

The Complexity of Counting Homomorphisms to Directed Acyclic Graphs

Martin Dyer

School of Computing
University of Leeds

BCTCS, Swansea

April 5, 2006

Joint work with Leslie Goldberg and Mike Paterson

Graphs

A **simple** graph is a graph without loops or parallel edges. The class of simple graphs will be denoted by \mathcal{S} .

The class of graphs without parallel edges, but possibly having **self-loops**, will be denoted by \mathcal{S}_0 .

A **multigraph** will be a graph which can have loops and parallel edges. The class of multigraphs will be denoted by \mathcal{M} .

The class of **digraphs** will be denoted by \mathcal{D} .

The class of **directed acyclic graphs** will be denoted by $\overrightarrow{\mathcal{D}}$.

Graphs

A **simple** graph is a graph without loops or parallel edges. The class of simple graphs will be denoted by \mathcal{S} .

The class of graphs without parallel edges, but possibly having **self-loops**, will be denoted by \mathcal{S}_0 .

A **multigraph** will be a graph which can have loops and parallel edges. The class of multigraphs will be denoted by \mathcal{M} .

The class of **digraphs** will be denoted by \mathcal{D} .

The class of **directed acyclic graphs** will be denoted by $\overrightarrow{\mathcal{D}}$.

Graphs

A **simple** graph is a graph without loops or parallel edges. The class of simple graphs will be denoted by \mathcal{S} .

The class of graphs without parallel edges, but possibly having **self-loops**, will be denoted by \mathcal{S}_0 .

A **multigraph** will be a graph which can have loops and parallel edges. The class of multigraphs will be denoted by \mathcal{M} .

The class of **digraphs** will be denoted by \mathcal{D} .

The class of **directed acyclic graphs** will be denoted by $\overrightarrow{\mathcal{D}}$.

Graphs

A **simple** graph is a graph without loops or parallel edges. The class of simple graphs will be denoted by \mathcal{S} .

The class of graphs without parallel edges, but possibly having **self-loops**, will be denoted by \mathcal{S}_0 .

A **multigraph** will be a graph which can have loops and parallel edges. The class of multigraphs will be denoted by \mathcal{M} .

The class of **digraphs** will be denoted by \mathcal{D} .

The class of **directed acyclic graphs** will be denoted by $\overrightarrow{\mathcal{D}}$.

Graphs

A **simple** graph is a graph without loops or parallel edges. The class of simple graphs will be denoted by \mathcal{S} .

The class of graphs without parallel edges, but possibly having **self-loops**, will be denoted by \mathcal{S}_0 .

A **multigraph** will be a graph which can have loops and parallel edges. The class of multigraphs will be denoted by \mathcal{M} .

The class of **digraphs** will be denoted by \mathcal{D} .

The class of **directed acyclic graphs** will be denoted by $\overrightarrow{\mathcal{D}}$.

Layered digraphs

A **layered digraph** H with ℓ **layers** will mean that \mathcal{V} can be partitioned into $(\ell + 1)$ **levels** \mathcal{V}_i ($i = 0, \dots, \ell$) so that $(u, u') \in \mathcal{E}$ implies $u \in \mathcal{V}_{i-1}, u' \in \mathcal{V}_i$ for some $1 \leq i \leq \ell$.

Then \mathcal{G}_ℓ will denote the class of ℓ -layer digraphs, and $\mathcal{G}_\infty = \bigcup_{\ell=0}^{\infty} \mathcal{G}_\ell$ will be the class of all layered digraphs.



A two-layer digraph

Layered digraphs

A **layered digraph** H with ℓ **layers** will mean that \mathcal{V} can be partitioned into $(\ell + 1)$ **levels** \mathcal{V}_i ($i = 0, \dots, \ell$) so that $(u, u') \in \mathcal{E}$ implies $u \in \mathcal{V}_{i-1}, u' \in \mathcal{V}_i$ for some $1 \leq i \leq \ell$.

Then \mathcal{G}_ℓ will denote the class of ℓ -layer digraphs, and $\mathcal{G}_\infty = \bigcup_{\ell=0}^{\infty} \mathcal{G}_\ell$ will be the class of all layered digraphs.

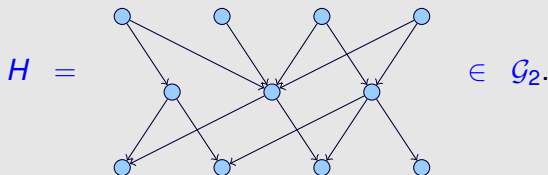


A two-layer digraph

Layered digraphs

A **layered digraph** H with ℓ **layers** will mean that \mathcal{V} can be partitioned into $(\ell + 1)$ **levels** \mathcal{V}_i ($i = 0, \dots, \ell$) so that $(u, u') \in \mathcal{E}$ implies $u \in \mathcal{V}_{i-1}, u' \in \mathcal{V}_i$ for some $1 \leq i \leq \ell$.

Then \mathcal{G}_ℓ will denote the class of ℓ -layer digraphs, and $\mathcal{G}_\infty = \bigcup_{\ell=0}^{\infty} \mathcal{G}_\ell$ will be the class of all layered digraphs.



A two-layer digraph

Graph homomorphisms

Let $H = (\mathcal{V}, \mathcal{E})$ be any fixed graph or digraph.

We view H as the **target** graph.

Let \mathcal{G} be any class of graphs or digraphs, and let

$G = (V, E) \in \mathcal{G}$. We view G as the **input** graph.

A **homomorphism** $f : G \rightarrow H$ is an adjacency-preserving function from G to H , i.e. we have

$f(e) \in \{e' \in \mathcal{E} : e' = (f(v), f(v'))\}$ for all $e = (v, v') \in E$.

Homomorphisms are also called **H -colourings** of G .

Let $\#H(G) = |\{f : G \rightarrow H \mid f \text{ is a homomorphism}\}|$

be the number of distinct homomorphisms from G to H .

Graph homomorphisms

Let $H = (\mathcal{V}, \mathcal{E})$ be any fixed graph or digraph.

We view H as the **target** graph.

Let \mathcal{G} be any class of graphs or digraphs, and let

$G = (V, E) \in \mathcal{G}$. We view G as the **input** graph.

A **homomorphism** $f : G \rightarrow H$ is an adjacency-preserving function from G to H , i.e. we have

$f(e) \in \{e' \in \mathcal{E} : e' = (f(v), f(v'))\}$ for all $e = (v, v') \in E$.

Homomorphisms are also called **H -colourings** of G .

Let $\#H(G) = |\{f : G \rightarrow H \mid f \text{ is a homomorphism}\}|$

be the number of distinct homomorphisms from G to H .

Graph homomorphisms

Let $H = (\mathcal{V}, \mathcal{E})$ be any fixed graph or digraph.

We view H as the **target** graph.

Let \mathcal{G} be any class of graphs or digraphs, and let

$G = (V, E) \in \mathcal{G}$. We view G as the **input** graph.

A **homomorphism** $f : G \rightarrow H$ is an adjacency-preserving function from G to H , i.e. we have

$f(e) \in \{e' \in \mathcal{E} : e' = (f(v), f(v'))\}$ for all $e = (v, v') \in E$.

Homomorphisms are also called **H -colourings** of G .

Let $\#H(G) = |\{f : G \rightarrow H \mid f \text{ is a homomorphism}\}|$

be the number of distinct homomorphisms from G to H .

Graph homomorphisms

Let $H = (\mathcal{V}, \mathcal{E})$ be any fixed graph or digraph.

We view H as the **target** graph.

Let \mathcal{G} be any class of graphs or digraphs, and let

$G = (V, E) \in \mathcal{G}$. We view G as the **input** graph.

A **homomorphism** $f : G \rightarrow H$ is an adjacency-preserving function from G to H , i.e. we have

$f(e) \in \{e' \in \mathcal{E} : e' = (f(v), f(v'))\}$ for all $e = (v, v') \in E$.

Homomorphisms are also called **H -colourings** of G .

Let $\#H(G) = |\{f : G \rightarrow H \mid f \text{ is a homomorphism}\}|$

be the number of distinct homomorphisms from G to H .

Graph homomorphisms

Let $H = (\mathcal{V}, \mathcal{E})$ be any fixed graph or digraph.

We view H as the **target** graph.

Let \mathcal{G} be any class of graphs or digraphs, and let

$G = (V, E) \in \mathcal{G}$. We view G as the **input** graph.

A **homomorphism** $f : G \rightarrow H$ is an adjacency-preserving function from G to H , i.e. we have

$f(e) \in \{e' \in \mathcal{E} : e' = (f(v), f(v'))\}$ for all $e = (v, v') \in E$.

Homomorphisms are also called **H -colourings** of G .

Let $\#H(G) = |\{f : G \rightarrow H \mid f \text{ is a homomorphism}\}|$

be the number of distinct homomorphisms from G to H .

Graph homomorphisms

Let $H = (\mathcal{V}, \mathcal{E})$ be any fixed graph or digraph.

We view H as the **target** graph.

Let \mathcal{G} be any class of graphs or digraphs, and let

$G = (V, E) \in \mathcal{G}$. We view G as the **input** graph.

