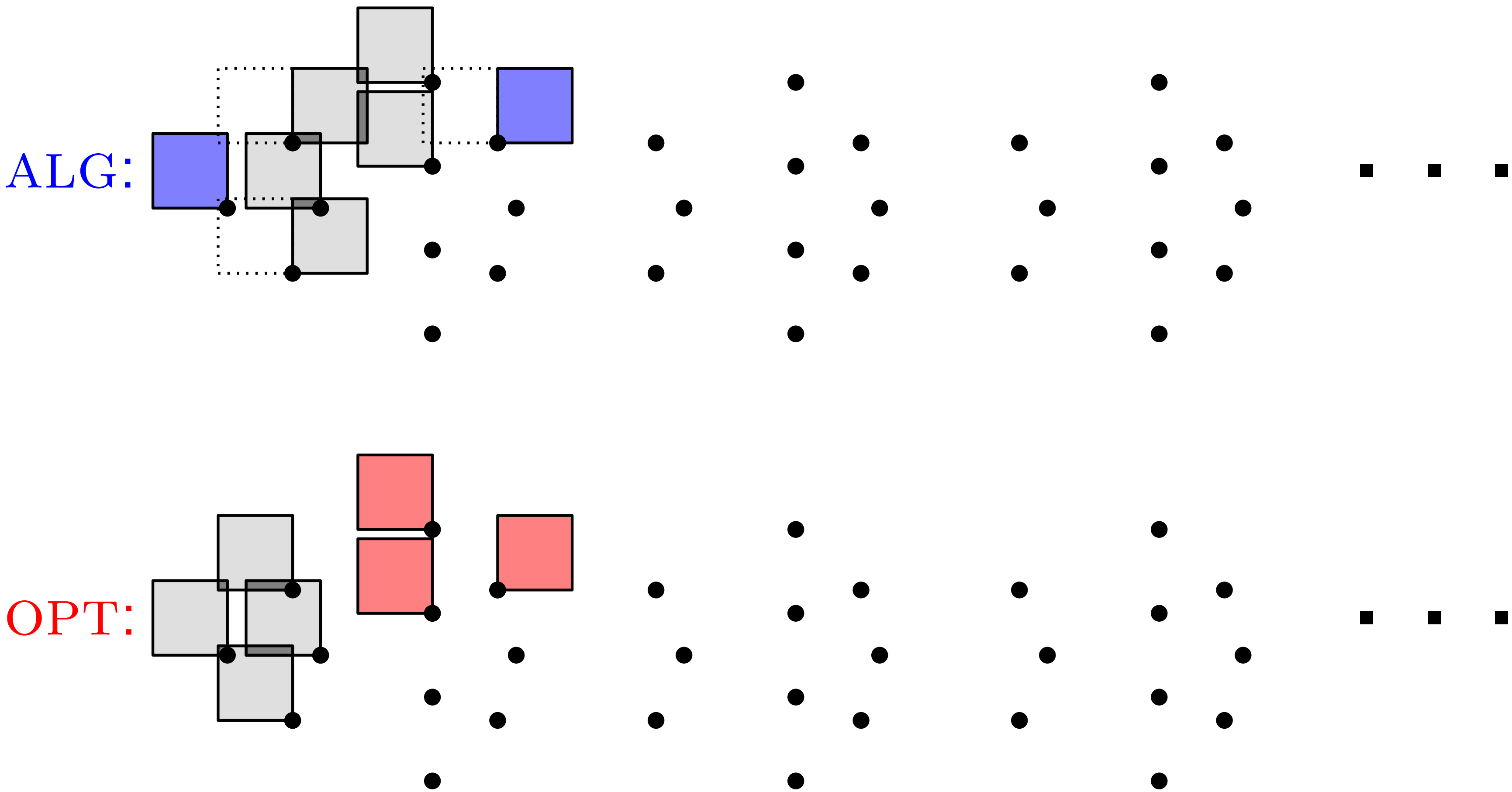
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.



$O(n \log n)$-time 1/4-approximation for 

For each point p from left to right:

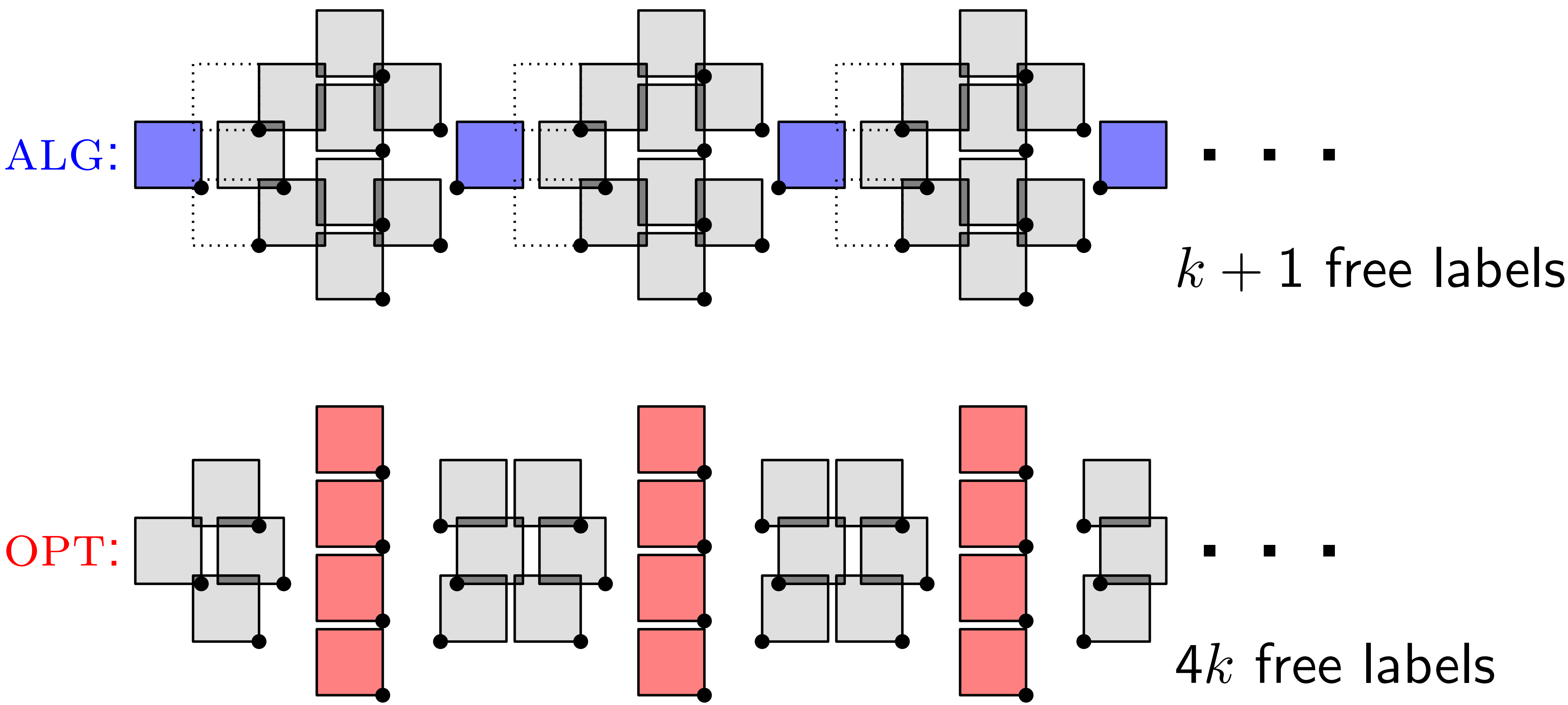
- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.

