# Deleting edges to save cows - an application of treewidth

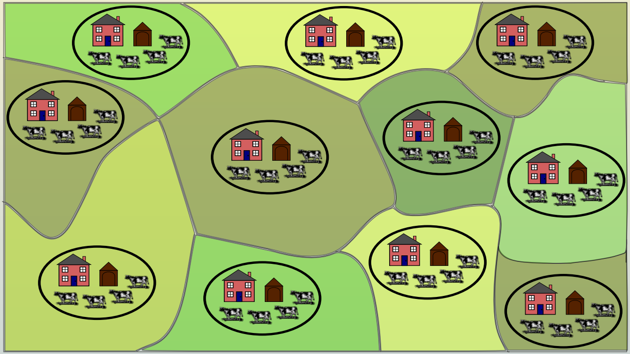
**UCLA, 14th February 2017**

**Kitty Meeks**

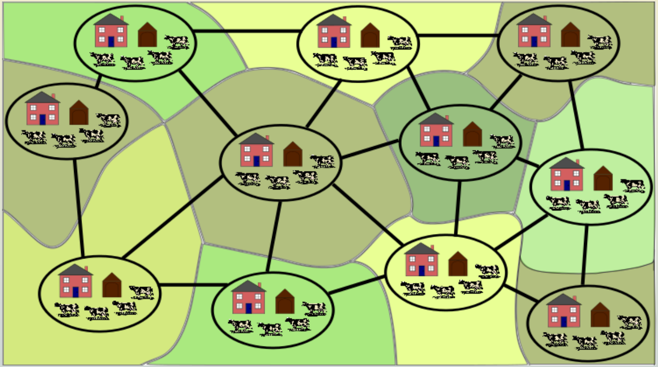
Joint work with Jessica Enright (University of Stirling)

60 YEARS OF COMPUTING AT GLASGOW

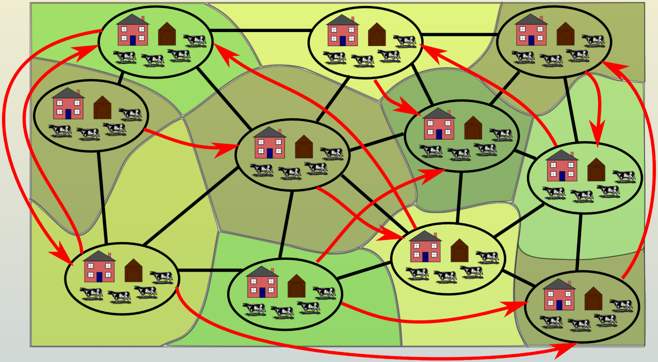
# The animal contact network



# The animal contact network



# The animal contact network



# The animal contact network

**Vertex-deletion**

**Vertex-deletion**

E.g. vaccinate all animals at a particular animal holding.

**Vertex-deletion**

E.g. vaccinate all animals at a particular animal holding.

**Edge-deletion**

# Modifying the network

**Vertex-deletion**
E.g. vaccinate all animals at a particular animal holding.

**Edge-deletion**
E.g.

- Double fence lines

## Modifying the network

**Vertex-deletion**

E.g. vaccinate all animals at a particular animal holding.

**Edge-deletion**

E.g.

- Double fence lines
- Testing or quarantine for animals on a particular trade route

# Modifying the network

**Vertex-deletion**
E.g. vaccinate all animals at a particular animal holding.

**Edge-deletion**
E.g.

- Double fence lines
- Testing or quarantine for animals on a particular trade route

**Cost of modifications** The cost of deleting individual vertices/edges may vary; this can be captured with a weight function on vertices and/or edges.

# How do we want to change the network?

# How do we want to change the network?

Desirable properties may include:

- Bounded component size

# How do we want to change the network?

Desirable properties may include:

- Bounded component size
- Bounded degree

# How do we want to change the network?

Desirable properties may include:

- Bounded component size
- Bounded degree
- Bounded *d*-neighbourhood

# How do we want to change the network?

Desirable properties may include:

- Bounded component size
- Bounded degree
- Bounded $d$-neighbourhood
- Low connectivity

# How do we want to change the network?

Desirable properties may include:

- Bounded component size
- Bounded degree
- Bounded $d$-neighbourhood
- Low connectivity

We may additionally want to:

- consider the total number of animals in e.g. a connected component, rather than just the number of animal holdings

# How do we want to change the network?

Desirable properties may include:

- Bounded component size
- Bounded degree
- Bounded $d$-neighbourhood
- Low connectivity

We may additionally want to:

- consider the total number of animals in e.g. a connected component, rather than just the number of animal holdings
- place more or less strict restrictions on individual animal holdings

# Bounding the component size by deleting edges

Let $\mathcal{C}_h$ be the set of all connected graphs on $h$ vertices.