A **homomorphism** $f : G \rightarrow H$ is an adjacency-preserving function from G to H , i.e. we have

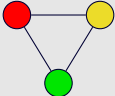


$f(e) \in \{e' \in \mathcal{E} : e' = (f(v), f(v'))\}$ for all $e = (v, v') \in E$.

Homomorphisms are also called **H -colourings** of G .

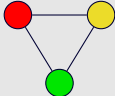
Let $\#H(G) = |\{f : G \rightarrow H \mid f \text{ is a homomorphism}\}|$

be the number of distinct homomorphisms from G to H .

Examples

- 1 If $\mathcal{G} = \mathcal{S}$ and $H =$  , the homomorphisms to H are **three-colourings** of G .
- 2 If $\mathcal{G} = \mathcal{S}$ and $H =$  , the homomorphisms to H are **independent sets** in G . Note $H \notin \mathcal{G}$.
- 3 If $\mathcal{G} = \mathcal{D}$ and $H =$  , a homomorphism to H is a **layering** of G , so that G becomes a layered digraph with (at most) three levels.

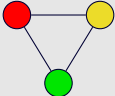


Examples

1 If $\mathcal{G} = \mathcal{S}$ and $H =$  , the homomorphisms to H are **three-colourings** of G .

2 If $\mathcal{G} = \mathcal{S}$ and $H =$  , the homomorphisms to H are **independent sets** in G . Note $H \notin \mathcal{G}$.

3 If $\mathcal{G} = \mathcal{D}$ and $H =$  , a homomorphism to H is a **layering** of G , so that G becomes a layered digraph with (at most) three levels.

Examples

- 1 If $\mathcal{G} = \mathcal{S}$ and $H =$  , the homomorphisms to H are **three-colourings** of G .
- 2 If $\mathcal{G} = \mathcal{S}$ and $H =$  , the homomorphisms to H are **independent sets** in G . Note $H \notin \mathcal{G}$.
- 3 If $\mathcal{G} = \mathcal{D}$ and $H =$  , a homomorphism to H is a **layering** of G , so that G becomes a layered digraph with (at most) three levels.

Complexity

Given H and an input graph G , two natural problems are:

- 1 **Decision**: The complexity of the question “ $\#H(G) > 0$?”.
- 2 **Counting**: The complexity of evaluating $\#H(G)$ (exactly).

More generally, we may consider all H in some class \mathcal{H} and ask

- 1 For which $H \in \mathcal{H}$ is the decision problem in P and for which H is it NP-Complete ?
- 2 For which $H \in \mathcal{H}$ is the counting problem in FP and for which H is it #P-Complete ?

When we can assign every $H \in \mathcal{H}$ into one of two classes (“easy” or “hard”), we say we have a **dichotomy theorem** for \mathcal{H} .

N.B. There is no dichotomy for the whole of NP (Ladner, 1975).

Complexity

Given H and an input graph G , two natural problems are:

- 1 **Decision**: The complexity of the question “ $\#H(G) > 0$?”.
- 2 **Counting**: The complexity of evaluating $\#H(G)$ (exactly).

More generally, we may consider all H in some class \mathcal{H} and ask

- 1 For which $H \in \mathcal{H}$ is the decision problem in P and for which H is it NP-Complete ?
- 2 For which $H \in \mathcal{H}$ is the counting problem in FP and for which H is it #P-Complete ?

When we can assign every $H \in \mathcal{H}$ into one of two classes (“easy” or “hard”), we say we have a **dichotomy theorem** for \mathcal{H} .

N.B. There is no dichotomy for the whole of NP (Ladner, 1975).

Complexity

Given H and an input graph G , two natural problems are:

- 1 **Decision**: The complexity of the question “ $\#H(G) > 0$?”.
- 2 **Counting**: The complexity of evaluating $\#H(G)$ (exactly).

More generally, we may consider all H in some class \mathcal{H} and ask

- 1 For which $H \in \mathcal{H}$ is the decision problem in P and for which H is it NP-Complete ?
- 2 For which $H \in \mathcal{H}$ is the counting problem in FP and for which H is it #P-Complete ?

When we can assign every $H \in \mathcal{H}$ into one of two classes (“easy” or “hard”), we say we have a **dichotomy theorem** for \mathcal{H} .

N.B. There is no dichotomy for the whole of NP (Ladner, 1975).

Complexity

Given H and an input graph G , two natural problems are:

- 1 **Decision**: The complexity of the question “ $\#H(G) > 0$?”.
- 2 **Counting**: The complexity of evaluating $\#H(G)$ (exactly).

More generally, we may consider all H in some class \mathcal{H} and ask

- 1 For which $H \in \mathcal{H}$ is the decision problem in P and for which H is it NP-Complete ?
- 2 For which $H \in \mathcal{H}$ is the counting problem in FP and for which H is it #P-Complete ?

When we can assign every $H \in \mathcal{H}$ into one of two classes (“easy” or “hard”), we say we have a **dichotomy theorem** for \mathcal{H} .

N.B. There is no dichotomy for the whole of NP (Ladner, 1975).

Complexity

Given H and an input graph G , two natural problems are:

- 1 **Decision**: The complexity of the question “ $\#H(G) > 0$?”.
- 2 **Counting**: The complexity of evaluating $\#H(G)$ (exactly).

More generally, we may consider all H in some class \mathcal{H} and ask

- 1 For which $H \in \mathcal{H}$ is the decision problem in \mathbf{P} and for which H is it NP-Complete ?
- 2 For which $H \in \mathcal{H}$ is the counting problem in \mathbf{FP} and for which H is it #P-Complete ?

When we can assign every $H \in \mathcal{H}$ into one of two classes (“easy” or “hard”), we say we have a **dichotomy theorem** for \mathcal{H} .

N.B. There is no dichotomy for the whole of \mathbf{NP} (Ladner, 1975).

Complexity

Given H and an input graph G , two natural problems are:

- 1 **Decision**: The complexity of the question “ $\#H(G) > 0$?”.
- 2 **Counting**: The complexity of evaluating $\#H(G)$ (exactly).

More generally, we may consider all H in some class \mathcal{H} and ask

- 1 For which $H \in \mathcal{H}$ is the decision problem in **P** and for which H is it **NP-Complete** ?
- 2 For which $H \in \mathcal{H}$ is the counting problem in **FP** and for which H is it **#P-Complete** ?

When we can assign every $H \in \mathcal{H}$ into one of two classes (“easy” or “hard”), we say we have a **dichotomy theorem** for \mathcal{H} .

N.B. There is no dichotomy for the whole of **NP** (Ladner, 1975).

Complexity

Given H and an input graph G , two natural problems are:

- 1 **Decision**: The complexity of the question “ $\#H(G) > 0$?”.
- 2 **Counting**: The complexity of evaluating $\#H(G)$ (exactly).

More generally, we may consider all H in some class \mathcal{H} and ask

- 1 For which $H \in \mathcal{H}$ is the decision problem in P and for which H is it NP-Complete ?
- 2 For which $H \in \mathcal{H}$ is the counting problem in FP and for which H is it #P-Complete ?

When we can assign every $H \in \mathcal{H}$ into one of two classes (“easy” or “hard”), we say we have a **dichotomy theorem** for \mathcal{H} .

N.B. There is no dichotomy for the whole of NP (Ladner, 1975).

Complexity

Given H and an input graph G , two natural problems are:

- 1 **Decision**: The complexity of the question “ $\#H(G) > 0$?”.
- 2 **Counting**: The complexity of evaluating $\#H(G)$ (exactly).

More generally, we may consider all H in some class \mathcal{H} and ask

- 1 For which $H \in \mathcal{H}$ is the decision problem in P and for which H is it NP-Complete ?
- 2 For which $H \in \mathcal{H}$ is the counting problem in FP and for which H is it #P-Complete ?

When we can assign every $H \in \mathcal{H}$ into one of two classes (“easy” or “hard”), we say we have a **dichotomy theorem** for \mathcal{H} .

N.B. There is no dichotomy for the whole of NP (Ladner, 1975).

Undirected graphs

When $\mathcal{G} = \mathcal{S}$ and $\mathcal{H} = \mathcal{S}_0$, we have the following dichotomies:

- 1 Hell & Nešetřil (1990) showed that the decision problem is in P if H has a loop or is bipartite. For all other H it is NP-Complete.
- 2 D & Greenhill (2000) showed that the counting problem for H is in FP if every component of H with more than one vertex is either a complete graph with all vertices looped, or a complete bipartite graph with no vertex looped. For all other H it is #P-Complete.

In both cases, the implied polynomial time algorithm is trivial.

Undirected graphs

When $\mathcal{G} = \mathcal{S}$ and $\mathcal{H} = \mathcal{S}_0$, we have the following dichotomies:

- 1 **Hell & Nešetřil (1990)** showed that the decision problem is in **P** if H has a loop or is bipartite. For all other H it is **NP-Complete**.
- 2 **D & Greenhill (2000)** showed that the counting problem for H is in **FP** if every component of H with more than one vertex is either a complete graph with all vertices looped, or a complete bipartite graph with no vertex looped. For all other H it is **#P-Complete**.

In both cases, the implied polynomial time algorithm is trivial.

