Fast exact algorithms for hamiltonicity in claw-free graphs

Hajo Broersma[1]
Fedor Fomin[2]
Pim van 't Hof[1]
Daniël Paulusma[1]

Durham University[1]

University of Bergen[2]

WG 2009

Introduction

$G = (V, E)$ is finite, undirected graph, no loops, no multiple edges.

G is claw-free if G does not contain an induced claw, which is a

$$K_{1,3} = (\{u, a, b, c\}, \{ua, ub, uc\}).$$

Goal: find a hamiltonian cycle of a claw-free graph.

Theorem (Li, Corneil & Mendelsohn, 2000)

Deciding if a graph has a hamiltonian cycle is NP-complete even for the class of 3-connected 3-regular claw-free planar graphs.

We aim for an exact algorithm.

Known Results

Theorem (Karp, 1982)

HAMILTONIAN CYCLE *can be solved using* $O^*(2^n)$ *time and polynomial space for any n-vertex graph.*

Here, O^*-notation suppresses factors of polynomial order.

Major open problem:

Can HAMILTONIAN CYCLE *be solved in* $O^*(\alpha^n)$ *time for* $\alpha < 2$*?*

Even unknown if polynomial space is dropped.

For the following graph classes, faster exact algorithms are known for solving HAMILTONIAN CYCLE:

- $O^*(c^{\sqrt{n}})$ time for some constant c for planar graphs [Deineko, Klinz & Woeginger, 2006]
- $O^*(1.251^n)$ time for cubic graphs [Iwama & Nakashima, 2007]
- $O^*(1.733^n)$ time for graphs with maximum degree 4 [Gebauer, 2008]
- $O^*((2-\epsilon)^n)$ time with $\epsilon > 0$ for graphs with bounded degree [Björklund, Husfeldt, Kaski & Koivisto, 2008]

The last three results are valid for the more general TRAVELING SALESMAN problem.

What about claw-free graphs?

We present two algorithms for Hamiltonian Cycle on n-vertex claw-free input graphs:

- an algorithm that uses $O^*(1.6818^n)$ time and exponential space
- an algorithm that uses $O^*(1.8878^n)$ time and polynomial space

Our Approach

We use the same approach for both algorithms:

1. Translate a (claw-free) instance of Hamiltonian Cycle to an instance of Dominating Closed Trail.
2. Solve Dominating Closed Trail.
3. Translate a solution of Dominating Closed Trail to a solution of Hamiltonian Cycle.

Performing 1 and 3 can be done in polynomial time; this is known already.

Performing 2 costs us exponential time; this is the new part and in this part the two algorithms are different from each other.

Outline of the Algorithms

Input: a claw-free graph G

Output: YES if G has a hamiltonian cycle, NO otherwise.

Step 1: compute the closure $cl(G)$ of G

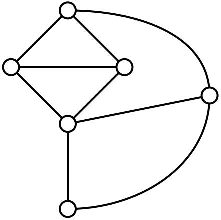
For each $x \in V_G$, the subgraph $G[N(x)]$ of G induced by $N(x)$ has at most two components.

- If $G[N(x)]$ has two components, both of them must be cliques. Do nothing.
- If $G[N(x)]$ is connected, add edges between all pairs of nonadjacent vertices in $N(x)$.

The *closure* $cl(G)$ of G is obtained by recursively repeating this operation, as long as this is possible.

Observe that $cl(G)$ is obtained in polynomial time.

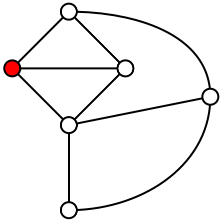
Example



G

Ryjáček [1997]:
$cl(G)$ is uniquely determined.

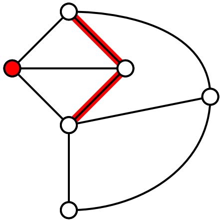
Example



G

Ryjáček [1997]:
$cl(G)$ is uniquely determined.

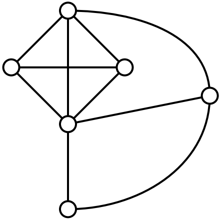
Example



G

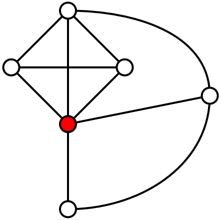
Ryjáček [1997]:
$cl(G)$ is uniquely determined.

