

# Block Sorting is Hard

Wolfgang W. Bein

Lawrence L. Larmore

University of Nevada, Las Vegas  
Department of Computer Science

I. Hal Sudborough

University of Texas at Dallas  
Department of Computer Science

Shahram Latifi

University of Nevada, Las Vegas  
Department of Electrical and Computer Engineering

May 2002



## What is Block Sorting?

How	did	they
do	it	?

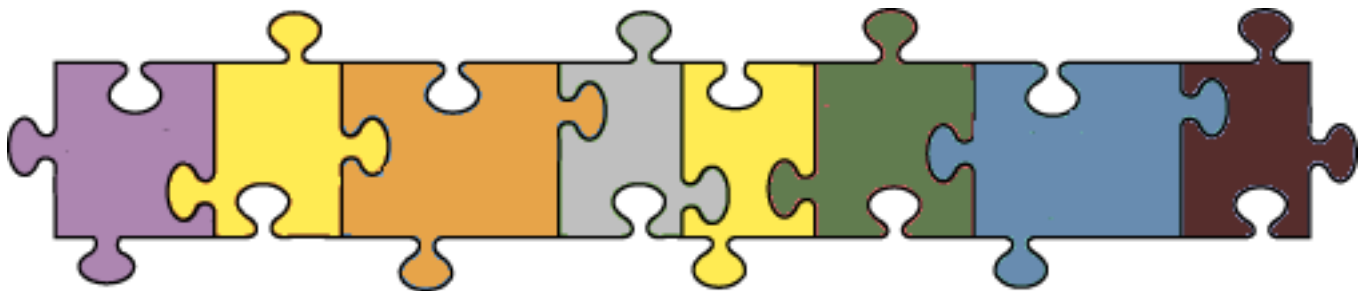
How ? they did it do

How they did it ? do

How they did  do

do

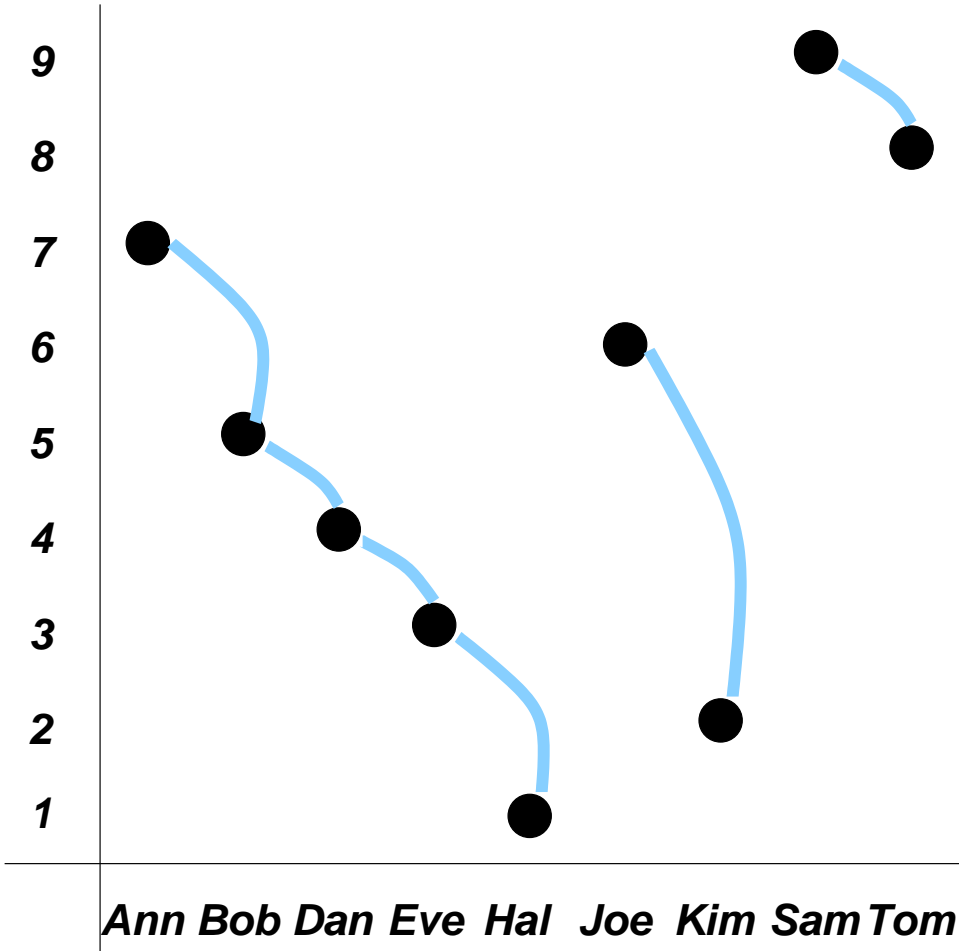
do



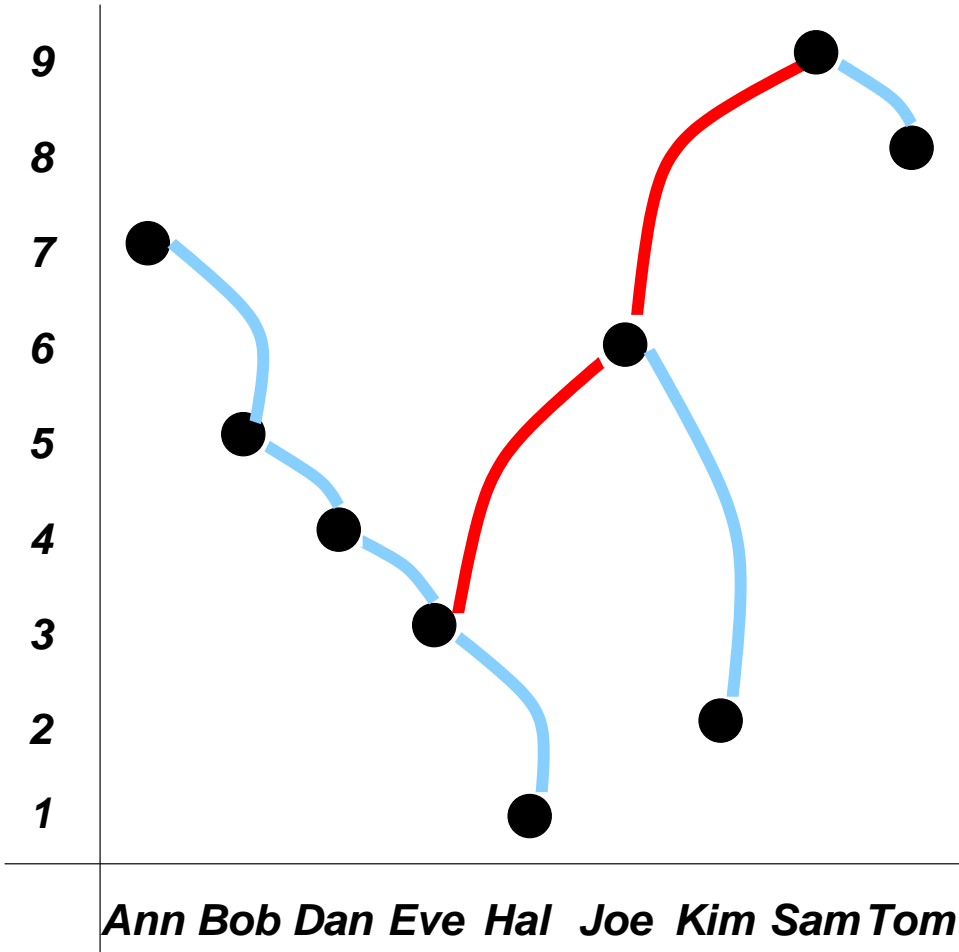


# Reversals

The number of reversals gives a lower bound on the number of moves



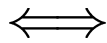
Perfect Sorting



Red edge between two elements (Hal, Kim):