Undirected graphs

When $\mathcal{G} = \mathcal{S}$ and $\mathcal{H} = \mathcal{S}_0$, we have the following dichotomies:

- 1 **Hell & Nešetřil (1990)** showed that the decision problem is in **P** if H has a loop or is bipartite. For all other H it is **NP-Complete**.
- 2 **D & Greenhill (2000)** showed that the counting problem for H is in **FP** if every component of H with more than one vertex is either a complete graph with all vertices looped, or a complete bipartite graph with no vertex looped. For all other H it is **#P-Complete**.

In both cases, the implied polynomial time algorithm is trivial.

Undirected graphs

When $\mathcal{G} = \mathcal{S}$ and $\mathcal{H} = \mathcal{S}_0$, we have the following dichotomies:

- 1 **Hell & Nešetřil (1990)** showed that the decision problem is in **P** if H has a loop or is bipartite. For all other H it is **NP-Complete**.
- 2 **D & Greenhill (2000)** showed that the counting problem for H is in **FP** if every component of H with more than one vertex is either a complete graph with all vertices looped, or a complete bipartite graph with no vertex looped. For all other H it is **#P-Complete**.

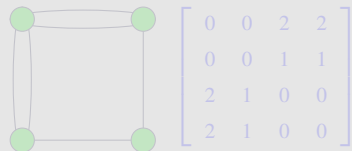
In both cases, the implied polynomial time algorithm is trivial.

Undirected multigraphs

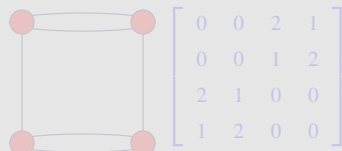
When $\mathcal{G} = \mathcal{S}$ and $\mathcal{H} = \mathcal{M}$, Bulatov & Grohe (2004) generalised the result of D & Greenhill (2000) as follows.

Let $\mathcal{V} = \{1, \dots, h\}$ and let $A = (a_{ij})$ be the (symmetric) $h \times h$ adjacency matrix of H , so that $a_{ij} \geq 0$ is the number of parallel edges between vertex i and vertex j .

Then computing $\#H$ is in FP (the algorithm is trivial) if A can be partitioned into blocks of rank 1. Otherwise it is $\#P$ -Complete.



Easy



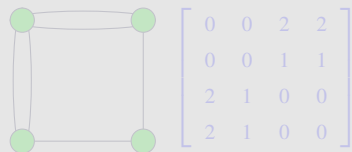
Hard

Undirected multigraphs

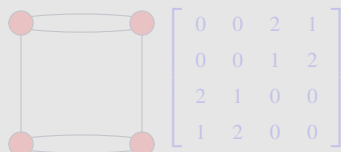
When $\mathcal{G} = \mathcal{S}$ and $\mathcal{H} = \mathcal{M}$, Bulatov & Grohe (2004) generalised the result of D & Greenhill (2000) as follows.

Let $\mathcal{V} = \{1, \dots, h\}$ and let $A = (a_{ij})$ be the (symmetric) $h \times h$ adjacency matrix of H , so that $a_{ij} \geq 0$ is the number of parallel edges between vertex i and vertex j .

Then computing $\#H$ is in FP (the algorithm is trivial) if A can be partitioned into blocks of rank 1. Otherwise it is $\#P$ -Complete.



Easy



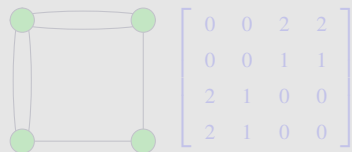
Hard

Undirected multigraphs

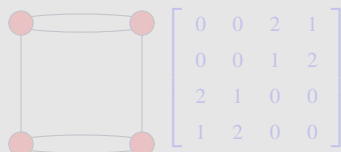
When $\mathcal{G} = \mathcal{S}$ and $\mathcal{H} = \mathcal{M}$, Bulatov & Grohe (2004) generalised the result of D & Greenhill (2000) as follows.

Let $\mathcal{V} = \{1, \dots, h\}$ and let $A = (a_{ij})$ be the (symmetric) $h \times h$ adjacency matrix of H , so that $a_{ij} \geq 0$ is the number of parallel edges between vertex i and vertex j .

Then computing $\#H$ is in **FP** (the algorithm is trivial) if A can be partitioned into blocks of rank 1. Otherwise it is **#P-Complete**.



Easy



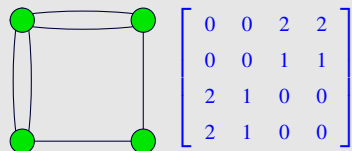
Hard

Undirected multigraphs

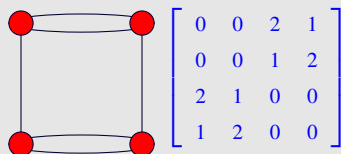
When $\mathcal{G} = \mathcal{S}$ and $\mathcal{H} = \mathcal{M}$, Bulatov & Grohe (2004) generalised the result of D & Greenhill (2000) as follows.

Let $\mathcal{V} = \{1, \dots, h\}$ and let $A = (a_{ij})$ be the (symmetric) $h \times h$ adjacency matrix of H , so that $a_{ij} \geq 0$ is the number of parallel edges between vertex i and vertex j .

Then computing $\#H$ is in FP (the algorithm is trivial) if A can be partitioned into blocks of rank 1. Otherwise it is $\#P$ -Complete.



Easy



Hard

Directed graphs

When $\mathcal{G} = \mathcal{D}$ and $\mathcal{H} = \mathcal{D}$, these problems remain unresolved.

- 1 No dichotomy theorem is known either for decision or counting. For **decision**, this is true even when H is the class of **oriented trees**. For **counting**, Bulatov & Dalmau (2003) gave a partial result and a general conjecture, but they later proved this to be incorrect.
- 2 Feder & Vardi (1993) conjectured decision dichotomy for the more general class of **constraint satisfaction problems** (CSPs), defined over arbitrary finite structures rather than graphs and digraphs. Many partial results.
- 3 Feder & Vardi (1998) showed that decision dichotomy for **all CSPs** would follow from dichotomy for \mathcal{G}_∞ . Even a dichotomy theorem for \mathcal{G}_4 would be enough.

Directed graphs

When $\mathcal{G} = \mathcal{D}$ and $\mathcal{H} = \mathcal{D}$, these problems remain unresolved.

- 1 No dichotomy theorem is known either for decision or counting. For **decision**, this is true even when H is the class of **oriented trees**. For **counting**, **Bulatov & Dalmau (2003)** gave a partial result and a general conjecture, but they later proved this to be incorrect.
- 2 **Feder & Vardi (1993)** conjectured decision dichotomy for the more general class of **constraint satisfaction problems** (CSPs), defined over arbitrary finite structures rather than graphs and digraphs. Many partial results.
- 3 **Feder & Vardi (1998)** showed that decision dichotomy for **all CSPs** would follow from dichotomy for \mathcal{G}_∞ . Even a dichotomy theorem for \mathcal{G}_4 would be enough.

Directed graphs

When $\mathcal{G} = \mathcal{D}$ and $\mathcal{H} = \mathcal{D}$, these problems remain unresolved.

- 1 No dichotomy theorem is known either for decision or counting. For **decision**, this is true even when H is the class of **oriented trees**. For **counting**, **Bulatov & Dalmau (2003)** gave a partial result and a general conjecture, but they later proved this to be incorrect.
- 2 **Feder & Vardi (1993)** conjectured decision dichotomy for the more general class of **constraint satisfaction problems** (CSPs), defined over arbitrary finite structures rather than graphs and digraphs. Many partial results.
- 3 **Feder & Vardi (1998)** showed that decision dichotomy for **all CSPs** would follow from dichotomy for \mathcal{G}_∞ . Even a dichotomy theorem for \mathcal{G}_4 would be enough.

Directed graphs

When $\mathcal{G} = \mathcal{D}$ and $\mathcal{H} = \mathcal{D}$, these problems remain unresolved.

- 1 No dichotomy theorem is known either for decision or counting. For **decision**, this is true even when H is the class of **oriented trees**. For **counting**, **Bulatov & Dalmau (2003)** gave a partial result and a general conjecture, but they later proved this to be incorrect.
- 2 **Feder & Vardi (1993)** conjectured decision dichotomy for the more general class of **constraint satisfaction problems** (CSPs), defined over arbitrary finite structures rather than graphs and digraphs. Many partial results.
- 3 **Feder & Vardi (1998)** showed that decision dichotomy for **all CSPs** would follow from dichotomy for \mathcal{G}_∞ . Even a dichotomy theorem for \mathcal{G}_4 would be enough.

Directed acyclic graphs

If $\mathcal{G} = \mathcal{D}$ and $\mathcal{H} = \vec{\mathcal{D}}$,

- 1 The result of Feder & Vardi (1998) means that decision for DAGs is equivalent to decision for general CSPs.
- 2 A dichotomy theorem for decision would not imply a dichotomy for counting, since the reductions for decision are not parsimonious.
- 3 We will outline a new dichotomy theorem for counting homomorphisms to directed acyclic graphs: D, Goldberg & Paterson (2005). Layered graphs will again turn out to have a central rôle.