# Bounding the component size by deleting edges

Let $\mathcal{C}_h$ be the set of all connected graphs on $h$ vertices.

$\mathcal{C}_h$-FREE EDGE DELETION
*Input:* A Graph $G = (V, E)$ and an integer $k$.
*Question:* Does there exist $E' \subseteq E$ with $|E'| = k$ such that $G \setminus E$ does not contain any $H \in \mathcal{C}_h$ as an induced subgraph?

# Bounding the component size by deleting edges

Let $\mathcal{C}_h$ be the set of all connected graphs on $h$ vertices.

$\mathcal{C}_h$-FREE EDGE DELETION
*Input:* A Graph $G = (V, E)$ and an integer $k$.
*Question:* Does there exist $E' \subseteq E$ with $|E'| = k$ such that $G \setminus E$ does not contain any $H \in \mathcal{C}_h$ as an induced subgraph?

This problem has also been called:

- Min-Max Component Size Problem
- Minimum Worst Contamination Problem
- Component Order Edge Connectivity

# Bounding the component size: what is known?

- This problem is NP-complete in general, even when $h = 3$.

# Bounding the component size: what is known?

- This problem is NP-complete in general, even when $h = 3$.

**Theorem (Cai, 1996)**

*$\mathcal{C}_h$-FREE EDGE DELETION can be solved in time $O(h^{2k} \cdot n^h)$, where $n$ is the number of vertices in the input graph.*

## Bounding the component size: what is known?

- This problem is NP-complete in general, even when $h = 3$.

**Theorem (Cai, 1996)**

*$\mathcal{C}_h$-FREE EDGE DELETION can be solved in time $O(h^{2k} \cdot n^h)$, where $n$ is the number of vertices in the input graph.*

**Theorem (Gross, Heinig, Iswara, Kazmiercaak, Luttrell, Saccoman and Suffel, 2013)**

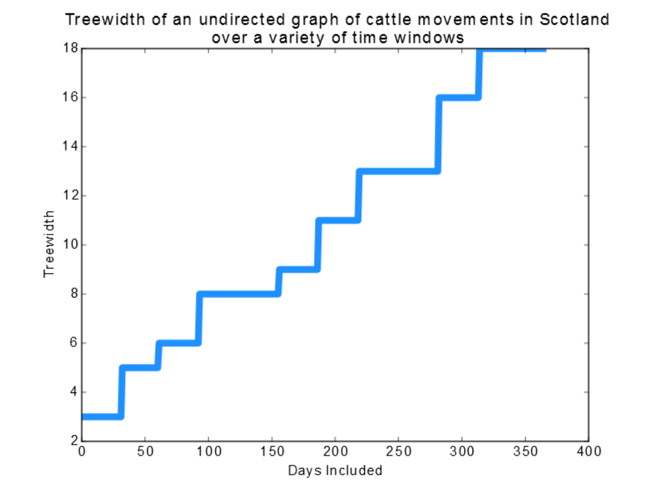
*$\mathcal{C}_h$-FREE EDGE DELETION can be solved in polynomial time when restricted to trees.*

# Treewidth is relevant

$(T, \mathcal{D})$ is a *tree decomposition* of $G$ if $T$ is a tree and $\mathcal{D} = \{\mathcal{D}(t) : t \in V(T)\}$ is a collection of non-empty subsets of $V(G)$ (or *bags*), indexed by the nodes of $T$, satisfying:

1. $V(G) = \bigcup_{t \in V(T)} \mathcal{D}(t)$,
2. for every $e = uv \in E(G)$, there exists $t \in V(T)$ such that $u, v \in \mathcal{D}(t)$,
3. for every $v \in V(G)$, if $T(v)$ is defined to be the subgraph of $T$ induced by nodes $t$ with $v \in \mathcal{D}(t)$, then $T(v)$ is connected.

# Treewidth is relevant




A plot of the treewidth of the largest component in an undirected version of the cattle movement graph in Scotland in 2009 over a number of different days included: all day sets start on January 1, 2009.

**Theorem (Enright and M., 2017)**

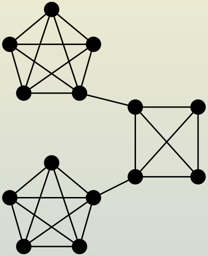
*There exists an algorithm to solve $\mathcal{C}_h$-*FREE EDGE DELETION *in time $O((wh)^{2w}n)$ on an input graph with $n$ vertices whose treewidth is at most $w$.*

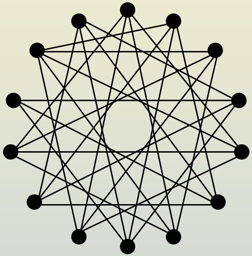
**Theorem (Enright and M., 2017)**

*There exists an algorithm to solve $\mathcal{C}_h$-*Free Edge Deletion *in time $O((wh)^{2w}n)$ on an input graph with $n$ vertices whose treewidth is at most $w$.*

We recursively compute the minimum number of edges we delete, for each combination of:

- a partition of the vertices in the bag, indicating which are allowed to belong to the same component, and
- a function from blocks of the partition to $[h]$, indicating the maximum number of vertices allowed so far in the component containing the block in question.





