Magnus Wahlström[1]

Abusing the Tutte Matrix

A Polynomial Compression for the K-set-cycle Problem

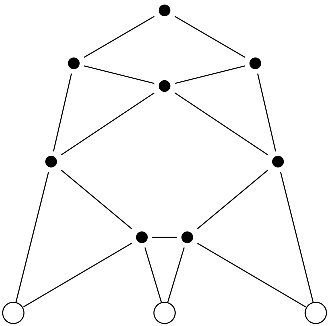
[1]Max-Planck-Institut für Informatik, Germany

March 1, 2013

The K-Cycle Problem

K-Cycle

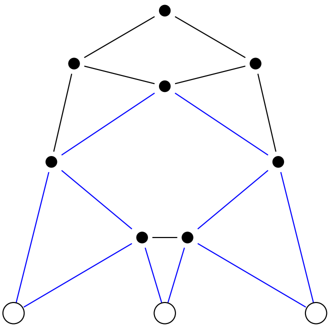
Given a graph $G = (V, E)$, with terminals $K \subseteq V$: Is there a (simple) cycle through all of K?



The K-Cycle Problem

K-Cycle

Given a graph $G = (V, E)$, with terminals $K \subseteq V$: Is there a (simple) cycle through all of K?



Previous work (algorithms)

- Ken-ichi Kawarabayashi (IPCO '08)
  - First $\mathcal{O}^*(f(|K|)$-time (FPT) algorithm
  - Graph minors-style algorithm; $f(k) = 2^{2^{k^{10}}}$ (estimated)
- Björklund, Husfeldt, Taslaman (SODA '12)
  - Algebraic algorithm (define (exponentially) large polynomial over $GF(2^\ell)$; test if all terms cancel).
  - Time $\mathcal{O}^*(2^k)$
- This talk:
  - Interpret $O(2^k)$-time algorithm as determinant sums procedure (Björklund, FOCS '10)
  - Derive polynomial compression

Previous work (algorithms)

- Ken-ichi Kawarabayashi (IPCO '08)
  - First $\mathcal{O}^*(f(|K|)$-time (FPT) algorithm
  - Graph minors-style algorithm; $f(k) = 2^{2^{k^{10}}}$ (estimated)
- Björklund, Husfeldt, Taslaman (SODA '12)
  - Algebraic algorithm (define (exponentially) large polynomial over GF(2^ℓ); test if all terms cancel).
  - Time $\mathcal{O}^*(2^k)$
- This talk:
  - Interpret $O(2^k)$-time algorithm as determinant sums procedure (Björklund, FOCS '10)
  - Derive polynomial compression

Kernelization and Compression

- **Kernelization:** Given input (G, K),
  - Produce output (G', K') of same problem
  - of total length $f(|K|)$,
  - in time poly$(|G|)$.
- **Compression:** Given input (G, K),
  - Produce output X of any (fixed) problem
  - of total length $f(|K|)$,
  - in time poly$(|G|)$.
- Both notions seemingly equivalent

Kernelization and Compression

- **Kernelization:** Given input (G, K),
  - Produce output (G', K') of same problem
  - of total length $f(|K|)$,
  - in time poly($|G|$).
- **Compression:** Given input (G, K),
  - Produce output X of any (fixed) problem
  - of total length $f(|K|)$,
  - in time poly($|G|$).
- Both notions seemingly equivalent...until now.

Why is a compression surprising?

- *k*-Cycle (find cycle of length at least *k*)
  - NO polynomial compression unless PH collapses
- Ordered *K*-Cycle (find cycle through *K* with specified order)
  - Equivalent to Disjoint Paths
  - NO polynomial compression unless PH collapses
- *K*-Cycle (this talk):
  - Polynomial (cubic) compression
  - Kernel unknown (compression not (known to be) within NP)

Why is a compression surprising?

- k-Cycle (find cycle of length at least k)
  - NO polynomial compression unless PH collapses
- Ordered K-Cycle (find cycle through K with specified order)
  - Equivalent to Disjoint Paths
  - NO polynomial compression unless PH collapses
- K-Cycle (this talk):
  - Polynomial (cubic) compression
  - Kernel unknown (compression not (known to be) within NP)

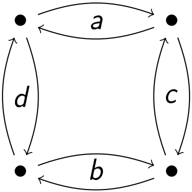
Why is a compression surprising?