Directed acyclic graphs

If $\mathcal{G} = \mathcal{D}$ and $\mathcal{H} = \vec{\mathcal{D}}$,

- 1 The result of **Feder & Vardi (1998)** means that decision for DAGs is equivalent to decision for general CSPs.
- 2 A dichotomy theorem for decision would not imply a dichotomy for counting, since the reductions for decision are not **parsimonious**.
- 3 We will outline a new dichotomy theorem for **counting** homomorphisms to directed acyclic graphs: **D, Goldberg & Paterson (2005)**. Layered graphs will again turn out to have a central rôle.

Directed acyclic graphs

If $\mathcal{G} = \mathcal{D}$ and $\mathcal{H} = \vec{\mathcal{D}}$,

- 1 The result of **Feder & Vardi (1998)** means that decision for DAGs is equivalent to decision for general CSPs.
- 2 A dichotomy theorem for decision would not imply a dichotomy for counting, since the reductions for decision are not **parsimonious**.
- 3 We will outline a new dichotomy theorem for **counting** homomorphisms to directed acyclic graphs: **D, Goldberg & Paterson (2005)**. Layered graphs will again turn out to have a central rôle.

Directed acyclic graphs

If $\mathcal{G} = \mathcal{D}$ and $\mathcal{H} = \vec{\mathcal{D}}$,

- 1 The result of **Feder & Vardi (1998)** means that decision for DAGs is equivalent to decision for general CSPs.
- 2 A dichotomy theorem for decision would not imply a dichotomy for counting, since the reductions for decision are not **parsimonious**.
- 3 We will outline a new dichotomy theorem for **counting** homomorphisms to directed acyclic graphs: **D, Goldberg & Paterson (2005)**. Layered graphs will again turn out to have a central rôle.

Homomorphisms to directed paths

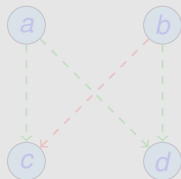
Bulatov & Grohe's theorem, with a simple k -path gadget to simulate the edges of an undirected multigraph, gives structural conditions for easy H :



In particular:

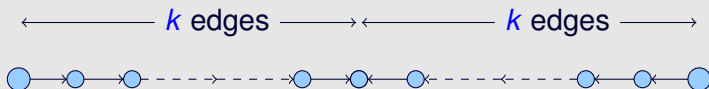
There cannot be an induced “N” in H , where the “edges” are k -paths.

(Bulatov & Dalmau (2003) gave a weaker one-layer version.)



Homomorphisms to directed paths

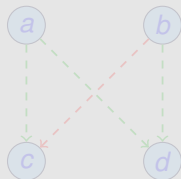
Bulatov & Grohe's theorem, with a simple k -path gadget to simulate the edges of an undirected multigraph, gives structural conditions for easy H :



In particular:

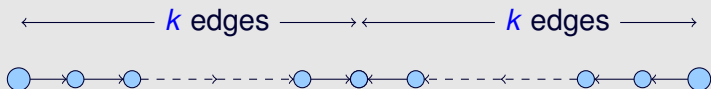
There cannot be an induced "N" in H , where the "edges" are k -paths.

(Bulatov & Dalmau (2003) gave a weaker one-layer version.)



Homomorphisms to directed paths

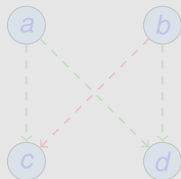
Bulatov & Grohe's theorem, with a simple k -path gadget to simulate the edges of an undirected multigraph, gives structural conditions for easy H :



In particular:

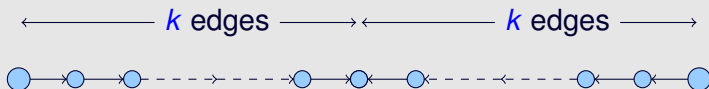
There cannot be an induced "N" in H , where the "edges" are k -paths.

(Bulatov & Dalmau (2003) gave a weaker one-layer version.)



Homomorphisms to directed paths

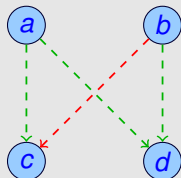
Bulatov & Grohe's theorem, with a simple k -path gadget to simulate the edges of an undirected multigraph, gives structural conditions for easy H :



In particular:

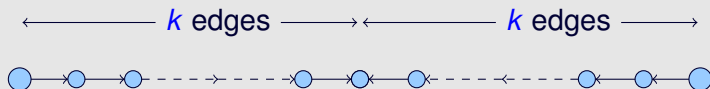
There cannot be an induced "N" in H , where the "edges" are k -paths.

(Bulatov & Dalmou (2003) gave a weaker one-layer version.)



Homomorphisms to directed paths

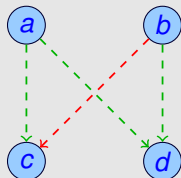
Bulatov & Grohe's theorem, with a simple k -path gadget to simulate the edges of an undirected multigraph, gives structural conditions for easy H :



In particular:

There cannot be an induced “N” in H , where the “edges” are k -paths.

(Bulatov & Dalmou (2003) gave a weaker one-layer version.)

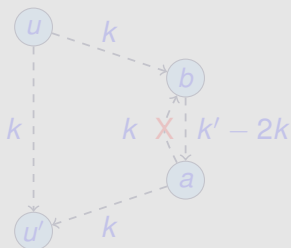


Reduction from acyclic to layered H

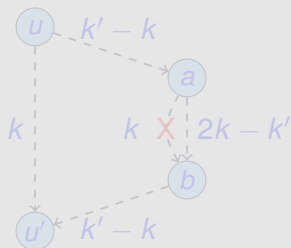
We use the absence of N 's to reduce to the layered case.

Suppose H has paths of length $k > 0$ and $k' > k$ from u to u' .
Choose k as small as possible and k' as large as possible.

(a) $k' \geq 2k$:



(b) $k' < 2k$:



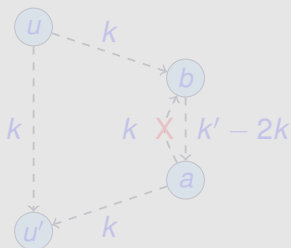
Thus $\#H$ is $\#P$ -Complete unless $H \in \mathcal{G}_\infty$.

Reduction from acyclic to layered H

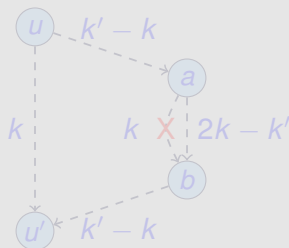
We use the absence of N's to reduce to the layered case.

Suppose H has paths of length $k > 0$ and $k' > k$ from u to u' .
Choose k as small as possible and k' as large as possible.

(a) $k' \geq 2k$:



(b) $k' < 2k$:



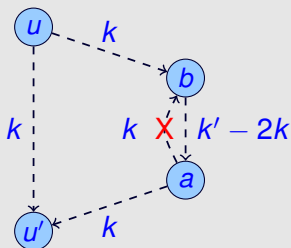
Thus $\#H$ is $\#P$ -Complete unless $H \in \mathcal{G}_\infty$.

Reduction from acyclic to layered H

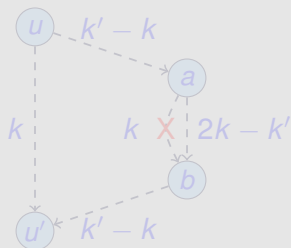
We use the absence of N's to reduce to the layered case.

Suppose H has paths of length $k > 0$ and $k' > k$ from u to u' .
Choose k as small as possible and k' as large as possible.

(a) $k' \geq 2k$:



(b) $k' < 2k$:



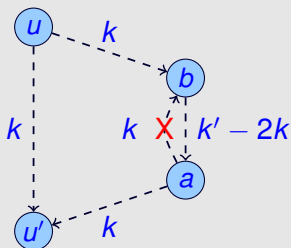
Thus $\#H$ is $\#P$ -Complete unless $H \in \mathcal{G}_\infty$.

Reduction from acyclic to layered H

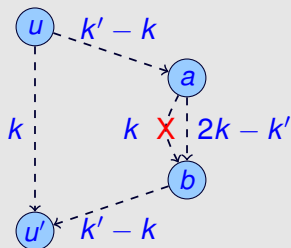
We use the absence of N's to reduce to the layered case.

Suppose H has paths of length $k > 0$ and $k' > k$ from u to u' .
Choose k as small as possible and k' as large as possible.

(a) $k' \geq 2k$:



(b) $k' < 2k$:



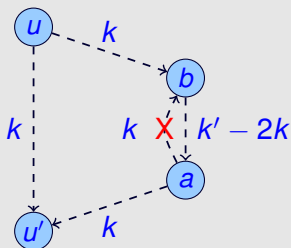
Thus $\#H$ is $\#P$ -Complete unless $H \in \mathcal{G}_\infty$.

Reduction from acyclic to layered H

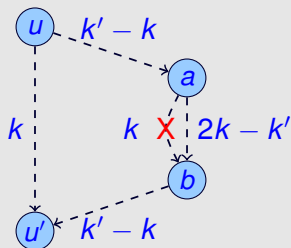
We use the absence of N's to reduce to the layered case.

Suppose H has paths of length $k > 0$ and $k' > k$ from u to u' .
Choose k as small as possible and k' as large as possible.

(a) $k' \geq 2k$:



(b) $k' < 2k$:



Thus $\#H$ is $\#P$ -Complete unless $H \in \mathcal{G}_\infty$.