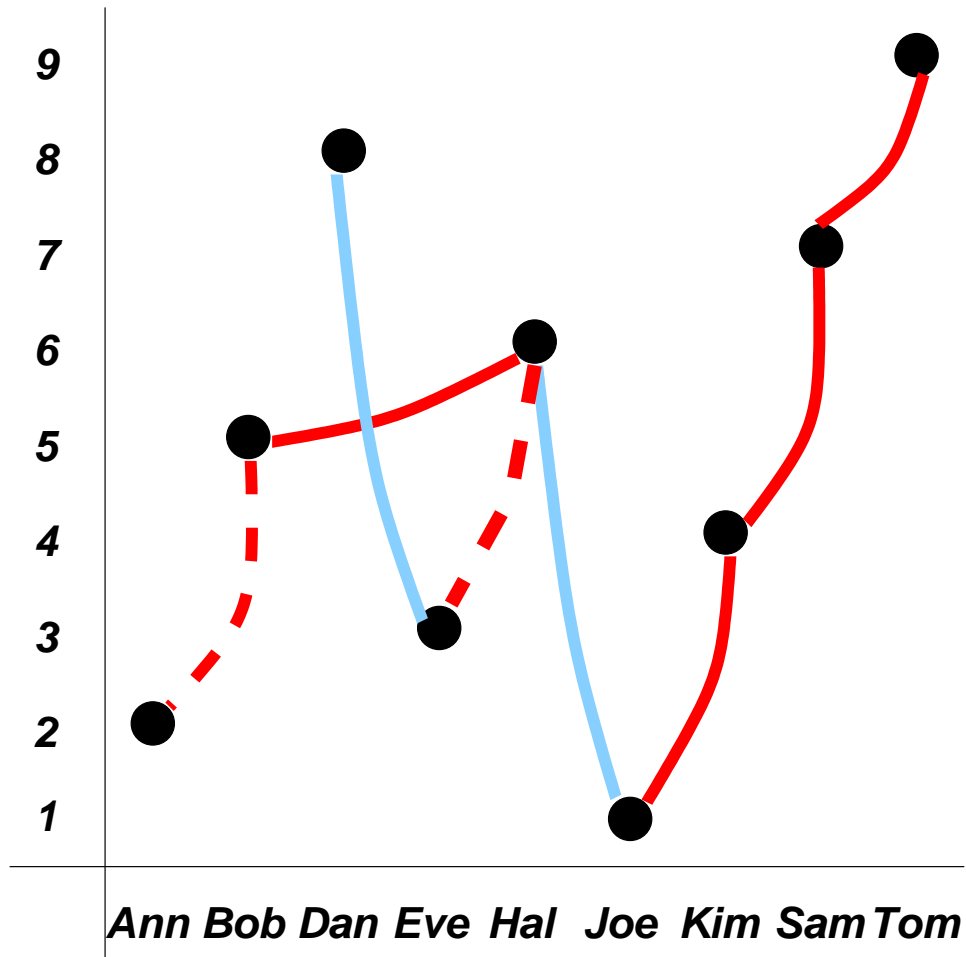
There is a sorting where Hal and Kim are joined before either is moved.

