Complexity of cycle-transverse matching problems

IWOCA 2011

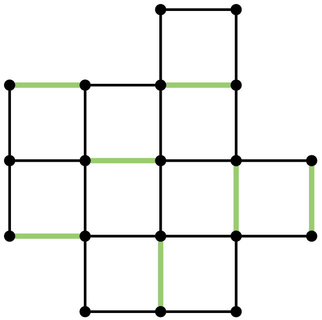
Ross Churchley

University of Victoria

June 18, 2011

Definition

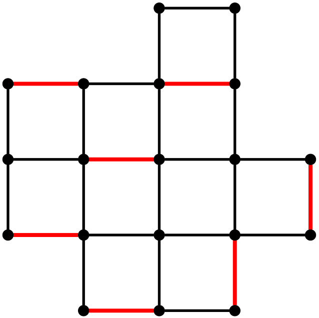
A C_k-**transverse matching** M is one which breaks every cycle of length k



A C_4-transverse matching.

Definition

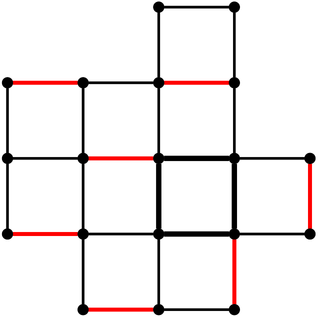
A C_k-**transverse matching** M is one which breaks every cycle of length k



Not a C_4-transverse matching.

Definition

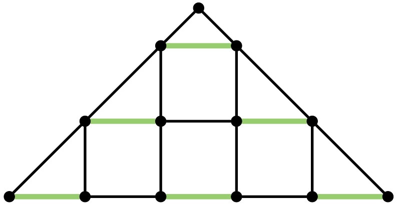
A C_k-**transverse matching** M is one which breaks every cycle of length k



Not a C_4-transverse matching.

Definition

A C_k-**transverse matching** M is one which breaks every cycle of length k



C_3- and C_4-transverse, but not C_5-transverse

tie-in with previous talk

Tatami tilings

Image from flickr user Joel Abroad

Tatami tilings

Image from Le Blog du Judo Club de St James

tatami mats meet in T, not +

Tatami tilings

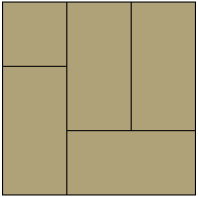
Image from Le Blog du Judo Club de St James

tatami mats meet in T, not +

"auspicious arrangements" date from Edo period (1603-1868)

Tatami tilings

Image from Le Blog du Judo Club de St James



tatami mats meet in T, not +

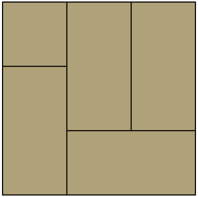
"auspicious arrangements" date from Edo period (1603-1868)

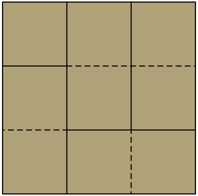
Tatami tilings

Image from Le Blog du Judo Club de St James

tatami mats meet in T, not +

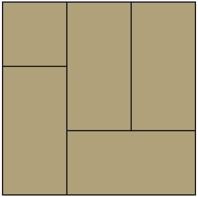
"auspicious arrangements" date from Edo period (1603-1868)

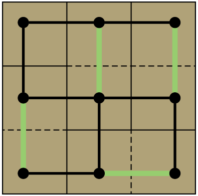




Tatami tilings

Image from Le Blog du Judo Club de St James





tatami mats meet in T, not +

"auspicious arrangements" date from Edo period (1603-1868)

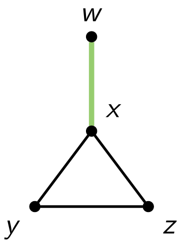
complexity of finding C_k-transverse matchings

complexity of finding C_k-transverse matchings

$$k = 3$$

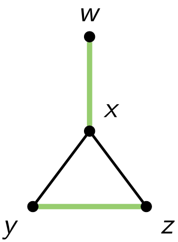
Finding C_3-transverse matchings

If M is a C_3-transverse matching containing wx



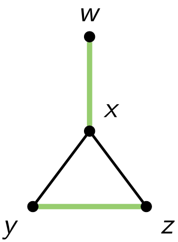
Finding C_3-transverse matchings

If M is a C_3-transverse matching containing wx then it also contains yz.



Finding C_3-transverse matchings

If M is a C_3-transverse matching containing wx then it also contains yz.



Definition / horrible pun

If M has this property, it is a **pawssible matching**.

Proposition

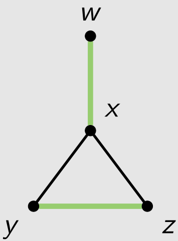
Let M, M_e be pawssible matchings such that $e \in M_e$ is minimal. If $M \cup e$ is a matching, then $M \cup M_e$ is a pawssible matching.