Layered H

Our proof for the layered case uses the result of [Bulatov and Grohe \(2004\)](#) and a result [Lovász \(1967\)](#) which, specialised to DAGs, is as follows.

Theorem (Lovász)

If $H_1, H_2 \in \vec{\mathcal{D}}$, and $\#H_1(G) = \#H_2(G) (\forall G \in \mathcal{D})$, then $H_1 \cong H_2$.

The proof also yields a very small witness G (a subgraph of H_1 or H_2) to $\exists G \in \mathcal{D} : \#H_1(G) \neq \#H_2(G)$ whenever $H_1 \not\cong H_2$.

We also rely on a [graph product](#) for layered graphs: the [layered cross-product](#) of [Even & Litman \(1997\)](#).

Layered H

Our proof for the layered case uses the result of [Bulatov and Grohe \(2004\)](#) and a result [Lovász \(1967\)](#) which, specialised to DAGs, is as follows.

Theorem (Lovász)

If $H_1, H_2 \in \overrightarrow{\mathcal{D}}$, and $\#H_1(G) = \#H_2(G) \ (\forall G \in \mathcal{D})$, then $H_1 \cong H_2$.

The proof also yields a very small witness G (a subgraph of H_1 or H_2) to $\exists G \in \mathcal{D} : \#H_1(G) \neq \#H_2(G)$ whenever $H_1 \not\cong H_2$.

We also rely on a [graph product](#) for layered graphs: the [layered cross-product](#) of [Even & Litman \(1997\)](#).

Layered H

Our proof for the layered case uses the result of [Bulatov and Grohe \(2004\)](#) and a result [Lovász \(1967\)](#) which, specialised to DAGs, is as follows.

Theorem (Lovász)

If $H_1, H_2 \in \overrightarrow{\mathcal{D}}$, and $\#H_1(G) = \#H_2(G) \ (\forall G \in \mathcal{D})$, then $H_1 \cong H_2$.

The proof also yields a very small witness G (a subgraph of H_1 or H_2) to $\exists G \in \mathcal{D} : \#H_1(G) \neq \#H_2(G)$ whenever $H_1 \not\cong H_2$.

We also rely on a [graph product](#) for layered graphs: the [layered cross-product](#) of [Even & Litman \(1997\)](#).

Layered H

Our proof for the layered case uses the result of [Bulatov and Grohe \(2004\)](#) and a result [Lovász \(1967\)](#) which, specialised to DAGs, is as follows.

Theorem (Lovász)

If $H_1, H_2 \in \overrightarrow{\mathcal{D}}$, and $\#H_1(G) = \#H_2(G) \ (\forall G \in \mathcal{D})$, then $H_1 \cong H_2$.

The proof also yields a very small witness G (a subgraph of H_1 or H_2) to $\exists G \in \mathcal{D} : \#H_1(G) \neq \#H_2(G)$ whenever $H_1 \not\cong H_2$.

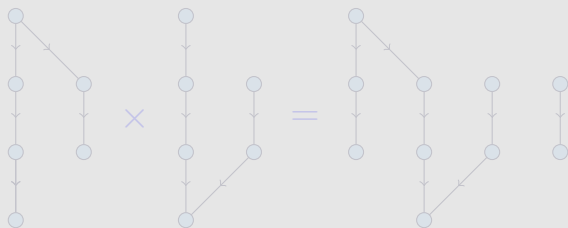
We also rely on a [graph product](#) for layered graphs: the [layered cross-product](#) of [Even & Litman \(1997\)](#).

Layered cross-product

The layered cross product $H = H_1 \times H_2$ of $H_1 = (\mathcal{V}_1, \mathcal{E}_1) \in \mathcal{G}_\ell$ and $H_2 = (\mathcal{V}_2, \mathcal{E}_2) \in \mathcal{G}_\ell$ is a digraph $H = (\mathcal{V}, \mathcal{E}) \in \mathcal{G}_\ell$ such that $\mathcal{V}_i = \mathcal{V}_{1i} \times \mathcal{V}_{2i}$ ($i = 0, \dots, \ell$), and we have $((u_1, u_2), (u'_1, u'_2)) \in \mathcal{E}$ if and only if $(u_1, u'_1) \in \mathcal{E}_1$ and $(u_2, u'_2) \in \mathcal{E}_2$.

We write $H_1 \times H_2$ simply as $H_1 H_2$. It is clear that $H_1 H_2$ is connected only if both H_1 and H_2 are connected.

The converse is not necessarily true.

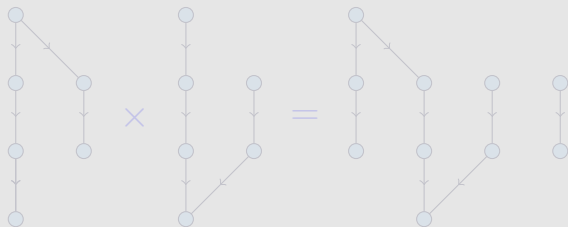


Layered cross-product

The layered cross product $H = H_1 \times H_2$ of $H_1 = (\mathcal{V}_1, \mathcal{E}_1) \in \mathcal{G}_\ell$ and $H_2 = (\mathcal{V}_2, \mathcal{E}_2) \in \mathcal{G}_\ell$ is a digraph $H = (\mathcal{V}, \mathcal{E}) \in \mathcal{G}_\ell$ such that $\mathcal{V}_i = \mathcal{V}_{1i} \times \mathcal{V}_{2i}$ ($i = 0, \dots, \ell$), and we have $((u_1, u_2), (u'_1, u'_2)) \in \mathcal{E}$ if and only if $(u_1, u'_1) \in \mathcal{E}_1$ and $(u_2, u'_2) \in \mathcal{E}_2$.

We write $H_1 \times H_2$ simply as $H_1 H_2$. It is clear that $H_1 H_2$ is connected only if both H_1 and H_2 are connected.

The converse is not necessarily true.

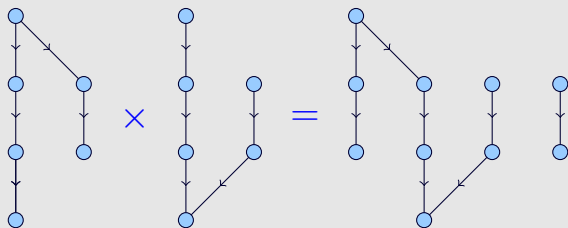


Layered cross-product

The layered cross product $H = H_1 \times H_2$ of $H_1 = (\mathcal{V}_1, \mathcal{E}_1) \in \mathcal{G}_\ell$ and $H_2 = (\mathcal{V}_2, \mathcal{E}_2) \in \mathcal{G}_\ell$ is a digraph $H = (\mathcal{V}, \mathcal{E}) \in \mathcal{G}_\ell$ such that $\mathcal{V}_i = \mathcal{V}_{1i} \times \mathcal{V}_{2i}$ ($i = 0, \dots, \ell$), and we have $((u_1, u_2), (u'_1, u'_2)) \in \mathcal{E}$ if and only if $(u_1, u'_1) \in \mathcal{E}_1$ and $(u_2, u'_2) \in \mathcal{E}_2$.

We write $H_1 \times H_2$ simply as $H_1 H_2$. It is clear that $H_1 H_2$ is connected only if both H_1 and H_2 are connected.

The converse is not necessarily true.



Relationship to counting

If $H_1, H_2 \in \mathcal{G}_\ell$, we write $H_1 \equiv H_2$ if they are isomorphic after deleting “short” components, i.e. those which are in \mathcal{G}_k for some $k < \ell$.

Now, if $H \equiv H_1 H_2$, it follows that $\#H(G) = \#H_1(G)\#H_2(G)$ for every $G \in \mathcal{G}_\ell$ with **exactly** ℓ layers.

We focus on certain induced subgraphs H_{st} of $H \in \mathcal{G}_\ell$.

We define H_{st} formally as follows. If $s \in \mathcal{V}_i$ and $t \in \mathcal{V}_j$ for $i < j$, then H_{st} is the subgraph of H induced by s and t and every component, to which both s and t are incident, of the subgraph induced by $\mathcal{V}_{i+1}, \dots, \mathcal{V}_{j-1}$. Otherwise H_{st} is empty.

Relationship to counting

If $H_1, H_2 \in \mathcal{G}_\ell$, we write $H_1 \equiv H_2$ if they are isomorphic after deleting “short” components, i.e. those which are in \mathcal{G}_k for some $k < \ell$.

Now, if $H \equiv H_1 H_2$, it follows that $\#H(G) = \#H_1(G)\#H_2(G)$ for every $G \in \mathcal{G}_\ell$ with **exactly** ℓ layers.

We focus on certain induced subgraphs H_{st} of $H \in \mathcal{G}_\ell$.

We define H_{st} formally as follows. If $s \in \mathcal{V}_i$ and $t \in \mathcal{V}_j$ for $i < j$, then H_{st} is the subgraph of H induced by s and t and every component, to which both s and t are incident, of the subgraph induced by $\mathcal{V}_{i+1}, \dots, \mathcal{V}_{j-1}$. Otherwise H_{st} is empty.

Relationship to counting

If $H_1, H_2 \in \mathcal{G}_\ell$, we write $H_1 \equiv H_2$ if they are isomorphic after deleting “short” components, i.e. those which are in \mathcal{G}_k for some $k < \ell$.