There is a perfect sorting



There exists a spanning tree of blue and red edges

## There is not always a Perfect Sorting



Red Edges must not “overlap” in either order, i.e. horizontally or vertically

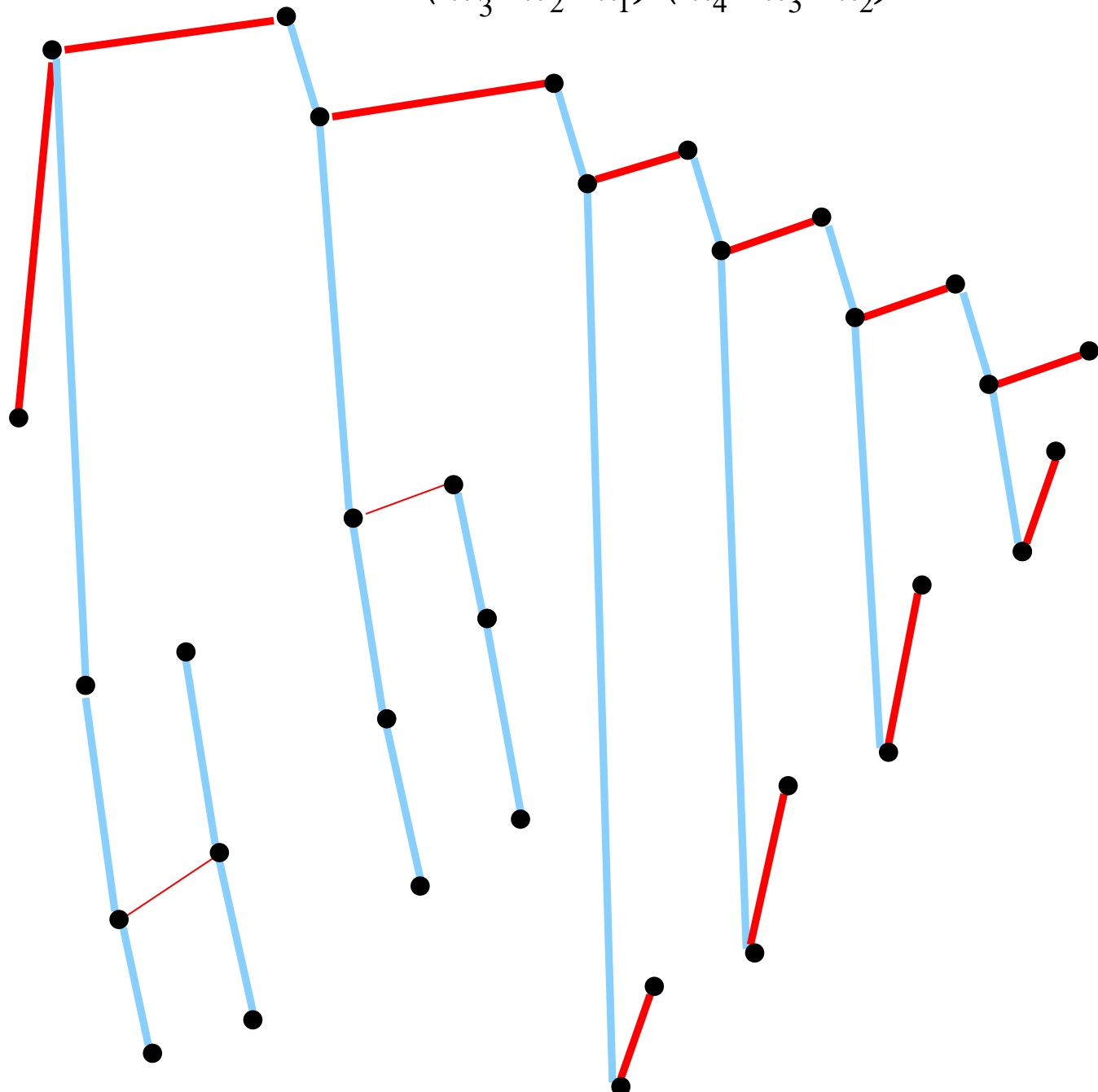
The problem of whether there exists a perfect sorting is  $\mathcal{NP}$ -complete.

Proof: 3-SAT can be reduced to Perfect Sorting. ■

$\implies$

Block Sorting is  $\mathcal{NP}$  complete.