- k-Cycle (find cycle of length at least k)
  - NO polynomial compression unless PH collapses
- Ordered K-Cycle (find cycle through K with specified order)
  - Equivalent to Disjoint Paths
  - NO polynomial compression unless PH collapses
- K-Cycle (this talk):
  - Polynomial (cubic) compression
  - Kernel unknown (compression not (known to be) within NP)

Determinants and Cycle Covers

Consider determinant over GF(2^ℓ).

$$A = \begin{pmatrix} 0 & a & d & 0 \\ a & 0 & 0 & c \\ d & 0 & 0 & b \\ 0 & c & b & 0 \end{pmatrix}$$



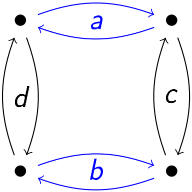
$$\det A = aabb + acdb + dabc + dcdc$$

After cancellation, enumerates matchings.

Determinants and Cycle Covers

Consider determinant over GF(2^ℓ).

$$A = \begin{pmatrix} 0 & a & d & 0 \\ a & 0 & 0 & c \\ d & 0 & 0 & b \\ 0 & c & b & 0 \end{pmatrix}$$



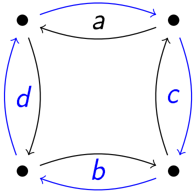
$$\det A = aabb + acdb + dabc + dcdc$$

After cancellation, enumerates matchings.

Determinants and Cycle Covers

Consider determinant over $GF(2^\ell)$.

$$A = \begin{pmatrix} 0 & a & d & 0 \\ a & 0 & 0 & c \\ d & 0 & 0 & b \\ 0 & c & b & 0 \end{pmatrix}$$



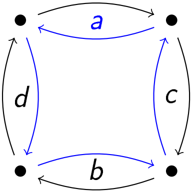
$$\det A = aabb + acdb + dabc + dcdc$$

After cancellation, enumerates matchings.

Determinants and Cycle Covers

Consider determinant over $\mathrm{GF}(2^\ell)$.

$$A = \begin{pmatrix} 0 & a & d & 0 \\ a & 0 & 0 & c \\ d & 0 & 0 & b \\ 0 & c & b & 0 \end{pmatrix}$$



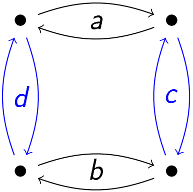
$$\det A = aabb + acdb + dabc + dcdc$$

After cancellation, enumerates matchings.

Determinants and Cycle Covers

Consider determinant over $GF(2^\ell)$.

$$A = \begin{pmatrix} 0 & a & d & 0 \\ a & 0 & 0 & c \\ d & 0 & 0 & b \\ 0 & c & b & 0 \end{pmatrix}$$



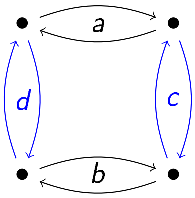
$$\det A = aabb + acdb + dabc + dcdc$$

After cancellation, enumerates matchings.

Determinants and Cycle Covers

Consider determinant over $GF(2^\ell)$.

$$A = \begin{pmatrix} 0 & a & d & 0 \\ a & 0 & 0 & c \\ d & 0 & 0 & b \\ 0 & c & b & 0 \end{pmatrix}$$



$$\det A = aabb + acdb + dabc + dcdc$$

After cancellation, enumerates matchings.

Determinants and Cycle Covers

Basic fact

Over GF(2^ℓ), $\det A$ enumerates cycle covers with only short or non-reversible cycles.

- Cycle C short: $|C| \leq 2$
- Cycle C non-reversible: $A(i,j) \neq A(j,i)$, some $ij \in C$
- **Proof:** Reverse first reversible long cycle (e.g., by vertex incidence). Bijection between identical terms.

Determinants and Cycle Covers

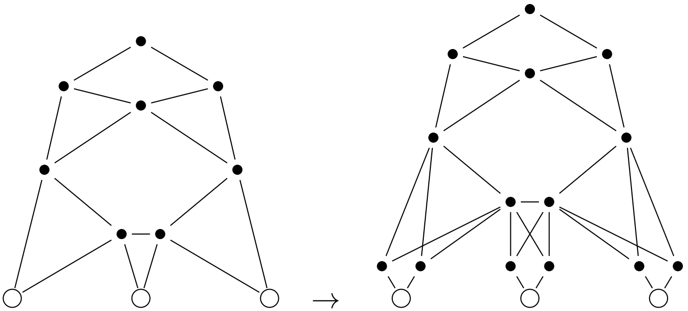
Basic fact

Over $GF(2^\ell)$, $\det A$ enumerates cycle covers with only short or non-reversible cycles.