Example



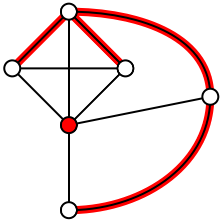
Ryjáček [1997]:
$cl(G)$ is uniquely determined.

Example



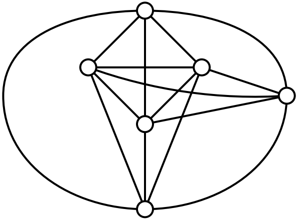
Ryjáček [1997]:
$cl(G)$ is uniquely determined.

Example



Ryjáček [1997]:
$cl(G)$ is uniquely determined.

Example



$$cl(G) = K_6$$

Ryjáček [1997]:
$cl(G)$ is uniquely determined.

Step 2: compute the triangle-free graph H with $L(H) = cl(G)$.

The line graph of a graph H with edges $e_1, \ldots, e_n$ is the graph $L(H)$ with vertices $u_1, \ldots, u_n$ such that

$$u_i u_j \in E_{L(H)} \Leftrightarrow e_i \text{ and } e_j \text{ share an end vertex in } H.$$

Ryjáček [1999]:
for a claw-free graph G, there is a triangle-free graph H such that $L(H) = cl(G)$.

We call H the preimage graph of G.

Due to Roussopolous [1973]:
H is unique and can be computed in polynomial time.

Step 3: detect a DCT of H

A graph is even if all its vertices have even degree.

A graph is a closed trail (or eulerian) if it is connected and even.

A closed trail T dominates H if $V_H \setminus V_T$ is independent set in H.

Then every edge of H has at least one vertex in T, so here:

dominating means *edge-dominating*.

In that case T is a dominating closed trail (DCT).

Due to Harary & Nash-Williams [1965]:
$cl(G)$ has a hamiltonian cycle if and only if H has a DCT.

How to detect a DCT of H will be explained later.

Step 4: translate DCT of H back into hamiltonian cycle of $cl(G)$

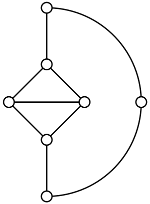
Traverse T using the polynomial-time algorithm that finds a eulerian tour in an even connected graph (cf. [Diestel, 2000]).

This way pick up edges ($=$ vertices in $cl(G)$) one by one and insert dominated edges as soon as an end vertex of a dominated edge is encountered.

Step 5: translate hamiltonian cycle in $cl(G)$ to one in G

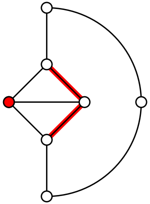
Use the polynomial time algorithm of Broersma & P. [2008].

Example.



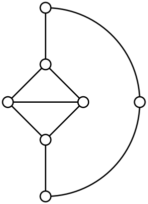
G

Example.

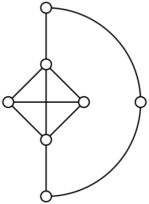


G

Example.

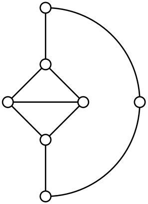


G

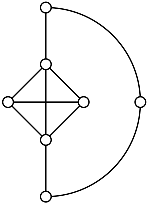


$cl(G) = L(H)$

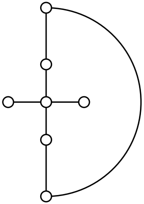
Example.



G

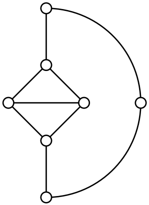


$cl(G) = L(H)$

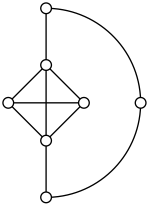


H

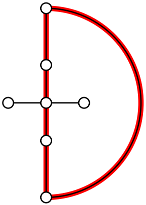
Example.



G

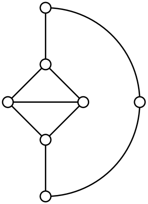


$cl(G) = L(H)$

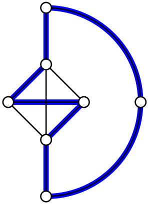


H

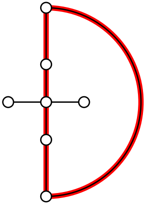
Example.