$$(\bar{x}_3 + x_2 + x_1) (x_4 + \bar{x}_3 + \bar{x}_2)$$

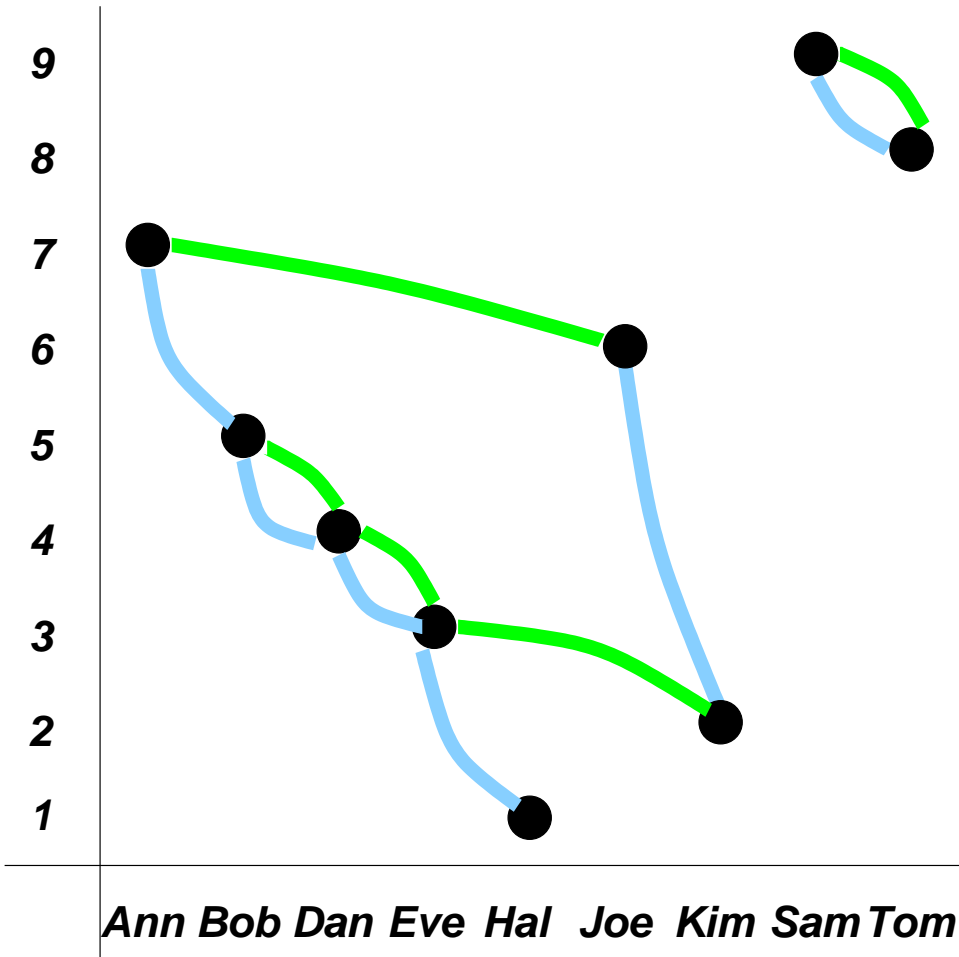


$s \quad l^1 \quad \bar{p}_3^1 \quad p_2^1 \quad p_1^1 \quad \bar{q}_3^1 \quad q_2^1 \quad q_1^1 \quad r^1 \quad l^2 \quad p_4^2 \quad \bar{p}_3^2 \quad \bar{p}_2^2 \quad q_4^2 \quad \bar{q}_3^2 \quad \bar{q}_2^2 \quad r^2 \quad l^3 \quad u^1 \quad v^1 \quad r^3 \quad l^4 \quad u^2 \quad v^2 \quad r^4 \quad l^5 \quad u^3 \quad v^3 \quad r^5 \quad l^6 \quad u^4 \quad v^4 \quad r^6$

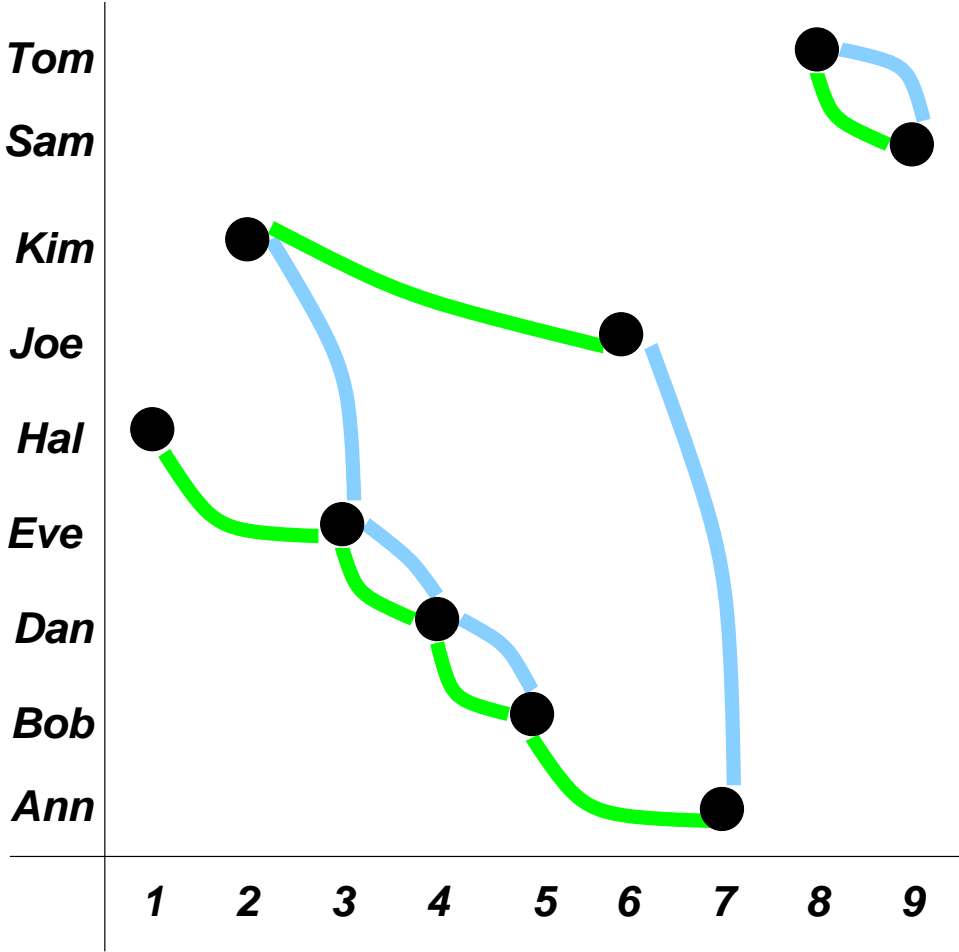
Since Block Sorting is  $\mathcal{NP}$ -complete, what can be done?