- Cycle C short: $|C| \leq 2$
- Cycle C non-reversible: $A(i,j) \neq A(j,i)$, some $ij \in C$
- **Proof:** Reverse first reversible long cycle (e.g., by vertex incidence). Bijection between identical terms.

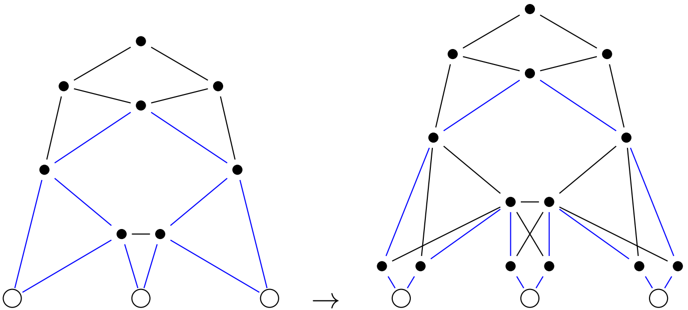
K-Cycle Preprocessing Step

Split every $v \in K$ such that $d(v) = 2$. Orient one edge. Add loops.



K-Cycle Preprocessing Step

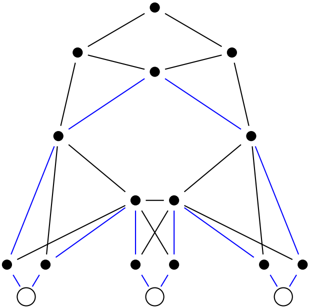
Split every $v \in K$ such that $d(v) = 2$. Orient one edge. Add loops.



Essential observations

Observation 1

Let $v \in K$, x_i, x_j labels of incident edges. A monomial m in $\det A$ is divided by $x_i x_j$ if and only if the corresponding cycle cover contains a long cycle through v.



Essential observations

Observation 1

Let $v \in K$, x_i, x_j labels of incident edges. A monomial m in $\det A$ is divided by $x_i x_j$ if and only if the corresponding cycle cover contains a long cycle through v.

Observation 2

Let $F \subseteq E$ be all edges incident on K. Then G contains a K-cycle iff $\prod_{e \in F} x_e$ divides some term of $\det A$.

Monomial detection

Let $P(x_1, \ldots, x_n)$ be a polynomial over $GF(2^\ell)$, $I \subseteq [n]$. Define

$$P_{S=0}(\mathbf{x}) = P(\mathbf{x} : x_i = 0, i \in S).$$

Then

$$R(\mathbf{x}) = \sum_{S \subseteq I} P_{S=0}(\mathbf{x})$$

is non-zero iff $T = \prod_{i \in I} x_i$ divides a monomial in P.

Proof sketch. Count contributions of monomials m in P.

- T divides m: Counted exactly once.
- Otherwise: Counted each time $(T \cap m)$ divides T.

The Algorithm

- Let $P(\mathbf{x}) = \det A$. Let $F \subseteq E$ be the edges incident on K.
- Define
$$R(\mathbf{x}) = \sum_{S \subseteq F} P_{S=0}(\mathbf{x}).$$
Then $R(\mathbf{x}) \equiv 0$ iff G has no K-cycle.
- Evaluate $R(\mathbf{x})$ randomly in e.g. $GF(2^{\log n + k})$.
  - Requires $\mathcal{O}^*(2^{2k})$ time.
- Schwartz-Zippel: With probability $1 - 1/2^k$, the result is correct.
- $\mathcal{O}^*(2^k)$-time algorithm similar (sum over orientations)

The Algorithm

- Let $P(\mathbf{x}) = \det A$. Let $F \subseteq E$ be the edges incident on K.
- Define

$$R(\mathbf{x}) = \sum_{S \subseteq F} P_{S=0}(\mathbf{x}).$$

Then $R(\mathbf{x}) \equiv 0$ iff G has no K-cycle.