Now, if $H \equiv H_1 H_2$, it follows that $\#H(G) = \#H_1(G)\#H_2(G)$ for every $G \in \mathcal{G}_\ell$ with **exactly** ℓ layers.

We focus on certain induced subgraphs H_{st} of $H \in \mathcal{G}_\ell$.

We define H_{st} formally as follows. If $s \in \mathcal{V}_i$ and $t \in \mathcal{V}_j$ for $i < j$, then H_{st} is the subgraph of H induced by s and t and every component, to which both s and t are incident, of the subgraph induced by $\mathcal{V}_{i+1}, \dots, \mathcal{V}_{j-1}$. Otherwise H_{st} is empty.

Relationship to counting

If $H_1, H_2 \in \mathcal{G}_\ell$, we write $H_1 \equiv H_2$ if they are isomorphic after deleting “short” components, i.e. those which are in \mathcal{G}_k for some $k < \ell$.

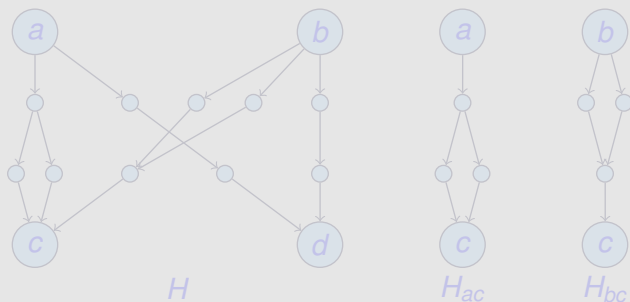
Now, if $H \equiv H_1 H_2$, it follows that $\#H(G) = \#H_1(G)\#H_2(G)$ for every $G \in \mathcal{G}_\ell$ with **exactly** ℓ layers.

We focus on certain induced subgraphs H_{st} of $H \in \mathcal{G}_\ell$.

We define H_{st} formally as follows. If $s \in \mathcal{V}_i$ and $t \in \mathcal{V}_j$ for $i < j$, then H_{st} is the subgraph of H induced by s and t and every component, to which both s and t are incident, of the subgraph induced by $\mathcal{V}_{i+1}, \dots, \mathcal{V}_{j-1}$. Otherwise H_{st} is empty.

Lovász goodness

Intuitively, H_{st} is the subgraph of H lying between s and t .

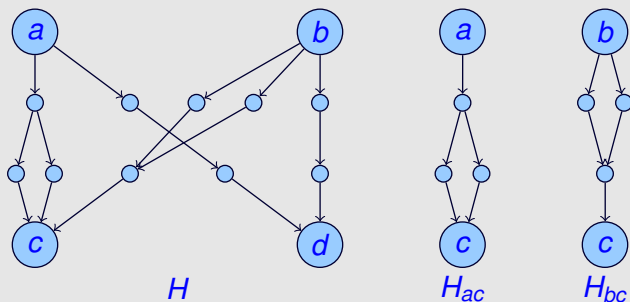


Four vertices $a, b, c, d \in \mathcal{V}$, with $a, b \in \mathcal{V}_i$ and $c, d \in \mathcal{V}_j$ ($0 \leq i < j \leq \ell$), are a **Lovász violation** if at most one of $H_{ac}, H_{bc}, H_{ad}, H_{bd}$ is empty and $H_{ac}H_{bd} \neq H_{ad}H_{bc}$.

We say H is **Lovász-good** if it has no Lovász violation.

Lovász goodness

Intuitively, H_{st} is the subgraph of H lying between s and t .

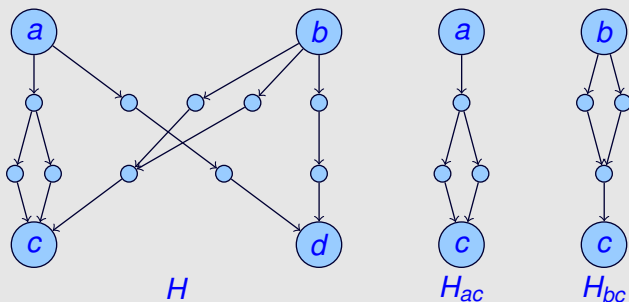


Four vertices $a, b, c, d \in \mathcal{V}$, with $a, b \in \mathcal{V}_i$ and $c, d \in \mathcal{V}_j$ ($0 \leq i < j \leq \ell$), are a **Lovász violation** if at most one of $H_{ac}, H_{bc}, H_{ad}, H_{bd}$ is empty and $H_{ac}H_{bd} \neq H_{ad}H_{bc}$.

We say H is **Lovász-good** if it has no Lovász violation.

Lovász goodness

Intuitively, H_{st} is the subgraph of H lying between s and t .

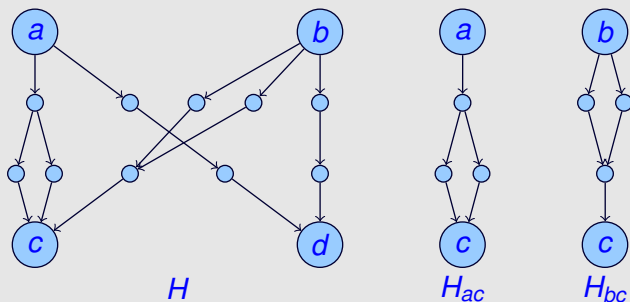


Four vertices $a, b, c, d \in \mathcal{V}$, with $a, b \in \mathcal{V}_i$ and $c, d \in \mathcal{V}_j$ ($0 \leq i < j \leq \ell$), are a **Lovász violation** if at most one of $H_{ac}, H_{bc}, H_{ad}, H_{bd}$ is empty and $H_{ac} H_{bd} \neq H_{ad} H_{bc}$.

We say H is **Lovász-good** if it has no Lovász violation.

Lovász goodness

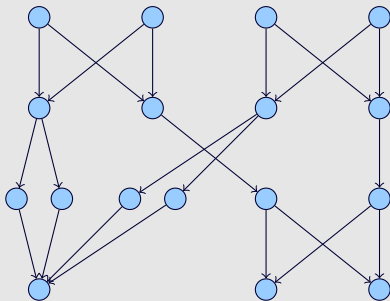
Intuitively, H_{st} is the subgraph of H lying between s and t .



Four vertices $a, b, c, d \in \mathcal{V}$, with $a, b \in \mathcal{V}_i$ and $c, d \in \mathcal{V}_j$ ($0 \leq i < j \leq \ell$), are a **Lovász violation** if at most one of $H_{ac}, H_{bc}, H_{ad}, H_{bd}$ is empty and $H_{ac}H_{bd} \neq H_{ad}H_{bc}$.

We say H is **Lovász-good** if it has no Lovász violation.

The dichotomy theorem



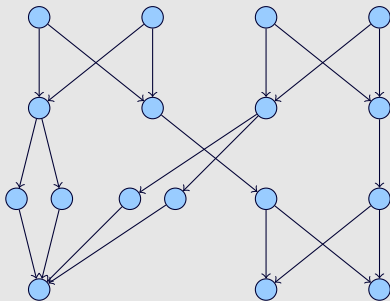
A Lovász-good H

We can now state the dichotomy theorem.

Theorem

Let H be a directed acyclic graph. Then $\#H$ is in FP if H is layered and Lovász-good. Otherwise $\#H$ is $\#P$ -Complete.

The dichotomy theorem



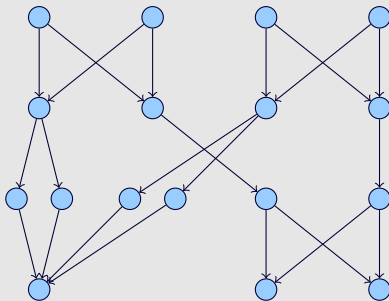
A Lovász-good H

We can now state the dichotomy theorem.

Theorem

Let H be a directed acyclic graph. Then $\#H$ is in FP if H is layered and Lovász-good. Otherwise $\#H$ is $\#P$ -Complete.

The dichotomy theorem



A Lovász-good H

We can now state the dichotomy theorem.

Theorem

Let H be a directed acyclic graph. Then $\#H$ is in FP if H is layered and Lovász-good. Otherwise $\#H$ is $\#P$ -Complete.

Proof outline

The proof is an induction based on the following approach

- 1 Factorising, in the layered cross-product, a Lovász-good H into graphs with a **single** source or sink.
- 2 Reducing the number of levels in each such factor.

Example



Proof outline

The proof is an induction based on the following approach

- 1 Factorising, in the layered cross-product, a Lovász-good H into graphs with a **single** source or sink.
- 2 Reducing the number of levels in each such factor.

Example



Proof outline

The proof is an induction based on the following approach

- 1 Factorising, in the layered cross-product, a Lovász-good H into graphs with a **single** source or sink.
- 2 Reducing the number of levels in each such factor.