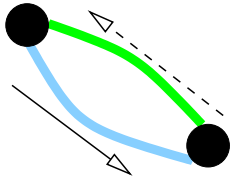
- Identify “good” moves
- Get numerous lower bounds
- Get a “competitive” algorithm

Primal Sorting Problem

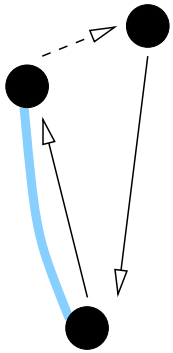


Dual Sorting Problem



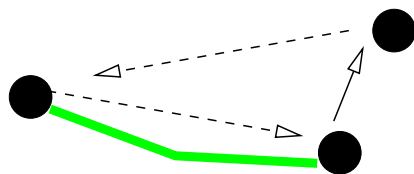


"primal-dual move"



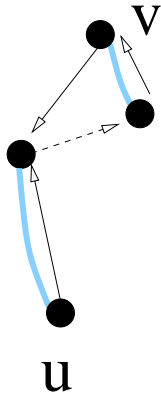
"primal obstruction"

u

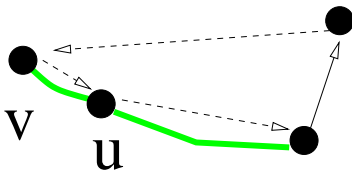


"dual obstruction"

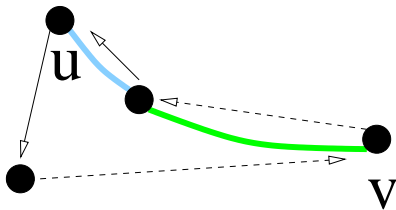
u



removal of  $u$  creates a primal obstruction at  $v$



removal of  $u$  creates a dual obstruction at  $v$



removal of  $u$  creates a dual obstruction at  $v$

## Local Property Algorithm

For each element (black dot) keep four pointers

x.left x.right x.down x.up

Type	Condition	Effect of Deletion
A	$x.r.u = x$	$n' \leq n - 1$
B	$x.d.r.d = x$	$n' \leq n - 2$
C	$x.l.u.l = x$	$n' \leq n - 2$
D	$x.l.u.l.l = x$	$n' \leq n - 1,$ x.r becomes type C
E	$x.d.r.d.d = x$	$n' \leq n - 1,$ x.u becomes type B
F	$x.l.u.u.l = x$	$n' \leq n - 1$ x.l.u becomes type B

- Can be implemented in  $O(1)$  per step

- Overall:  $O(n \log n)$

A lower bound

*k*      *blue edges*

*l*      *green edges*

$$\# \text{ moves} \geq \max \begin{cases} n - k - l - 1 \\ k \\ l \end{cases}$$

$\Rightarrow$

$$\# \text{ moves} \geq \left\lceil \frac{n-1}{3} \right\rceil$$



## Competitiveness

How good does an algorithm behave in the worst case compared to an optimal scheme?

Local Property algorithms are 3 - competitive

Open Problem:

Is there a polynomial-time 2 - competitive algorithm?