- Evaluate $R(\mathbf{x})$ randomly in e.g. $GF(2^{\log n+k})$.
  - Requires $\mathcal{O}^*(2^{2k})$ time.
- Schwartz-Zippel: With probability $1 - 1/2^k$, the result is correct.
- $\mathcal{O}^*(2^k)$-time algorithm similar (sum over orientations)

The Algorithm

- Let $P(\mathbf{x}) = \det A$. Let $F \subseteq E$ be the edges incident on K.
- Define
$$R(\mathbf{x}) = \sum_{S \subseteq F} P_{S=0}(\mathbf{x}).$$
Then $R(\mathbf{x}) \equiv 0$ iff G has no K-cycle.
- Evaluate $R(\mathbf{x})$ randomly in e.g. $GF(2^{\log n + k})$.
  - Requires $\mathcal{O}^*(2^{2k})$ time.
- Schwartz-Zippel: With probability $1 - 1/2^k$, the result is correct.
- $\mathcal{O}^*(2^k)$-time algorithm similar (sum over orientations)

The Algorithm

- Let $P(\mathbf{x}) = \det A$. Let $F \subseteq E$ be the edges incident on K.
- Define
$$R(\mathbf{x}) = \sum_{S \subseteq F} P_{S=0}(\mathbf{x}).$$
Then $R(\mathbf{x}) \equiv 0$ iff G has no K-cycle.
- Evaluate $R(\mathbf{x})$ randomly in e.g. $GF(2^{\log n + k})$.
  - Requires $\mathcal{O}^*(2^{2k})$ time.
- Schwartz-Zippel: With probability $1 - 1/2^k$, the result is correct.
- $\mathcal{O}^*(2^k)$-time algorithm similar (sum over orientations)

The Algorithm

- Let $P(\mathbf{x}) = \det A$. Let $F \subseteq E$ be the edges incident on K.
- Define
$$R(\mathbf{x}) = \sum_{S \subseteq F} P_{S=0}(\mathbf{x}).$$
Then $R(\mathbf{x}) \equiv 0$ iff G has no K-cycle.
- Evaluate $R(\mathbf{x})$ randomly in e.g. $GF(2^{\log n + k})$.
  - Requires $\mathcal{O}^*(2^{2k})$ time.
- Schwartz-Zippel: With probability $1 - 1/2^k$, the result is correct.
- $\mathcal{O}^*(2^k)$-time algorithm similar (sum over orientations)

Towards a compression

What parts can we do in polynomial time?

- $R(\mathbf{x}) \not\equiv 0$ iff K-cycle in G.
- Let α be random assignment to $\mathbf{x}$ in $GF(2^\ell)$.
- Want to compute

$$R(\alpha) = \sum_{S \subseteq F} P_{S=0}(\alpha) = \sum_{S \subseteq F} \det A(S=0)$$

- The matrix A now contains no variables!

Towards a compression

What parts can we do in polynomial time?

- $R(\mathbf{x}) \not\equiv 0$ iff K-cycle in G.
- Let α be random assignment to $\mathbf{x}$ in $GF(2^\ell)$.
- Want to compute

$$R(\alpha) = \sum_{S \subseteq F} P_{S=0}(\alpha) = \sum_{S \subseteq F} \det A(S=0)$$

- The matrix A now contains no variables!

Towards a compression

What parts can we do in polynomial time?

- $R(\mathbf{x}) \not\equiv 0$ iff K-cycle in G.
- Let α be random assignment to $\mathbf{x}$ in $GF(2^\ell)$.
- Want to compute

$$R(\alpha) = \sum_{S \subseteq F} P_{S=0}(\alpha) = \sum_{S \subseteq F} \det A(S = 0)$$

- The matrix A now contains no variables!

Towards a compression

What parts can we do in polynomial time?

- $R(\mathbf{x}) \not\equiv 0$ iff K-cycle in G.
- Let α be random assignment to $\mathbf{x}$ in GF(2^ℓ).
- Want to compute

$$R(\alpha) = \sum_{S \subseteq F} P_{S=0}(\alpha) = \sum_{S \subseteq F} \det A(S=0)$$

- The matrix A now contains no variables!

Illustration

$$A = \begin{pmatrix} 0 & F & A_1 \\ F' & 0 & A_2 \\ A_1 & A_2 & A_3 \end{pmatrix}$$

Block sizes $k + 2k + (n - 3k)$. Only F changes (parts set to 0).

Introduce variables $\mathbf{y}$ on F:

$$A(\mathbf{y}) = \begin{pmatrix} 0 & F(\mathbf{y}) & A_1 \\ F'(\mathbf{y}) & 0 & A_2 \\ A_1 & A_2 & A_3 \end{pmatrix}$$

Illustration

$$A = \begin{pmatrix} 0 & F & A_1 \\ F' & 0 & A_2 \\ A_1 & A_2 & A_3 \end{pmatrix}$$

Block sizes $k + 2k + (n - 3k)$. Only F changes (parts set to 0).
Introduce variables $\mathbf{y}$ on F:

$$A(\mathbf{y}) = \begin{pmatrix} 0 & F(\mathbf{y}) & A_1 \\ F'(\mathbf{y}) & 0 & A_2 \\ A_1 & A_2 & A_3 \end{pmatrix}$$

Compression

- Want to compute:

$$R(\alpha) = \sum_{\mathbf{y} \in \{0,1\}^{2k}} \det A(\mathbf{y}).$$

- Only tiny $3k \times 3k$-part of A depends on $\mathbf{y}$.
- Recall: Row, column operations on A preserve $\det A$
- Reduce $A(\mathbf{y})$ to $3k \times 3k$-matrix $A'(\mathbf{y})$ with identical determinant polynomial.