Example

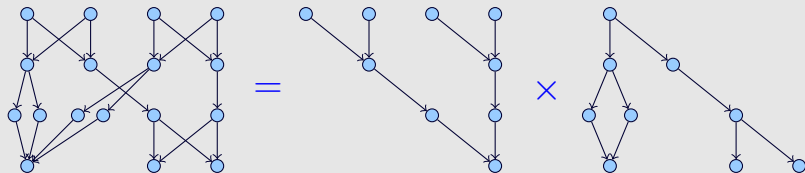


Proof outline

The proof is an induction based on the following approach

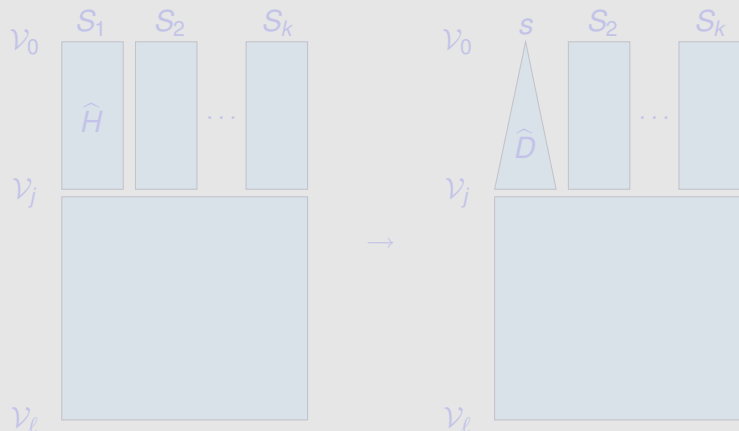
- 1 Factorising, in the layered cross-product, a Lovász-good H into graphs with a **single** source or sink.
- 2 Reducing the number of levels in each such factor.

Example



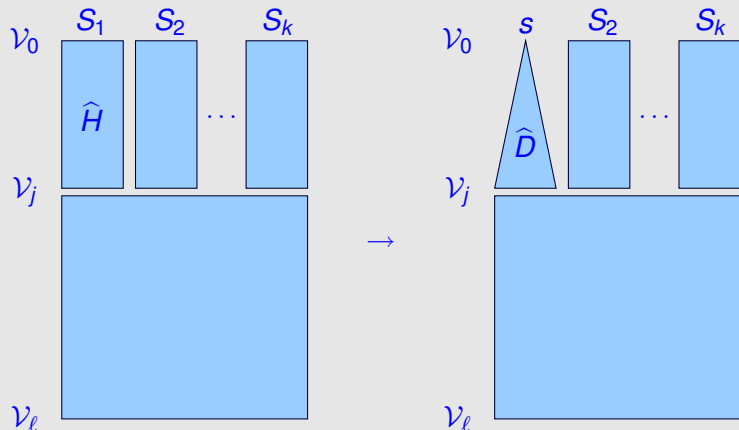
Removing local factors

Recursively simplify the local structure by factoring subgraphs:



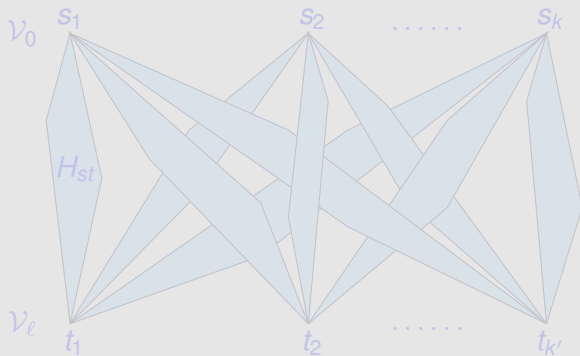
Removing local factors

Recursively simplify the local structure by factoring subgraphs:



Disjoint case

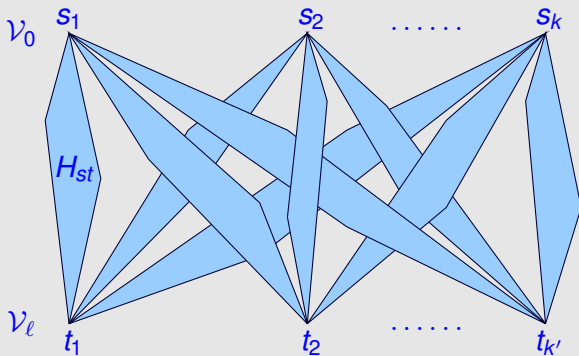
Eventually we arrive at an H which is disjoint except at top-level sources and bottom-level sinks.



Lovász-goodness almost implies that such a graph must factor, but unfortunately this is not quite true.

Disjoint case

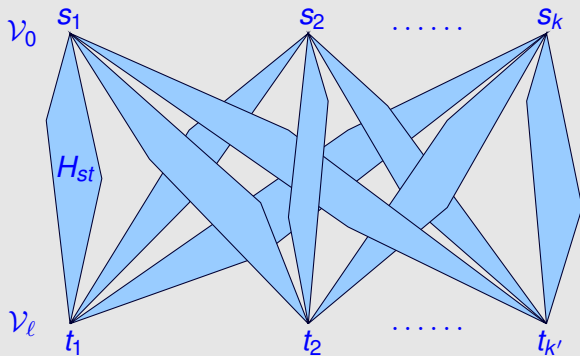
Eventually we arrive at an H which is disjoint except at top-level sources and bottom-level sinks.



Lovász-goodness almost implies that such a graph must factor, but unfortunately this is not quite true.

Disjoint case

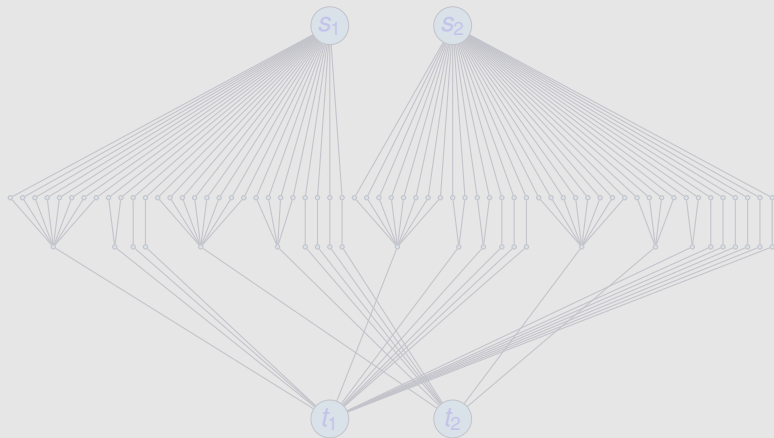
Eventually we arrive at an H which is disjoint except at top-level sources and bottom-level sinks.



Lovász-goodness almost implies that such a graph must factor, but unfortunately this is not quite true.

Irreducible Lovász-good graphs

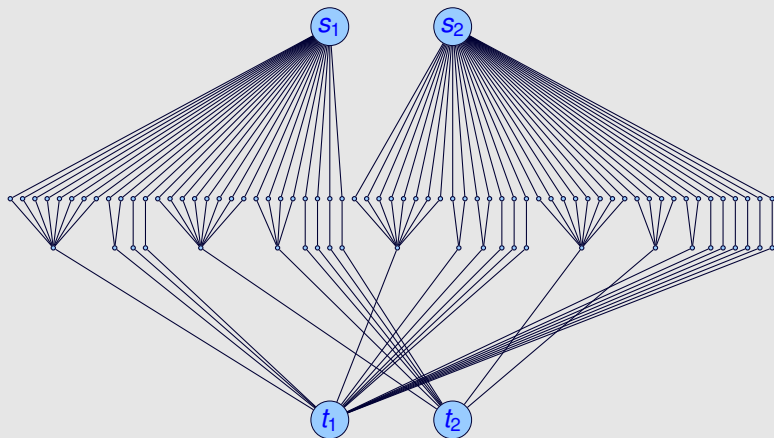
There exist Lovász-good H which have no nontrivial factors.



Algebraically, this is due to the the LCP lacking the property of **unique factorisation into primes**.

Irreducible Lovász-good graphs

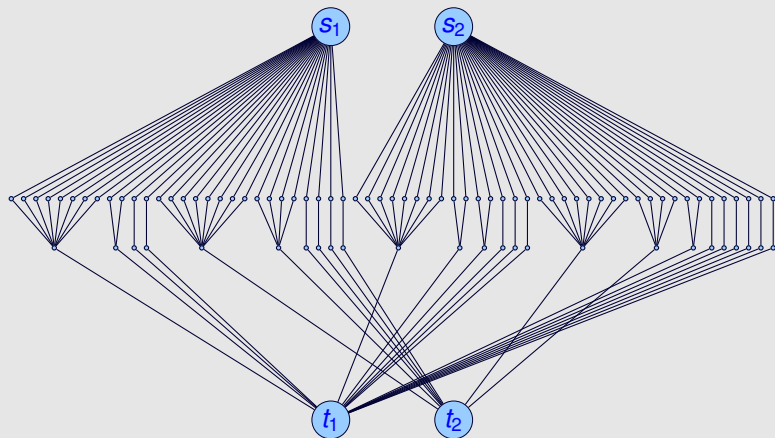
There exist Lovász-good H which have no nontrivial factors.



Algebraically, this is due to the the LCP lacking the property of **unique factorisation into primes**.

Irreducible Lovász-good graphs

There exist Lovász-good H which have no nontrivial factors.



Algebraically, this is due to the the LCP lacking the property of **unique factorisation into primes**.

Preconditioning H

This is handled as follows. Suppose the sources are s_1, s_2, \dots, s_k and the sinks are $t_1, t_2, \dots, t_{k'}$, and the H_{st} are disjoint other than at the sources and sinks.