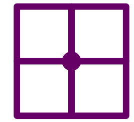
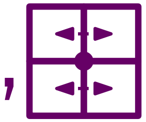
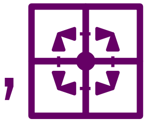


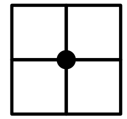
$O(n \log n)$-time 1/4-approximation for ,

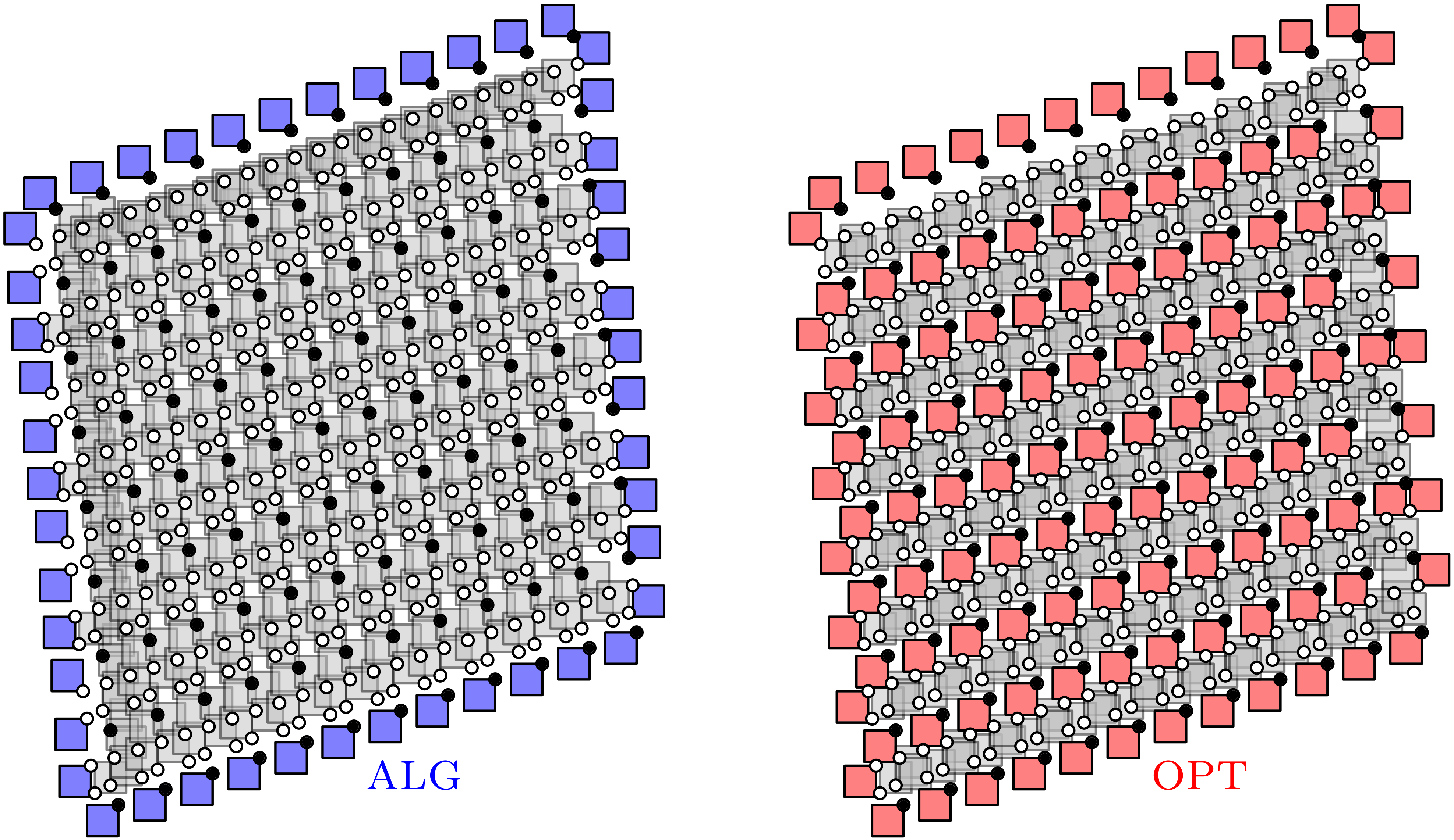
For each point p from left to right:

- If we can give p a label that is and will remain free, do so.
- Otherwise, give p a label that does not need to remain free.
- On ties, pick the leftmost label candidate.

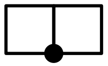
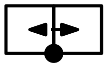
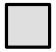
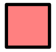
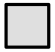


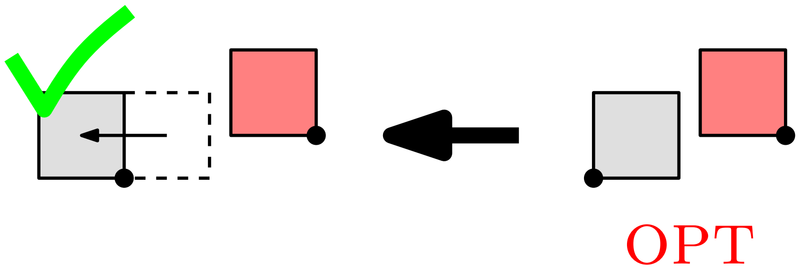
No O(1)-approximation for , , 

Same greedy algorithm for ? $\rightarrow O(1/\sqrt{n})$-approximation!

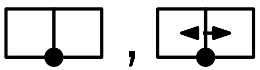


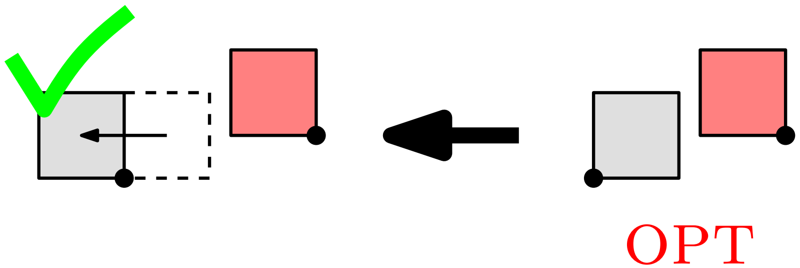
Why not? What is the difference?

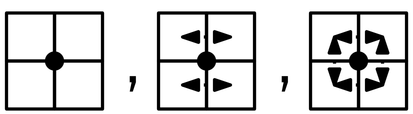
, : Take OPT and make some label  leftmost. No free label  right of  can become intersected.

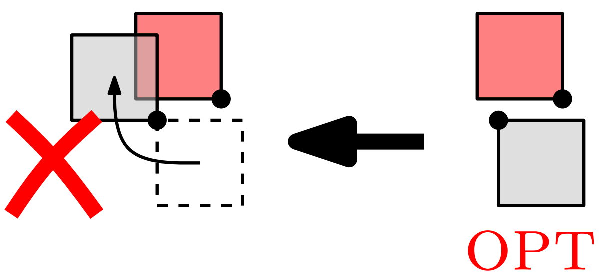


Why not? What is the difference?