Compression

- Want to compute:

$$R(\alpha) = \sum_{\mathbf{y} \in \{0,1\}^{2k}} \det A(\mathbf{y}).$$

- Only tiny $3k \times 3k$-part of A depends on $\mathbf{y}$.
- Recall: Row, column operations on A preserve $\det A$
- Reduce $A(\mathbf{y})$ to $3k \times 3k$-matrix $A'(\mathbf{y})$ with identical determinant polynomial.

Compression

- Want to compute:

$$R(\alpha) = \sum_{\mathbf{y} \in \{0,1\}^{2k}} \det A(\mathbf{y}).$$

- Only tiny $3k \times 3k$-part of A depends on $\mathbf{y}$.
- Recall: Row, column operations on A preserve $\det A$
- Reduce $A(\mathbf{y})$ to $3k \times 3k$-matrix $A'(\mathbf{y})$ with identical determinant polynomial.

Compression

- Want to compute:

$$R(\alpha) = \sum_{\mathbf{y} \in \{0,1\}^{2k}} \det A(\mathbf{y}).$$

- Only tiny $3k \times 3k$-part of A depends on $\mathbf{y}$.
- Recall: Row, column operations on A preserve $\det A$
- Reduce $A(\mathbf{y})$ to $3k \times 3k$-matrix $A'(\mathbf{y})$ with identical determinant polynomial.

Gaussian reduction

1. $A = \begin{pmatrix} F(\mathbf{y}) & B \\ C & D \end{pmatrix}$
2. $A' = \begin{pmatrix} F'(\mathbf{y}) & 0 \\ 0 & D' \end{pmatrix}$ via row/column operations
3. $A'' = (F'(\mathbf{y})) \cdot \det D'$ since D' is variable-free
4. $\det A'' = \det A$.

Total size $\mathcal{O}(3k \cdot 3k \cdot \ell) = \mathcal{O}(k^3)$ (assume $\log n \leq k$).

Gaussian reduction

1. $A = \begin{pmatrix} F(\mathbf{y}) & B \\ C & D \end{pmatrix}$
2. $A' = \begin{pmatrix} F'(\mathbf{y}) & 0 \\ 0 & D' \end{pmatrix}$ via row/column operations
3. $A'' = (F'(\mathbf{y})) \cdot \det D'$ since D' is variable-free
4. $\det A'' = \det A$.

Total size $\mathcal{O}(3k \cdot 3k \cdot \ell) = \mathcal{O}(k^3)$ (assume $\log n \leq k$).

Gaussian reduction

1. $A = \begin{pmatrix} F(\mathbf{y}) & B \\ C & D \end{pmatrix}$
2. $A' = \begin{pmatrix} F'(\mathbf{y}) & 0 \\ 0 & D' \end{pmatrix}$ via row/column operations
3. $A'' = (F'(\mathbf{y})) \cdot \det D'$ since D' is variable-free
4. $\det A'' = \det A$.

Total size $\mathcal{O}(3k \cdot 3k \cdot \ell) = \mathcal{O}(k^3)$ (assume $\log n \leq k$).

Gaussian reduction

1. $A = \begin{pmatrix} F(\mathbf{y}) & B \\ C & D \end{pmatrix}$
2. $A' = \begin{pmatrix} F'(\mathbf{y}) & 0 \\ 0 & D' \end{pmatrix}$ via row/column operations
3. $A'' = (F'(\mathbf{y})) \cdot \det D'$ since D' is variable-free
4. $\det A'' = \det A$.

Total size $\mathcal{O}(3k \cdot 3k \cdot \ell) = \mathcal{O}(k^3)$ (assume $\log n \leq k$).

Conclusions

- K-Cycle solved via determinant sums
- Implies $\mathcal{O}(|K|^3)$-sized instance compression
- NO polynomial kernel/witness known
- Only known problem where polynomial kernel and polynomial instance compression seem to diverge