G

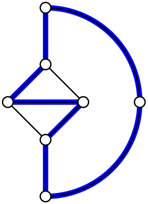


$cl(G) = L(H)$

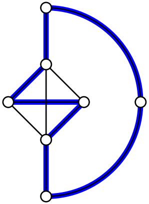


H

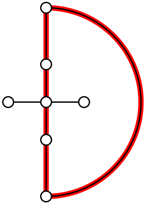
Example.



G



$cl(G) = L(H)$



H

Remaining Task

We need to find the DCT of a preimage graph H in Step 3 in

- $O^*(1.6818^n)$ time and exponential space (algorithm 1)
- $O^*(1.8878^n)$ time and polynomial space (algorithm 2)

Recall that $n = |E_H| = |V_G|$.

We obtain such algorithms by making use of

- a structural result by which we only have to search for a DCT with relatively few edges.

Structural Result on Closed Trails

For $k \geq 1$, a graph H is k-degenerate if every non-empty subgraph of H has a vertex of degree at most k.

H is k-ordered if H allows a vertex ordering $\pi = v_1, \ldots, v_{|V_H|}$ such that for $1 \leq i \leq |V_H|$:

- $H[\{v_1, \ldots, v_i\}]$ is connected
- v_i has at most k neighbors in $H[\{v_1 \ldots, v_i\}]$.

Theorem

Every graph with a spanning closed trail contains a 2-degenerate 3-ordered spanning closed trail.

So if such a graph has p vertices, it contains a spanning closed trail with at most $2p$ edges (due to the 2-degeneracy).

Algorithm 1 for detecting DCT in H

Phase 1. As long as H contains a vertex v with $d(v) \leq 4$:

1. Guess the set of DCT edges incident with v.
2. Adjust the parities of the neighbors of v.
3. Remove v.

This leads to a phase-2 tuple $(H', W(H'), \ell)$, where

- H' is the remaining graph (with minimum degree at least 5).
- $W(H')$ is the total set of guessed DCT edges
- ℓ is the parity labeling of the vertices in $V(H')$.

With each phase-2 tuple $(H', W(H'), \ell)$ enter Phase 2.

Phase 2. Use dynamic programming:

Given a pair (S, ℓ), how can $v \in V(H')$ be connected to $S \subset V_{H'}$ with DCT edges?

Lemma

Phase 1 creates $O^*(1.6818^{n_1})$ phase-2 tuples $(H', W(H'), \ell)$, where n_1 is the total number of deleted edges, and $|V_{H'}| \leq \frac{2(n-n_1)}{5}$.

Lemma

We may assume that in Phase 2 a vertex $v \in V(H')$ will be connected to a set $S \subset V_{H'}$ with at most three edges.

The last lemma follows from our *3-ordered* result on closed trails.

Together they ensure that Algorithm 1 uses $O^*(1.6818^n)$ time.

Due to Phase 2, Algorithm 1 may use exponential space.

Algorithm 2 for detecting DCT in H

Phase 1. As long as H contains a vertex v with $d(v) \leq 12$:

1. Guess the set of DCT edges incident with v.
2. Adjust the parities of the neighbors of v.
3. Remove v.

This leads to a phase-2 tuple $(H', W(H'), \ell)$, where

- H' is the remaining graph (with minimum degree at least 13).
- $W(H')$ is the total set of guessed DCT edges
- ℓ is the parity labeling of the vertices in $V(H')$.

With each phase-2 tuple $(H', W(H'), \ell)$ enter Phase 2.

Phase 2. Guess the remaining edges of a DCT.

Lemma

Phase 1 creates $O^*(1.8878^{n_1})$ *phase-2 tuples* $(H', W(H'), \ell)$*, where* n_1 *is the total number of deleted edges, and* $|V_{H'}| \leq \frac{2(n-n_1)}{13}$*.*

Lemma

We may assume that the set of guessed edges in Phase 2 has size at most $2|V_{H'}|$*.*

The last lemma follows from our *2-degenerate* result on closed trails.

Together they ensure that Algorithm 2 uses $O^*(1.8878^n)$ time and polynomial space.

Open problems

1. Can we speed up the algorithms by making use of the triangle-freeness of the preimage graph H?

2. Can Traveling Salesman be solved for claw-free graphs in $O^*(\alpha^n)$ time for some constant $\alpha < 2$?

3. Can Hamiltonian Cycle be solved in $O^*(\alpha^n)$ time for some constant $\alpha < 2$ for
   - chordal bipartite graphs?
   - $K_{1,4}$-free graphs?