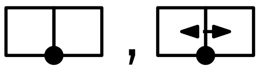
: Take OPT and make some label leftmost. No free label right of can become intersected.

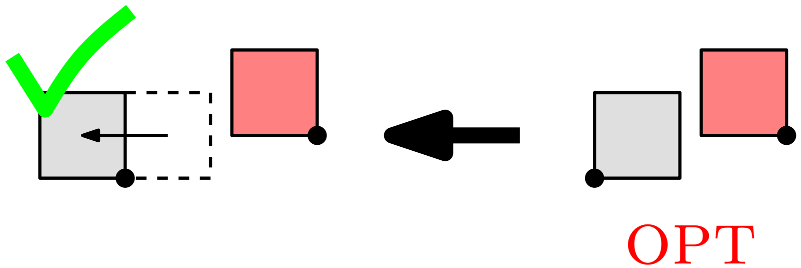


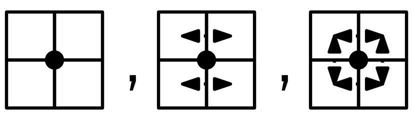
: No longer true...

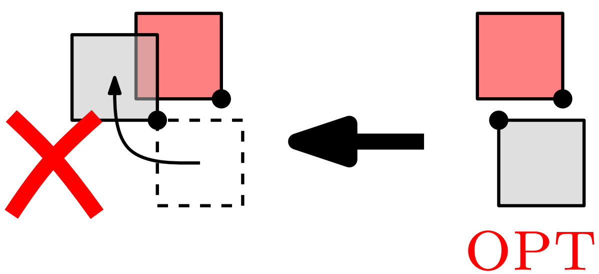


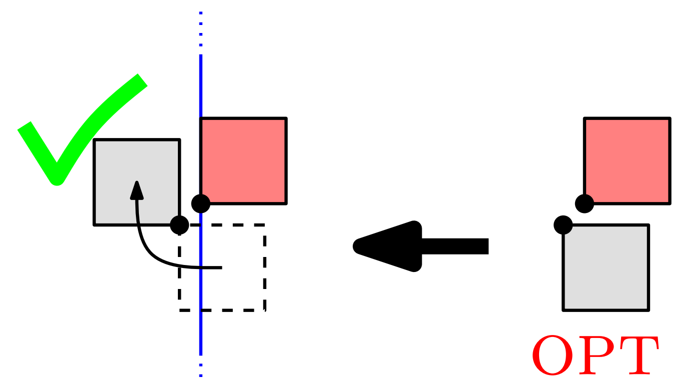
Why not? What is the difference?

: Take OPT and make some label [grey square] leftmost. No free label [red square] right of [grey square] can become intersected.

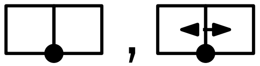


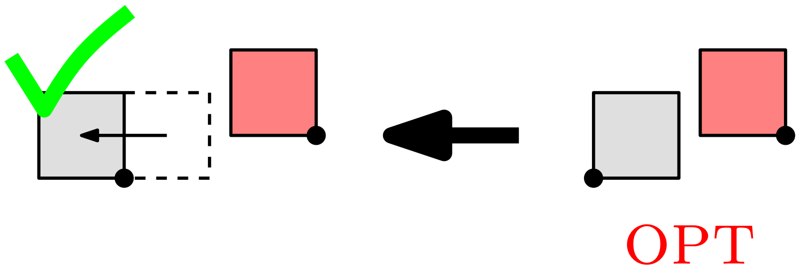
: No longer true. . . unless the free labels of OPT are rightmost.

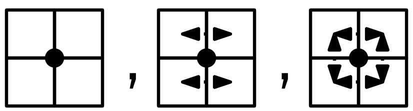


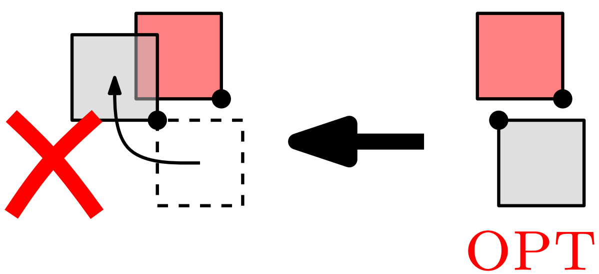


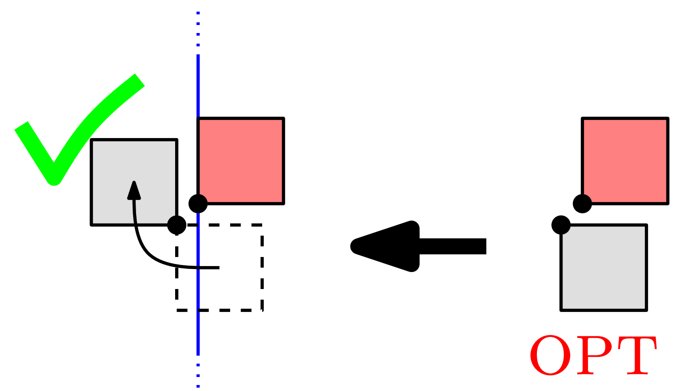
Why not? What is the difference?

: Take OPT and make some label leftmost. No free label right of can become intersected.



: No longer true...unless the free labels of OPT are rightmost.

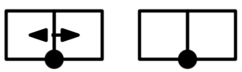
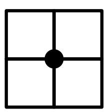
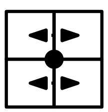
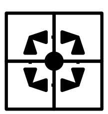




Idea: Take the best solution over several sweep directions.
A fraction of OPT's free labels will have the right position.

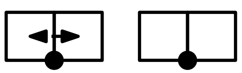
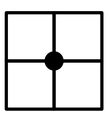
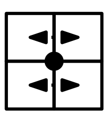
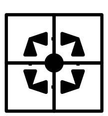
O(1)-approximations for all label models

Idea: Take the best solution over several sweep directions.

Model	Algorithm	Approximation ratio
	Sweep $1\times$ $(\rightarrow)$	$1/1 \cdot 1/4 = 1/4$
	Sweep $2\times$ $(\leftarrow, \rightarrow)$	$1/2 \cdot 1/4 = 1/8$?
	Sweep $2\times$ $(\uparrow, \downarrow)$	$1/2 \cdot 1/4 = 1/8$?
	Sweep $4\times$ $(\leftarrow, \uparrow, \downarrow, \rightarrow)$	$1/4 \cdot 1/4 = 1/16$?