Proposition

Let M, M_e be pawssible matchings such that $e \in M_e$ is minimal. If $M \cup e$ is a matching, then $M \cup M_e$ is a pawssible matching.

Proof

Let $e = wx$ as shown; then $e' = yz \in M_e$.

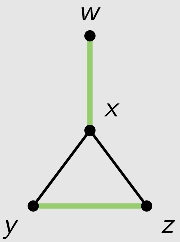


Proposition

Let M, M_e be pawssible matchings such that $e \in M_e$ is minimal. If $M \cup e$ is a matching, then $M \cup M_e$ is a pawssible matching.

Proof

Let $e = wx$ as shown; then $e' = yz \in M_e$. Either $e' \in M$ or M contains no edge incident with x, y, z.

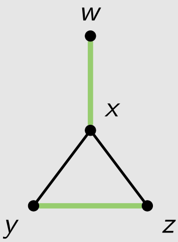


Proposition

Let M, M_e be pawssible matchings such that $e \in M_e$ is minimal. If $M \cup e$ is a matching, then $M \cup M_e$ is a pawssible matching.

Proof

Let $e = wx$ as shown; then $e' = yz \in M_e$. Either $e' \in M$ or M contains no edge incident with x, y, z. In any case, $M \cup \{e, e'\}$ is a matching.



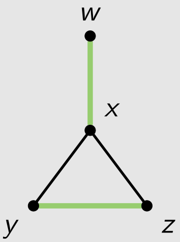
Proposition

Let M, M_e be pawssible matchings such that $e \in M_e$ is minimal. If $M \cup e$ is a matching, then $M \cup M_e$ is a pawssible matching.

Proof

Let $e = wx$ as shown; then $e' = yz \in M_e$. Either $e' \in M$ or M contains no edge incident with x, y, z. In any case, $M \cup \{e, e'\}$ is a matching.

Since M_e is minimal, we can repeat this argument to show that $M \cup M_e$ is a matching.



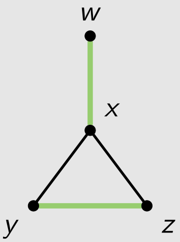
Proposition

Let M, M_e be pawssible matchings such that $e \in M_e$ is minimal. If $M \cup e$ is a matching, then $M \cup M_e$ is a pawssible matching.

Proof

Let $e = wx$ as shown; then $e' = yz \in M_e$. Either $e' \in M$ or M contains no edge incident with x, y, z. In any case, $M \cup \{e, e'\}$ is a matching.

Since M_e is minimal, we can repeat this argument to show that $M \cup M_e$ is a matching. In fact, it is a pawssible matching.



Proposition

Let M, M_e be pawssible matchings such that $e \in M_e$ is minimal. If $M \cup e$ is a matching, then $M \cup M_e$ is a pawssible matching.

Corollary

If M is a pawssible matching, then it can be extended to a C_3-transverse matching (provided G admits one).

Proposition

Let M, M_e be pawssible matchings such that $e \in M_e$ is minimal. If $M \cup e$ is a matching, then $M \cup M_e$ is a pawssible matching.

Corollary

If M is a pawssible matching, then it can be extended to a C_3-transverse matching (provided G admits one).

Theorem

It can be determined in polynomial time whether a graph G admits a C_3-transverse matching.

complexity of finding C_k-transverse matchings

$$k \geq 4$$

Theorem

It is NP-complete to determine whether a graph G admits a C_k-transverse matching, for every $k \geq 4$.

Theorem

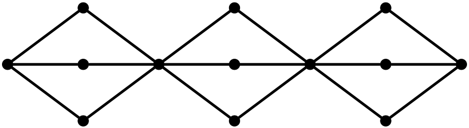
It is NP-complete to determine whether a graph G admits a C_k-transverse matching, for every $k \geq 4$.

Proof is a reduction from $\lceil \frac{k}{2} \rceil$-SAT (for $k \geq 5$).

Theorem

It is NP-complete to determine whether a graph G admits a C_k-transverse matching, for every $k \geq 4$.

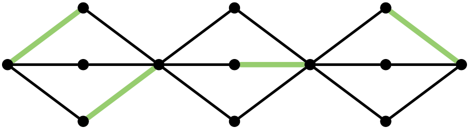
Proof is a reduction from $\lceil \frac{k}{2} \rceil$-SAT (for $k \geq 5$).



Theorem

It is NP-complete to determine whether a graph G admits a C_k-transverse matching, for every $k \geq 4$.

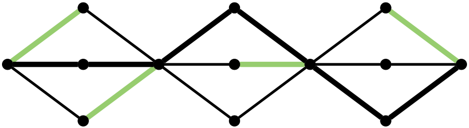
Proof is a reduction from $\lceil \frac{k}{2} \rceil$-SAT (for $k \geq 5$).



Theorem

It is NP-complete to determine whether a graph G admits a C_k-transverse matching, for every $k \geq 4$.