Premultiply the whole graph by $H_{s_1 t_1}$, and note that

$$H_{s_1 t_1} H_{st} = H_{s_1 t} H_{s_1 t_1} \quad \text{by Lovász-goodness of } H.$$

Thus $H_{s_1 t_1} H$ factors into primes as



Preconditioning H

This is handled as follows. Suppose the sources are s_1, s_2, \dots, s_k and the sinks are $t_1, t_2, \dots, t_{k'}$, and the H_{st} are disjoint other than at the sources and sinks.

Premultiply the whole graph by $H_{s_1 t_1}$, and note that

$$H_{s_1 t_1} H_{st} = H_{s_1 t} H_{s_1 t_1} \quad \text{by Lovász-goodness of } H.$$

Thus $H_{s_1 t_1} H$ factors into primes as



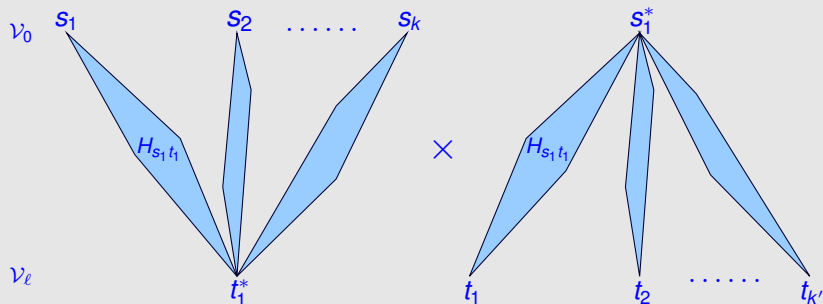
Preconditioning H

This is handled as follows. Suppose the sources are s_1, s_2, \dots, s_k and the sinks are $t_1, t_2, \dots, t_{k'}$, and the H_{st} are disjoint other than at the sources and sinks.

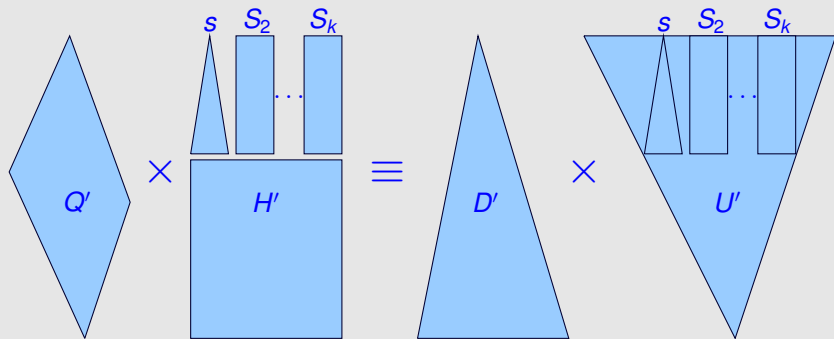
Premultiply the whole graph by $H_{s_1 t_1}$, and note that

$$H_{s_1 t_1} H_{st} = H_{s_1 t} H_{s_1 t_1} \quad \text{by Lovász-goodness of } H.$$

Thus $H_{s_1 t_1} H$ factors into primes as



Factorisation

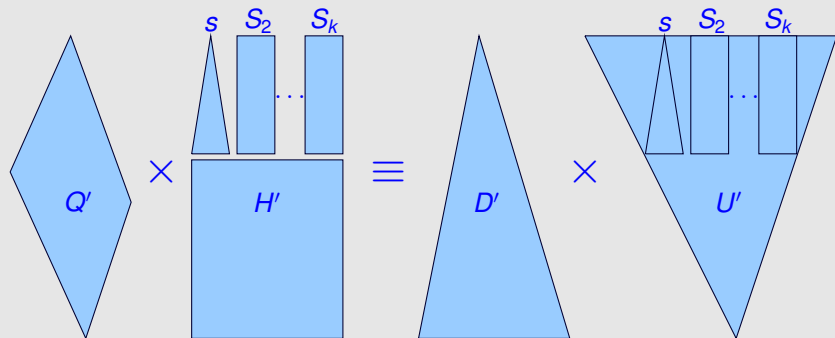


Calculate $\#H'(G) = \#D'(G)\#U'(G)/\#Q'(G)$.

$\#Q'(G)$, $\#D'(G)$, $\#U'(G)$ are determined recursively by reducing the number of levels.

The counting algorithm for Lovász-good H is clearly non-trivial.

Factorisation

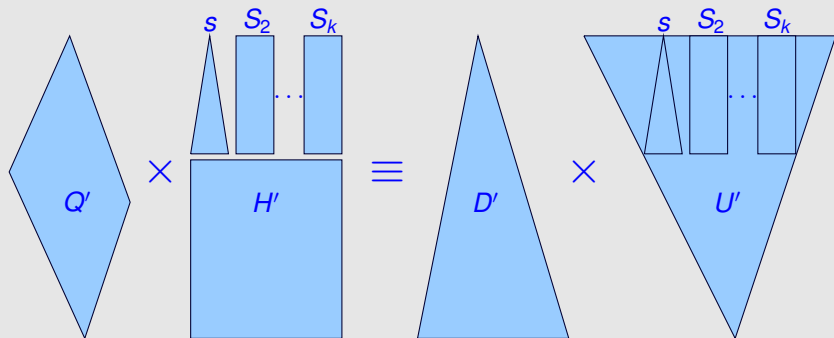


Calculate $\#H'(G) = \#D'(G)\#U'(G)/\#Q'(G)$.

$\#Q'(G)$, $\#D'(G)$, $\#U'(G)$ are determined recursively by reducing the number of levels.

The counting algorithm for Lovász-good H is clearly non-trivial.

Factorisation

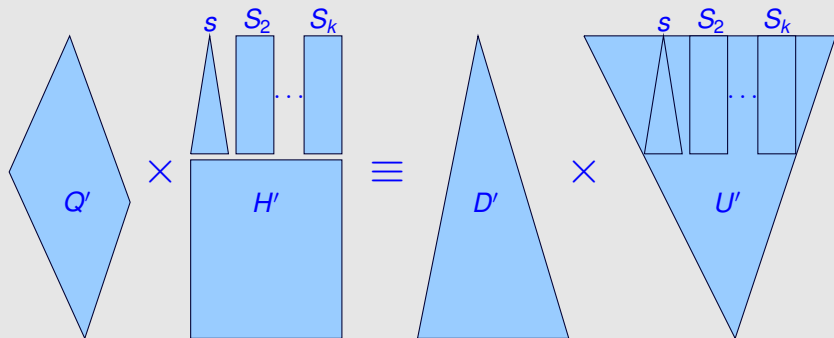


Calculate $\#H'(G) = \#D'(G)\#U'(G)/\#Q'(G)$.

$\#Q'(G), \#D'(G), \#U'(G)$ are determined recursively by reducing the number of levels.

The counting algorithm for Lovász-good H is clearly non-trivial.

Factorisation



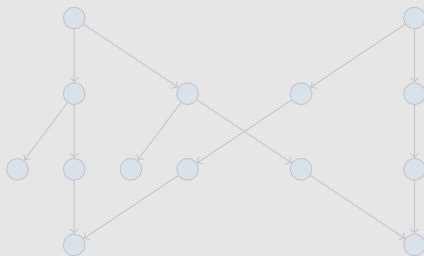
Calculate $\#H'(G) = \#D'(G)\#U'(G)/\#Q'(G)$.

$\#Q'(G)$, $\#D'(G)$, $\#U'(G)$ are determined recursively by reducing the number of levels.

The counting algorithm for Lovász-good H is clearly non-trivial.

Danglers

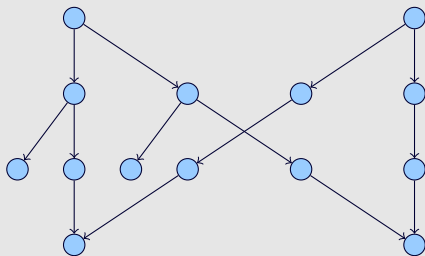
There is another problem in making this idea work:
not all sources need be in \mathcal{V}_0 , nor sinks in \mathcal{V}_ℓ .



We call these sources and sinks “danglers”. They complicate an induction on the number of levels, but fortunately don’t cause insurmountable difficulties.

Danglers

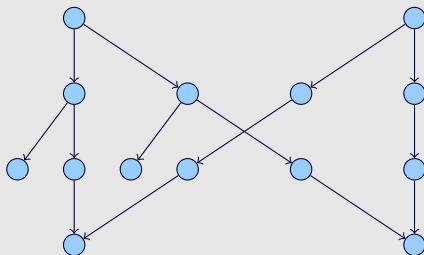
There is another problem in making this idea work:
not all sources need be in \mathcal{V}_0 , nor sinks in \mathcal{V}_ℓ .



We call these sources and sinks “danglers”. They complicate an induction on the number of levels, but fortunately don’t cause insurmountable difficulties.

Danglers

There is another problem in making this idea work:
not all sources need be in \mathcal{V}_0 , nor sinks in \mathcal{V}_ℓ .



We call these sources and sinks “danglers”. They complicate an induction on the number of levels, but fortunately don’t cause insurmountable difficulties.