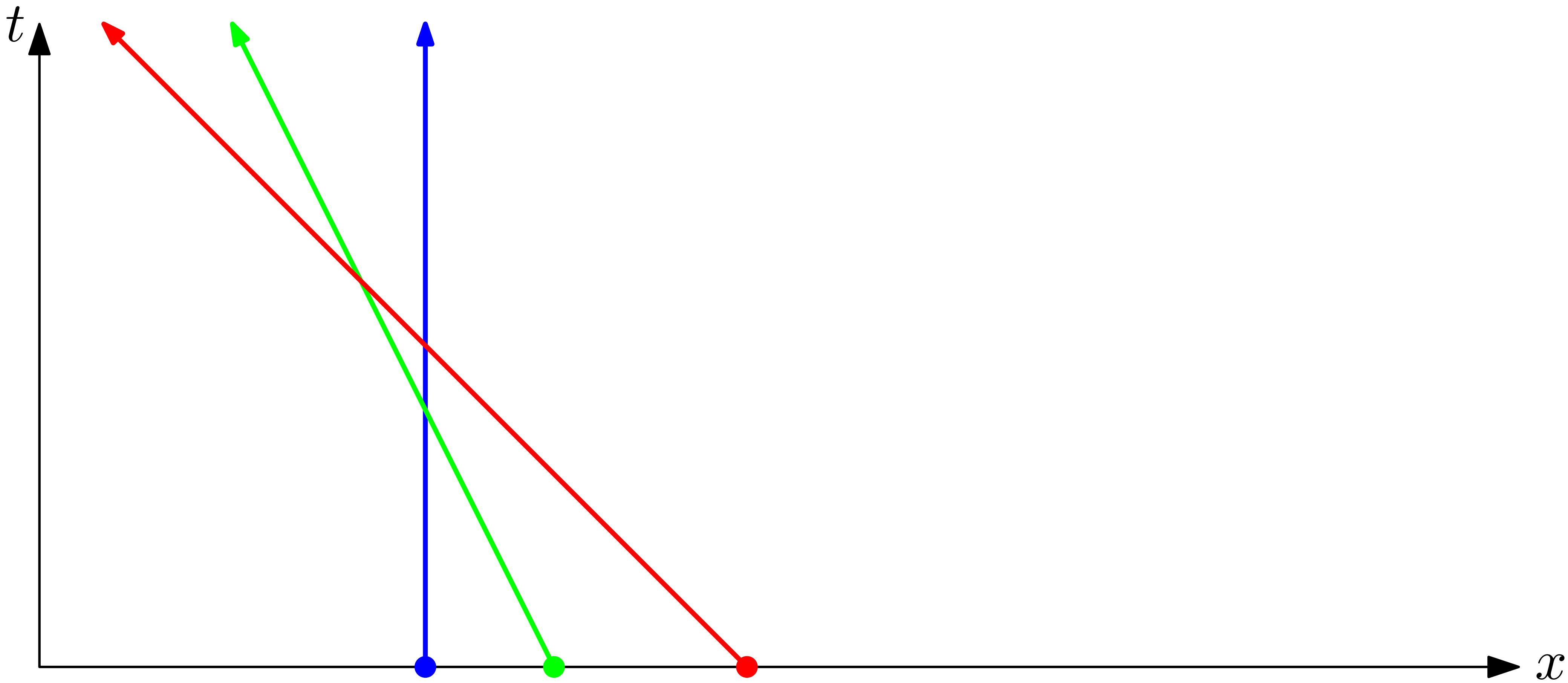
O(1)-approximations for all label models

Idea: Take the best solution over several sweep directions.

Model	Algorithm	Approximation ratio
	Sweep $1\times$ $(\rightarrow)$	$1/1 \cdot 1/4 = 1/4$
	Sweep $2\times$ $(\leftarrow, \rightarrow)$	$1/2 \cdot 1/8 = 1/16$
	Sweep $2\times$ $(\uparrow, \downarrow)$	$1/2 \cdot 1/6 = 1/12$
	Sweep $4\times$ $(\leftarrow, \uparrow, \downarrow, \rightarrow)$	$1/4 \cdot 1/6 = 1/24$