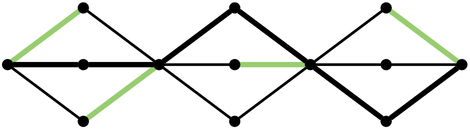
Proof is a reduction from $\lceil \frac{k}{2} \rceil$-SAT (for $k \geq 5$).



Theorem

It is NP-complete to determine whether a graph G admits a C_k-transverse matching, for every $k \geq 4$.

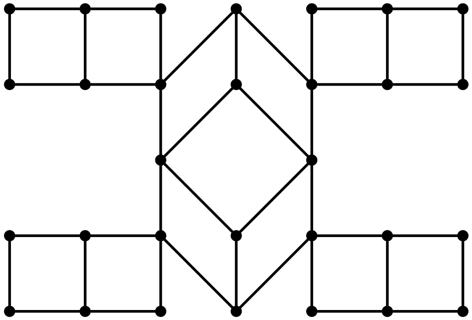
Proof is a reduction from $\lceil \frac{k}{2} \rceil$-SAT (for $k \geq 5$).



Build a cycle out of a path P and an "accordion path." Then any C_k-transverse matching contains at least one edge from P.

C_4-transverse matchings

Unfortunately, accordion paths don't work for $k = 4$. Reduction is from 4-SAT with ad-hoc gadgets:



graph partition problems

Problem: partition the vertices of a graph into A, B inducing F_A-free and F_B-free subgraphs, respectively.

Problem: partition the vertices of a graph into A, B inducing F_A-free and F_B-free subgraphs, respectively.

F_A	F_B	Chordal	Perfect	Line	Claw-free	All graphs
K_2	P_3	P	NPC	P	P	NPC

Problem: partition the vertices of a graph into A, B inducing F_A-free and F_B-free subgraphs, respectively.

F_A	F_B	Chordal	Perfect	Line	Claw-free	All graphs
K_2	P_3	P	NPC	P	P	NPC
K_2	C_3					
K_2	C_{k+4}					

Problem: partition the vertices of a graph into A, B inducing F_A-free and F_B-free subgraphs, respectively.

F_A	F_B	Chordal	Perfect	Line	Claw-free	All graphs
K_2	P_3	P	NPC	P	P	NPC
K_2	C_3					
K_2	C_{k+4}	Y	Y/NPC	NPC	NPC	NPC

Problem: partition the vertices of a graph into A, B inducing F_A-free and F_B-free subgraphs, respectively.

F_A	F_B	Chordal	Perfect	Line	Claw-free	All graphs
K_2	P_3	P	NPC	P	P	NPC
K_2	C_3			P		
K_2	C_{k+4}	Y	Y/NPC	NPC	NPC	NPC

Problem: partition the vertices of a graph into A, B inducing F_A-free and F_B-free subgraphs, respectively.

F_A	F_B	Chordal	Perfect	Line	Claw-free	All graphs
K_2	P_3	P	NPC	P	P	NPC
K_2	C_3			P	?	NPC
K_2	C_{k+4}	Y	Y/NPC	NPC	NPC	NPC

Problem: partition the vertices of a graph into A, B inducing F_A-free and F_B-free subgraphs, respectively.

F_A	F_B	Chordal	Perfect	Line	Claw-free	All graphs
K_2	P_3	P	NPC	P	P	NPC
K_2	C_3	P	P	P	?	NPC
K_2	C_{k+4}	Y	Y/NPC	NPC	NPC	NPC

Problem: partition the vertices of a graph into A, B inducing F_A-free and F_B-free subgraphs, respectively.

F_A	F_B	Chordal	Perfect	Line	Claw-free	All graphs
K_2	P_3	P	NPC	P	P	NPC
K_2	C_3	P	P	P	?	NPC
K_2	C_{k+4}	Y	Y/NPC	NPC	NPC	NPC
K_2	K_{k+4}	P	P	P	?	NPC

Problem: partition the vertices of a graph into A, B inducing F_A-free and F_B-free subgraphs, respectively.

F_A	F_B	Chordal	Perfect	Line	Claw-free	All graphs
K_2	P_3	P	NPC	P	P	NPC
K_2	C_3	P	P	P	?	NPC
K_2	C_{k+4}	Y	Y/NPC	NPC	NPC	NPC
K_2	K_{k+4}	P	P	P	?	NPC
K_2	co-P_3	P	P	P	P	P
K_2	co-C_3	P	P	P	P	P

Problem: partition the vertices of a graph into A, B inducing F_A-free and F_B-free subgraphs, respectively.

F_A	F_B	Chordal	Perfect	Line	Claw-free	All graphs
K_2	P_3	P	NPC	P	P	NPC
K_2	C_3	P	P	P	?	NPC
K_2	C_{k+4}	Y	Y/NPC	NPC	NPC	NPC
K_2	K_{k+4}	P	P	P	?	NPC
K_2	co-P_3	P	P	P	P	P
K_2	co-C_3	P	P	P	P	P
K_2	$2C_3$	?	?	?	?	?

Complexity of cycle-transverse matching problems

IWOCA 2011

Ross Churchley

University of Victoria

June 18, 2011