# New results

**Theorem (Enright and M., 2015+)**

*Let* $\Pi$ *be a monotone graph property defined by the set of forbidden subgraphs* $\mathcal{F}$*, where* $\max\{|F| : F \in \mathcal{F}\}$ *exists and is equal to* $h$*. Then* Edge Deletion To $\Pi$ *can be solved in time* $f(h, w) \cdot n$ *on an input graph with* $n$ *vertices whose treewidth is at most* $w$*, where* $f$ *is an explicit computable function.*

# Algorithms in practice

| Graph information | | | | Tree decomposition method | | CP method | | |
|---|---|---|---|---|---|---|---|---|
| Graph ID | v(G) | e(G) | tw(G) | Minimum deletion found | Time | Minimum deletion found | Time | Time to confirmation |
| 2010-0 | 104 | 110 | 4 | 38 | 53 | 90 | 27000 | - |
| 2010-3 | 45 | 45 | 3 | 11 | 4 | 25 | 21600 | - |
| 2010-4 | 38 | 38 | 3 | 20 | 7 | 24 | 25200 | - |
| 2010-5 | 37 | 40 | 4 | 7 | 64 | 7 | 1032 | 1046 |
| 2012-0 | 97 | 119 | 5 | 58 | 5400 | 93 | 7200 | - |
| 2012-1 | 72 | 74 | 3 | 20 | 11 | 38 | 32400 | - |
| 2012-4 | 31 | 30 | 2 | 12 | 3 | 12 | 1082 | - |
| 2012-5 | 49 | 52 | 3 | 15 | 7 | 30 | 19800 | - |
| 2013-1 | 45 | 47 | 4 | 20 | 35 | 30 | 19800 | - |
| 2013-3 | 61 | 62 | 4 | 19 | 38 | 36 | 7200 | - |
| 2013-4 | 35 | 38 | 4 | 15 | 5 | 20 | 3600 | - |
| 2013-6 | 39 | 41 | 3 | 14 | 4 | 17 | 7200 | - |
| 2014-0 | 32 | 49 | 4 | 28 | 445 | 28 | 2738 | - |
| 2014-1 | 47 | 47 | 3 | 18 | 6 | 24 | 7200 | - |
| 2014-2 | 57 | 59 | 4 | 17 | 10 | 33 | 19800 | - |
| 2014-3 | 41 | 40 | 2 | 15 | 5 | 31 | 360 | - |
| 2014-5 | 31 | 35 | 4 | 8 | 18 | 8 | 412 | 476 |
| 2014-6 | 48 | 49 | 3 | 21 | 11 | 21 | 10800 | - |
| 2014-7 | 31 | 32 | 3 | 16 | 4 | 16 | 7 | - |

# Future directions

- Why do trade networks have small treewidth?

# Future directions

- Why do trade networks have small treewidth?
- Budget as parameter, rather than desired component size

## Future directions

- Why do trade networks have small treewidth?
- Budget as parameter, rather than desired component size
- Geographic networks - planar graphs, or powers of planar graphs

# Future directions

- Why do trade networks have small treewidth?
- Budget as parameter, rather than desired component size
- Geographic networks - planar graphs, or powers of planar graphs
- Extra structure in directed graphs

## Future directions

- Why do trade networks have small treewidth?
- Budget as parameter, rather than desired component size
- Geographic networks - planar graphs, or powers of planar graphs
- Extra structure in directed graphs
- Temporal connectivity

# Future directions

- Why do trade networks have small treewidth?
- Budget as parameter, rather than desired component size
- Geographic networks - planar graphs, or powers of planar graphs
- Extra structure in directed graphs
- Temporal connectivity
- New parameters to capture structure of the input

# Future directions

- Why do trade networks have small treewidth?
- Budget as parameter, rather than desired component size
- Geographic networks - planar graphs, or powers of planar graphs
- Extra structure in directed graphs
- Temporal connectivity
- New parameters to capture structure of the input
- More complicated or less costly goals

## Future directions

- Why do trade networks have small treewidth?
- Budget as parameter, rather than desired component size
- Geographic networks - planar graphs, or powers of planar graphs
- Extra structure in directed graphs
- Temporal connectivity
- New parameters to capture structure of the input
- More complicated or less costly goals

**THANK YOU**