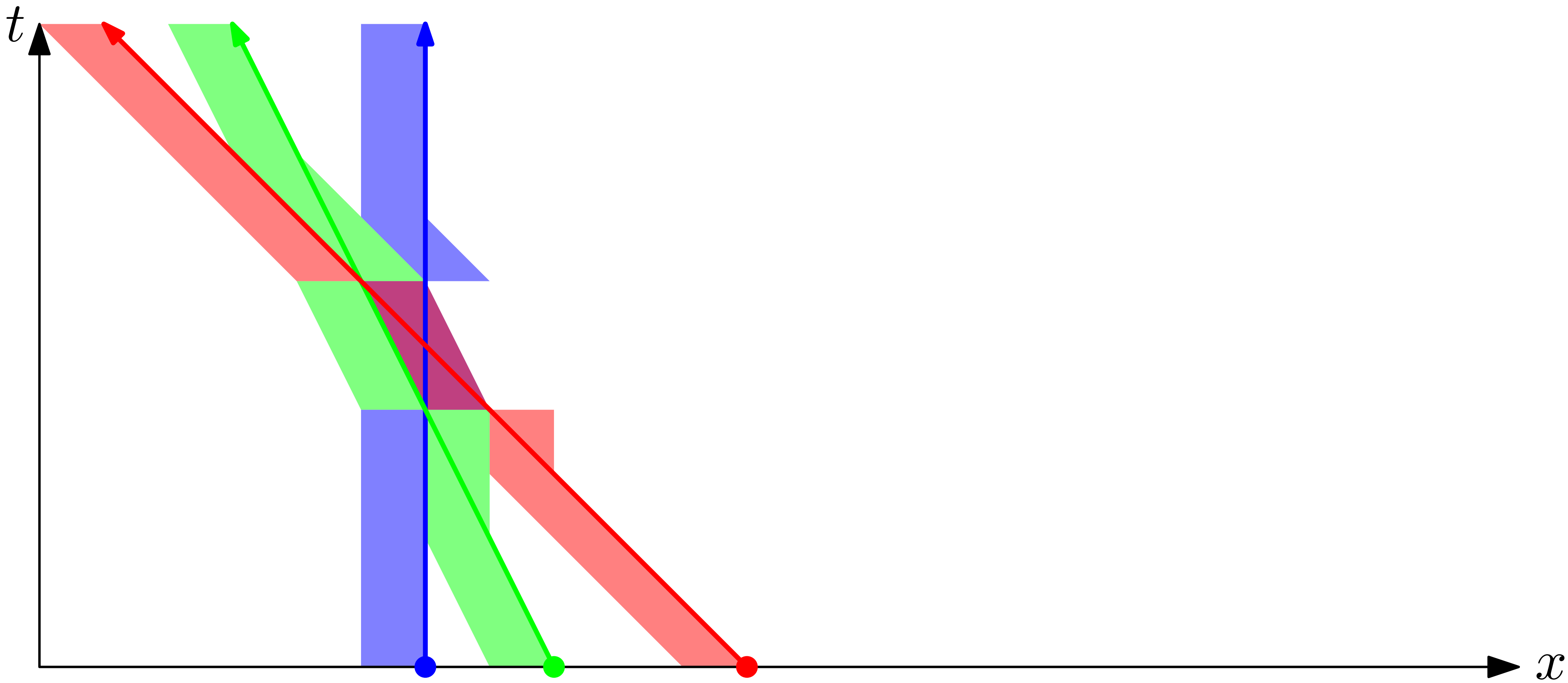
Towards labeling moving points

Idea: Kinetically maintain our greedy solution



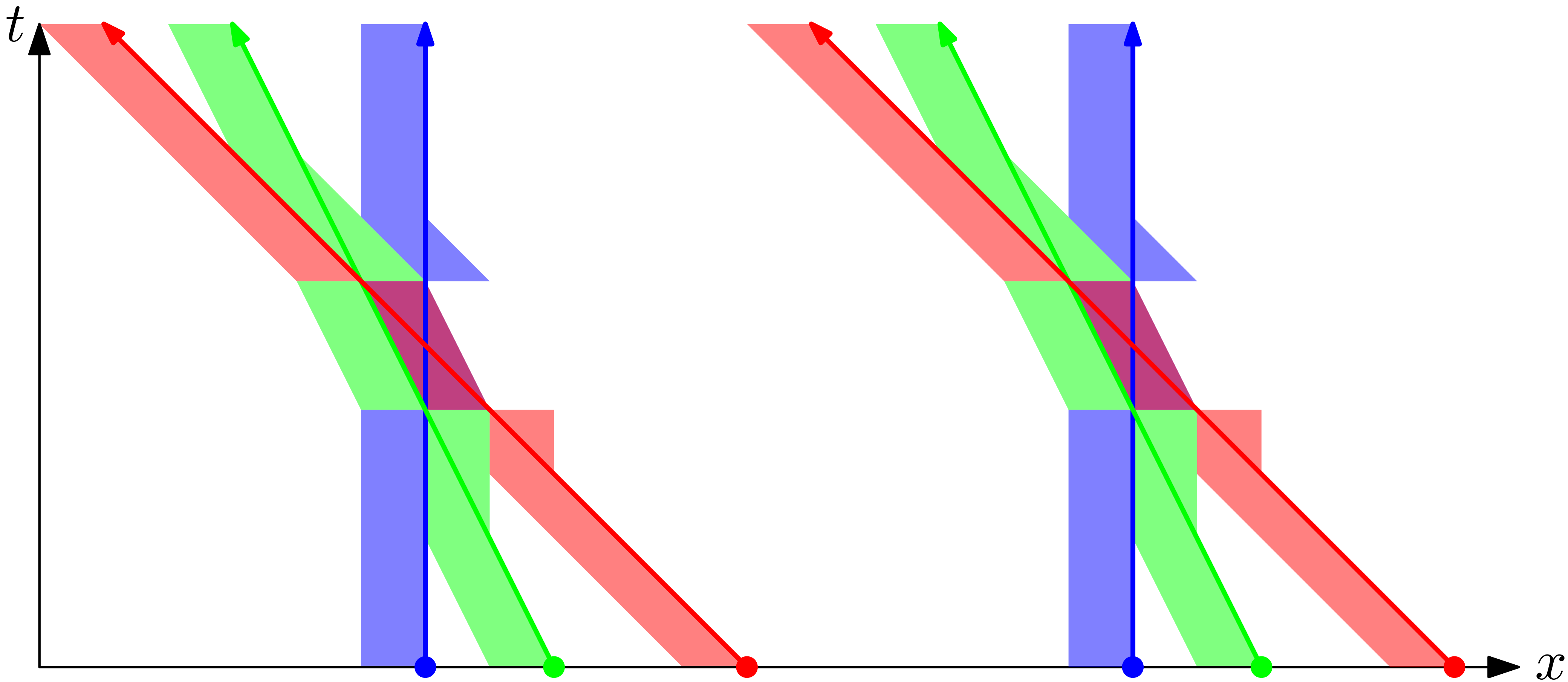
Towards labeling moving points

Idea: Kinetically maintain our greedy solution



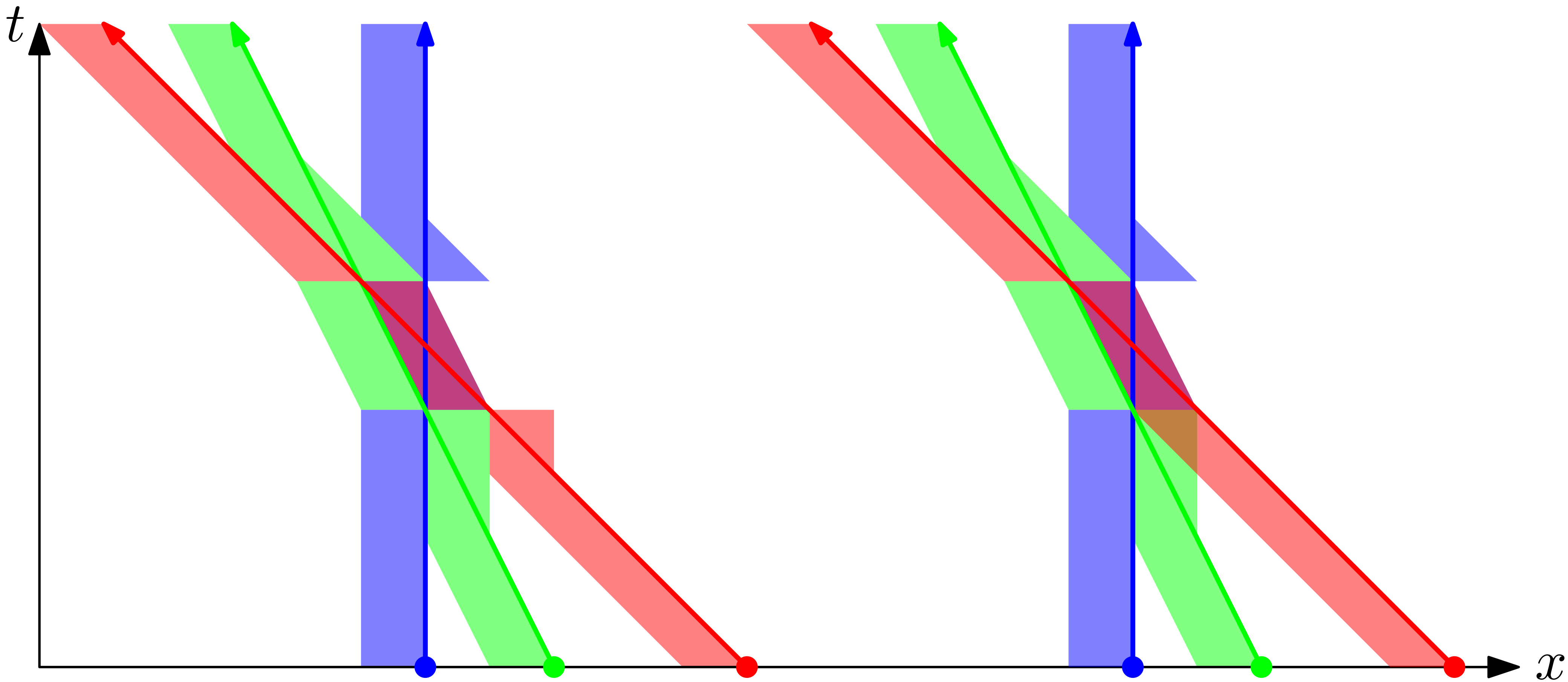
Towards labeling moving points

Idea: Kinetically maintain our greedy solution, removing any discontinuities.



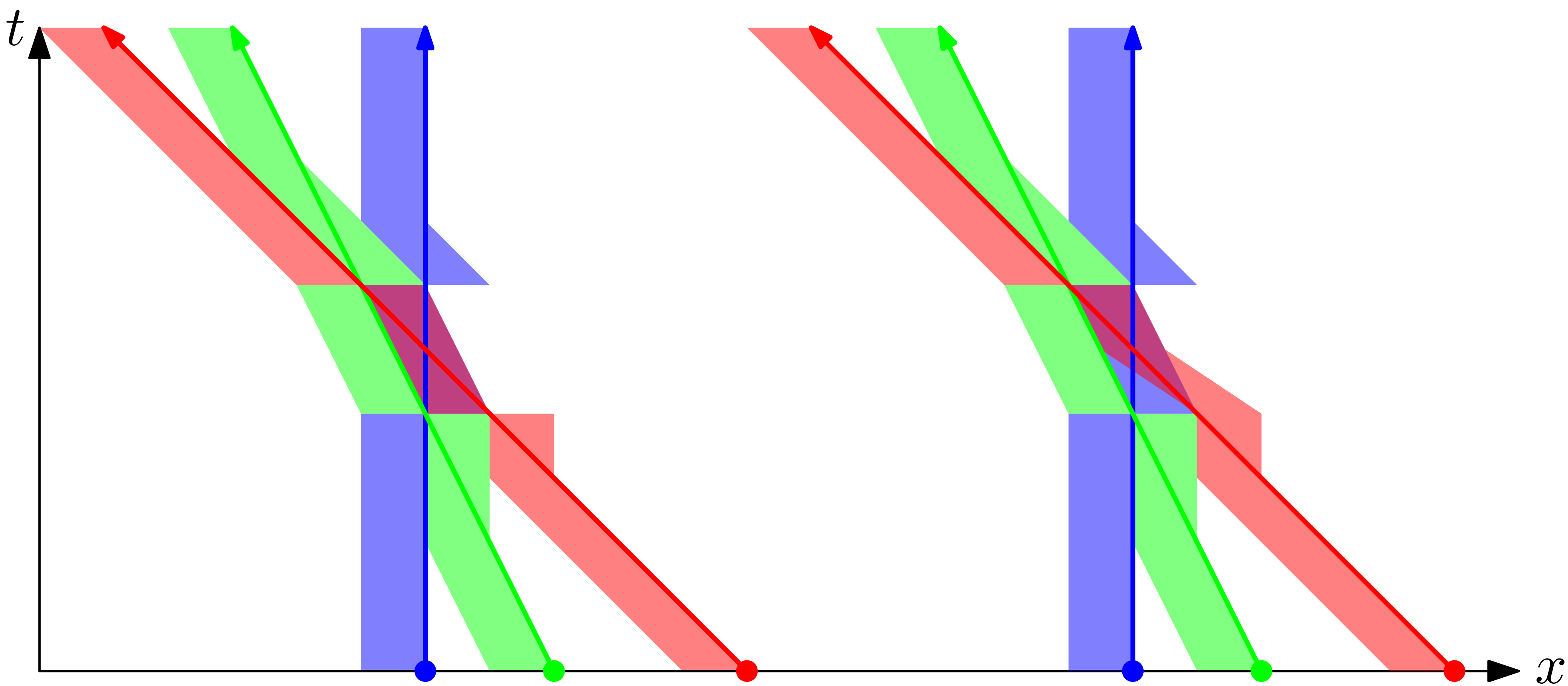
Towards labeling moving points

Idea: Kinetically maintain our greedy solution, removing any discontinuities.



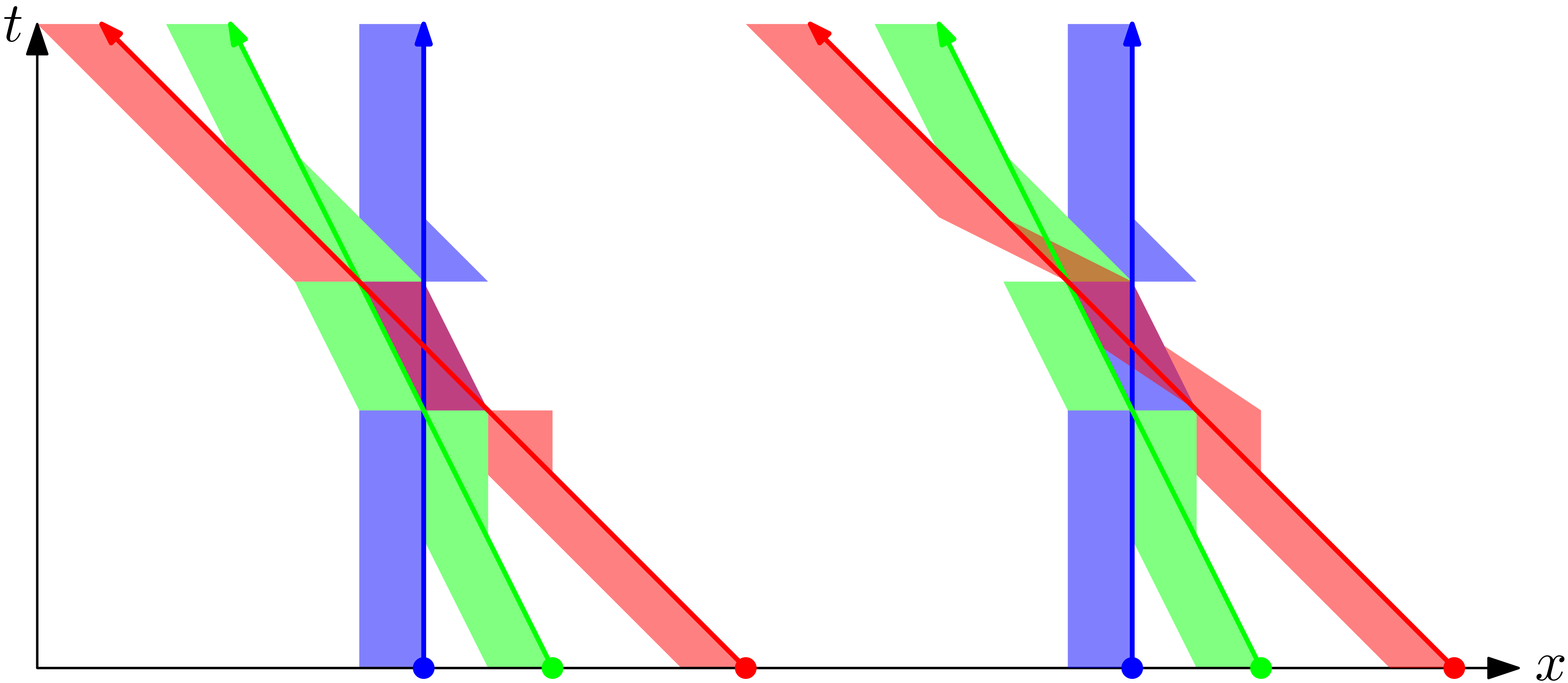
Towards labeling moving points

Idea: Kinetically maintain our greedy solution, removing any discontinuities.



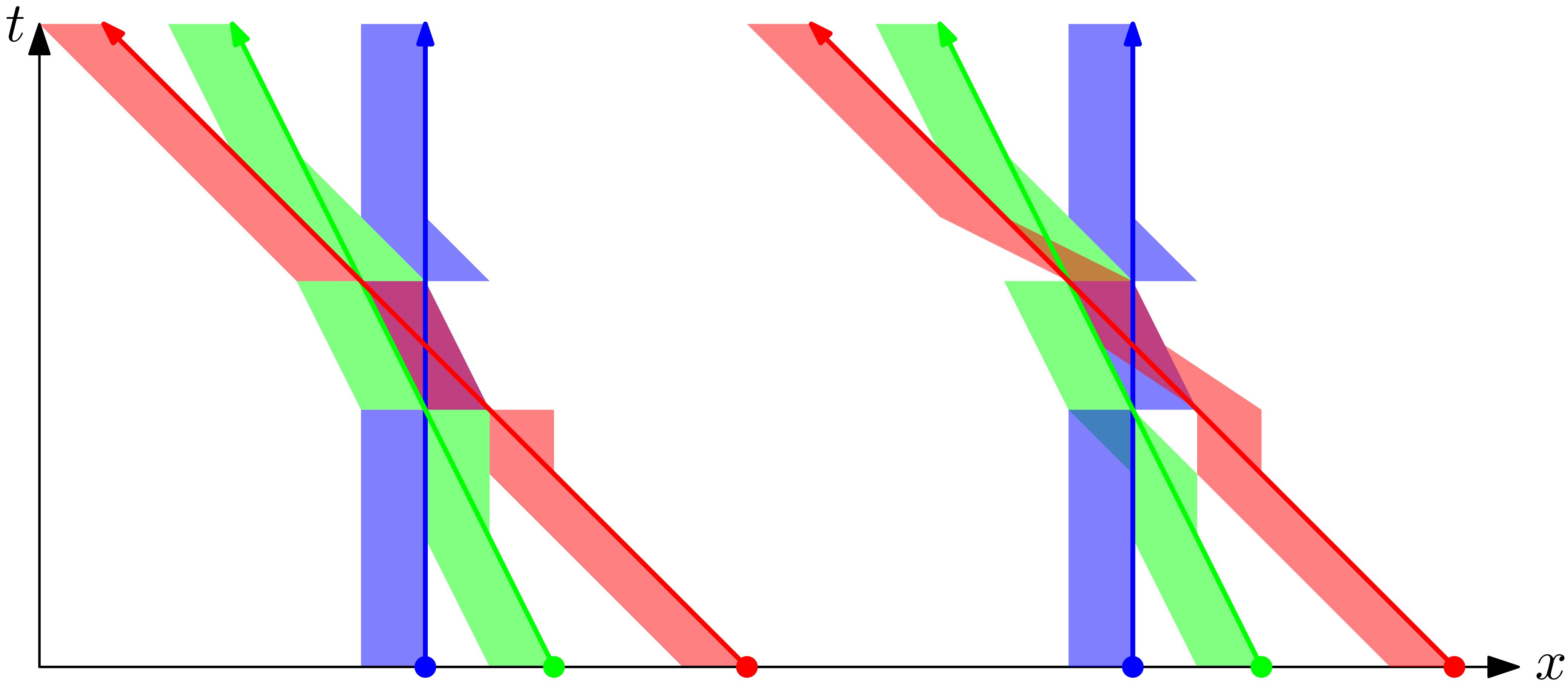
Towards labeling moving points

Idea: Kinetically maintain our greedy solution, removing any discontinuities.



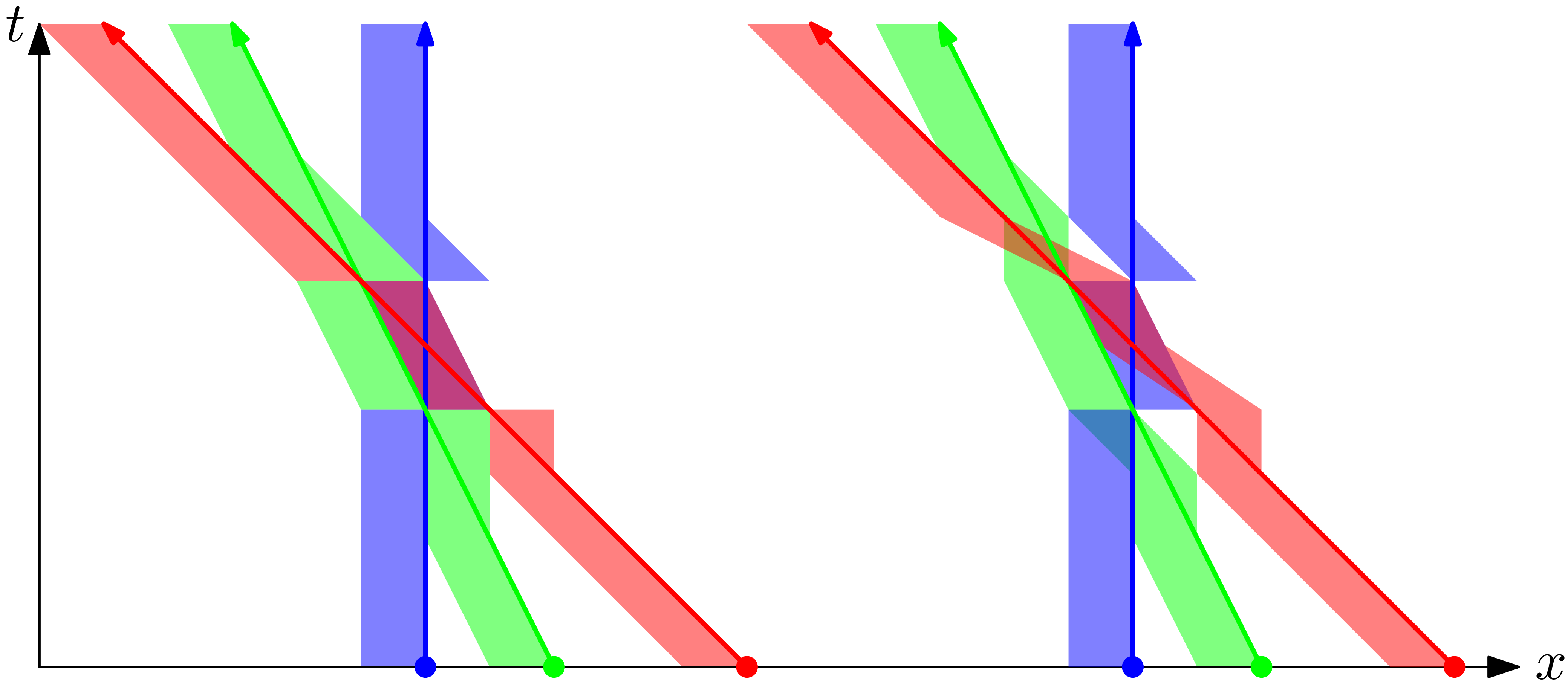
Towards labeling moving points

Idea: Kinetically maintain our greedy solution, removing any discontinuities.



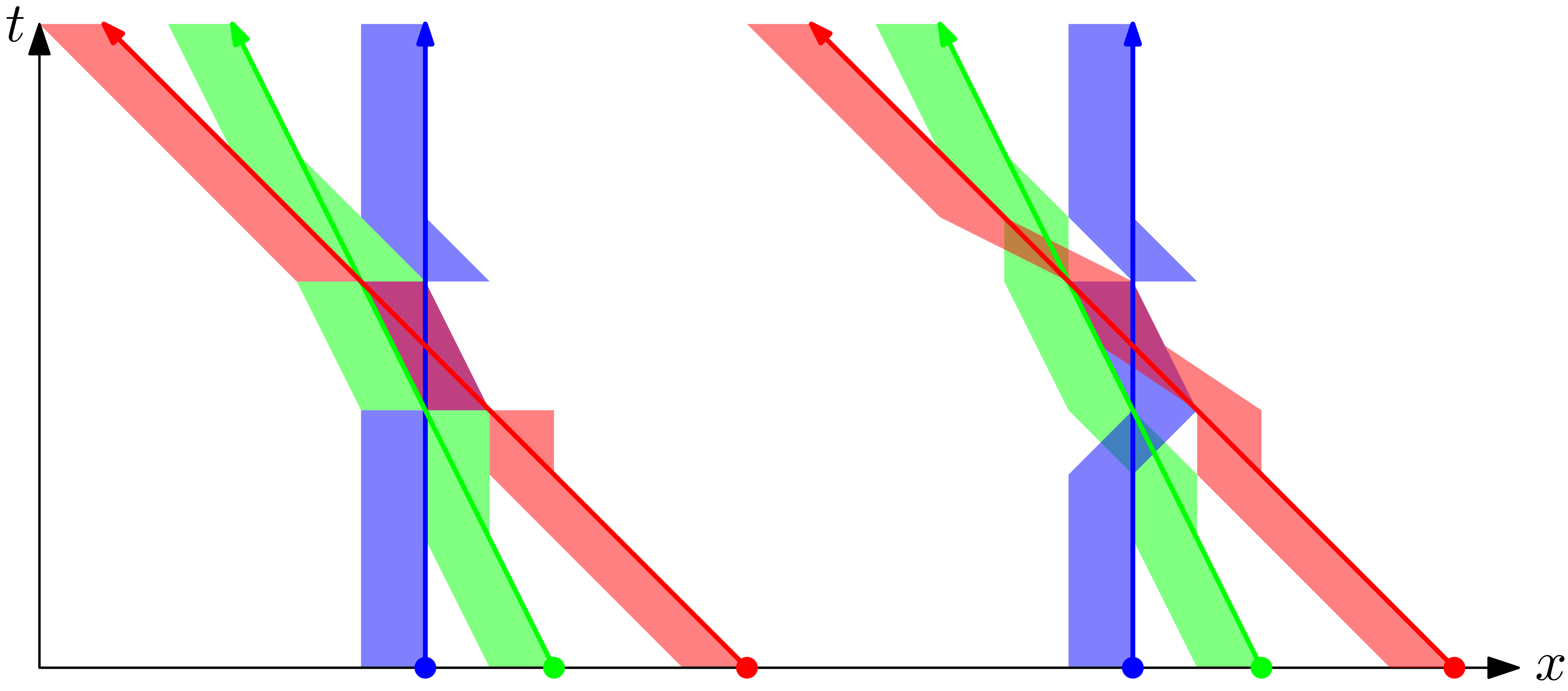
Towards labeling moving points

Idea: Kinetically maintain our greedy solution, removing any discontinuities.



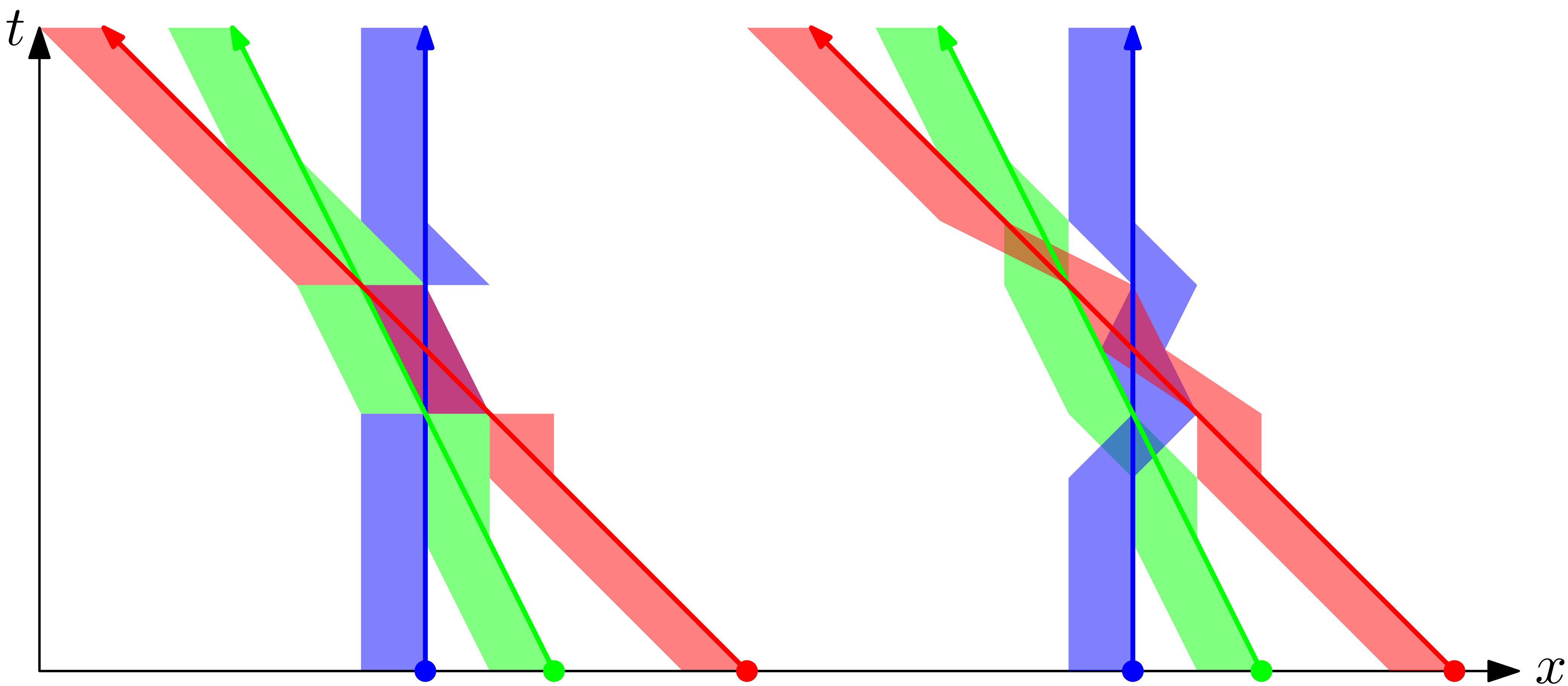
Towards labeling moving points

Idea: Kinetically maintain our greedy solution, removing any discontinuities.



Towards labeling moving points

Idea: Kinetically maintain our greedy solution, removing any discontinuities.



Towards labeling moving points

Idea: Kinetically maintain our greedy solution, removing any discontinuities.

Questions: How to remove discontinuities in a good way?
What does "good" even mean?